Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor

A MIP-based heuristic for a single trade routing and scheduling problem in roll-on roll-off shipping

Jone R. Hansen [a], Kjetil Fagerholt [a,*], Frank Meisel [b]

[a] Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology, Norway
[b] Faculty of Business, Economics and Social Sciences, Christian-Albrechts-University Kiel, Germany

ARTICLE INFO

Keywords:
Maritime transportation
Roll-on roll-off shipping
Evenly spread requirement
Speed optimization
Transit time constraints
Heuristic

ABSTRACT

We study a single trade ship routing and scheduling problem for a roll-on roll-off shipping company. Along the given trade, there is a number of contracts for transportation of cargoes between port pairs. Each contract states a minimum service frequency where the services should be evenly separated in time and possibly transit time requirements. Current planning practice is to visit all ports along the trade every time it is serviced. Here, we aim instead at determining the sailing route and schedule of each voyage along the trade, i.e., which ports to visit when, which contracts to serve, and the sailing speeds, so that all contract requirements are satisfied at minimum cost. To solve this problem, we have developed a three-phase MIP-based heuristic, where each phase consists of solving dedicated a mixed-integer programming (MIP) model. The heuristic constructs solutions by first identifying the most promising candidate routes along the trade. Next, a candidate route is allocated to each available vessel. Finally, the heuristic determines the allocation of cargoes between the vessels, as well as sailing speeds and arrival times. Computational tests show that the heuristic outperforms a commercial MIP-solver and provides high-quality solutions to realistically sized instances in reasonable time.

1. Introduction

The ocean shipping industry is a major mode of transportation carrying around 90% of the world trade (Christiansen et al., 2020). Liner shipping is a mode of operation within the maritime industry that is often used for transporting goods, with vessels sailing according to a published schedule. Container shipping is the dominant segment within liner shipping, and reviews on related optimization problems are provided by Meng et al. (2014) and Christiansen et al. (2020). For rolling cargo, such as cars, trucks, and other equipment on wheels, Roll-on Roll-off (RoRo) vessels are the preferred choice. These vessels have a large ramp where the cargo is rolled on board and multiple decks where the rolling cargo is placed and fixed during the transport. With a total of around 5000 vessels in the world fleet, RoRo-shipping is an important segment within the maritime industry (ISL, 2017).

In this paper, we consider the single trade ship routing and scheduling problem (STSRSP) for a shipping company in the RoRo-segment, which was introduced by Hansen et al. (2019). In the STSRSP, a shipping company has entered into a large number of agreements on the transport of goods, contracts of affreightments, with different customers. These contracts describe the quantity of goods to be transported between specified ports within a given time frame. A contract also specifies that the total volume should be split into several services or shipments, referred to as *partial cargoes* in the following. These partial cargoes for each contract must be fairly *evenly spread* in time. We consider the planning scenario for a single trade. Typically, a trade connects two geographical regions, where a set of ports may be called in each region, e.g., US–Europe. A number of vessels which are available in the origin region of the given trade during the following planning period are to be deployed. The objective is to minimize the sailing, charter, and port costs while ensuring that all contractual terms are fulfilled. The number of vessels and the exact sailing frequency are not given a priori but are decisions to be made in the STSRSP. This means that the trade may not necessarily be serviced with a fixed frequency (e.g., one vessel per week), where each vessel visits each port along the trade as is common in other shipping segments. In container liner shipping, the trades are usually serviced on a weekly basis, see for example Brouer et al. (2013), Ng (2015), Wang and Meng (2017), and Wetzel and Tierney (2021). With vessels arriving at each port at a specific time each week, the planning becomes simpler for the container shipping companies and more predictable for the customers. However, this regularity may come at a cost, such as unnecessary port calls and lower capacity utilization. Compared to container liner shipping, RoRo-shipping is usually more lenient both to how often each port is visited and the frequency of sailings during the planning horizon. In the

* Corresponding author.
E-mail address: kjetil.fagerholt@ntnu.no (K. Fagerholt).

https://doi.org/10.1016/j.cor.2022.105904
Received 21 June 2021; Received in revised form 3 June 2022; Accepted 3 June 2022
Available online 17 June 2022

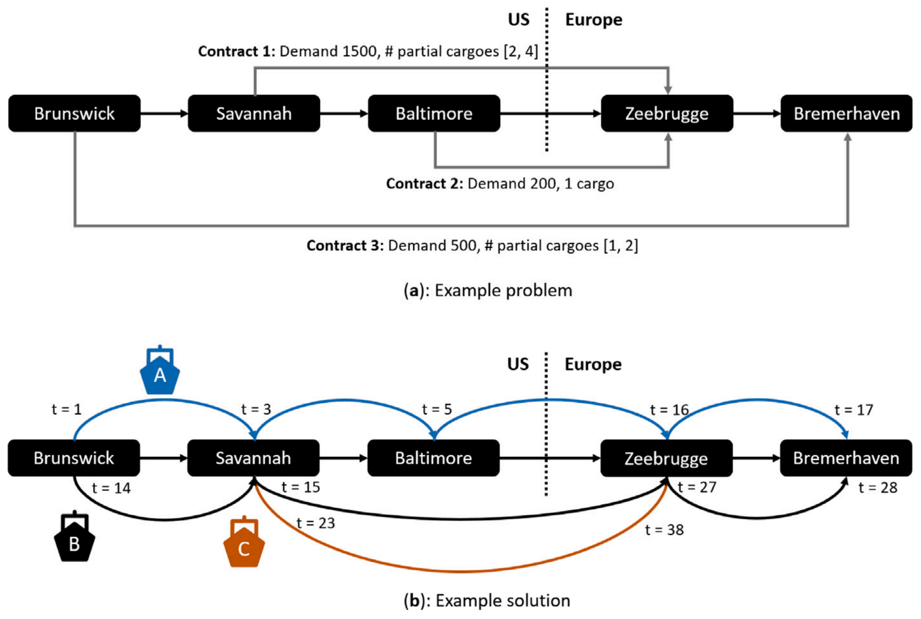

Fig. 1. On top, an example of the STSRSP for a trade from US to Europe with five ports and three contracts (**a**). On the bottom, a possible solution to the problem (**b**).

STSRSP, we aim to utilize this planning flexibility to reduce operational costs while fulfilling all contractual demands. In order to do so, we aim at determining the sailing route and schedule of each voyage along the trade, i.e., which ports to visit when, which contracts to serve, and the sailing speeds, so that all contract requirements are satisfied at minimum cost.

Fig. 1a shows a tiny example of the STSRSP. The example is considerably downscaled and its purpose is only to demonstrate the evenly spread requirement. Therefore, we simplify the example and do not consider capacities of vessels nor speed decisions along the routes. The example considers the US–Europe trade, and the shipping company must satisfy three contractual agreements along this trade. Contract 1 has a total demand of 1500 units to be transported from Savannah to Zeebrugge. The total demand must be split into either two, three, or four partial cargoes over the planning horizon, which is assumed to be a month in this example ($t = 30$ days). Contract 2 from Baltimore to Zeebrugge has a total demand of 200 and must be transported as one cargo. Finally, contract 3 has a demand of 500 units which can be split into one or two partial cargoes to be loaded in Brunswick and unloaded in Bremerhaven.

Fig. 1b shows a possible solution to the example problem in Fig. 1a. All sailings have to start within the planning horizon. In the solution, three vessels are used to handle the demand. Vessel A visits all five ports, starting the voyage from Brunswick on day 1. Vessel B visits all ports except Baltimore, while vessel C only calls at Savannah and Zeebrugge. Contract 1 is split among all vessels, resulting in three partial cargoes, contract 2 is handled by vessel A (i.e., one partial cargo), which is the only vessel to visit Baltimore, and contract 3 is split into two partial cargoes serviced by vessels A and B.

One important aspect for the customers/contracts is the evenly spread requirement. The vessels servicing contract 1 call at Savanna (the loading port of contract 1) on days 3, 15, and 23 to pickup partial cargoes of that contract. This is regarded as fairly evenly spread over the planning horizon. Similarly, we see that the pickups of the two partial cargoes of contract 3 are fairly evenly spread with 13 days between the pickups. The spread is only measured using the pickup time of each partial cargo of a contract for the following two reasons: First, as many of the customers are manufacturers of vehicles, evenly pickups of vehicles will maintain an even inventory at the loading port. Second, if the pickups are evenly spread, the deliveries will show a similar spread. This is due to the inherent trade structure, where the sailing times between the loading and unloading port are rather similar for each sailing. The threshold for what is considered fairly evenly regarding the spread requirements and the modeling of these requirements are described in more details in the next section.

Ensuring evenly spread pickups is an important aspect of many maritime transportation problems. In previous studies, it has usually been handled by either introducing time windows or voyage separation constraints. Fagerholt et al. (2009), Andersson et al. (2015), and Dong et al. (2020) use time windows for when each voyage along a trade should start. For example, with time windows of 1 week, they ensure that only one vessel starts sailing along a trade every week. However, with this approach, one could get sailings on days 7, 8, 21, and 22, alternating between starting the sailings at the end and the beginning of the time windows. However, a customer may not consider this as fairly evenly spread. The other commonly used approach is voyage separation constraints, e.g., Norstad et al. (2015), Bakkehaug et al. (2016), and Vilhelmsen et al. (2017). All these studies introduce separation requirements with a minimum acceptable time between the start of two consecutive voyages on a given trade. With the starting times of voyages being separated in time, the problem that could occur with time windows as described above is eliminated. If the separated voyages visit the same ports, this method could be sufficient for ensuring fairly evenly spread pickups. However, in the STSRSP, vessels may skip ports along the voyage. This means that while the separation constraints can ensure that the start of each voyage is fairly evenly spread, they do not necessarily ensure a fairly evenly spread on a specific contract's pickups, i.e., the service of the contract's partial cargoes. Hansen et al. (2019), which introduced the STSRSP, handled the evenly spread requirements on a contractual level instead of the more commonly used aggregated voyage level. This approach gives the shipping company more flexibility regarding the starting time of each voyage, the number of voyages to sail along a given trade, and which ports to visit along each voyage while ensuring that all contractual terms are complied with.

