Transportation Research Part E 176 (2023) 103153

Contents lists available at ScienceDirect

Transportation Research Part E

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

An adaptive heuristic for Feeder Network Design with optional transshipment

Morten Bergmann [a], Mohamed Kais Msakni [b,*], Ahmad Hemmati [a], Kjetil Fagerholt [b]

[a] Department of Informatics, University of Bergen, Bergen, Norway
[b] Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology, Trondheim, Norway

ARTICLE INFO

Keywords:
Maritime transportation
Liner shipping
Feeder network design
Optional transshipment
Adaptive heuristic

ABSTRACT

This paper studies the Feeder Network Design Problem (FNDP), which considers the design of a minimum cost liner shipping network for the transportation of cargo (containers) between a given hub port and a set of feeder ports. In addition to determining the services to operate (i.e., the routes), the FNDP deals with deciding the fleet of vessels to deploy on these services. In contrast to most FNDPs previously studied in the literature, the feeder network can, if found beneficial, be a hub-and-spoke system where cargo can be transshipped at any feeder ports. Thus, we denote our problem as the Feeder Network Design with Optional Transshipment (FND-OT). To solve the FND-OT, we propose a novel adaptive Heuristic with a special data structure for solution representation, which gives significant speed-ups. Furthermore, a new escape algorithm is used to escape from local optima. We show that the adaptive heuristic outperforms an existing solution algorithm in the literature on a set of realistic test instances. We also present results for a new set of instances adapted from a previously published benchmark suite (LINER-LIB) and show that including the possibility of having cargo transshipment in the FNDP can give significant benefits and reduced costs.

1. Introduction

Maritime transportation is a key element for the current world trade growth. With more than 200 countries open to container ships, connection between continents, countries and markets is made possible at a scale not previously seen. Container ships have the capacity to carry thousands of twenty-foot equivalent container units (TEUs) containers, e.g., the capacity of a Triple E class vessel is more than 18 000 TEUs. This large capacity and economies of scale make container shipping attractive for transporting goods in terms of costs. In addition to that, large ships are more energy-efficient and have a lesser impact on the environment than other modes of transportation. All these factors have contributed to establishing maritime transportation as the most important transportation mode for global trade, accounting for around 80% in terms of volume and 70% in terms of value. The container liner shipping industry is particularly important with a value of more than four trillion USD, accounting for around two-thirds of the world's maritime trade (World Shipping Council, 2020).

The container liner shipping sector is characterized by regular scheduled services that cover several geographical regions. A particular service represents a single trade route formed by a sequence of ports visited in the same order. It is a common practice in liner shipping to have weekly visits of each port of a service. The shipping companies announce their services and schedules several weeks to months in advance, which offers predictability for customers. Also, the regularity of port visits simplifies the transit of

* Correspondence to: Department of Industrial Economics and Technology Management, NTNU, 7491 Trondheim, Norway.
E-mail address: kais.msakni@ntnu.no (M.K. Msakni).

https://doi.org/10.1016/j.tre.2023.103153
Received 14 November 2022; Received in revised form 10 March 2023; Accepted 9 May 2023
Available online 7 June 2023

Table 1
Summary of the literature on Feeder Network Design problems. For the cargo type, "P&D" refers to pickup and delivery from and to the hub port, and "Rejection" designates the possibility of cargo rejection.

Paper	Main problem characteristics					Approach
	Fleet	Cargo	Time window	Speed	Transshipment	
Fagerholt (1999)	Homogeneous					MILP
Sambracos et al. (2004)	Homogeneous					Metaheuristic
Fagerholt (2004)	Heterogeneous					MILP
Karlaftis et al. (2009)	Heterogeneous	P&D	✓			Hybrid GA
Santini et al. (2018)	Heterogeneous	P&D		✓		Branch-and-Price
Holm et al. (2018)	Mother & Daughter	P&D			Specific locations at sea	MILP,
Medbøen et al. (2020)	Mother & Daughter	P&D			Specific locations at sea	Simulation-Optimization
Akbar et al. (2020)	Mother & daughter	P&D			Specific local ports	MILP
Msakni et al. (2020)	Heterogeneous	P&D			Any local port	MILP
Hellsten et al. (2021)	Heterogeneous	P&D ; Rejection				Branch-and-Price
Jin et al. (2021)	Heterogeneous	P&D	✓	✓	Synchronization with long haul services	Branch-and-price
This work	Mother & daughter	P&D			Any local port	Adaptive heuristic

containers between services until they reach their final destinations. According to World Shipping Council (2020), there are around 500 liner shipping services operated around the world by more than 6,000 ships.

