Liner Shipping Network Design

Marielle Christiansen and Erik Hellsten and David Pisinger and David Sacramento and Charlotte Vilhelmsen

1 Introduction

Maritime transportation enables transportation of large volumes at relatively low costs and is therefore a fundamental part of the world trade. It is estimated that around 90 percent of world trade today is carried by the international shipping industry. Additionally, efficient port structures make it possible to combine sea transportation with land-based modes of transportation.

Roughly speaking, liner shipping is the service of transporting large volumes of cargo by means of high-capacity vessels that follow regular routes on fixed schedules. Within this type of shipping service, the variety of vessel types can be split into vessels designed for Lift-on/Lift-off (LoLo) operations, and Roll-on/Roll-off (RoRo) operations. In LoLo operations, quay cranes located on the docks are used to load and unload the vessels' cargo, while RoRo vessels are designed to carry cargo that can be rolled on and off the vessels.

Marielle Christiansen
Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology, Trondheim, Norway, e-mail: marielle.christiansen@iot.ntnu.no

Erik Hellsten
DTU Management, Technical University of Denmark, Akademivej 358, DK-2800 Kgs. Lyngby, Denmark, e-mail: erohe@dtu.dk

David Pisinger
DTU Management, Technical University of Denmark, Akademivej 358, DK-2800 Kgs. Lyngby, Denmark, e-mail: dapi@dtu.dk

David Sacramento
DTU Management, Technical University of Denmark, Akademivej 358, DK-2800 Kgs. Lyngby, Denmark, e-mail: dsle@dtu.dk

Charlotte Vilhelmsen
DTU Management, Technical University of Denmark, Akademivej 358, DK-2800 Kgs. Lyngby, Denmark, e-mail: chaandtu.dk

In this chapter, we mainly focus on *containerised* liner shipping network design, i.e., networks using LoLo operations, though we also briefly introduce network design for RoRo liner shipping. Although we have tried to cover most of the relevant models and algorithms dealing with containerised liner shipping network design, we have chosen to focus on those that seem to be applicable in practice.

This chapter is organised as follows: In Section 2 we give a brief introduction to containerised liner shipping, RoRo liner shipping, and network design. We also introduce the LINER-LIB test instances for network design in containerised liner shipping. The following sections are focused on containerised liner shipping. In Section 3 we discuss the challenges in designing a liner shipping network, and give an introduction to the models and algorithms, which will be presented in the following sections. In Section 4 we give an overview of integrated Mixed Integer Programming (MIP) models, while Section 5 studies two-stage algorithms where the service generation and the flowing of containers are separated in two steps. Section 6 presents the bibliographical notes of the topics addressed in this chapter. The chapter is concluded in Section 7 with a short discussion of future trends and challenges. Finally, an overview of the notation used throughout the chapter is found in Appendix 8. Some parts of this chapter are based on the authors' previous survey work presented in Hellsten et al. (2019) and Christiansen et al. (2019).

2 Overview of Liner Shipping and Liner Shipping Network Design

Maritime transportation is a cost-effective means of transportation, as it offers the transport of large volumes of cargo all around the world. The cargo vessels in the maritime industry are generally divided into the following major categories: general cargo ships, bulk cargo carriers, tankers, container ships and RoRo ships. This section presents the network design for containerised liner shipping, which are based on LoLo operations, as well as a brief description of RoRo liner shipping.

2.1 Containerised liner shipping

The liner shipping industry is a vital part of the global economy, constituting one of the cheapest modes of cargo transportation. During the last three decades, the volume of containerised cargo has grown by more than 8% per year, and more than 5,200 container vessels were in operation worldwide in 2019. Standard containers come in two different sizes, twenty and forty feet, which have given rise to the standard measures of containerised cargo, *twenty foot equivalent units* (TEU) and *forty foot equivalent units* (FFE). The largest vessels carry more than 20,000 TEU and during 2016, a container volume of around 140,000,000 TEU was estimated to pass through the vast liner shipping network (Unctad, 2018, 2019). Clearly, any improvement in the network design in the liner shipping industry will correspond to enormous savings.

The liner shipping industry is built up by so called *services*. A service is a fixed cyclic itinerary, sailed by a number of similar vessels. Services usually have weekly or biweekly departures to add consistency and regularity for the customers. The vessels are operated by shipping companies called carriers, where the largest carriers operate over 600 vessels. The trend is to build ever larger vessels, as they are significantly more energy efficient, see Figure 1. To efficiently utilise those tremendous liner vessels, each region typically has a few larger ports, called *hubs*, where the liner vessels pick up and deliver containers. From the hubs, the containers are then transported to other ports by smaller, more flexible, so called *feeder vessels*. The transfer of a container, from one vessel to another in a port, is called a *transshipment*. Transshipments occur both between larger vessels and smaller vessels, but also between larger vessels when no suitable service connects the origin and destination hub. While transshipments add flexibility, they tend to be costly, as the cargo needs to be unloaded, stored until the arrival of the new vessel and then reloaded again. Finally, to protect the national trade business, many countries forbid foreign carriers to ship cargo between two ports within the country. This is called *cabotage rules*.

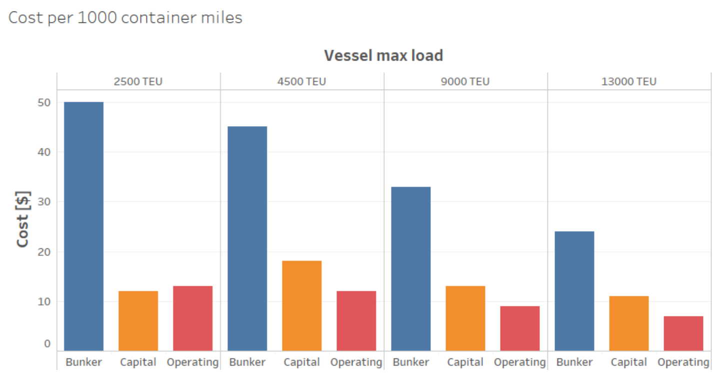

Fig. 1 Estimated cost per 1,000 container miles for different vessel sizes. The vessels are assumed to sail at 19 knots and the bunker fuel price is estimated as 750 $/tonne. We see that bunker represents the largest cost and that transporting containers on larger vessels requires significantly less fuel. Data has been obtained from Germanische Lloyd (2017).

The major costs for the carriers are vessel acquirement and bunker fuel. However, other costs, like canal fees, port costs, and transshipment costs, are also highly significant. The fuel consumption is frequently estimated as a cubic function of the speed, as seen in Figure 2. As the speed has such an impact on the fuel consumption, *slow steaming* is often used to reduce the consumption. Especially after the financial crisis in 2008, maritime shipping companies implemented slow steaming policies for cost-cutting purposes. The obvious drawback of slow steaming is that

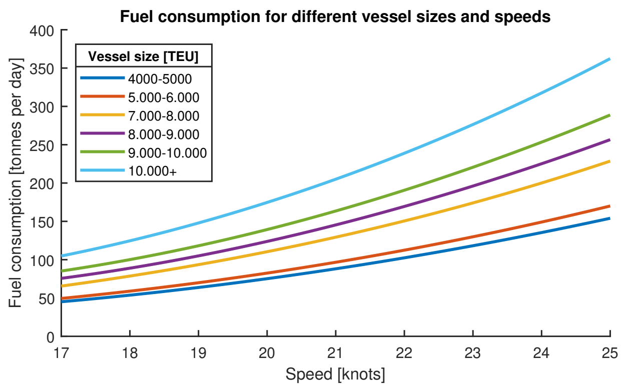

Fig. 2 Estimated fuel consumption as a function of steaming speed and vessel size. Data has been obtained from Notteboom and Vernimmen (2009).

more vessels are required to transport the same amount of cargo and also, that transit times become longer, yielding a lower level of service for the customers. In general, services have two directions, *head-* and *back-haul*, where most of the cargo is transported in the head haul direction. A good example of this is the trade between Asia and Europe, where most of the goods are delivered from Asia to Europe. In this case, vessels are slow steaming in the back-haul direction where less customers are affected by the increased transit time.

2.2 Containerised liner shipping network design

The Liner Shipping Network Design Problem (LSNDP) can be defined as follows: Given a collection of ports, a fleet of container vessels and a group of origin-destination demands. A set of services is constructed for the container vessels such that the overall operational expenses are minimised, while ensuring that all demands can be routed through the resulting network from their origin to their destination, respecting the capacity of the vessels.

In the following, we present some notation of the LSNDP that will be used throughout the chapter, introducing the necessary notation when required. A table of notation can be found in Table 2 in the Appendix.

The set of ports is denoted by N and represents the set of physical ports in the problem. The set of arcs A represents all possible sailings between ports. The set of commodities is denoted K and for each commodity $k \in K$, there is an origin port o_k, a destination port d_k, as well as a quantity q_k measured in TEU. Furthermore, the corresponding unit-cost for transporting a unit of commodity k through arc $(i, j) \in A$ is defined as c_{ij}^k. Finally, the set V denotes the set of *vessel classes*. For each $v \in V$ there is a corresponding cargo capacity u_v in number of TEUs, an available fleet quantity m_v, and additional speed limitations and fuel consumption parameters.

Furthermore, for convenience, the demand of the commodities in each port $i \in N$ is defined as:

$$\xi_i^k = \begin{cases} q_k & \text{if port } i \text{ is the origin port of commodity } k \\ -q_k & \text{if port } i \text{ is the destination port of commodity } k \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

There is a limited fleet of container vessels, but not all vessels need to be used. The deployment of a vessel $v \in V$ has an associated *charter cost* c^v. Additionally, there are other costs related with the resulting network, such as the *sailing cost* c_{ij}^v associated with each vessel and each arc, which is given as a combination of the port call cost and the fuel consumption for the corresponding arc. When containers of commodity $k \in K$ are transferred from one vessel to another in port $i \in N$, there is a *transshipment cost* c_{ik}^{T} for each container. Furthermore, there is an associated sailing time t_{ij}^v for each container vessel v sailing between ports i and j, which is given as a combination of its design speed and the corresponding distance between the ports. Moreover, each port $i \in N$ has an associated berthing time b_i.

One of the main traits of the liner shipping industry is the regular operation of services under a pre-established schedule. Sometimes it is possible to define the set of candidate services in advance. In these cases, let S be the set of feasible services in the model. Notice that S can be exponentially large. Each service $s \in S$ has an associated operational cost c_s. As the set of services is defined beforehand, the operational cost is given as a combination of the sailing cost of the arcs on the service route and the corresponding port-call costs. Moreover, it is required that all services should have *weekly operations*, meaning that if a round trip takes eight weeks to complete, then eight similar vessels need to be deployed to the service in order to ensure that each port is visited once a week. Therefore, the required number of vessels from vessel class v to maintain the weekly frequency is defined as m_v^s. The structure of the service can be divided into several types according to the number of times a port is visited during the service. A *simple service* or a *circular service* visits each port in the service exactly once. However, a service is often allowed to be non-simple, meaning that a port can be visited several times, as this may improve transit times. Nodes (or ports in the sequence) that are visited several times in the service are denoted *butterfly nodes* and a service containing a single butterfly node defines a *butterfly service*. Another common service type is the *pendulum service*, in which each port is visited twice, once in each direction. Examples of the different type of services for some European ports are illustrated in Figures 3, 4 and 5. Moreover, in some cases, the structure of the service can naturally be defined by the geographical distribution of the ports. For instance, if the ports are located along the coastline, it can be convenient to define services following an outbound-inbound principle. Such services are similar to pendulum services; however, they may be asymmetric, as some ports can be omitted in each direction, as illustrated in Figure 6.