The aspect of ensuring services to be fairly evenly spread in time also appears in other contexts of maritime transportation, as well as in other modes of transportation, such as for example the routing of supply vessels in the offshore oil and gas industry (Borthen et al., 2018; Kisialiou et al., 2018), the periodic vehicle routing problem (Campbell and Wilson, 2014), and in the airline industry (Ho-Huu et al., 2020).

Two other important characteristics of the STSRSP are the transit time requirements and speed optimization. In the STSRSP, some contracts have requirements on the maximum transit time for the transportation between the contracts' pickup and delivery ports. In the example above, there could for example be a transit time requirement for contract 1 of 15 days, meaning that no partial cargo for this contract should spend more than this time in transit from its pickup port, Savannah, to the corresponding delivery port, Zeebrugge. In the example, we can see that this requirement is fulfilled as the three partial cargoes of contract 1 spend 13, 12 and 15 days in transit, respectively. Transit time requirements are also important in container shipping (e.g., Reinhardt et al. (2020)). Speed optimization, where optimal speeds along all voyages' sailing are determined, is also an important part of the STSRSP, like in many other maritime routing problems (e.g., Reinhardt et al. (2020), Andersson et al. (2015) and Eide et al. (2020)).

Two mixed integer programming (MIP) models for the STSRSP were proposed by Hansen et al. (2019) and they showed that significant gains could be obtained by utilizing the inherent planning flexibility compared to the current planning practice of visiting all ports every time a trade is serviced. However, Hansen et al. (2019) also showed that large realistically sized instances could not be solved by commercial solvers using any of these two models due to the complexity of the STSRSP. Therefore, our main contribution in this paper is to introduce a novel three-phase MIP-based heuristic for the STSRSP. We show that the heuristic is capable of providing high-quality solutions to large realistically sized instances in a short amount of time. As a second contribution, based on a real situation for our case company, we also present a study where we evaluate the potential gains of a merger between two equally sized shipping companies. We study a case where both companies operate their own trade between the same regions and evaluate the effects of merging the two trades and operate as one new merged company. Using the proposed heuristic, we show that the operational costs can be substantially reduced if the potential merger is performed and the potential benefits from the more flexible way of planning is utilized.

The remainder of this paper is organized as follows: A description of the problem is given in Section 2. In Section 3, the proposed solution method is presented. Section 4 provides the computational study, while concluding remarks are given in Section 5.

2. Problem definition and mathematical formulation

In this section, we give a formal definition of the STSRSP along with the mathematical formulation developed by Hansen et al. (2019). The MIP-based heuristic presented in Section 3 heavily relies on this mathematical formulation, which is why we include it here for the sake of completeness. The notation that is used for this model is summarized in the tables in Appendix.

The shipping company is concerned with fulfilling the transportation tasks along a given trade within the planning period while respecting frequency requirements. The problem consists of a set of contracts (or cargoes) $\mathcal{C}$ to be transported along the trade. Let $\mathcal{P}$ be the set of product types, while $\mathcal{P}^S_p$ is the subset of product types that can be stored in the same space as product type p. As an example, cars can be stored on decks facilitated for storing breakbulk cargo, but not the other way around. See for example Pantuso et al. (2016) for more details. Each contract (cargo) c is a transportation task of one and only one product type p. Let D_{cp} be the quantity of products of type p in contract c to be transported during the planning horizon T^{PH}. Let $l(c)$ and $u(c)$ be the loading and unloading port of contract c, respectively. The set C^L_i consists of all cargoes that are to be loaded at port i. Similarly, let C^U_i be the set of cargoes to be unloaded at port i.

Whenever a contract c is serviced by a vessel, the amount transported from that contract, i.e., a partial cargo, must be within a minimum and maximum bound denoted by $\underline{Q}_{cp}$ and $\bar{Q}_{cp}$, respectively. Furthermore, some contracts impose maximum transit times for the transportation between the contract's pickup and delivery ports. Let $\mathcal{C}^T \subseteq \mathcal{C}$ denote the set of transit time contracts and let T^T_c be the maximum transit time of contract c. Most of the contracts are so-called evenly spread contracts, given by the set $\mathcal{C}^E \subseteq \mathcal{C}$. For these contracts, the contractual terms state that the pickups of their partial cargoes should be fairly evenly separated throughout the planning horizon. Additionally, the terms state a lower and an upper limit on the number of partial cargoes the contract may be split into, represented by $[\underline{P}_c, \bar{P}_c]$. Hence, selecting the number of pickups (or partial cargoes) of these contracts is a decision to be made within the STSRSP. Let L be the evenly spread threshold, i.e., the maximum total deviation in days from the evenly spread requirement for all contracts.

The STSRSP can be defined on a graph $G = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$, indexed by i, is the set of nodes and $\mathcal{A} \subset \mathcal{N} \times \mathcal{N}$ is the set of arcs. The arcs describe the feasible movements of vessels between the nodes. As the vessels are to sail along trades, the ports may only be called in a fixed and given order. Hence, the graph G is both directed and acyclic. Let $\mathcal{K}$ be the set of vessels that are available for being deployed on this trade in the following planning horizon. Hence, the problem deals with determining how we could utilize these vessels in the best possible way to service the cargo contracts along the trade, while maintaining all the contractual requirements, e.g., regarding evenly spread and transit time. A starting position $o(k)$ and an artificial ending position $d(k)$ is associated with each vessel k. The set $\mathcal{N}^P \subset \mathcal{N}$ gives the possible port calls but does not include the starting and artificial ending positions. The set $\mathcal{N}^P_k \subseteq \mathcal{N}^P$ consists of the ports which vessel k may call, as some vessels may not call at a certain port due to shallow water and draft limitations. Further, let $\mathcal{N}_k = \mathcal{N}^P_k \cup \{o(k), d(k)\}$, i.e., all nodes that vessel k may use and $\mathcal{A}_k \subset \mathcal{N}_k \times \mathcal{N}_k$ be the possible sailing arcs for vessel k.

A vessel that is deployed and sails along a trade performs a *voyage*. Let $\mathcal{V}$ be the set of voyages. Since each voyage can only be serviced by one vessel and each vessel can also service at most one voyage within the planning horizon due to the long sailing times, we define one voyage for each vessel, so that $|\mathcal{V}| = |\mathcal{K}|$, such that one voyage is designed in the STSRSP for every available vessel. It should be pointed out that (at least) one of these voyages can turn out in the optimal solution to become an "empty" voyage, which does not include any ports, i.e., it goes directly from $o(k)$ to $d(k)$ and has zero cost. If such a voyage is selected, it simply means that a vessel is not used. The voyages are ordered, such that voyage v_1 starts before voyage v_2. All voyages succeeding voyage v are included in the set $\mathcal{V}^S_v$. The fleet of vessels is heterogeneous, with each vessel having different capabilities. Some vessels are designed to carry high and heavy product types, while other vessels may only transport vehicles. Ship k's capacity of a certain product type p is given by K^V_{kp}. The capacity intended for large bulk products may be used for storing vehicles, but not the other way around. The loading/unloading time per unit of product type p is given by the handling time parameter T^H_p. When chartering vessel k, a daily charter rate of C^C_k is imposed. Further, each vessel may sail with speeds in the interval $[\underline{g}_k, \bar{g}_k]$. We use the method proposed by Andersson et al. (2015) to model the speed and fuel consumption. The speed alternatives are discretized, represented by the set of discrete speed alternatives $\mathcal{S}$, indexed by s. The sailing time from node i to node j for vessel k using speed alternative s is given by T^S_{ijks}. The corresponding cost of sailing this arc for the given vessel and speed combination is given by C^{SC}_{ijks}, which includes both fuel and time charter costs. Let C^V_i be the cost of calling port i. Due to duties on other trades, the vessels may not be available at the start of the planning horizon. The time a vessel becomes available is given by T^A_k. The required piloting time at port i is given by T^P_i.

Let binary variable x_{ijv} be 1 if voyage v uses the arc that connects nodes i and j, 0 otherwise. Further, let binary decision variable y_{vk} define whether voyage v is sailed by vessel k or not. Let w_{ijvks} be a continuous variable, representing the weight of speed alternative s used on voyage v by vessel k, when sailing arc (i, j). Note that defining the variable for each arc (i, j) allows to decide on an individual speed for each leg of a voyage, which helps to meet the evenly spread requirement. The load of product type p on the arc (i,j) on voyage v is given by l_{ijvp}. The binary variable δ_{vc} defines whether contract c is handled on voyage v or not. Let ϕ_{nc} be 1 if contract c is split into n partial cargoes, 0 otherwise. The variable q_{vcp} defines the quantity of product p in contract c that is transported on voyage v. Let s_c be the maximum number of days contract c deviates from the selected pickup interval. The start of service at node i on voyage v is represented by the time variable t_{iv}. The continuous variable t_k^{HW} give the total waiting and handling time for vessel k. Finally, let the binary variable z_{vwc} define whether voyage w is the next voyage after voyage v where contract c is serviced or not. If $z_{vwc} = 1$, we will refer to the pair of voyages (v,w) as a spread pair for contract c.

With this notation, the problem can be described as follows:

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

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

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

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

$$x_{ijv} = \sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}} w_{ijvks}, \qquad \forall (i,j)\in\mathcal{A}, v\in\mathcal{V} \tag{5}$$

$$\sum_{j\in\mathcal{N}_k}\sum_{v\in\mathcal{V}} x_{o(k)jv} = 1, \qquad \forall k\in\mathcal{K} \tag{6}$$

$$\sum_{i\in\mathcal{N}_k}\sum_{v\in\mathcal{V}} x_{id(k)v} = 1, \qquad \forall k\in\mathcal{K} \tag{7}$$

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

$$\sum_{v\in\mathcal{V}} y_{vk} = 1, \qquad \forall k\in\mathcal{K} \tag{9}$$

$$\sum_{k\in\mathcal{K}} y_{vk} = 1, \qquad \forall v\in\mathcal{V} \tag{10}$$

$$0 \le l_{ijvp} \le \sum_{k\in\mathcal{K}} K^V_{kp} y_{vk} - \sum_{p'\in\mathcal{P}^S_p} l_{ijvp'}, \qquad \forall (i,j)\in\mathcal{A}, v\in\mathcal{V}, p\in\mathcal{P} \tag{11}$$

$$l_{ijvp} \le M^C_p x_{ijv}, \qquad \forall (i,j)\in\mathcal{A}, v\in\mathcal{V}, p\in\mathcal{P} \tag{12}$$

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

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

$$\underline{P}_c \le \sum_{v\in\mathcal{V}} \delta_{vc} \le \overline{P}_c, \qquad \forall c\in C \tag{15}$$

$$\delta_{vc} \le \sum_{i\in\mathcal{N}} x_{il(c)v}, \qquad \forall v\in\mathcal{V}, c\in C \tag{16}$$

$$\delta_{vc} \le \sum_{i\in\mathcal{N}} x_{iu(c)v}, \qquad \forall v\in\mathcal{V}, c\in C \tag{17}$$

$$\underline{Q}_{cp}\delta_{vc} \le q_{vcp} \le \overline{Q}_{cp}\delta_{vc}, \qquad \forall v\in\mathcal{V}, c\in C, p\in\mathcal{P} \tag{18}$$

$$\sum_{v\in\mathcal{V}} q_{vcp} = D_{cp}, \qquad \forall c\in C, p\in\mathcal{P} \tag{19}$$

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

$$t_{iv} + T^P_i x_{ijv} + \sum_{c\in C^L_i\cup C^U_i}\sum_{p\in\mathcal{P}} T^H_p q_{vcp} + \sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}} T^S_{ijks} w_{ijvks} \le t_{jv} \qquad \forall (i,j)\in\mathcal{A}, v\in\mathcal{V} \tag{21}$$

$$t_{l(c)v} + T^T_c + M^T_c(1-\delta_{vc}) \ge t_{u(c)v}, \qquad \forall v\in\mathcal{V}, c\in C^T \tag{22}$$

$$t_{jv} - M^S_{jk}(1 - x_{o(k)jv}) \le T^{PH}, \qquad \forall k\in\mathcal{K}, j\in\mathcal{N}^P_k, v\in\mathcal{V} \tag{23}$$

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

$$\sum_{k\in\mathcal{K}} t^{HW}_k \ge 2\cdot\sum_{c\in C}\sum_{p\in\mathcal{P}} T^H_p D_{cp}, \tag{25}$$

$$\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{N}^P_k} x_{o(k)i(v+1)} \le \sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{N}^P_k} x_{o(k)iv}, \qquad \forall v\in\mathcal{V}\setminus\{|\mathcal{V}|\} \tag{26}$$

$$\sum_{v\in\mathcal{V}}\sum_{w\in\mathcal{V}^S_v} z_{vwc} \ge \sum_{v\in\mathcal{V}} \delta_{vc} - 1, \qquad \forall c\in C^E \tag{27}$$

$$\sum_{w\in\mathcal{V}^S_v} z_{vwc} \le \delta_{vc}, \qquad \forall v\in\mathcal{V}, c\in C^E \tag{28}$$

$$\sum_{w\in\mathcal{V}\setminus(\mathcal{V}^S_v\cup\{v\})} z_{wvc} \le \delta_{vc}, \qquad \forall v\in\mathcal{V}, c\in C^E \tag{29}$$

$$\sum_{n=\underline{P}_c}^{\bar{P}_c} n\phi_{nc} = \sum_{v\in\mathcal{V}} \delta_{vc}, \qquad \forall c\in C^E \tag{30}$$

$$\sum_{n=\underline{P}_c}^{\bar{P}_c} \phi_{nc} = 1, \qquad \forall c\in C^E \tag{31}$$

$$\sum_{n=\underline{P}_c}^{\bar{P}_c} \frac{T^{PH}\phi_{nc}}{n} - s_c - M^E_c(1-z_{vwc}) \le t_{l(c)w} - t_{l(c)v}, \qquad \forall v\in\mathcal{V}, w\in\mathcal{V}^S_v, c\in C^E \tag{32}$$

$$\sum_{n=\underline{P}_c}^{\bar{P}_c} \frac{T^{PH}\phi_{nc}}{n} + s_c + M^E_c(1-z_{vwc}) \ge t_{l(c)w} - t_{l(c)v}, \qquad \forall v\in\mathcal{V}, w\in\mathcal{V}^S_v, c\in C^E \tag{33}$$

$$\sum_{c\in C^E} s_c \le L \tag{34}$$

$$z_{vwc}\in\{0,1\}, \qquad \forall v\in\mathcal{V}, w\in\mathcal{V}^S_v, c\in C^E, \tag{35}$$

$$x_{ijv}\in\{0,1\}, \qquad \forall (i,j)\in\mathcal{A}, v\in\mathcal{V} \tag{36}$$

$$y_{vk}\in\{0,1\}, \qquad \forall v\in\mathcal{V}, \forall k\in\mathcal{K} \tag{37}$$

$$\delta_{vc}\in\{0,1\}, \qquad \forall v\in\mathcal{V}, c\in C \tag{38}$$

$$0\le w_{ijvks}\le 1, \qquad \forall k\in\mathcal{K}, (i,j)\in\mathcal{A}_k, v\in\mathcal{V}, s\in\mathcal{S} \tag{39}$$

$$\phi_{nc}\in\{0,1\}, \qquad \forall c\in C^E, n=\underline{P}_c..\bar{P}_c \tag{40}$$

$$s_c\ge 0, \qquad \forall c\in C^E \tag{41}$$

The objective function (1) is to minimize the total costs, i.e., sailing costs, ports fees, and time charter costs. Constraints (2)–(4) ensure that that each voyage v flows on the predefined network. Constraints (5) connect the flow and speed variables by setting the sum of speed variables to 1 if arc (i,j) is used on voyage v, 0 otherwise. Each vessel k has to start at its origin and end at the artificial destination node $d(k)$, see Constraints (6)–(7). Constraints (8) ensure that the speed variables of a vessel k on the voyage v take positive values only if the vessel k is deployed on the given voyage v. Constraints (9)–(10) ensure that each vessel is allocated to a voyage, and each voyage is sailed by only one vessel, respectively. Constraints (11)–(12) guarantee that the capacity limit of each product type on each vessel k is respected. The parameter M_p^C, which appears in Constraints (12), is an appropriately set big-M value. It should be noted that the total capacity of the vessel is given by the capacity of the product type "cars", which is the lightest product type which can be placed anywhere on the vessels. As an example, assume a total vessel capacity of 5000 (e.g., square meters). Assume further that we have a product capacity of cars of 5000 and of "high and heavy" of 1000 square meters. Then, we could for example either load (a) 4000 square meters of cars and 1000 square meters of high and heavy, (b) 5000 square meters of cars, or (c) any combination in between (a) and (b). However, we cannot load 5000 square meters of cars **and** 1000 square meters of high and heavy. Please note that in Constraints (11), we subtract the load for larger product types to accommodate for this.

There exists a load variable for each arc, voyage, product type combination. The connections between these variables are handled by the load balance Constraints (13). Constraints (14) define the initial load to be 0 for all voyages and product types. Constraints (15) guarantee compliance with the minimum and maximum number of partial cargoes. Constraints (16) and (17) ensure that a contract c is only picked up on a voyage if the voyage calls at both the loading and unloading port of contract c. Constraints (18) guarantee that if a partial cargo of contract c is picked up, the quantity transported is within the minimum and maximum quantity of each partial cargo. Constraints (19) require that all contracted demand is transported within the planning period. Constraints (20) give the time vessel k may start sailing. Constraints (21) link the time-variables such that the time a vessel starts servicing node j must be greater than or equal to the start of service at the previous node i, plus the sailing time, piloting time at node i, and cargo handling time. Constraints (22) ensure that the transit time restrictions are respected. M_c^T is the upper bound on the maximum time a ship may use between nodes $l(c)$ and $u(c)$, when transporting contract c. Constraints (23) guarantee that every vessel deployed on the trade starts sailing before the end of the planning horizon. An upper bound on M_{ik}^S is given by the maximum time ship k may use from its origin $o(k)$ to port i. Constraints (24) set the time each vessel uses on handling and waiting, where an upper bound on M_k^L is given by the latest time ship k may arrive at its artificial destination $d(k)$. Constraint (25) are included to tighten the formulation by defining a lower bound on the minimum time used to handle the contracts. The logic behind this is that all cargo needs to be serviced over the planning horizon. This means that the total handling time given by the left-hand-side of the constraints cannot be less than the time it takes to load and unload all cargo given by the right-hand-side, where the factor 2 is due to that all cargo needs to be both loaded and unloaded. The constraints are greater-than-or-equal since waiting can occur due to the evenly spread requirements. Symmetry-breaking constraints (26) ensure that voyages that visit ports are placed first in the voyage ordering. Please note again that it is possible to have an "empty" voyage, which means that a vessel k goes directly from its origin $o(k)$ to its artificial destination $d(k)$ at zero cost. This can happen when it is possible and optimal to handle all contractual requirements with a reduced number of voyages. Any empty voyages are placed last in the voyage ordering.

Constraints (27)–(34) concern the evenly spread requirements. Constraints (27) ensure that the sum of spread pairs for a contract c is at least the number of partial cargoes for the contract minus 1. Constraints (28) and (29) ensure that a voyage v may only be included in a spread pair for contract c if the contract is serviced on voyage v, while also ensuring that voyage v is present in at most two spread pairs for each contract. Constraints (30) and (31) guarantee that ϕ_{nc} is 1 if contract c is picked up n times. Constraints (32) and (33) ensure that the evenly spread requirement is complied with if partial cargoes of contract c is picked up on the two voyages v and w and (v, w) is a spread pair for contract c. If the desired spread is not achieved, the spread slack variable s_c may take a positive value to correct for the deviation from the desired spread. An upper bound on M_c^E is the latest time contract c may be serviced plus T^{PH} minus the earliest time contract c may be serviced. The sum of deviations for all contracts are limited by the service level Constraint (34). Constraints (35)–(41) define the bounds and requirements of the variables.

3. Three-phase MIP-based heuristic

In Hansen et al. (2019), it was shown that the proposed model could be used to solve small and medium-sized instances by a commercial MIP solver. However, this approach was of limited use for larger instances, despite allowing prohibitively large run-times. In order to solve larger instances and reduce the computational times to a practically acceptable level, we propose here a novel three-phase MIP-based heuristic for the STSRSP.