When designing a shipping network, different variants of the LSNDP can be considered, varying mainly in the following four respects:

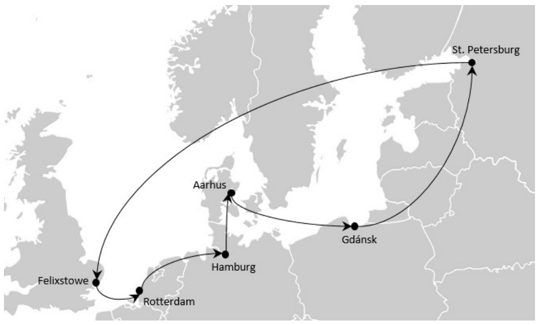

Fig. 3 Example of a simple service, where each port is visited exactly once.

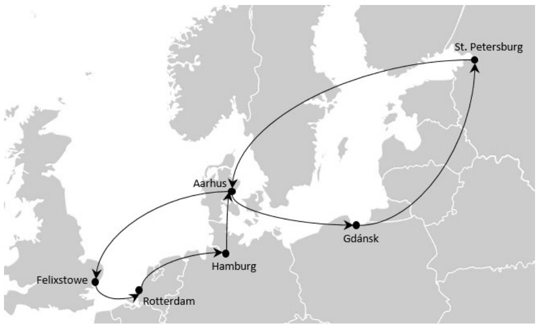

Fig. 4 Example of a butterfly service, where Aarhus is the butterfly node.

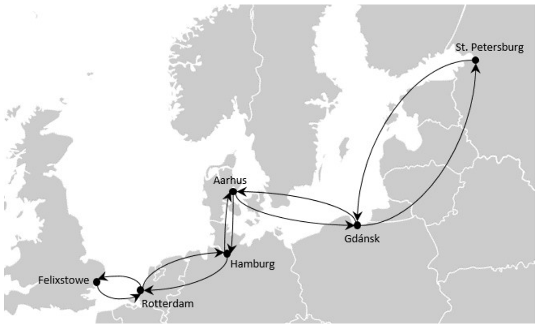

Fig. 5 Example of a pendulum service, where each port can be visited both on the head-haul and back-haul trip.

- *Transit time constraints.* The demands may be subject to transit times and hence have an associated time limit that must be respected. If the transit time is not respected, perishable goods may become spoiled.
- *Transshipment costs.* The costs of transshipments are a significant part of the operational costs, so it is generally important to represent these costs in the model.

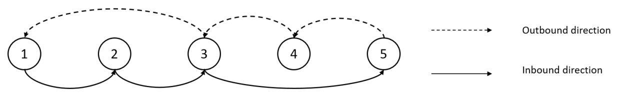

Fig. 6 Example of an asymmetric service with five ports with the outbound-inbound principle.

- *Rejected demands.* Although the standard formulation of LSNDP states that all demands must be flowed through the network, many models allow rejection of demands by imposing a penalty.
- *Speed optimisation*. There are three main approaches to model speed optimisation: Models which have constant speed for all services, models which choose a speed for each service, and models which choose a speed on each individual leg in each service. As the fuel consumption depends non-linearly on the speed, it is common to choose between a number of discrete speed alternatives, each with a corresponding cost.

Most models for LSNDP design the network without a specific schedule. Hence the service for each vessel is defined, but not the exact day of arrival/departure. This is typically done in a later step, where port availabilities are negotiated and transshipment times at ports are adjusted.

2.3 RoRo network design

RoRo shipping is an important segment within liner shipping. The RoRo ships are vessels designed to carry wheeled cargo, such as cars, trucks, semi-trailer trucks, and railroad cars, that can be driven on and off the ship on their own wheels. In addition, RoRo ships may carry complex cargo that is placed on trolleys and rolled on and off the ships, such as boats, helicopters, and heavy plant equipment. RoRo shipping is often the only viable method of ocean freight transportation for these oversized vehicles, as they may not fit in standard containers. There exist various types of RoRo ships, such as ferries, cruise ferries, cargo ships, and barges. In this subsection, we consider the RoRo ships used for transporting cars, trucks and complex general cargo across oceans known as Pure Car Carriers (PCC), Pure Truck & Car Carriers (PCTC) and general RoRo ships, respectively. A typical PCTC has a carrying capacity in the range of 5,500 to 8,000 RT43. Here, RT43 is a capacity measure in the RoRo business and corresponds to the size of a 1966 Toyota Corona. In 2016, the world fleet of RoRo ships consists of around 5,000 ships with a total capacity of more than 24 million deadweight tons (ISL, 2016).

The trades to be serviced in RoRo shipping are usually designed based on a large number of contracts for transportation of cargo between the different port pairs along a trade route. Hence, trade routes are defined as transportation arrangements from one geographical region to another, where the world is divided into a number of geographical regions. Each trade route has a number of loading ports in one region and a number of discharging ports in the other. In Figure 7, two trade routes are illustrated by solid lines and the ports are shown as filled circles. These trade routes are similar to the services in container network design. As seen in Figure 7, the trade routes are not traditionally circular. After a ship has sailed one voyage on a trade route it often needs to reposition to start on the next one due to trade imbalances. This repositioning means ballast sailing, i.e., sailing without cargo, which of course should be reduced as much as possible. The ballast sailing between the two trade routes in Figure 7 is illustrated by a dashed line. Differences in contractual requirements and variety in the types of cargo transported on the various trade routes may restrict which vessels can be assigned to a particular trade route, regarding both capacity and vessel type.

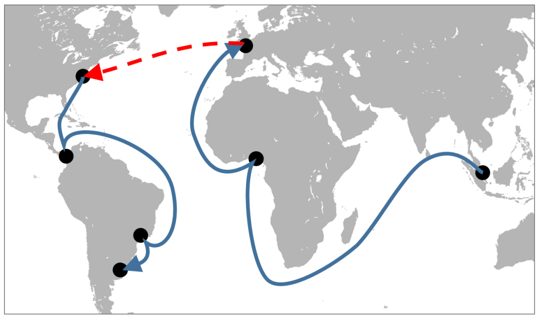

Fig. 7 Illustration of two trade routes sailed in sequence, Oceania to Europe and North to South America, with associated ballast sailing from Europe to North America. World map by San Jose under a CC-BY-SA-3.0 license.

Each trade route is sailed regularly, for example weekly, fortnightly, 3 times per months, depending on demand and contractual obligations. Each sailing on a trade route is called a voyage, and normally there is a time window to start sailing a voyage. Due to contractual obligations, these voyages are mandatory and must be covered either by a ship in the RoRo shipping company's own fleet or by a chartered ship. A RoRo shipping company owns and operates a heterogeneous fleet of ships having different cargo capacities, sailing speed ranges, and bunker consumption profiles, and serves a given set of trade routes.

The operations within RoRo shipping deviates from container shipping in several ways as well as in the cargo and ships. In container shipping, each ship is normally assigned to a single route, while in RoRo shipping a ship may sail several trade routes during a planning horizon. Ship types, instead of individual ships, are often considered in container shipping, while in RoRo shipping a route for each ship is determined. This also means that in RoRo shipping each trade route may be serviced by different ship types. Furthermore, there is a great variation in when to start each voyage, as well as when and how often to visit each port along the trade. Therefore, existing studies within fleet deployment in RoRo shipping have used time windows for when each voyage along each trade should start. This flexibility is in contrast to container shipping, as each service is usually served on a strict weekly basis, and each voyage along the trade visits all ports in the same order. Finally, transshipment rarely exist in RoRo shipping in contrast to container shipping. This makes the network design easier.

2.4 The LINER-LIB test instances

In order to make it easier to compare algorithms for LoLo liner shipping network design, Brouer et al. (2014a) introduced the LINER-LIB benchmark suite. The test instances in LINER-LIB are based on real-life data from leading shipping companies along with several other industry and public stakeholders. The benchmark suite contains data on ports including port call cost, cargo handling cost and draft restrictions, distances between ports considering draft and canal traversal, vessel related data for capacity, cost, speed interval and bunker consumption, and finally a commodity set with quantities, revenue, and maximal transit time. The commodity data is intended to reflect the differentiated revenue associated with the current imbalance of world trade.

The LINER-LIB benchmark suite consists of seven instances described in Brouer et al. (2014a) and is available at *http://www.linerlib.org*. The instances range from smaller networks suitable for being solved by exact solution methods to large scale instances spanning the globe. Table 1 gives an overview of these instances.

Instance	Category	$\lvert N\rvert$	$\lvert K\rvert$	$\lvert V\rvert$	min v	max v
Baltic	Single-hub	12	22	2	5	7
WestAfrica	Single-hub	19	38	2	33	51
Mediterranean	Multi-hub	39	369	3	15	25
Pacific	Trade-Lane	45	722	4	81	119
AsiaEurope	Trade-Lane	111	4,000	6	140	212
WorldSmall	Multi-hub	47	1,764	6	209	317
WorldLarge	Multi-hub	197	9,630	6	401	601

Table 1 The seven test instances included in LINER-LIB with indication of the number of ports ($\lvert N\rvert$), the number of origin-destination pairs ($\lvert K\rvert$), the number of vessel classes ($\lvert V\rvert$), the minimum (min v) and maximum number of vessels (max v) in each class.

Each of the instances can be used in a low, base, and high capacity case depending on the fleet of the instance. For the low capacity case, the fleet quantity and the weekly vessel costs are adjusted to fewer vessels with a higher vessel cost. For the high capacity case the adjustments are reversed.

Currently, most papers only report results for the base capacity case. Furthermore, most often only the six first instances are considered, with Krogsgaard et al. (2018) being the only paper to report results for the *WorldLarge* instance.

3 Overview of models and algorithms

Designing a liner shipping network is a difficult task, embracing several decisions: Not only do we need to construct the individual services, but we should also deploy vessels of the right size to each service and ensure that there is sufficient capacity in the network to transport all containers from their origin to their destination. Designing the individual services is an *NP*-hard problem. Furthermore, routing the containers through a given network subject to time constraints for each container, can be recognised as a time-constrained multi-commodity flow problem, which is also *NP*-hard.

The problem is further complicated by the fact that ports are often visited several times in the same service. This allows containers to quickly be transshipped to other services, and it frees up capacity. However, formulating the problem with multiple visits to a port as a MIP model becomes more difficult.