3.1. Heuristic overview

The general idea behind the proposed MIP-based heuristic is to divide the solution process into three phases which the algorithm iterates between: (1) Selection of candidate routes (Section 3.2), (2) allocation of vessels to routes (Section 3.3), and (3) solving a reduced version of the problem (Section 3.4). The motivation behind this approach is that it is possible to get a useful estimate of the total cost of a solution just by knowing the routes (phase 1) and then the vessels to sail these routes (phase 2), without dealing with the very complicating decisions related to the evenly spread requirements (handled in phase 3).

The model presented in the previous section is an arc-flow model, where the binary variables define whether a vessel sails on an arc between two nodes (ports) or not. Another commonly used approach for formulating routing models is the path-flow formulation. Here, the binary variables determine whether a vessel sails a certain preprocessed route or not. A route is here defined as a sequence of port calls. Our approach combines these formulations in order to solve the problem. In phases 1 and 2, we generate several possible routes and the path-flow formulation is used to determine the route each vessel should sail. In phase 3, we use the results from the previous phases to fix multiple variables and solve a reduced version of the STSRSP model. A flow chart of the heuristic and how it iterates between the three phases is shown in Fig. 2. Control parameters and implementation details are discussed in Section 3.5.

3.2. Phase 1: Generation and selection of promising candidate routes

The first phase of the heuristic is to identify routes that may be used in a feasible, and hopefully good, solution to the problem. For this, we first enumerate all feasible routes. We exclude routes that only visit loading ports, or only visit unloading ports, as such routes cannot both pickup and deliver partial cargoes for any contract, which is why they never be part of a useful solution to the problem. Let $\mathcal{R}$ be the set of all feasible routes, including an empty route that visits no ports for vessels that may not be used in a solution. For larger instances, this set may be huge. However, only a subset of these routes may be relevant for compiling good solutions to the problem. The goal of phase 1 is to identify such promising candidate routes, in order to avoid spending computational time on the evaluation of poor routes in the later stages. Let $\mathcal{R}'$ be the set of these candidate routes, initially an empty set. The model presented subsequently is used for selecting the candidate routes, and the routes in the solution are added to this set.

At this stage, we aim to select a number of routes equal to the fleet size (number of voyages) $|\mathcal{V}|$. However, we are not yet concerned with determining which route each vessel should sail. Recall that for each contract c, there is a minimum pickup requirement, or number of partial cargoes, which the selected routes must comply with. The model then selects routes together with the number of times the routes should

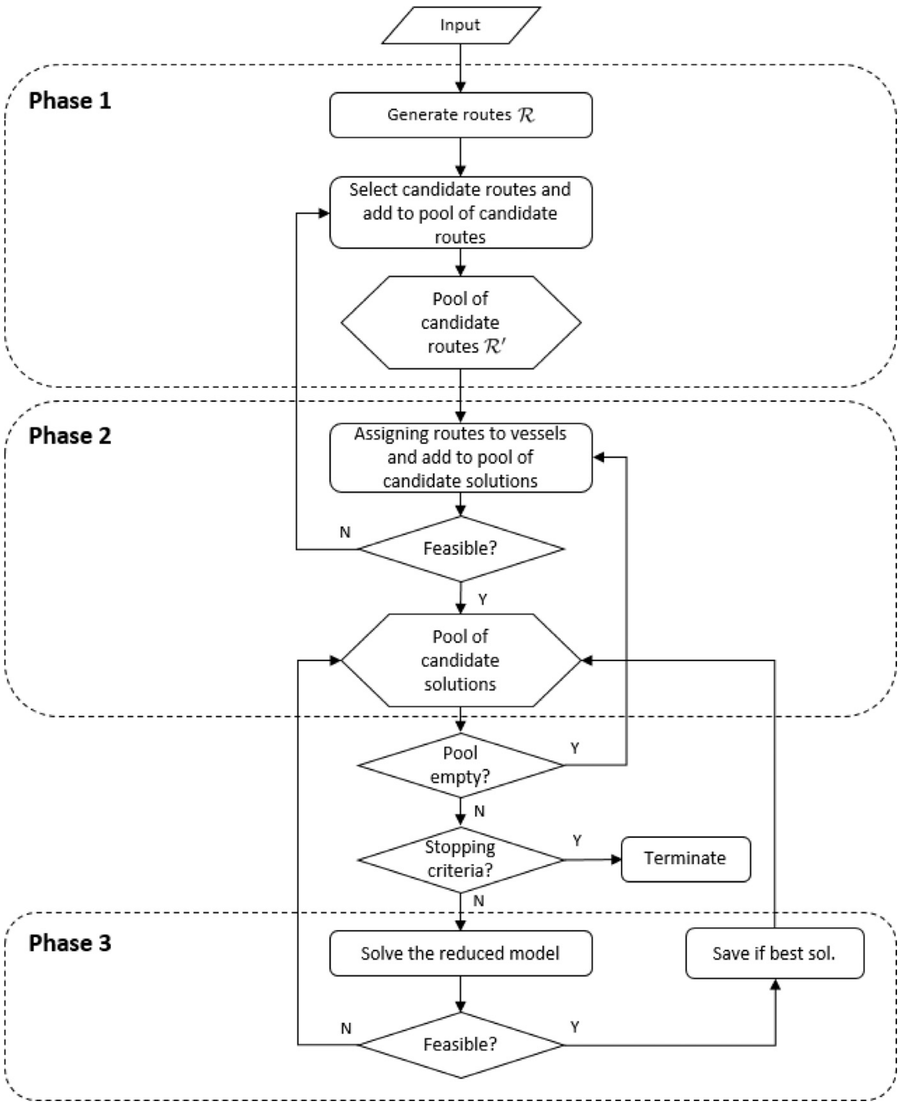
Input
Phase 1
Generate routes $\mathcal{R}$
Select candidate routes and add to pool of candidate routes
Pool of candidate routes $\mathcal{R}'$
Phase 2
Assigning routes to vessels and add to pool of candidate solutions
Feasible?
N
Y
Pool of candidate solutions
Pool empty?
Y
N
Stopping criteria?
Y
Terminate
N
Phase 3
Solve the reduced model
Save if best sol.
Feasible?
N
Y

Fig. 2. Flow chart of the MIP-based heuristic.

be sailed in order to minimize the cost of sailing cost, while respecting the minimum pickup requirements.

In addition to the previously defined sets, we define the following further parameters. Let C_r be the minimum cost of sailing route r with a vessel, including sailing costs, port visit costs, and charter costs. This minimum cost is calculated using the cheapest vessel sailing at the speed at which the sum of fuel and charter costs are minimized. Let A_{rc} be a binary parameter equal to 1 if route r visits both the loading and the unloading port of contract c. Similarly, binary parameter A^T_{rc} defines whether route r visits both the loading and unloading port within the transit time requirement of contract c. It is here assumed that the fastest possible sailing speed is used, as opposed to the calculation of the minimum costs, to preserve the entire solution space. Let the integer decision variable x_r define the number of times route r is sailed. The candidate route selection model is then as follows:

$$\min\ z = \sum_{r\in\mathcal{R}} C_r x_r \tag{42}$$

subject to:

$$\sum_{r\in\mathcal{R}} A_{rc} x_r \geq \underline{P}_c, \qquad c \in \mathcal{C} \tag{43}$$

$$\sum_{r\in\mathcal{R}} A^T_{rc} x_r \geq \underline{P}_c, \qquad c \in \mathcal{C}^T \tag{44}$$

$$\sum_{r\in\mathcal{R}} x_r = |\mathcal{V}| \tag{45}$$

$$\sum_{r\in\mathcal{R}'} x_r \leq |\mathcal{V}| - 1 \tag{46}$$

$$x_r \in \mathbb{Z}_{\geq 0} \qquad r \in \mathcal{R} \tag{47}$$

The objective function (42) minimizes the sailing cost. Constraints (43) ensure that the selected candidate routes are sufficient to meet contract c's minimum number of visits $\underline{P}_c$ in the later phases of the heuristic. Constraints (44) are similar to Constraints (43), but consider transit time contracts. A route can be used to serve contract c if it visits both the loading and unloading port within the transit time requirement of contract c. Constraint (45) ensures that the number of selected routes equals the number of voyages to be sailed over the planning horizon. Constraint (46) guarantees that at least one route is used in the solution, which is not already in the pool of candidate solutions $\mathcal{R}'$. This constraint plays a role in later stages of the heuristic, where model (42)–(47) is solved repeatedly to diversify the search by adding further routes to the candidate pool $\mathcal{R}'$. Finally, x_r is defined as a positive integer by Constraints (47). Further details on the orchestration of the generation of the candidate route set $\mathcal{R}'$ are provided in Section 3.5.

3.3. Phase 2: Assigning routes to vessels

The next phase of the heuristic is to assign a route to each vessel. From phase 1, we now have a set consisting of promising routes $\mathcal{R}'$. Given the routes in this set, the objective is to determine which candidate route each available vessel should sail to minimize the sailing costs, while respecting the minimum number of partial cargoes and capacity requirements.

Let C_{rk} be the minimum cost of sailing route r with vessel k, including sailing costs, port visit costs, and charter costs. As in phase 1, we have to ensure that we respect the minimum number partial cargoes. Let binary parameter V_{rkc} be 1 if route r visits both the loading and the unloading port of contract c and ship k is able to carry the product type of contract c, 0 otherwise. Similarly, let V^T_{rkc} be 1 if route r visits both the loading and the unloading port of contract $c \in C^T$, vessel k is able to carry the product type, and vessel k is able to comply with the transit time requirement of this contract if sailing route r at maximum speed. To ensure that the minimum capacity required is respected, the following parameters are defined. Let $\underline{Q}^{\mathrm{T}}_p$ be the minimum capacity of product type p required to fulfill the demand. Next, let $\underline{Q}^{\mathrm{P}}_{ip}$ be the accumulated minimum capacity of product type p required for the vessels visiting port i, in order to fulfill the demand. These parameters establish lower bounds on the required capacities of those ships and routes that are selected here in phase 2, which is a prerequisite for finding a feasible overall solution in the subsequent phase 3. They are computed by assuming a vessel with infinite capacity that sails along the trade, visits all ports, and picks up all cargoes along the trade. The observed load at each port for each product type then serves as $\underline{Q}^{\mathrm{P}}_{ip}$ while $\underline{Q}^{\mathrm{T}}_p$ is an aggregation of these values over all ports i. Finally, let parameter F_{ir} be 1 if route r visits port i, 0 otherwise. We define the binary decision variable x_{rk} to be 1 if vessel k sails route r, 0 otherwise. The route allocation model can be formulated as follows:

$$\min z_m = \sum_{r\in\mathcal{R}'}\sum_{k\in\mathcal{K}} C_{rk}x_{rk} \tag{48}$$

subject to:

$$\sum_{r\in\mathcal{R}'} x_{rk} = 1, \qquad k \in \mathcal{K} \tag{49}$$

$$\sum_{r\in\mathcal{R}'}\sum_{k\in\mathcal{K}} V_{rkc}x_{rk} \geq \underline{P}_c, \qquad c \in \mathcal{C} \tag{50}$$

$$\sum_{r\in\mathcal{R}'}\sum_{k\in\mathcal{K}} V^T_{rkc}x_{rk} \geq \underline{P}_c, \qquad c \in \mathcal{C}^T \tag{51}$$

$$\sum_{r\in\mathcal{R}'}\sum_{k\in\mathcal{K}} K^V_{kp}x_{rk} \geq \underline{Q}^{\mathrm{T}}_p, \qquad p \in \mathcal{P} \tag{52}$$

$$\sum_{r\in\mathcal{R}'}\sum_{k\in\mathcal{K}} F_{ir}K^V_{kp}x_{rk} \geq \underline{Q}^{\mathrm{P}}_{ip}, \qquad i \in \mathcal{N}, p \in \mathcal{P} \tag{53}$$

$$x_{rk} \in \{0,1\}, \qquad r \in \mathcal{R}', k \in \mathcal{K} \tag{54}$$

The objective function (48) minimizes the sailing cost. Constraints (49) ensure that exactly one route r is assigned to each vessel k. Constraints (50) and (51), similar to Constraints (43) and (44), ensure that both the loading and unloading port are visited at least the minimum number of partial cargoes for each contract. Additionally, they guarantee that this is fulfilled by vessels able to carry the product type p of contract c. Constraints (52) ensure that the selected vessels have enough capacity to transport the total demand of each product type p. For this, the minimum required capacity $\underline{Q}^{\mathrm{T}}_p$ has to be met by the fleet of ships that is selected in the solution of the model. Constraints (53) guarantee that the selected vessels have the capacity to transport the demand at port i of product type p, for which only those ships are taken into account that actually visit port i on their assigned routes. Finally, binary restrictions are given by Constraints (54).

The route allocation model is solved multiple times in this heuristic framework. Let $\mathcal{M}$ be the set of so far conducted iterations, indexed by m'. Let $\mathcal{O}_{m'} = \{(r,k) \in \mathcal{R}' \times \mathcal{K} : x^{m'}_{rk} = 1\}$ be the solution from iteration m', where $x^{m'}_{rk}$ is the equivalent of the variable x_{rk} in iteration m'. For example, if the solution in iteration 4 is that vessel 1 sails route 14 and vessel 2 sails route 7, then $O_4 = \{(14,1),(7,2)\}$. As we aim to generate a new candidate solution for each iteration, all previously found solutions have to be removed from the solution space. For this purpose, Constraints (55) are added to the route allocation model to guarantee that a new candidate solution is found in the next iteration $m = |\mathcal{M}| + 1$, i.e., that the solution differs in at least one of the route-allocation decisions from all previous solutions m'.

$$\sum_{(r,k)\in\mathcal{O}_{m'}} x_{rk} \leq |\mathcal{O}_{m'}| - 1, \qquad m' \in \mathcal{M} \tag{55}$$

3.4. Phase 3: Solving the reduced model

In this section we describe a modified and simplified version of the mathematical model presented in Section 2. The general idea is to use a solution $\mathcal{O}_m$ from phase 2 to fix variables in the MIP-model, which then reduces the computational time of solving the MIP. The MIP is solved several times, i.e., once for each iteration m.

Given a solution $\mathcal{O}_m$ from the mth iteration in phase 2, some of the variables in the MIP-model may be fixed. Recall that an element (r,k) in this set gives the route r vessel k sails. As route r defines a sequence of port calls, we use each element in $\mathcal{O}_m$ to fix variables x_{ijv} and y_{vk} in the MIP-model. Let r_0 be the route with zero port calls. If there is one or more elements $(r_0,k) \in \mathcal{O}_m$, these vessels k are placed last in the voyage ordering, as the vessels are not used, see Constraint (26). For all vessels that sail any route $r \neq r_0$, the vessels are allocated to voyages in ascending order with respect to the time T^A_k at which the vessels become available. Variables x_{ijv} and y_{vk} are fixed based on this voyage ordering and the elements in $\mathcal{O}_m$. As these variables are fixed, the following constraints may be removed from the model: (2)–(4), (6), (7), (9), (10). With the routing and vessel selection fixed, many binary variables are fixed or removed from the problem, which drastically reduces the computational time of solving the problem. The only remaining binary variables in the model are: Contract handled on a voyage (δ_{vc}), the number of cargo splits per contract ϕ_{nc}, and the spread pair variables (z_{vwc}).

The reduced model is solved once in each iteration. If the objective value is better than the best objective value, the solution is stored. Else, the solution is rejected. A new candidate solution from phase 2 is then selected, and the reduced model is solved again.

3.5. Control parameters and implementation details

While Sections 3.1–3.4 give a general outline of the method, this sections provides further insights of the implementation of the MIP-based heuristic. We do not intend to describe all aspects of the implementation but rather address essential control parameters and technical clarifications. Various parameters control the heuristic, such as runtimes, size of solution pools, optimality gaps, and others. The parameter values were obtained after preliminary testing where the parameter values were varied within reasonable intervals. The ones selected in the end were the combinations that performed best. We certainly do not claim to have identified the optimal set of parameter choices, but the selected values have shown to be well-functioning for all test instances.

In the first phase, we repeatedly solve the model (42)–(47) until 20 candidate routes have been selected. Preliminary testing showed that this is usually a sufficient number of candidate routes to ensure that a feasible solution can be compiled from these routes. If the candidate route selection model is recalled in a later iteration, five new candidate routes are identified and passed on to phase 2. As such, the number of routes in the candidate pool is always a multiple of five. If the model in phase 1 fails to provide a new candidate route as the problem instance does not allow for further distinct routes, the model will never be called again, as the model will then not achieve any progress at any later stage. This is for example the case for the smallest instances considered in the later experiments, who only allow for 16 different routes, such that even the first call of phase 1 creates a candidate route set with a size of below 20. In other words, for such an instance, the candidate route set involves all relevant routes and does not even constitute a heuristic reduction of the solution process in itself. The solution time of the model of phase 1 is negligible.

In phase 2, the route allocation model is used to generate candidate solutions $\mathcal{O}_m$. Whenever new candidate routes are added to the pool of candidate routes in the first phase, 100 iterations of the route allocation model are initiated. For each iteration, the resulting candidate solution $\mathcal{O}_m$ is stored in the pool of candidate solutions. The objective of the model of phase 2 associated with each candidate solution is a lower bound to the objective value for the corresponding solution in phase 3. Therefore, only solutions with better lower bound than the best-known solution are accepted. This means that any feasible solution to the problem may be a new best solution in phase 3. As this stage aims at identifying candidate solutions, we do not necessarily need to solve for the optimal candidate solution in each iteration. Hence, we accept any feasible solution to the problem in each iteration, which greatly reduces the computational time for each iteration. As for phase 1, the solution time of the model is negligible.

Preliminary testing showed that accepting any feasible solution both improves the final solution quality and reduces the run-time of the heuristic. The main reason for this is because a candidate solution may not be feasible when put in the phase 3 model. The feasible low-cost candidate solutions from phase 2 often have a higher chance of violating constraints added in phase 3 than the more expensive solutions that, e.g., have more port calls. For small instances, the choice does not matter, as all candidate solutions are evaluated within few seconds. For larger instances, selecting any feasible candidate solution is the better choice as the best bound gradually improves, instead of evaluating a large number of optimal candidate solutions that all fail in phase 3.

The MIP-based heuristic is designed to utilize multiple threads when running on a multi-core computer. One thread is used to handle phases 1 and 2. Thus, as the 100 iterations of the route allocation model are finished in phase 2, the model loops back to phase 1 and generates five new candidate routes. Next, 100 new iterations of the route allocation model are initiated. This procedure continues until a stopping criterion is met. If the route allocation model cannot find a feasible solution in any iteration, new candidate routes are generated. When the route allocation model has failed ten consecutive times, no further candidate solutions are added to the pool of candidate solutions. If the pool of candidate solutions now becomes empty, the heuristic terminates. The heuristic will also terminate if a preset maximum running time is met.

All other available threads are allocated for evaluating candidate solutions in phase 3. For each available thread, a candidate solution is pulled from the pool of candidate solutions and solved using the model described in Section 3.4, prioritizing the solution with best lower bound. The maximum running time is set equal to the remaining available time of the global heuristic running time. In most cases, only a fraction of the maximum time limit is used for this. As in phase 2, a constraint is added to ensure that the objective value is less than the so far best-known objective value. Most often, no feasible integer solution will be found, as it requires an optimal solution with a lower objective value than the best-known objective. If the objective value is better than the best objective value, the solution is stored, and the best bound is updated. A new candidate solution is then evaluated, and this procedure repeats until a stopping criterion is met. There are two stopping criteria in phase 3. First, the procedure terminates if the maximum running time is met. Second, the search is stopped if the pool of candidate solutions is empty and both the models of phases 1 and 2 fail to generate new solutions. While designed to utilize multiple threads, the heuristic may be modified to run on a single thread by solving the reduced model of phase 3 after each iteration in phase 2.

Table 1
Overview of trades.

Size	Regions	# Loading ports	# Unloading ports
Small (S)	US–Japan	4	1
Medium (M)	Asia–Europe	5	5
Large (L)	Europe–US	5	10

4. Computational study

We perform several computational experiments in order to evaluate the performance of the MIP-based heuristic. The generation of the test instances is explained in Section 4.1. In Section 4.2, we compare the heuristic with the best results reported in Hansen et al. (2019). Finally, in Section 4.3, we study the potential gains of a merger of two equally sized companies that serve the same trade regions.