Finally, one should notice that transshipment costs represent the majority of the cost of routing the containers through the network according to Psaraftis and Kontovas (2015). It is therefore important to carefully model which containers might be transshipped and at which costs. This adds further complexity to the problem, and makes a graph formulation huge and difficult to solve.

First, we will focus on how the problem can be modeled. Generally speaking, *models* for liner shipping network design can roughly be divided into the following two groups:

- *Service selection models*. The idea behind these models is to use a large set of promising candidate services and then select a subset of them to create a network. The set of promising candidate services can both incorporate services designed by experienced planners as well as services internally generated by an algorithm. Many shipping companies and customers do not want the network to be completely restructured, in which case proposing small variations to each service may be a sensible method.
- *Arc formulation models*. These unified MIP models design services and flow containers through the resulting network. In order to handle this task, two sets of variables are needed: Binary variables to select edges in a service, and integer variables to denote the flow on each edge. If multiple visits to a node are allowed (butterfly nodes) then an additional index is needed to indicate the visit number at each node.

Many of the MIP models can in principle solve the LSNDP to optimality. However, due to the intrinsic complexity, only small instances can be solved to proven optimality within a reasonable time. The service selection based models more easily solve the problem to optimality given that only the proposed candidate services are valid. In practice, however, there may be an exponential number of valid services, and we cannot expect to get all services as input. Hence, the found solution will often be sub-optimal.

Large real world problems, in most cases, cannot be solved directly as MIP models. Therefore, it is required to use specialised algorithms to design the liner shipping network. Within this category, one of the most commonly used algorithms are the *two-stages algorithms*, which benefit from the decomposition of the original problem into two tightly related problems: the vessel service network design and the container flow problem. These algorithms can be roughly divided into the following two groups:

- *Service-first and flow-second algorithms*. As the name suggests, these algorithms model and solve the problem in two steps: Designing the services, and flowing containers through the resulting network. Frequently, these algorithms contain a feed-back mechanism, where output from the second-stage flow model is used as input to improve the services in the first stage.
- *Backbone flow algorithms*. It can be difficult to design the individual services without knowing how the containers will flow through the network. Hence, another approach is to reverse the order of the sub-problems in the two-stage algorithms, and start by finding an initial flow (a so-called backbone network) where cargo is flowed through a complete network with all connections between ports available. The connections are priced such that they are expensive at low loads and cheap at high loads, in order to make the cargo gather at few connections. The initial flow can be seen as an accomplishment of the *physical internet* (Montreuil, 2011) where point-to-point transport has been replaced by multi-segment intermodal transport.

In liner shipping network design problems, each stage corresponds to a single layer of decision: service selection decisions for the network design problem and continuous decisions for the container flow problem. It is important to notice that the two-stage algorithms are both heuristics, since they first solve one stage, and then optimise the second stage with the first-stage decisions fixed. However, this decomposition by stages make the problem more tractable for larger instances.

Most of the MIP models solve the network design problem by dealing simultaneously with the two layers of decisions. On the other hand, the two-stage algorithms, and most of the heuristic approaches, separate these layers when designing the shipping network. The way in which the two layers interact is what defines the core of the algorithm. For example, when container flow decisions are postponed to the second-stage, the service selection decisions of the first-stage can be made based on estimates for serving the ports. At each iteration of the algorithm, the estimates can be updated based on the current configuration of the resulting network after flowing the containers. In other cases, such as for models or algorithms based on Column Generation, new services can be iteratively constructed using the information obtained from the dual variables of a restricted problem.

4 Models for the LSNDP

In this section we present graph-based models to define the LSNDP. Different formulations are briefly introduced to model the network design problem in liner shipping and a summary of the main mathematical formulations proposed in the literature is presented under different assumptions.

4.1 Service selection formulations

This section introduces service flow formulations for the LSNDP, where the set of all feasible services is defined in advance for the model. This reduces the network design to the selection of feasible services.

Let us begin with introducing a basic service formulation. We use the terminology presented in Section 2.2 with the addition of the following definitions: Let $G = (N, A)$ be a directed graph. Define for each service s and for each arc (i, j), the associated capacity u^s_{ij}, which is given by the maximum cargo capacity of the corresponding vessel class assigned to the service. Finally, let x^{ks}_{ij} be a continuous variable denoting the amount of commodity k transported in service s on arc (i, j), and y_s a binary variable for the selection of service s in the network. Now, the service formulation of the LSNDP can be expressed as:

$$\min \quad \sum_{s\in S} c_s y_s + \sum_{k\in K}\sum_{(i,j)\in A} c^k_{ij} \sum_{s\in S} x^{ks}_{ij} \tag{2a}$$

$$\text{s.t.} \quad \sum_{s\in S}\sum_{j:(i,j)\in A} x^{ks}_{ij} - \sum_{s\in S}\sum_{j:(j,i)\in A} x^{ks}_{ji} = \xi^k_i \qquad i\in N, k\in K \tag{2b}$$

$$\sum_{k\in K} x^{ks}_{ij} \le u^s_{ij} y_s \qquad s\in S, (i,j)\in A \tag{2c}$$

$$\sum_{s\in S} m^s_v y_s \le m_v \qquad v\in V \tag{2d}$$

$$x^{ks}_{ij} \ge 0 \qquad (i,j)\in A, k\in K, s\in S \tag{2e}$$

$$y_s \in \{0,1\} \qquad s\in S. \tag{2f}$$

The objective function (2a) minimises the total operational cost of the network. The first term accounts for the fixed cost of the selected services, whereas the second term constitutes the sailing cost of shipping the demand. Constraints (2b) are the flow conservation constraints, and the flow of commodities on the arcs has to respect the capacity of the vessel deployed in the selected service s as formulated in constraints (2c). For each vessel class $v \in V$, constraints (2d) ensure that the number of deployed vessels on the selected services using vessel class v does not exceed the maximum availability m_v. Finally, the domain of the variables is defined in constraints (2e) and (2f).

For a better utilisation of the capacity of the deployed vessels, the model can also allow the rejection of cargo by incurring a penalty. Hence, extra continuous variables can be defined to account for the demand that is rejected by the liner company.

As stated, the model requires all demand to be served, but cargo rejection can be included by adding additional variables. Finally, to account for transshipment costs, additional variables f_i^{ks} could be added, denoting the transshipment of commodity k at port i from service s, with a corresponding term in the objective function. The following constraints (3) could be used to enforce the sought behaviour, though only if we consider simple services.

$$f_i^{ks} \geq \sum_{\substack{j \in N \\ j \neq i}} x_{ji}^{ks} - x_{ij}^{ks}, \qquad i \in N \setminus \{o_k, d_k\}, k \in K, s \in S. \tag{3}$$

This formulation allows designing networks considering only a subset of promising candidate services. It can easily be seen that, as the size of the problem increases, the number of feasible services grows exponentially, making the model intractable to solve. One possible approach to solve this problem for large instances is to apply a Column Generation algorithm.

4.1.1 A sub-path service formulation with limited transshipments

In the previous model, the flow of each commodity, on each sailing arc, is modelled explicitly. This leads to a compact formulation, but it turns out to be difficult to add transshipments together with complex service structures. Another approach, which allows limiting the number of transshipments for each commodity, while still permitting complex structures, is to use *sub-paths*. A sub-path is defined as the section of a service travelled by a group of containers. If this section spans the arcs from port i to port j on service s, the sub-path is denoted $\langle i, j, s \rangle$. The set H_s denotes the full set of sub-paths for service s, i.e., the set contains one sub-path $\langle i, j, s \rangle$ for each combination of ports i and j included in service s, and u_s denotes the capacity of service s.

Now, these sub-paths are used to introduce an augmented multi-commodity flow network. The augmented network contains one node for each port and one link for each sub-path of each service. The sub-path structure also extends to more complex services, e.g., butterfly services. Let A_{ij}^s denote the set of sailing arcs of service s included in sub-path $\langle i, j, s \rangle$. The cost of routing one container of commodity k on sub-path $\langle i, j, s \rangle$ is denoted c_{ijs}^k. Finally, r_k denotes the maximum allowed number of sub-paths on which commodity k can travel. By limiting the maximum number of sub-paths allowed for a commodity, we also implicitly limit the number of transshipments.

We now present a multi-commodity model based on flows along sub-paths in the augmented network. The binary variable y_s is equal to 1 if service $s \in S$ is selected, and 0 otherwise. The flow of commodity k using sub-path $\langle i, j, s \rangle$ as the h^{th} stage is defined by the variable x_{ijs}^{hk} for $s \in S$, $\langle i, j, s \rangle \in A_s$, and $h = 1, 2, \ldots, r_k$. Finally, z_k is equal to the unmet demand (number of containers) for commodity $k \in K$.

The sub-path service formulation can then be written as:

$$\min \sum_{s\in S} c_s y_s + \sum_{k\in K}\sum_{s\in S}\sum_{\langle i,j,s\rangle \in A_s}\sum_{h=1}^{r_k} c^k_{ijs} x^{hk}_{ijs} + \sum_{k\in K} c^R_k z_k \tag{4a}$$

$$\text{s.t.} \sum_{s\in S}\sum_{\langle o_k,j,s\rangle\in H_s} x^{1k}_{o_k js} + z_k = q_k \qquad \forall k \in K, \tag{4b}$$

$$\sum_{h=1}^{r_k}\sum_{s\in S}\sum_{\langle j,d_k,s\rangle\in H_s} x^{hk}_{jd_k s} + z_k = q_k \qquad \forall k \in K, \tag{4c}$$

$$\sum_{s\in S}\sum_{i:\langle i,j,s\rangle\in H_s} x^{hk}_{ijs} - \sum_{s\in S}\sum_{l:\langle j,l,s\rangle\in H_s} x^{h+1,k}_{jls} = 0 \quad \forall k \in K, j \in N\setminus\{o_k,d_k\}, h = 1,\dots,r_k-1, \tag{4d}$$

$$\sum_{k\in K}\sum_{h=1}^{r_k}\sum_{\langle i,j,s\rangle\in H_s: a\in A^s_{ij}} x^{hk}_{ijs} \le u_a y_s \qquad \forall s \in S, a \in A_s \tag{4e}$$

$$\sum_{s\in S} m^s_v y_s \le m_v \qquad \forall v \in V, \tag{4f}$$

$$x^{hk}_{ijs} \ge 0 \qquad \forall k \in K, s \in S, \langle i,j,s\rangle \in H_s, h = 1,\dots,r_k, \tag{4g}$$

$$z_k \ge 0 \qquad \forall k \in K, \tag{4h}$$

$$y_s \in \{0,1\} \qquad \forall s \in S. \tag{4i}$$

The objective function (4a) minimises the total cost comprised of fixed costs for the selected services, the cost of transporting commodities along each sub-path, and finally the penalties incurred for rejected demand. By including penalties, the problem is formulated as a cost minimisation problem as opposed to a profit maximisation problem where c^k_R would instead represent the revenue for transporting one unit of commodity k.