For the experiments, the MIP-based heuristic was implemented in Java, while the different models of the algorithm were solved with Gurobi Optimizer version 8.1.0. We have used a computer with two Intel Xeon E5-2670v3 2.3 GHz processors and 64 GB of RAM for the testing.

4.1. *Test instances*

To evaluate the performance of our heuristic, we use the instances of Hansen et al. (2019). These instances represent realistic planning situations and are generated based on data provided by the case company. The instances are grouped into sets where three parameters are used to define each set:

1. The size of the trade (S, M, L), see Table 1.
2. The number of contracts (50 or 100). For each contract, the following parameters are defined: Loading and unloading port, total demand, and minimum and maximum pickup quantity, i.e., size of each partial cargo. For some contracts, a maximum transit time may be given. For the evenly spread contracts, a minimum and a maximum number of partial cargoes are set. 40% of the contracts have evenly spread requirements, 20% have transit time requirements, while the remaining contracts do not have any specific service requirements.
3. The service level requirement L with regards to the evenly spread of the contracts (N, M, H). For the instances with no service level requirement (N), the evenly spread constraints are relaxed by setting L to a large value. For the medium (M) and high (H) service level instances, a restricting service level threshold L is set for each instance where lower values of L give a higher service level, see constraint (34).

The combination of three trade sizes, two amounts of contracts, and three service levels results in 18 sets of instances. Each set contains five instances, resulting in a total of 90 test instances, i.e., 30 instances for each of the trades shown in Table 1. The sets are named according to the parameters used in the instance generation in the format'size-contracts-service level'. Hence, set S-100-M refers to the set of instances belonging to the small trade, 100 contracts, and a medium service level requirement. The five instances within this set are labeled S-100-M-n, where n is a number between one and five. The service level threshold L is instance specific and is set in the following way for the high service level (H): First, a given instance is solved (with the MIP-based heuristic) without any service level restriction (N). The number of vessels used in the optimal solution, N^*, is recorded. Next, a new constraint is added to the problem, limiting the number of vessels used in any solution to N^*. Finally, the problem is solved once more, with a new objective function:

Table 2
Computational results for a comparison of the optimization models and the heuristic.

Set of instances	VoyMod (1800 s)			VoyMod (10,800 s)			MIP-based heuristic (1800 s)							
	Sol time	# Feasible	Gap%	Sol time	# Feasible	Gap%	Sol time	# Feasible	Gap%	Worse	Equal	Better	Rel.imp.%	Cost per contract
S-50-N	4.2	5	0.0	4.2	5	0.0	0.6	5	0.0	0	5	0	0.0	75.0
S-50-M	9.8	5	0.0	9.8	5	0.0	2.5	5	0.2	1	4	0	0.2	76.4
S-50-H	23.1	5	0.0	23.1	5	0.0	5.0	5	4.0	2	3	0	3.9	85.3
S-100-N	9.7	5	0.0	9.7	5	0.0	0.8	5	0.0	0	5	0	0.0	38.0
S-100-M	26.2	5	0.0	26.2	5	0.0	14.0	5	0.0	0	5	0	0.0	39.3
S-100-H	46.1	5	0.0	46.1	5	0.0	18.0	5	1.1	2	3	0	1.0	42.1
M-50-N	198.1	5	0.0	198.1	5	0.0	55.8	5	0.0	0	5	0	−0.4	129.1
M-50-M	868.0	5	0.0	868.0	5	0.0	946.9	5	0.2	1	4	0	−0.3	131.2
M-50-H	1800.0	5	4.6	6 154.8	5	4.1	1800.0	5	9,6	3	0	2	5.1	156.6
M-100-N	555.1	5	0.0	602.5	5	0.0	55.8	5	0.0	0	5	0	0.0	66.9
M-100-M	1754.0	5	1.9	4 870.3	5	0.3	571.9	5	0.3	0	1	4	−1.5	67.9
M-100-H	1800.0	1	81.7	10 799.9	5	19.0	1800.0	5	20.8	0	0	5	−1.5	83.4
L-50-N	1800.0	5	2.4	7 311.8	5	1.5	1498.4	5	2.1	3	1	1	−0.2	139.2
L-50-M	1800.0	5	14.1	10 800.0	5	6.9	1800.0	5	6.3	0	0	5	−6.6	144.1
L-50-H	1800.0	1	87.0	10 800.0	2	69.8	1800.0	5	26.0	0	0	5	−10.0	175.0
L-100-N	1484.5	5	1.7	8 078.3	5	1.5	1108.6	5	1.5	1	2	2	−0.2	74.6
L-100-M	1800.0	3	49.9	9 637.1	4	28.5	1800.0	5	5.9	0	0	5	−10.7	76.8
L-100-H	1800.0	0	100.0	10 800.0	0	100.0	1440.8	4	41.9	0	1	4	–	92.2
Avg./Sum	976.6	4.2	19.1	4 502.2	4.5	12.9	817.7	4.9	6.7	13	44	33	−1.2	94.2

min $L = \sum_{c \in C} s_c$, for each instance. As such, the values of L for the high service level instances are the best possible service level that can be offered using N^* vessels. The medium service level threshold is set in the following way: $L^M = L^H + (L^N - L^H)/3$, where L^N and L^H is the service level threshold for no service level and high service level, respectively. The vessels' characteristics are provided by the case company, and all vessels may visit all ports in the test instances, i.e., no shallow water or draft limitations. For all test instances, the planning horizon is set to 30 days. Within this time limit, all sailings must begin. We refer to Hansen et al. (2019) for further details on the test instances.

4.2. Computational results

To test the capabilities of the MIP-based heuristic, we have conducted computational experiments on all 90 test instances. Hansen et al. (2019) present two models for solving the problem, namely the vessel- and the voyage-model. Thereby, the vessel-model is based on a four-index decision variable w_{ijks} whereas the voyage-model is based on a five-index decision variable w_{ijvks}. However, despite having more decisions variables, the results show that the voyage-model, hereafter referred to as VoyMod and which corresponds to the model presented in Section 2, outperformed the vessel-model in computational experiments. Hence, we only compare the results from VoyMod solved with the commercial solver with the results from the heuristic. The maximum running time used in Hansen et al. (2019) was 10,800 s (3 h) per instance. For the heuristic, we set a maximum running time, including the time for all three phases, of 1800 s (0.5 h). To evaluate the heuristic's performance in a fair way, we also consider the results achieved by VoyMod after a running time of 1800 s. The corresponding results for all sets of test instances are summarized in Table 2. For each set of instances, we report the average solution times over the five instance ('Sol time'), the number of instances for which we have obtained feasible integer solutions ('# Feasible'), and the relative gap of the obtained solutions. The gap is computed for all methods as follows:

$$\text{Gap} = \frac{\text{best solution - best lower bound from VoyMod after 10,800 s}}{\text{best lower bound from VoyMod after 10,800 s}}$$

To further analyze the heuristic solutions, we present five additional measures for them in Table 2. Columns 'Worse', 'Equal', and 'Better' show the number of instances per set where the quality of the heuristic's solutions is worse, equal, or better compared to the solutions of the optimization model under an identical runtime limit of 1800 s per instance. Column 'Rel.imp.%' shows the relative improvement of the heuristic solutions over the solutions of the optimization model, where negative values indicate that a heuristic solution has lower cost than the non-optimal solution achieved from solving the model under the given runtime limit. This measure is only computed in case that both approaches deliver an integer feasible solution, which is why no value is reported for instance set L-100-H where the model does not deliver a single feasible solution. Finally, column 'Cost per contract' shows the average logistics cost per served contract in the solutions of the heuristic.

For the sets of instances based on the small trade, all 30 instances within these sets are solved to optimality by VoyMod. The heuristic finds the optimal solution for 25 of these instances. The average computational time used by the heuristic is 1/3 of VoyMod, with an average gap of 0.9%. On the medium-sized trade, the heuristic performs clearly better than VoyMod under the same running time limit of 1800 s, though with a few exceptions. The average gap is 14.7% for VoyMod and only 5.2% for the heuristic, where the heuristic has a clear advantage in the most challenging medium-sized instances of set M-100-H. Out of the 30 instances for the medium trade size, the optimal solution is known for 22 as obtained by VoyMod (10,800 s). The heuristic finds 18 out of these 22 optimal solutions. Finally, for the largest trade, the heuristic outperforms VoyMod even more clearly. Here, the average gaps are 42.5% for VoyMod and13.9% for the heuristic. Only six large instances are solved to optimality by VoyMod, and the heuristic finds the optimal solution to four of these. Furthermore, the MIP-solver does not even find feasible solutions for 15 instances with a runtime of 1800 s and nine instances with a runtime of 10,800 s whereas the heuristic fails for only one single instance. From columns 'Worse', 'Equal', and 'Better', we find that the heuristic solutions are worse in only 13 cases, equally good in 44 cases, and better than the optimization model's solutions in 33 cases. In more detail, for the small instances the heuristic finds the same optimal solution as the optimization model for 25 out of 30 instances. For the medium instances, it even finds 11 better solutions in contrast to only four worse solutions. For the largest instances, the heuristic clearly reveals its advantageousness by producing better solutions in 22 out of 30 cases. This finding is supported by the relative improvements of the heuristic solutions, which reveal an average relative cost reduction of 1.2% over all instances. We observe only four instance sets where the heuristic performs on average worse than the optimization model (positive values in column 'Rel.imp.%) whereas it provides solutions of lower average cost for nine instance sets. In particular the solutions to the largest instances reveal significant cost savings of up to 10.7% (instance set L-100-M).

Table 3
Average results, grouped by service level.

Set of instances	VoyMod (1800 s)			VoyMod (10,800 s)			MIP-based heuristic (1800 s)		
	Sol time	# Feasible	Gap%	Sol time	# Feasible	Gap%	Sol time	# Feasible	Gap%
--N	675.3	30	0.7	2700.8	30	0.5	453.3	30	0.6
--M	1043.0	28	11.0	4368.6	29	5.9	856.0	30	2.1
--H	1211.5	17	45.5	6437.3	22	32.2	1144.3	29	17.2
Average	976.6	25.0	19.1	4502.2	27.0	12.9	817.9	29.7	6.7

The asterisks (*) denote all instances within a certain set.