Constraints (4b) and (4c) ensure that the flow of each commodity k is assigned to sub-paths incident to the corresponding origin port o_k and the destination port d_k. They also ensure that the flow out of the origin port in combination with the unmet demand for commodity k adds up to the total demand for commodity k. Constraints (4d) are flow-balancing constraints for intermediate ports. Together with constraints (4b) and (4c) these constraints ensure that for each commodity k, the demand, minus any unmet demand, will arrive at the destination port using at most r_k sub-paths, i.e., fulfilling the constraint on a maximum number of transshipments.

Constraints (4e) impose capacity constraints on the sailing arcs and ensure that only sub-paths from the selected services are used. Constraints (4f) ensure that no more than the available vessels are used. Finally, constraints (4g)-(4i) impose non-negativity and binary restrictions on the respective decision variables.

The model formulation is flexible enough to allow incorporation of several practical container routing issues such as cabotage rules, regional policies and embargoes. The incorporation of many of these constraints can be handled during preprocessing simply by removing sub-paths that are no longer permitted. Balakrishnan and Karsten (2017) show that the problem is *NP*-hard.

4.2 Arc formulations

The main problem with a service-based formulation is that generating all services S is prohibitive, due to the high number of combinatorial possibilities. Therefore, an alternative compact formulation is introduced in this section, which is based on an arc formulation. The set of predefined services S is no longer considered in the model, but the services are designed as the problem is solved.

We first present a basic mathematical model based on an arc formulation. For this we again use the notation presented in Section 2.2 with small extensions. Let $G=(N,A)$ be a directed graph. Moreover, let S^v be an index set for the services for vessel class v, indexed by s. Let x_{ij}^{ks} be a continuous variable denoting the flow of commodity k on arc (i,j) by service s, and y_{ij}^{s} a binary variable for the selection of arc (i,j) in service s operated by vessel class v. The binary variable γ_i^s is equal to 1 if port i is the hub port in service S. Moreover, we define τ_i^s as a continuous variable for the time in service s of vessel class v departing from port i, and w_s as an integer variable indicating the number of vessels from class v needed to maintain the weekly frequency in service s. Then, the arc formulation of the LSNDP can be expressed as follows:

$$\min \sum_{v\in V}\sum_{s\in S^v} c^v w_s + \sum_{v\in V}\sum_{s\in S^v}\sum_{(i,j)\in A} c_{ij}^v y_{ij}^s + \sum_{k\in K}\sum_{(i,j)\in A} c_{ij}^k \sum_{v\in V}\sum_{s\in S^v} x_{ij}^{ks} \tag{5a}$$

$$\text{s.t.} \sum_{v\in V}\sum_{s\in S^v}\sum_{j:(i,j)\in A} x_{ij}^{ks} - \sum_{v\in V}\sum_{s\in S^v}\sum_{j:(j,i)\in A} x_{ji}^{ks} = \xi_i^k \qquad i\in N, k\in K \tag{5b}$$

$$\sum_{j:(i,j)\in A} y_{ij}^s - \sum_{j:(j,i)\in A} y_{ji}^s = 0 \qquad i\in N, v\in V, s\in S^v \tag{5c}$$

$$\sum_{i\in N} \gamma_i^s = 1 \qquad v\in V, s\in S^v \tag{5d}$$

$$\sum_{k\in K} x_{ij}^{ks} \le u_v y_{ij}^s \qquad (i,j)\in A, v\in V, s\in S^v \tag{5e}$$

$$\tau_j^s \ge (\tau_i^s + t_{ij}^v + b_j)(1-\gamma_j^s) y_{ij}^s \qquad i,j\in N, v\in V, s\in S^v \tag{5f}$$

$$\sum_{(i,j)\in A} (t_{ij}^s + b_j) y_{ij}^s \le 24\cdot 7\cdot w_s \qquad v\in V, s\in S^v \tag{5g}$$

$$\sum_{s\in S^v} w_s \le m_v \qquad v\in V \tag{5h}$$

$$x_{ij}^{ks} \ge 0 \qquad (i,j)\in A, k\in K, v\in V, s\in S^v \tag{5i}$$

$$y^s_{ij} \in \{0,1\} \qquad (i,j) \in A, v \in V, s \in S^v \tag{5j}$$
$$w_s \in \mathbb{Z}^+ \qquad v \in V, s \in S^v \tag{5k}$$
$$\tau^s_i \geq 0 \qquad i \in N, v \in V, s \in S^v. \tag{5l}$$

The objective function (5a) minimises the cost of deploying the vessels and designing the services, and the cost of transporting the demand quantities through the network. The flow conservation constraints for the cargo variables are given by constraints (5b), whereas the flow conservation constraints for the routing variables are given by constraints (5c). Constraints (5d) ensure that there is only a single hub port for each service. The flow of cargo on an edge (i,j) cannot exceed the capacity u_v of a vessel class, as expressed in (5e). If the service does not use a given edge in the graph, i.e., $y^s_{ij} = 0$, then the capacity is zero. The time schedule constraints for the routing variables are given by the time variables in constraints (5f). Note that it is necessary to linearise these constraints, as they are non-linear. Moreover, these constraints also ensure the elimination of sub-tours when designing the liner network, and prevents non-simple services. The weekly frequency of the services and the deployment of the fleet are expressed by constraints (5g). The availability of the fleet is limited by constraints (5h). Finally, the domain of the variables is defined by constraints (5i)-(5l).

Furthermore, the arc formulation may also include the rejection of cargo by defining continuous variables that account for the overall demand that is not flowed. Moreover, as this formulation only allows simple services, constraints (3) can also be included to account for transshipments.

This model is a simple representation of the arc formulation for the LSNDP, but it can be extended to incorporate the various constraints and considerations encountered in liner shipping. One of the main drawbacks with the formulation is that only simple services can be modelled. Next, we present an extended MIP model based on an arc-flow formulation which incorporates the network design and fleet assignment and accounts correctly for the transshipment cost in intermediate ports. Moreover, the model can handle the inclusion of butterfly services, where it is allowed to visit a single port at most twice during the service, and the capacity of the services dependent on the time horizon and the service length.

Let $G = (N,A)$ be a directed graph. Define the set V as the set of vessels, instead of the set of vessel classes. We consider each vessel v to belong to its own vessel class. Let t_{max} be the length of the time horizon. The design of the network is modelled with the binary variables y^v_{ij} for the utilisation of an arc (i,j) in the service for vessel v. Similarly, as proposed by Miller et al. (1960), positive integer variables e^v_{ij} are defined for enumerating the arcs used in the vessel service and avoid subtours in services. As the model allows the definition of butterfly services, the binary variables γ^v_i and z^v_{ij} are required to, respectively, identify the unique center-point, i.e., the hub port in the vessel service, and to allow the possibility of identifying the first and last arc visiting the hub port. These variables are used for modelling the transshipment of cargo in hub ports. The routing of containers through the network is modelled with continuous variables x^{kv}_{ij}, and extra continuous variables are defined to count the amount of the transshipped containers in intermediate ports within the same service. Let the continuous variables f_j^{kv} define the amount of commodity k transshipped by vessel v at port j, while the continuous variables f_{jih}^{kv} denote the amount of commodity k, arriving at port i through arc (j,i), in vessel v, and not leaving in arc (i,h). Furthermore, the model also considers the fleet deployment. The problem is defined for a heterogeneous fleet and the service capacities depend on the deployed vessels as well as the frequency at which they sail. The deployment of a vessel is controlled by the binary variable λ^v, whereas the continuous variables τ_v limit the service length of the vessels. Then, the arc-flow model can be defined as:

$$\min \sum_{k\in K} \sum_{(i,j)\in A} c_{ij}^k \sum_{v\in V} x_{ij}^{kv} + \sum_{k\in k} \sum_{i\in N} c_{ik}^{\mathrm{T}} \sum_{v\in V} f_i^{kv} + \sum_{v\in V} c^v \lambda^v \tag{6a}$$

$$\text{s.t.} \sum_{v\in V} \sum_{j:(i,j)\in A} x_{ij}^{kv} - \sum_{v\in V} \sum_{j:(j,i)\in A} x_{ji}^{kv} = \xi_i^k \qquad i\in N, k\in K \tag{6b}$$

$$f_i^{kv} \geq \sum_{j:(i,j)\in A} x_{ij}^k - \sum_{j:(j,i)\in A} x_{ji}^k \qquad k\in K, i\in N, v\in V \tag{6c}$$

$$f_i^{kv} \geq \sum_{j,h\in N} \sum_{v\in V} f_{jih}^{kv} - M_1(1-\gamma_i^v) \qquad k\in K, i\in N, v\in V \tag{6d}$$

$$f_{jih} \geq x_{ji}^{kv} - x_{ih}^{kv} - M_2(2 - y_{ji}^v - y_{ih}^v + z_{ji}^v + z_{ih}^v) \qquad k\in K, j,i,h\in N, v\in V \tag{6e}$$

$$f_{jih} \geq x_{ji}^{kv} - x_{ih}^{kv} - M_3(4 - z_{ji}^v - z_{ih}^v - y_{ji}^v - y_{ih}^v) \qquad k\in K, j,i,h\in N, v\in V \tag{6f}$$

$$\sum_{i\in N} \gamma_i^v = 1 \qquad v\in V \tag{6g}$$

$$\sum_{(i,j)\in A} z_{ij}^v = 2 \qquad v\in V \tag{6h}$$

$$\gamma_i^v - \sum_{j:(i,j)\in A} z_{ij}^v \leq 0 \qquad i\in N, v\in V \tag{6i}$$

$$\gamma_i^v - \sum_{j:(j,i)\in A} z_{ji}^v \leq 0 \qquad i\in N, v\in V \tag{6j}$$

$$\sum_{j:(i,j)\in A} y_{ij}^v - \sum_{j:(j,i)\in A} y_{ji}^v = 0 \qquad i\in N, v\in V \tag{6k}$$

$$\sum_{j:(i,j)\in A} y_{ij}^v - \gamma_i^v \leq 1 \qquad i\in N, v\in V \tag{6l}$$

$$e_{ji}^v - e_{ih}^v + M_4(y_{ih}^v + y_{ji}^v - 2 - z_{ji}^v - z_{ih}^v) \leq -1 \qquad i,j,h\in N, v\in V \tag{6m}$$

$$y_{ij}^v - \lambda^v \leq 0 \qquad (i,j)\in A, v\in V \tag{6n}$$

$$\tau_v \leq t_{max} \qquad v\in V \tag{6o}$$

$$\tau_v = \sum_{(i,j)\in A} y_{ij}^v (t_{ij}^v + b_j) \qquad v\in V \tag{6p}$$

$$\frac{t_{max}}{\tau_v} u_v y_{ij}^v \geq \sum_{k\in K} x_{ij}^{kv} \qquad (i,j)\in A, v\in V \tag{6q}$$

$$z_{ij}^{v}, y_{ij}^{v} \in \{0,1\} \qquad (i,j) \in A, v \in V \tag{6r}$$
$$f_{jih}^{kv} \geq 0 \qquad k \in K, j,i,h \in N, v \in V \tag{6s}$$
$$f_{j}^{kv} \geq 0 \qquad k \in K, j \in N, v \in V \tag{6t}$$
$$e_{ij}^{v} \in \mathbb{Z}^{+} \qquad i,j \in N, v \in V \tag{6u}$$
$$x_{ij}^{kv} \geq 0 \qquad (i,j) \in A, k \in K, v \in V \tag{6v}$$
$$\gamma_{i}^{v} \in \{0,1\} \qquad i \in N, v \in V \tag{6w}$$
$$\lambda^{v} \in \{0,1\} \qquad v \in V \tag{6x}$$
$$\tau_{v} \geq 0 \qquad v \in V. \tag{6y}$$

The objective function (6a) minimises the total cost of transporting the cargo through the network, the transshipment costs of the demand in hub ports, and the associated cost for deploying the vessels. Furthermore, the flow conservation constraints (6b) ensure that all demand is satisfied. Constraints (6c) account for the amount of containers transshipped in intermediate ports; however, if the corresponding service is non-simple, the model requires constraints (6d)-(6f) for updating the commodities transshipped by the same vessel in butterfly services. Constraints (6g)-(6j) are used to handle butterfly services. Constraints (6g) identify the unique butterfly node for the vessel service and constraints (6h)-(6j) find the adjacent arcs corresponding to the first or last visit to the butterfly node of the vessel service. Moreover, constraints (6k) are the flow conservation constraints for the network design of the vessel service, and constraints (6l) control the number of times a vessel visits the hub port in a service. In addition, constraints (6m) use the previous information for correctly enumerating the order in which the vessel traverses the arcs in the vessel service. Additionally, the fleet deployment is controlled by constraints (6n), and the corresponding service length is computed in constraints (6o) and (6p). It can also be seen that the service length is included in the capacity constraints (6q), as was first introduced in Agarwal and Ergun (2008). However, the model does not require weekly departures for all ports. The inclusion of time for the service in the calculation of the capacity results in a non-linear model. Therefore, it is necessary to linearise the corresponding constraints (6q) together with (6o)-(6p) in order to obtain a MIP formulation. Finally, constraints (6r)-(6y) define the domain of the decision variables.

The high number of details in the model allows the representation of a fairly realistic problem, making it possible to design efficient services reducing the overall operational costs. Nonetheless, it can easily be observed that the compact model is computationally hard to solve. The model presents several "big-M" constraints, which produce a very weak relaxation, and Branch-and-Bound techniques provide large integrality gaps and poor bounds. One proposed method to solve this problem is Branch-and-Cut, as it has presented good results to the VRP and other transportation network design problems. The idea is to solve the relaxed problem without the transshipment constraints, (6d)-(6f), and the connectivity constraints (6h)-(6j) and (6m) in butterfly nodes and then, gradually add cuts to the formulation when those constraints are violated. This was used by Reinhardt and Pisinger (2012), to solve instances of up to 10 ports to proven optimality.

4.3 Considering non-simple services in the formulation

The majority of models for LSNDP are defined using an arc formulation. These formulations are suitable when the structure of the services can be assumed to be simple. However, these formulations become problematic when formulating non-simple services, as it requires the inclusion of many extra variables in the model. In this section, we present different modelling approaches that can be used when studying the design of more complex services.

4.3.1 Port-call formulations

The general idea of this formulation is to define services as sequences of port-calls, instead of by the arcs that connect the physical ports in the graph, as presented in Section 4.1. The formulation keeps track of the order in which the ports are visited during a service, which makes it possible to distinguish between multiple visits to a port by the same service. This, in turn, allows the inclusion of non-simple services, which better reflects how services are designed in practice.

The port-call formulation defines service flow variables to model how containers are transported within and between services. Therefore, the transshipment of cargo can then be modelled by the service flows. Hence, not only can this approach capture the cost of transshipment between different services, but also within the same service. Furthermore, the model can also account for the demand that is rejected by considering the cargo that is not flowed in the services.

Nevertheless, the decision variables of this formulation must be extended with an extra sequence index to identify the corresponding physical port visited at certain port-call by the service. Since the complexity of the problem increases with the addition of the extra index, it is common to impose a maximum number of port-calls within a service.

4.3.2 Layer-networks for complex services structures

One way of managing the difficulty of handling complex service structures together with transshipment restrictions, is by using a graph network with multiple nodes for each port.

The graph network is modelled with multiple layers, where each layer is a complete sub-graph, containing exactly one copy of each port. This way, each visit to a port is considered a separate node, and the maximum number of visits to each port is limited by the number of nodes representing that port. The layers are only connected between nodes representing the same port so that it is possible to switch layers at any given point. A graphical representation of the network can be seen in Figure 8. The key is that, by separating the different visits to a port, complex services can be modelled, together with transshipment costs for transshipments within the same service.

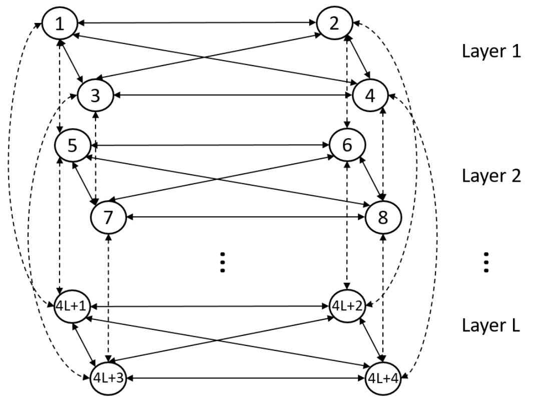

Fig. 8 Visual representation of the layer-network with four ports and L layers.

However, the drawback with this modelling approach is that it is significantly more complex. In addition to using more nodes and edges, symmetries can easily arise, as a service could swap layers more or less arbitrarily without changing the represented solution. Some of this can be counteracted by symmetry breaking constraints, but to this day, only smaller instances of up to around 15 ports have been solved with modelling approach. Additionally, to keep down the complexity, it can be beneficial to impose a limitation of two layers in the defined graph network. This is a quite realistic limitation, as a service seldom visits a port more than twice. Hence, the problem allows the creation of simple, butterfly and pendulum services.

4.3.3 Time-Space models

The LSNDP can also be represented by a time-space graph. The key of this modelling approach is to include the temporal aspects into the graph structure. Therefore, this graph structure not only allows the representation of any type of service, but also integrates the vessel scheduling into the modelling.

Let H be the set of time-units after discretising the weekly planning horizon into uniform time steps. This discretisation is used to define the graph structure of the problem. Define $G = (\tilde{N}, A)$ as a directed graph, where $\tilde{N}$ and A are the set of nodes and arcs, respectively.

The set of nodes contains a copy of each port for each of the time steps. Hence, each node $(i,h) \in \tilde{N}$ represents the departure time of a vessel from port $i \in N$ at a specific time $h \in H$. The set of arcs A is composed by two set of arcs; the set of *sailing arcs* A^S and the set of *transshipment arcs* A^T. Each sailing arc $a \in A^S$ connects the departure of a vessel from two different port-nodes, i.e., connects two nodes (i_1, h_1) and (i_2, h_2) such as $i_1 \neq i_2$. Note that, due to the weekly frequency of the services, the times should be computed modulo 7 days,and hence, services are modelled as closed cycles in the graph. Moreover, each transshipment arc $a \in A^T$ represents the cargo transshipment between two services at the same port, i.e., the arc connects two nodes (i_1, h_1) and (i_2, h_2) such as $i_1 = i_2$. As an example, Figure 9 illustrates a time-space graph for four ports with two services. The transshipment of cargo between services can be carried out at ports B and C. Moreover, port B is visited twice by Service 1, allowing cargo to be transshipped within the same service.

One of the main advantages with the time-space models, is that in addition to generating the routes for the services, they also generate the schedule. They also make it handy to model more complex operations, such as speed optimisation and transshipment times. Furthermore, if we disallow services to visit the same port twice at the same time slot, we get the same benefits as the layered networks in terms of handling transshipment together with complex services. This, however, again comes at the price of having significantly more nodes and edges in the graph.

An important design choice, when working with time-space graphs is the time discretisation granularity. By defining small time steps, the model can better reflect the aforementioned operations; however, the graph would require a large number of nodes, which increase the overall complexity of the problem. On the other hand, large time steps may be too restrictive, and the model may not capture the additional constraints as accurately.

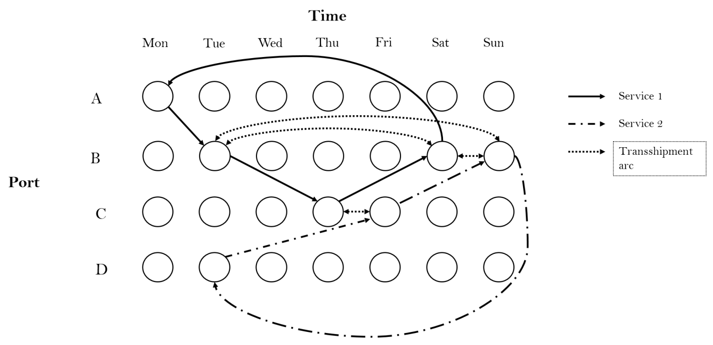

Fig. 9 Example of a time-space graph G with four ports with a planning horizon discretised by the days of the week. The graph depicts two services, where Service 1 visits port B twice during the same rotation. Transshipment arcs are represented by dashed arcs, connecting the nodes for the same port.

5 Two-stage algorithms

The LSNDP consists of two tightly interrelated problems — the vessel service network design and the container flow problem. One of the most successful approaches so far for finding good solutions to the LSNDP, has been to use heuristics exploiting this two-tier structure.

In general, the two stage algorithms can be divided into two categories: *service first* and *flow first*. In *service first*, the service generation is the leading procedure, and the containers are then flowed, given the generated services. It is then common to use information from the container flow to update the services. This way a feedback loop is created, iteratively improving the services and solving the container flow.

In *flow first*, the containers are instead first flowed without any information regarding the services, and the services are subsequently created to match that flow. The key concept is to try to aggregate the container flows onto a small subset of the arcs, to facilitate the ensuing service generation.

5.1 The container flow problem

Before going into the full two-stage algorithms, let us briefly discuss the container flow problem, which is the lower tier problem in the LSNDP two-tier structure. In general, for a given set of services, the container flow problem reduces to a *multi-commodity flow* problem (MCFP) with fractional flows allowed.