Eventually, column 'Cost per contract' shows the cost increase per served contract under stricter service levels, which reveals to a company the marginal cost of promising such service quality features to the customers. We also observe that the cost per contract reduces significantly when turning from 50 contracts to 100 contracts for a same trade, which reveals substantial economies of scale for the solutions. The cost per contract also confirms the good quality of the heuristic solutions. For example, the cost per contract increases by about 21% when turning from M-50-N to the stricter service level of M-50-H where the lower bound gap of the heuristic solutions for M-50-H is 9.6%. When turning from L-100-N to L-100-H the cost per contract increases by a similar percentage value (24%) even though the lower bound gap of L-100-H is as high as 41.9%. This is a strong indicator that the large gaps are caused by a weak lower bound whereas the heuristic solutions are of consistently high quality even for the largest instances.

Table 3 shows the average results where instances are grouped by service level. Recall that the only difference between instances S-50-N-n, S-50-M-n, and S-50-H-n is the service level threshold, given by the parameter L. A lower value of L implies a higher service level, i.e., more evenly spread services of the partial cargoes for the different contracts. From the results in Table 3, we see that the service level significantly affects all three performance measures: solution times, number of feasible solutions, and gaps. We see that all methods perform well for the instances without any service level requirement (*-*-N), where the heuristic has a slight advantage with respect to solution time. For the medium service level instances (*-*-M), the solution times increase for all solution methods. The heuristic provides feasible solutions to all instances, with both considerably lower average gap and solution time than VoyMod.

We see that for the high service level instances (*-*-H), the heuristic provides feasible solutions to 29 of the 30 instances within the group. Within the same running time limit, VoyMod found only 17 feasible solutions. Furthermore, the heuristic achieves a much lower average gap compared to VoyMod for the high service level instances even though the gap significantly higher than for no requirement and medium service level instances. An inferior lower bound may possibly explain some of this larger gap. Additionally, we expect these instances to have very few feasible solutions within a reasonable range of the optimal solution value. Recall from Section 1 that for a given high service level instance, we use the best possible service level that can be offered using N^* vessels, i.e., the number of vessels used in the optimal solution for the corresponding no service level instance. For many instances, the heuristic is able to find a solution using N^* vessels. However, for some of the instances, $N^* + 1$ vessels are used, which results in large gaps.

Overall, the heuristic is shown to be a very good solution method for the STSRSP. It provides average gaps of 0.9% and 5.2% compared to the lower bounds of the commercial solver for the small and medium instances, respectively, compared to 0.0% and 14.7% for VoyMod. For the largest instances, the heuristic excelled, with impressive computational performance concerning both time and solution quality. For some of the instances with very high service level demands, the heuristic could not find near-optimal solutions. However, for a more normal service level, the heuristic achieved an average gap of 2.1%, which is considerably lower than VoyMod, with 11.0% and 5.9% for a computational time limit of 1800 s and 10,800 s, respectively.

Fig. 3 illustrates the cost breakdown and the average number of vessels (shown on top of each bar chart) for all the different instance sets. In average over all instances, the total cost is distributed as 37% sailing (fuel) costs, 18% port costs, and 45% charter costs. More interestingly, these results tell us that more ships are used to achieve a high service level, which gives an increased charter costs of 27.4% for the high service level instances (i.e., *-*-H) compared to the medium ones (i.e., *-*-M). If we look further into the solutions, we also see that the vessels in the high service level instances have more waiting and sail at somewhat lower speeds to meet the time slots at the different ports (i.e., to meet the strict service requirements). We can also note from Fig. 3 that the average port costs are 14.4% higher for the high service level instances compared to the medium ones. This is because more port calls are needed to maintain the high service level. All this information could be very useful for the case company when negotiating new contracts with potential customers, as they can now quantify the cost of having strict service level requirements.

4.3. *Merger case*

At the time of this study, the case company was in the process of merging with another equally sized RoRo-shipping company. The main motivation for this was to achieve economies of scale, especially since the two shipping companies were servicing several of the same trades and ports. Therefore, it was of great interest to see whether the merged company could service given trades and their combined cargo contracts along these trades more efficiently than the two companies separately, e.g., by using a smaller number of vessels and voyages. In this case study, the MIP-based heuristic is used to evaluate possible merger scenarios, which is one of the main reasons for developing the heuristic, as it was shown in preliminary testing that VoyMod struggled with finding feasible solutions to these large-scale merger instances.

The case is to evaluate the potential gains of merging two equally sized companies with regards to operating costs, which was a relevant study for the case company at the time this research was started. Both companies operate similar trades and transport similar amounts of cargo using the same types of vessels. We have generated 30 instances to evaluate this merger case's potential gains. Each instance was created as follows: First, two separate instances I^A and I^B were generated, one for each of the companies A and B, respectively. Next, a merged instance I^M was generated by combining instances I^A and I^B. For example, the merger instance "Merger-S-200-M" was created by merging two instances of type S-100-M, representing the instances for each of the two separate shipping companies within a single merged instance. Note that while the trades size (S) and the service level (M) are the same, the number of contracts in the merged instances is doubled. The service level for these instances is set to medium. All further instance data is generated using the procedure from Hansen et al. (2019). The potential gains of a merger are evaluated by comparing the objective values obj(I^M) (Merger) and obj(I^A)+obj(I^B) (Individually). The results are presented in Table 4, reported as the cost reduction in percentage of the merger solution compared to the individual solutions, defined as follows:

$$\text{Cost reduction (\%)}: \frac{\text{obj}(I^M) - (\text{obj}(I^A) + \text{obj}(I^B))}{\text{obj}(I^A) + \text{obj}(I^B)}$$

Table 4
Relative reduction in operating costs caused by a merger.

Set of instances	Total costs	Charter costs	Sailing costs	Port fees
Merger-S-100-M	−21%	−17%	−20%	−36%
Merger-S-200-M	−24%	−22%	−23%	−36%
Merger-M-100-M	−8%	−9%	−3%	−18%
Merger-M-200-M	−8%	−6%	−5%	−18%
Merger-L-100-M	−14%	−14%	−13%	−15%
Merger-L-200-M	−18%	−17%	−19%	−18%
Average	−16%	−14%	−14%	−23%

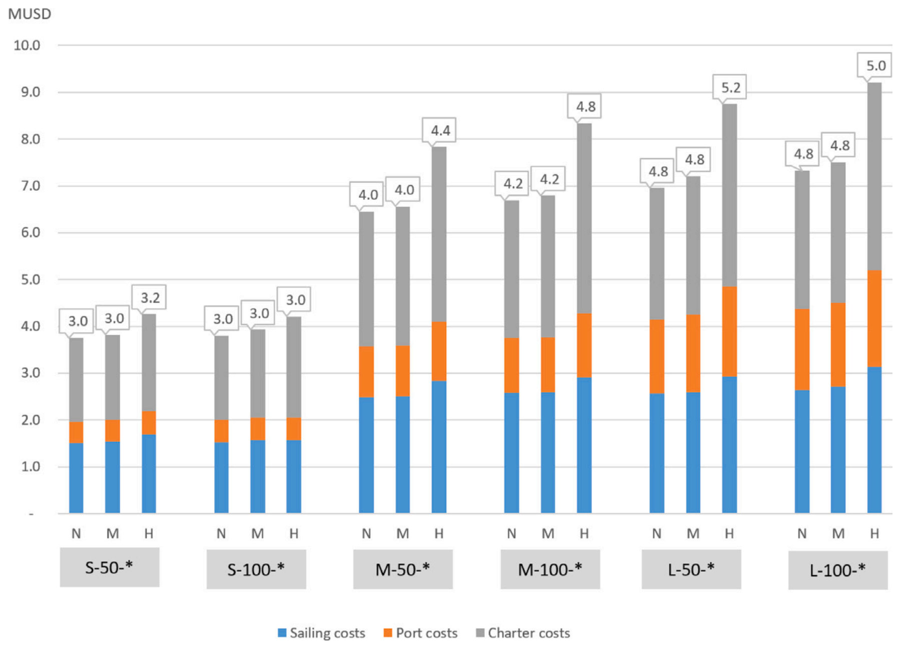

Fig. 3. Cost breakdown and average number of ships (number above the stacked bars).

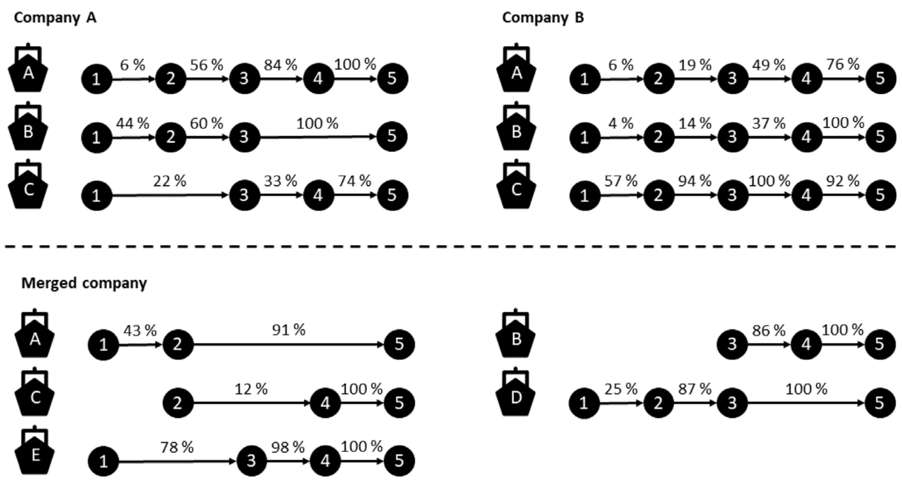

Fig. 4. On top, solutions to two separate instances are shown. At the bottom, a solution to the corresponding merger-instance is shown. Each circle represents a port visit. The number above each arrow gives the ship's capacity utilization on the sailing leg between the ports.