In addition to the notation defined in Section 2.2, we need an additional parameter; to each arc $(i,j) \in A$ define its corresponding flow capacity, u_{ij}. The arc set A and its corresponding costs c_{ij}^k and capacities u_{ij} are defined by the vessel services, designed in the upper-tier problem. Also, let x_{ij}^k be a continuous variable denoting the flow of commodity k through arc (i,j). The MCFP can then be expressed as:

$$\min \quad \sum_{(i,j)\in A} \sum_{k\in K} c_{ij}^k x_{ij}^k \tag{7a}$$

$$\text{s.t.} \quad \sum_{j\in N:(i,j)\in A} x_{ij}^k - \sum_{j\in N:(j,i)\in A} x_{ji}^k = \xi_i^k \qquad i \in N, k \in K \tag{7b}$$

$$\sum_{k\in K} x_{ij}^k \leq u_{ij} \qquad (i,j) \in A \tag{7c}$$

$$x_{ij}^k \geq 0 \qquad (i,j) \in A, k \in K. \tag{7d}$$

Here, the objective, (7a), is to minimise the total cost. Constraints (7b) are the flow conservation constraints, constraints (7c) are the capacity constraints, and constraints (7d) define the domain of the variables x_{ij}^k.

When fractional flows are allowed, the MCFP is solvable in polynomial time. For larger instances it is, however, still computationally demanding. As the model generally has to be solved a multitude of times in the presented two-tier solutions to the LSNDP, efficient solution methods to the MCFP are essential.

One of the most common solution approaches is to exploit its block-angular constraints matrix and apply Dantzig-Wolfe Decomposition (Ahuja et al., 1993). The problem should first be reformulated as a path flow formulation, where the goal is to allocate the commodities to a number of flow paths from the commodities' origins to their destinations, while respecting the capacity constraints on the arcs. Let P^k be the set of all paths for commodity $k \in K$, from o_k to d_k, and let P^k_a be the set of paths for commodity k, which uses arc a. Then we define

$$P_a = \bigcup_{k \in K} P^k_a,$$

to be the set of all paths going through arc $a \in A$. For each path, p, for commodity $k \in K$, define its cost $c^k_p = \sum_{a \in A: p \in P^k_a} c^k_a$, and a corresponding decision variable f^k_p, deciding the flow through path p. The path flow formulation can then be expressed as:

$$\begin{aligned}
\min \quad & \sum_{k \in K} \sum_{p \in P^k} c^k_p f^k_p & & \text{(8a)}\\
\text{s.t.} \quad & \sum_{p \in P^k} f^k_p = \xi^k_{o_k} & k \in K \quad & \text{(8b)}\\
& \sum_{p \in P_a} f^k_p \le u_a & a \in A \quad & \text{(8c)}\\
& f^k_p \ge 0 & k \in K, p \in P^k. \quad & \text{(8d)}
\end{aligned}$$

The objective function, (8a), minimises the cost. Constraints (8b) ensure that all commodities are delivered and constraints (8c) assert that the arc capacity cannot be exceeded. Lastly, constraints (8d) define the domain for the variables.

The path formulation has a very large number of variables, but generally, only a few of them are needed for the optimal solution. Using column generation, the problem can be restricted to only consider a limited amount of paths for each commodity and new paths can then be generated dynamically. In this way, the path formulation can generally be solved faster than the arc formulation, described earlier. The path formulation makes it relatively easy to implement transit time constraints as they can be handled in the pricing problem.

5.2 *Service first methods*

While the lower-tier container flow problem is solvable in polynomial time (when no transit time constraints are imposed), the upper-tier service selection problem is NP-hard, and just calculating the objective value of a given solution, requires solving the container flow problem. This makes the service selection problem difficult to solve to optimality and instead several matheuristics have been developed to find good solutions to larger instances. A matheuristic is a method that employs heuristics together with methods from linear and integer programming. In the case of the LSNDP, the most common procedure is to use linear programming tools to solve the MCFP and then various heuristics to update the vessel services.

Typically, we keep a set of services S, each with a designated speed and vessel class, and sufficient vessels to keep the weekly frequency. When solving the container flow for those services, we can see which services are under or over utilised and change the speed and vessel classes of the services accordingly. Looking at the dual variables for the capacity constraints, such as constraints (7c), we get information about which arcs would need more capacity to lower the costs. This can then be used to modify or create new services. Then, based on this information, we gradually modify the services, most commonly within a tabu-search framework.

An alternative approach is to instead use linear programming for updating the services. When modifying a service, the evaluation of the resulting change in time and revenue require the full solution of the commodity flow problem. This is generally too expensive, if we would like to compare multiple modifications, and the idea is therefore to instead work with estimates of those changes. We will iteratively update the services one by one, and for each service we find a set of potential ports to either include or exclude from the service. The key is then to, for each of those ports, estimate the change in revenue for the problem and the change in time for the service, if this port would be included or excluded. Where in the services to insert new ports is decided using a greedy heuristic. Estimating the change in service time is straightforward, but for estimating the change in revenue, we need an estimate of how the cargo flow would be affected. For this we solve a shortest path problem for each affected commodity to evaluate the approximate cost of routing it through the resulting network.

With the estimates in place, we can then define an integer program to optimise the service changes. In addition to the notation from the Section 2.2, let τ_s be the time length of a service s, let $\Delta_{is}^{\mathrm{R}+}$ ($\Delta_{is}^{\mathrm{R}-}$) be the estimated revenue change, and $\Delta_{is}^{\mathrm{T}+}$ ($\Delta_{is}^{\mathrm{T}-}$) be the estimated duration change from including (excluding) port $i \in N$ in (from) service $s \in S$. Also, let η_s^+ (η_s^+) be the maximum number of inclusions (removals) allowed and let $\bar{N}_s$ denote the set of ports which can be included. Let $\hat{m}_v$ denote the number of free vessels of class v, such that

$$\hat{m}_v = m_v - \sum_{s \in S} m_v^s.$$

Lastly, let us define the binary variables x_{is}^+ and x_{is}^-, which control the inclusion and removal, respectively, of port i from service s, and the integer variables ζ_s, which denote the number of vessels to add to/subtract from service s. For each service $s \in S$, with corresponding vessel class v, we can then define the following integer program:

$$\max \quad \sum_{i \in N_s} \Delta_{is}^{\mathrm{R}+} x_{is}^{+} + \sum_{i \in \bar{N}_s} \Delta_{is}^{\mathrm{R}-} x_{is}^{-} - c_v \zeta_s \tag{9a}$$

$$\text{s.t.} \quad \tau_s + \sum_{i \in N^s} \Delta_{is}^{\mathrm{T}+} x_{is}^{+} + \sum_{i \in \bar{N}_s} \Delta_{is}^{\mathrm{T}-} x_{is}^{-} \leq 24 \cdot 7 \cdot (m_v^s + \zeta_s) \tag{9b}$$

$$\zeta_s \leq \hat{m}_v \tag{9c}$$

$$\sum_{i \in \bar{N}_s} x_{is}^{+} \leq \eta_s^{+} \tag{9d}$$

$$\sum_{i \in N_s} x_{is}^{-} \leq \eta_s^{-} \tag{9e}$$

$$\sum_{j \in L_i^+} x_{js}^{-} \leq |L_i^+|(1 - x_{is}^{+}) \qquad i \in \bar{N}_s \tag{9f}$$

$$\sum_{j \in L_i^-} x_{js}^{-} \leq |L_i^-|(1 - x_{is}^{-}) \qquad i \in N_s \tag{9g}$$

$$x_{is}^{+} \in \{0,1\} \qquad i \in \bar{N}_s \tag{9h}$$

$$x_{is}^{-} \in \{0,1\} \qquad i \in N_s \tag{9i}$$

$$\zeta_s \in \mathbb{Z}. \tag{9j}$$

Here, the objective (9a) is to maximise the increase in revenue. Constraint (9b) ensures that there is enough vessels assigned to keep the weekly frequency, and constraint (9c) says that no more than the number of free vessels can be added to the service. Constraints (9d) and (9e) set a limit on the number of insertions and removals, while (9f) and (9g) prevent certain combinations of insertions and removals. The sets L_i^+ are defined such that if a port i is to be inserted, then no port in L_i^+ is allowed to be removed. If instead a port i is removed, then every port in L_i^- must remain. If a new port call is inserted in between two ports, then neither of those are allowed to be removed, and if inserting a new port means that a new commodity is transported, then the origin and destination nodes of this commodity are not allowed to be removed. Constraints (9d)–(9g) are defined to limit the amount of changes which can be applied, as the revenue and time change estimates are made for one or a few changes and deteriorates rapidly when multiple changes are applied. Lastly, (9h)–(9j) define the domain of the variables.

The algorithm works such that each service, one by one, is updated according to the solution of the above defined mixed integer problem. Then the MCFP is solved to update the total revenue and the effect of new changes is once again estimated with the shortest path procedure. To diversify the solutions, in every tenth iteration the services with lowest utilisation are removed and new services are created using the greedy creation heuristic. A more thorough description of the algorithm can be seen in Brouer et al. (2015). They also use the algorithm to find good solution to all the LINERLIB instances, except for *World Large*.

5.3 Backbone flow

The main idea in backbone flow algorithms is to reverse the order of two-phase algorithms by first flowing the containers, and then constructing services that cover the flow.

For the first step in the backbone flow algorithms, which flow the containers, a directed graph $G = (N, A)$ is used. There are no capacities associated with the edges, as there are no services designed at this stage. To design a backbone flow, which can later be effectively covered by a set of services, it is important to aggregate the flows onto a few arcs. This can be achieved by using for example a fixed charge cost. Another way would be to use a concave edge cost function $c(x)$ of the flow x, reflecting the economy of scale for flowing more containers. There is a large cost associated with opening an arc (i.e., deploying a vessel), while the cost per container decreases as the flow (and hence vessel size) increases. See Figure 1 for an illustration of the costs.

Let x^k_{ij} denote the flow of commodity k on edge (i,j). Then the backbone flow problem becomes a non-linear MCFP as given by

$$\min \sum_{(i,j)\in A} c\big(\sum_{k\in K} x^k_{ij}\big) \tag{10a}$$

$$\text{s.t.} \sum_{(i,j)\in A} x^k_{ij} - \sum_{(j,i)\in A} x^k_{ji} = \xi^k_i \qquad i \in N, k \in K \tag{10b}$$

$$x^k_{ij} \geq 0 \qquad (i,j) \in A, k \in K. \tag{10c}$$

As before, the objective, (10a), is to minimise the total cost, and constraints (10b) are the flow conservation constraints. Constraints (10c) define the domain of the variables.

Since the model is non-linear and non-convex, many of the classic optimisation techniques are not applicable. Instead, it is commonplace to use various heuristics and approximation methods, such as, for example, dual ascent methods or tabu search. One classic method is to use successive linearisation of the cost function, where the concave cost function is estimated with a linear function. Typically, this would be done by iteratively routing some or all containers, by solving a number of shortest path problems, based on the current arc costs, and then update the costs according to the new solution. If the arc costs are updated before all containers are routed, generally, the first containers to be routed are more decisive for which arcs are used in the final solution. To circumvent this, the algorithm is run several times, with a random order of the containers, to achieve a reasonable average picture of the backbone flow.

An example of running ten iterations for the demand matrix of the *WorldSmall* instance gives the average arc loads shown in Figure 10. The figure clearly shows that only a fraction of the possible arcs is used in the solution.

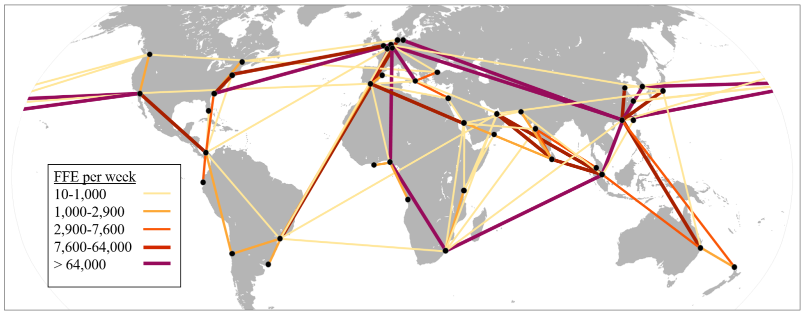

Fig. 10 Typical backbone flow for the *WorldSmall* instance. Data from Krogsgaard et al. (2018) and World map by San Jose under a CC-BY-SA-3.0 license.

5.3.1 From backbone flow to network design

Having found a backbone flow, the goal is to design services which satisfies it. This becomes a kind of arc-covering problem, but with the addition that the commodities have to be assigned to the different services for the purpose of modelling transshipments and handling capacity constraints. Alternatively, if the aim is to just create a decent initial solution, the service generation can be split in two parts: one where the routes are generated, trying to cover as much as possible of the flow, and one where the containers are assigned to the services, trying to minimise the transshipments while satisfying the capacity constraints. Due to the rather limited amount of arcs, on which containers flow, the pure arc-covering problem can be solved rather effectively using greedy construction heuristics. The idea is to create services by greedily inserting arcs one at a time, until the maximum service time is reached.

In a computational study by Krogsgaard et al. (2018), it is shown that usable solutions can be found in relatively short time. Using the *WorldSmall* instance, the authors generate 20 different sets of services by running the above algorithm where the containers are flown in random order. This can be done in about 80 seconds, and results in a profitable network, although the resulting network is far from optimal.

While it is unlikely that the backbone flow perfectly matches the structure of the service network, this methodology can be used to quickly generate good initial solutions. The container paths can then be updated to better fit the services, which could be the starting step in a continued two-stage algorithm.

6 Bibliographic notes

A good introduction to the liner-shipping business along with a description of its main assets and infrastructure can be found in Brouer et al. (2014a). This paper also presents in details a complete model for the LSNDP with all side-constraints and introduces the LINER-LIB test instances. For a detailed review of the research on liner shipping optimisation problems, see the survey papers Ronen (1983, 1993), Christiansen et al. (2004), Christiansen et al. (2013), Meng et al. (2014), Tran and Haasis (2015), Brouer et al. (2016, 2017), and Lee and Song (2017).

Planning problems within RoRo shipping are far less studied in the OR literature compared to container shipping. As far as we know, there is no literature contribution considering RoRo network design. However, other strategic planning issues involving decisions regarding fleet size and mix can be found in Pantuso et al. (2015). On the tactical planning level, the fleet deployment problem consists of assigning ships in the fleet to voyages that must be performed repeatedly on given trade routes. In addition to the ship-voyage assignment, the results from fleet deployment are sailing routes for the ships in the fleet. Deployment models for RoRo shipping are presented by Fagerholt et al. (2009) and Andersson et al. (2015).

In Section 4.1, the service formulation for the LSNDP has been used by Álvarez (2009) and Balakrishnan and Karsten (2017). Álvarez (2009) studied the LSNDP at the tactical level, considering the joint routing and deployment of container vessels. The formulation presented in the paper is based on the set of all feasible services, which are given as a combination of vessel class, operating speed and route structure. Moreover, the idea of selecting a subset of sailing services from a pool of candidate services was originally proposed by Balakrishnan and Karsten (2017), who also presented the multi-commodity model based on flows along sub-paths in the augmented network.

The arc formulation model, in Section 4.2, was originally presented by Reinhardt and Pisinger (2012), in which network design and fleet assignment were combined. The authors were among the first to study exact methods for the network design problem in liner shipping with transshipment operations and butterfly services. They developed a branch-and-cut algorithm and reported results for instances up to 15 ports.

In Section 4.3, different approaches for modelling complex-services are presented. The port-call formulation for the LSNDP is based on Plum et al. (2014). The model proposed in this paper has been used to solve the two smallest LINER-LIB instances using CPLEX. The authors reported promising solutions for these instances, but optimal solutions were not achieved due to the large number of constraints and decision variables. The layer-network for complex services structures is based on Thun et al. (2017). The problem is solved to integer optimality using a branch-and-price algorithm, reporting results for small instances of between 5 and 7 ports with up to 14 demands. Finally, the time-space graph is based on Koza et al. (2020). The authors developed a column generation matheuristic and reported very good results for LINER-LIB instances of up to 45 ports.

Section 5 is dedicated to two-stage algorithms, where column generation and Benders' decomposition have been used by Agarwal and Ergun (2008), and various matheuristics have been used by Álvarez (2009), Brouer et al. (2014b) and Karsten et al. (2017). A good introduction to the transit time constrained multi-commodity flow problem can be found in Karsten et al. (2015).

The general service-first method, in Section 5.2, is inspired by Álvarez (2009) and the model (9a)-(9h) is based on Brouer et al. (2015). The authors report satisfactory solutions for 6 out of 7 LINER-LIB instances for the case with transit time requirements for the commodities.

In Section 5.3 the main idea of backbone flow was presented by Krogsgaard et al. (2018). Sun and Zheng (2016) also use a concave function to optimise the container flow. Moreover, the greedy heuristic for generating services presented in the same Section 5.3.1 was also presented in Krogsgaard et al. (2018). Promising computational results are reported for all LINER-LIB instances without the transit time requirements, where the best results are achieved for 4 out of the 7 instances, namely *WestAfrica*, *Pacific*, *WorldSmall* and *WorldLarge*.

7 Concluding remarks and future challenges

Liner shipping is the backbone of international trade, hence it is important to develop decision support tools that can help design more cost-efficient services, and balance several objectives. This includes finding the right trade-off between speed, transportation times, number of transshipments, and operational costs.

Slow steaming together with larger vessels has proven to be an efficient tool for reducing energy consumption. However, slow steaming decreases the capacity of vessels, since they cannot transport as much cargo per time unit as before. Hence, more vessels are needed in order to maintain the same capacity, straining the environment. Bigger vessels tend to be more energy efficient per container, but the increased capacity results in longer port stays, making it necessary to speed up between the port stays. It is therefore necessary to design services such that fewer transshipments are needed, while still ensuring a good utilisation of the mega vessels.

Although liner shipping generally is one of the most energy-efficient modes of transportation per kilometer, the shipping industry emits large quantities of SO_x and NO_x. In the future, we will see container vessels operating with new, greener, propulsion types. Electric vessels may operate shorter services, while liquid natural gas (LNG) may be used for operating longer services. The new propulsion types will make it necessary to completely rethink service network design, since refueling/recharging will be more complicated, and vessels will have a more limited range of operation.

The introduction of autonomous vessels in the container shipping industry may also significantly change the way a network is designed and operated. In particular for feeder-lines we will see more smaller vessels sailing on-demand, depending on the cargo. This will turn the network design process into a dynamic routing problem. It would also be interesting to investigate whether fuel savings are achievable if autonomous vessels are sailing in convoys close to each other. If this is the case, the network design should take these convoys into account.

Nearly every vessel will be delayed in one or more ports during a round trip. Instead of just speeding up (and hence using more energy) advanced disruption management tools need to be developed that can ensure timely arrival to the end customer with the lowest possible energy consumption. Some studies along this path include Brouer et al. (2014a) but much more work needs to be done in this area.

Vessel sharing agreements are an important tool for making it possible to operate larger and more energy-efficient vessels. In a vessel sharing agreement two or more companies share the capacity of a vessel throughout the full rotation or on certain legs. Vessel sharing agreements, however, substantially increase the complexity of designing a network, since some legs and capacities are locked according to the agreement.

References

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

RK Ahuja, TL Magnanti, and JB Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

JF Álvarez. Joint routing and deployment of a fleet of container vessels. *Maritime Economics & Logistics*, 11(2):186–208, 2009.

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

A Balakrishnan and CV Karsten. Container shipping service selection and cargo routing with transshipment limits. *European Journal of Operational Research*, 263(2):652–663, 2017.

BD Brouer, JF Álvarez, CEM Plum, D Pisinger, and MM Sigurd. A base integer programming model and benchmark suite for liner-shipping network design. *Transportation Science*, 48(2):281–312, 2014a.

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

BD Brouer, G Desaulniers, CV Karsten, and D Pisinger. A matheuristic for the liner shipping network design problem with transit time restrictions. In F Corman, S Voß, and RR Negenborn, editors, *Computational Logistics*, pages 195–208. Springer, 2015.

BD Brouer, CV Karsten, and D Pisinger. Big data optimization in maritime logistics. In A Emrouznejad, editor, *Big Data Optimization: Recent Developments and Challenges*, volume 18 of *Studies in Big Data*, pages 319–344. Springer International Publishing, 2016.

BD Brouer, CV Karsten, and D Pisinger. Optimization in liner shipping. *4OR*, 15 (1):1–35, 2017.

M Christiansen, K Fagerholt, and D Ronen. Ship routing and scheduling: Status and perspectives. *Transportation science*, 38(1):1–18, 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, 2013.

M Christiansen, E Hellsten, D Pisinger, D Sacramento, and C Vilhelmsen. Liner shipping network design. *European Journal of Operational Research*, 286(1): 1–20, 2019.

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

Germanische Lloyd. http://www.balkanlloyd.com/news/96-germanischer-lloyd-has-conducted-research-showing-that-new-and-efficient-4-500-teu-panamax, 2017. [Online; accessed February 20, 2017].

E Hellsten, D Pisinger, D Sacramento, and C Vilhelmsen. Green liner shipping network design. In *Sustainable Shipping*, pages 307–337. Springer, 2019.

ISL. Shipping statistics and market review. Technical report, Institute of Shipping Economics and Logistics, 2016.

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

CV Karsten, BD Brouer, G Desaulniers, and D Pisinger. Time constrained liner shipping network design. *Transportation Research. Part E: Logistics and Transportation Review*, 105:152–162, 2017.

DF Koza, G Desaulniers, and S Ropke. Integrated liner shipping network design and scheduling. *Transportation Science*, 54(2):512–533, 2020.

A Krogsgaard, D Pisinger, and J Thorsen. A flow-first route-next heuristic for liner shipping network design. *Networks*, 72(3):358–381, 2018.

CY Lee and DP Song. Ocean container transport in global supply chains: Overview and research opportunities. *Transportation Research Part B: Methodological*, 95: 442–474, 2017.

Q Meng, S Wang, H Andersson, and K Thun. Containership routing and scheduling in liner shipping: overview and future research directions. *Transportation Science*, 48:265–280, 2014.