The results show that the merger of companies A and B gives an average cost reduction of 16% for the merger instances. For the instances based on the smallest trade, the cost savings are on average 22%. For many of these small instances, we observe that the merger-solutions use one vessel/voyage less compared to the sum of vessels/voyage used when solving for each company individually. Fig. 4 shows an example of such a solution to an instance of type Merger-S-100-M. When solved for the companies individually, we see that each company deploys three vessels along three voyages. Company A skips only two port calls, company B does not skip a single port call. At the bottom of Fig. 4, we show the solution with coordinated freight operations as they would result from the merger. Here, the merged company fulfills the same demands, with the same service level requirements, and the same planning horizon. We see major differences between the solutions. First, the merged company uses one less vessel/voyage for transporting the goods. One reason for this is that the merger solution uses other vessels

Table A.1
Notation used in the model.

Mathematical sets:	
$\mathcal{K}$	Set of available vessels.
$\mathcal{N}$	Set of nodes.
$\mathcal{N}_k$	Set of nodes that can be visited by vessel k.
$\mathcal{N}^P$	Set of ports along the trade route.
$\mathcal{N}_k^P$	Set of ports that can be visited by vessel k.
$\mathcal{A}$	Set of feasible arcs.
$\mathcal{A}_k$	Set of feasible arcs for vessel k.
$\mathcal{C}$	Set of given contracts for cargoes to be transported along the trade during the planning horizon.
$\mathcal{C}^T$	Set of contracts with transit time restrictions.
$\mathcal{C}^E$	Set of contracts with evenly spread requirements.
$\mathcal{C}_i^L$	Set of cargoes that are to be loaded at port i.
$\mathcal{C}_i^U$	Set of cargoes that are to be unloaded at port i.
$\mathcal{P}$	Set of product types.
$\mathcal{P}_p^S$	Set of product types that can be stored in the same space as product type p.
$\mathcal{S}$	Set of discrete speed alternatives.
$\mathcal{V}$	Set of voyages.
$\mathcal{V}_v^S$	Set of voyages succeeding voyage v.
Parameters:	
K_{kp}^V	Capacity for product p on ship k.
D_{cp}	Demand for the whole planning period for product type p for contract c.
$\underline{Q}_{cp}$	Minimum pickup quantity for product type p for contract c.
$\bar{Q}_{cp}$	Maximum pickup quantity for product type p for contract c.
$\underline{P}_c$	Minimum number of pickups of contract c.
$\bar{P}_c$	Maximum number of pickups of contract c.
T_{ijks}^S	Sailing time from a node (port) i to node (port) j for vessel k using speed alternative s.
T_k^A	The time vessel k becomes available at its origin.
T_i^P	Piloting time at port i.
T_p^H	The time used to handle, i.e. load or unload, one unit of product type p.
T_c^T	Maximum transit time for contract c.
T^{PH}	Length of the planning horizon.
L	Evenly spread service level threshold.
$o(k)$	Initial position of vessel k.
$d(k)$	Artificial ending position of vessel k.
$l(c)$	Loading port of contract c.
$u(c)$	Unloading port of contract c.
C_k^C	Daily charter rate for vessel k.
C_i^V	Cost of calling port i.
C_{ijks}^{SC}	Sailing and chartering costs corresponding to the piloting and sailing time from node i to j with vessel k using speed alternative s.
Big-M parameters:	
M_p^C	The maximum capacity of product type p, for all vessels k.
M_c^T	Upper bound on the maximum time a ship may use between nodes $l(c)$ and $u(c)$, when transporting contract c.
M_{ik}^S	Upper bound on the maximum time ship k may use from its origin $o(k)$ to port i.
M_k^L	Upper bound on the latest time ship k may arrive at its artificial destination $d(k)$.
M_c^E	Upper bound on the latest time contract c may be serviced plus T^{PH} minus the earliest time contract c may be serviced.
Decision variables:	
x_{ijv}	1 if voyage v uses the arc between nodes i and j, 0 otherwise.
y_{vk}	1 if vessel k sails voyage v, 0 otherwise.
δ_{vc}	1 if voyage v serves contract c, 0 otherwise.
w_{ijvks}	Weight of speed alternative s for vessel k on the arc (i,j) on voyage v.
l_{ijvp}	Load of product type p on voyage v on the arc (i,j).
q_{vcp}	Quantity of product p in contract c that is picked up on voyage v.
t_{iv}	Start time of service at node i on voyage v.
t_k^{HW}	Total time used on handling and waiting by vessel k.
z_{vwc}	1 if voyage w is the next voyage after voyage v, picking up contract c, 0 otherwise.
ϕ_{nc}	1 if contract c is picked up n times during the planning horizon, 0 otherwise.
s_c	Maximum number of days contract c deviates from the evenly spread requirement.

from the available pool of vessels, increasing the average capacity per vessel by 8% compared to the vessels used in the individual solutions. This increased capacity comes at the cost of higher charter and sailing costs per distance unit. Still, the total charter and sailing costs are reduced by 19% in the merger case because of the reduced number of used vessels. Second, the merger drastically reduces the number of port calls by about 40%. This reduction furthermore reduces the overall time spent on sailing and piloting at ports, reflected in both the charter and sailing costs. The total cost reduction for this specific merger solution is 22%.

For the medium and large instances, the merger-solutions deploy the same number of vessels along the trade. However, we still see a significant reduction in total costs of about 8% for medium sized instances and about 14% to 18% for large instances. There are several reasons for this: fewer port visits, shorter voyage lengths, lower sailing speeds, and less time spent on piloting. Of the three cost types distinguished in the table, the highest relative reduction is achieved on port call fees. However, as these fees only contribute about 10% to 15% to the total operating costs, charter and sailing costs have a higher impact on the total costs.

Given the prerequisites for this case study, it is clear from our analysis that there are significant gains of merging companies A and B with regards to operational costs. However, the potential gains from such a merger depend also on several other factors not modeled here, which may be affected in either direction. For example, fewer port calls may also reduce the total administrative costs associated with this.

5. Conclusions

In this paper we have presented a solution method for a single trade ship routing and scheduling problem for a RoRo shipping company. The goal is to determine which vessels to deploy along the trade within the planning horizon to fulfill the company's long-term contractual obligations at minimum cost. To solve this problem, we proposed a three-phase MIP-based heuristic. The heuristic solves two path-flow models in the first two phases to generate potential routing solutions. Next, these solutions are sent to a mathematical model that attempts to create a feasible schedule while minimizing costs. The computational results show that the heuristic provides high-quality solutions to instances that represent real planning situations in a short amount of time. The heuristic is especially useful for instances where commercial solvers are even unable to provide feasible solutions.

One important reason for developing this heuristic was to evaluate a possible merger case, which induced the need for solving larger instances than what is typically solved in this industry. We considered a potential merger of two equally sized companies and evaluated the potential gains of planning for both companies combined, compared to individually planning for each company. The results showed that the merged company could achieve an average operational cost reduction of 16% for this specific case. With operational costs of several million USD within each planning period, these potential savings are substantial.

CRediT authorship contribution statement

Jone R. Hansen: Conceptualization, Methodology, Software, Formal analysis, Investigation, Validation, Writing – original draft. **Kjetil Fagerholt:** Conceptualization, Methodology, Formal analysis, Investigation, Writing – review & editing. **Frank Meisel:** Methodology, Formal analysis, Investigation, Writing – review & editing.

Appendix. Notation used in the model

See Table A.1.

References

Andersson, H., Fagerholt, K., Hobbesland, K., 2015. Integrated maritime fleet deployment and speed optimization: Case study from RoRo shipping. Comput. Oper. Res. 55, 233–240.

Bakkehaug, R., Rakke, J.G., Fagerholt, K., Laporte, G., 2016. An adaptive large neighborhood search heuristic for fleet deployment problems with voyage separation requirements. Transp. Res. C 70, 129–141.

Borthen, T., Loennechen, H., Wang, X., Fagerholt, K., Vidal, T., 2018. A genetic search-based heuristic for a fleet size and periodic routing problem with application to offshore supply planning. EURO J. Transp. Logist. 7 (2), 121–150.

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

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

Christiansen, M., Hellsten, E., Pisinger, D., Sacramento, D., Vilhelmsen, C., 2020. Liner shipping network design. European J. Oper. Res. 286 (1), 1–20.

Dong, B., Christiansen, M., Fagerholt, K., Bektaş, T., 2020. Combined maritime fleet deployment and inventory management with port visit flexibility in roll-on roll-off shipping. Transp. Res. E 140, 101988.

Eide, L., Årdal, G.C., Evsikova, N., Hvattum, L.M., Urrutia, S., 2020. Load-dependent speed optimization in maritime inventory routing. Comput. Oper. Res. 123, 105051.

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

Hansen, J.R., Fagerholt, K., Meisel, F., Rakke, J.G., 2019. Planning interrelated voyages with separation requirements in roll-on roll-off shipping. EURO J. Transp. Logist. 8 (5), 633–659.

Ho-Huu, V., Hartjes, S., Péres-Castán, H., Curran, R., 2020. A multilevel optimization approach to route design and flight allocation taking aircraft sequence and separation constraints into account. Transp. Res. C 117, 102684.

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

Kisialiou, Y., Gribkovskaia, I., Laporte, G., 2018. The periodic supply vessel planning problem with flexible departure times and coupled vessels. Comput. Oper. Res. 94, 52–64.

Meng, Q., Wang, S., Andersson, H., Thun, K., 2014. Containership routing and scheduling in liner shipping: overview and future research directions. Transp. Sci. 48 (2), 265–380.

Ng, M., 2015. Container vessel fleet deployment for liner shipping with stochastic dependencies in shipping demand. Transp. Res. B 74, 79–87.

Norstad, I., Fagerholt, K., Hvattum, L.M., Arnulf, H.S., Bjørkli, A., 2015. Maritime fleet deployment with voyage separation requirements. Flex. Serv. Manuf. J. 27 (2–3), 180–199.

Pantuso, G., Fagerholt, K., Wallace, S.W., 2016. Uncertainty in fleet renewal: a case from maritime transportation. Transp. Sci. 50 (2), 390–407.

Reinhardt, L.B., Pisinger, D., Sigurd, M.M., Ahmt, J., 2020. Speed optimizations for liner networks with business constraints. European J. Oper. Res. 285 (3), 1127–1140.

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

Wang, S., Meng, Q., 2017. Container liner fleet deployment: A systematic overview. Transp. Res. C 77, 389–404.

Wetzel, D., Tierney, K., 2021. Integrating fleet deployment into liner shipping vessel repositioning. Transp. Res. E 143, 102101.