CE Miller, AW Tucker, and RA Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)*, 7(4):326–329, 1960.

B Montreuil. Towards a physical internet: meeting the global logisitcs sustainability grand challenge. *Logistics Research*, 3:71–87, 2011.

TE 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.

G Pantuso, K Fagerholt, and SW Wallace. Uncertainty in fleet renewal: a case from maritime transportation. *Transportation Science*, 50(2):390–407, 2015.

CEM Plum, D Pisinger, and MM Sigurd. A service flow model for the liner shipping network design problem. *European Journal of Operational Research*, 235(2): 378–386, 2014.

HN Psaraftis and CA Kontovas. Slow steaming in maritime transportation: Fundamentals, trade-offs, and decision models. In CY Lee and Q Meng, editors, *Handbook of Ocean Container Transport Logistics: Making Global Supply Chains Effective*, pages 315–358. Springer International Publishing, 2015.

LB Reinhardt and D Pisinger. A branch and cut algorithm for the container shipping network design problem. *Flexible Services and Manufacturing Journal*, 24(3): 349–374, 2012.

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

D Ronen. Ship scheduling: The last decade. *European Journal of Operational Research*, 71(3):325–333, 1993.

Z Sun and J Zheng. Finding potential hub locations for liner shipping. *Transportation Research Part B: Methodological*, 93:750–761, 2016.

K Thun, H Andersson, and M Christiansen. Analyzing complex service structures in liner shipping network design. *Flexible Services and Manufacturing Journal*, 29(3-4):535–552, 2017.

NK Tran and HD Haasis. Literature survey of network optimization in container liner shipping. *Flexible Services and Manufacturing Journal*, 27:139–179, 2015.

Unctad. Review of maritime transport. Technical report, United Nations Conference on Trade and Development, 2018.

Unctad. UnctadSTAT. https://unctadstat.unctad.org/wds/ReportFolders/reportFolders.aspx, 2019. [Online; accessed December 9, 2019].

8 Notation used in this chapter

We have tried to use an uniform mathematical notation throughout the chapter, although each section needs some additional symbols. The notation has been gathered and presented in the following tables. First, the general notation presented in the introduction to be commonly used in the mathematical models of the chapter is presented. Next, the extra notation needed to define the models of each specific section is presented.

Table 2 General notation from the Introduction

		Sets
N	:=	Set of Ports
A	:=	Set of Sailing Arcs
K	:=	Set of Commodities
V	:=	Set of Vessel Classes
S	:=	Set of Services
		Parameters
o_k	:=	Origin port for commodity $k \in K$
d_k	:=	Destination port for commodity $k \in K$
q_k	:=	Quantity of commodity $k \in K$
u_v	:=	Cargo capacity for vessel class $v \in V$
m_v	:=	Available fleet quantity of vessel class $v \in V$
ξ_i^k	:=	$\begin{cases} q_k & \text{if port } i \in N \text{ is the origin port of demand } k \in K \\ -q_k & \text{if port } i \in N \text{ is the destination port of demand } k \in K \\ 0 & \text{otherwise.} \end{cases}$
c_{ij}^k	:=	Unit-cost for transporting a unit of commodity $k \in K$ through arc $(i, j) \in A$
c^v	:=	Cost for deployment of a vessel from vessel class $v \in V$
c_{ij}^v	:=	Sailing cost for vessel class $v \in V$ traversing arc $(i, j) \in A$
c_{ik}^T	:=	Cost per unit of commodity $k \in K$ transshipped in port $i \in N$
t_{ij}^v	:=	Sailing time for vessel class $v \in V$ traversing arc $(i, j) \in A$
b_i	:=	Berthing time for the port call $i \in N$
c_s	:=	Cost for operation service $s \in S$.
m_v^s	:=	Required number of vessels from vessel class $v \in V$ to maintain the weekly frequency in service $s \in S$

Table 3 Additional notation for Section 4.1

		Parameters
u_{ij}^s	:=	Capacity for service $s \in S$ along arc $(i, j) \in A$
		Decision variables
$x_{ij}^{ks} \in \mathbb{R}^+$	:=	Amount of commodity $k \in K$ transported in service $s \in S$ along arc $(i, j) \in A$
$y_s \in \{0, 1\}$	:=	Equal to 1 if service $s \in S$ is selected in the network, and 0 otherwise

Table 4 Additional notation for Section 4.1.1 (Balakrishnan and Karsten, 2017)

		Sets
A_s	:=	Set of sailing arcs for service $s \in S$
H_s	:=	Full set of sub-paths for service $s \in S$ (a sub-path is a part of a route in which the container travels on a single service and is denoted $\langle i,j,s \rangle$)
A^s_{ij}	:=	Set of sailing arcs of service $s \in S$ included in sub-path $\langle i,j,s \rangle$
		Parameters
u_a	:=	Capacity for arc $a \in A_s$
c^R_k	:=	Penalty cost per container for rejected demand of commodity $k \in K$
c^k_{ijs}	:=	Cost of routing one container of commodity $k \in K$ on sub-path $\langle i,j,s \rangle$
h_k	:=	Maximum allowed number of sub-paths on which commodity $k \in K$ can travel
		Decision variables
$y_s \in \{0,1\}$	:=	Equal to 1 if service $s \in S$ is selected, and 0 otherwise
$x^{hk}_{ijs} \in \mathbb{R}^+$	:=	Flow of commodity $k \in K$ using sub-path $\langle i,j,s \rangle$ as the h^{th} stage for $s \in S$, $\langle i,j,s \rangle \in A_s$, and $h = 1,2,\ldots,h_k$
$z_k \in \mathbb{R}^+$	:=	Unmet demand (number of containers) for commodity $k \in K$

Table 5 Additional notation for Section 4.2

		Sets
S^v	:=	Set of maximum number of services for vessel class $v \in V$
		Decision variables
$x^{ks}_{ij} \in \mathbb{R}^+$	:=	Amount of commodity $k \in K$ transported in service $s \in S^v$ along arc $(i,j) \in A$
$\tau^s_i \in \mathbb{R}^+$	:=	Departure time from port $i \in N$ of the vessel operating service $s \in S^v$
$w^s \in \mathbb{Z}^+$	:=	Number of deployed vessels to maintain a weekly frequency in service $s \in S^v$
$\gamma^s_i \in \{0,1\}$	:=	Equal to 1 if port $i \in N$ is the hub port in the service $s \in S^v$, and 0 otherwise
$y^s_{ij} \in \{0,1\}$	:=	Equal to 1 if arc $(i,j) \in A$ is selected on the service $s \in S^v$, and 0 otherwise

Table 6 Additional notation for Section 4.2 (Reinhardt and Pisinger (2012))

		Parameters
t_{max}	:=	Length of the time horizon
		Decision variables
$x^{kv}_{ij} \in \mathbb{R}^+$	:=	Amount of commodity $k \in K$ transported by vessel $v \in V$ along arc $(i,j) \in A$
$f^{kv}_j \in \mathbb{R}^+$	:=	Amount of commodity $k \in K$ transshipped at port $j \in N$ by vessel $v \in V$
$f^{kv}_{jih} \in \mathbb{R}^+$	:=	Amount of commodity $k \in K$ arriving at port $i \in N$ through arc $(j,i) \in A$ and not leaving in arc $(i,h) \in A$ by vessel $v \in V$
$e^v_{ij} \in \mathbb{Z}^+$	:=	Position of arc $(i,j) \in A$ in the service of vessel $v \in V$
$\tau_v \in \mathbb{R}^+$	:=	Route length of the service operated by vessel $v \in V$
$y^v_{ij} \in \{0,1\}$	:=	Equal to 1 if arc $(i,j) \in A$ is selected in the service for vessel $v \in V$, and 0 otherwise
$z^v_{ij} \in \{0,1\}$	:=	Equal to 1 if arc $(i,j) \in A$ is either the first or the last arc in the service for vessel $v \in V$, and 0 otherwise
$\gamma^v_i \in \{0,1\}$	:=	Equal to 1 if port $i \in N$ is the hub port in the service for vessel $v \in V$, and 0 otherwise
$\lambda^v \in \{0,1\}$	:=	Equal to 1 if vessel $v \in V$ is deployed for operating a service, and 0 otherwise

Table 7 Additional notation for Section 4.3

	Sets
H	:= Set of time-units after discretising the weekly planning horizon
$\tilde{N}$	:= Set nodes in the time-space graph
A^S	:= Set of sailing arcs in the time-space graph
A^T	:= Set of transshipment arcs in the time-space graph

Table 8 Additional notation for Section 5.1

	Sets
P^k	:= Set of paths for commodity $k \in K$
P_a	:= Set of paths using arc $a \in A$
	Parameters
u_{ij}/u_a	:= Flow capacity through arch $a = (i,j) \in A$
c_p^k	:= Unit flow cost of commodity $k \in K$ through path $p \in P_k$
	Decision variables
$x_{ij}^k \in \mathbb{R}^+$	:= Flow of commodity $k \in K$ through arc $(i,j) \in A$
$f_p^k \in \mathbb{R}^+$	:= Flow of commodity $k \in K$ through path $p \in P_k$

Table 9 Additional notation for Section 5.2

	Sets
N_s	:= Set of ports in service $s \in S$
$\bar{N}_s$	:= Set of ports available for inclusion into service $s \in S$
L_i	:= Lockset containing the ports which are forbidden to remove, if port $i \in N_s \cup \bar{N}_s$ is included (removed)
	Parameters
$\Delta_{is}^{\mathrm{R}+}$	:= Estimated revenue change from including port $i \in \bar{N}_s$ into service $s \in S$
$\Delta_{is}^{\mathrm{R}-}$	:= Estimated revenue change from removing port $i \in N_s$ from service $s \in S$
$\Delta_{is}^{\mathrm{T}+}$	:= Estimated duration change from including port $i \in \bar{N}_s$ into service $s \in S$
$\Delta_{is}^{\mathrm{T}-}$	:= Estimated duration change from removing port $i \in N_s$ from service $s \in S$
$\hat{m}_v$	:= Number of free vessels of vessel class $v \in V$
τ_s	:= Round trip time for service $s \in S$
η_s^+	:= Maximum number of inclusions into service $s \in S$
η_s^-	:= Maximum number of removals into service $s \in S$
	Decision variables
$x_{is}^+ \in \{0,1\}$	:= Equal to 1 if port $i \in \bar{N}_s$ is included into service $s \in S$, and 0 otherwise
$x_{is}^- \in \{0,1\}$	:= Equal to 1 if port $i \in N_s$ is removed from service $s \in S$, and 0 otherwise
$\zeta_s \in \mathbb{Z}$	:= Change in number of vessels in service $s \in S$

Table 10 Additional notation for Section 5.3

	Parameters
$c(x)$	:= Concave cost function of using an edge $(i,j) \in A$ for the flow x
	Decision variables
$x_{ij}^k \in \mathbb{R}^+$	:= Flow of commodity $k \in K$ on edge $(i,j) \in A$