The problem of creating services or shipping routes is known in the literature as the liner shipping network design problem (LSNDP), which is known to be a very complex planning problem. The LSNDP involves high operational costs, including vessel costs (i.e., time charter costs) and variable sailing costs, such as fuel costs, port costs, and other cargo handling costs. The objective of the LSNDP is to satisfy customer demands at competitive prices. Hence, achieving high capacity utilization for the deployed ships is crucial for liner shipping companies.

There exist several studies on developing operations research and optimization models and methods for different versions of the LSNDP, e.g., Agarwal and Ergun (2008), Álvarez (2009), Wang and Meng (2014), Zheng et al. (2015), Thun et al. (2017), Karsten et al. (2017), Balakrishnan and Karsten (2017), Koza et al. (2020), and Ameln et al. (2021). In particular, Brouer et al. (2014) present a real case of the LSNDP based on the largest liner shipping company in the world, from which a set of benchmark instances, called LINER-LIB, is derived.

This paper presents an LSNDP for a *feeder network* where the cargo (i.e., containers) is to be transported from the hub to the feeder ports and vice versa. This network structure gives a particular case of the LSNDP known as the Feeder Network Design Problem (FNDP). The objective is, as for the LSNDP, to establish routes and assign vessels (number of each ship type) to deploy on these routes so that the weekly demand of all ports is satisfied while minimizing the operational costs. Typically, the FNDP is designed to transport goods between a major hub port and local ports in a restricted geographical area, e.g., between the port of Rotterdam to ports along the Norwegian coast. A common assumption in the FNDP is that the cargoes are directly transported between the hub and the feeder ports with no possibility of transshipment at feeder ports (Meng et al., 2014). However, a recent study by Msakni et al. (2020) shows that allowing transshipment at any feeder port and relaying the shipping system with even smaller vessels to serve local ports can reduce total operational costs.

Table 1 gives an overview of the literature in the FNDP. One of the first studies to address this problem is the one by Fagerholt (1999), where the objective is to find weekly route services at minimum operational costs. Fagerholt (2004) includes the possibility to deploy a heterogeneous fleet of vessels that can run at different speeds. Karlaftis et al. (2009) consider a feeder network with no service frequency and time deadlines in ship arrivals. The problem is formulated as a variant of the vehicle routing problem with pickups and deliveries. Santini et al. (2018) develop a branch-and-price algorithm to solve the FNDP. The results are reported for instances derived from LINER-LIB. A recent contribution is by Hellsten et al. (2021), who consider the FNDP for a weekly shipping system with cargo rejection at a penalty cost. The problem is solved using a branch-and-price framework, which can solve instances with up to 12 ports to proven optimality, as well as the BALTIC instance from LINER-LIB. To reduce the cargo transit time, Jin et al. (2021) considers synchronizing feeder vessel schedules at the hub port with the schedules of the long-haul services. Specifically, the arrival time of a shipment at the hub port from a feeder port must fall within a predefined time slot.

There are also a few recent studies of the FNDP, which also include the option of having transshipment of cargo at feeder ports, resulting in a possible hub-and-spoke feeder system. Holm et al. (2018) study a particular case of such an FNDP, in which the liner shipping network is composed of mother ships that sail a single mother route serving main (hub) ports only, while daughter ships can visit small (spoke) ports as well as the main ports not visited by the mother route. This system has the peculiarity of transshipping cargo between mother and daughter ships at sea and not at regular ports. A critical operation to consider while creating the routes is that synchronization has to be ensured to guarantee that both mother and daughter ships meet at the same location at the same time at sea for the transshipment. This problem was later extended by Medbøen et al. (2020) with a simulation engine embedded within the optimization algorithm to handle uncertainty in sailing times. A similar study is performed by Akbar et al. (2020) with the same mother and daughter route structure. However, daughter routes are sailed by autonomous ships, and the transshipment is done at regular ports. The aim was to study the economic impact of introducing autonomous ships in a short-sea feeder shipping network.

Msakni et al. (2020) examine a case study of a Norwegian liner shipping company operating a feeder network. The paper proposes a model to select the best alternative between an FNDP and a hub-and-spoke network where transshipments can take place at any feeder port. The objective is to examine which alternative between the two types of network, as well as simple and butterfly route structures, perform better for the liner shipping company. In addition, the shipping system allows for splitting up deliveries and pickups. Msakni et al. (2020) formulate the underlying problem using a mathematical model and derive from it a simple heuristic that is shown to provide good results for the case study, which has a special geographical structure where the ports are located more or less along a coastline.

This paper studies a similar version of the FNDP as Msakni et al. (2020), and does as such explore the possibility of including cargo transshipment at feeder ports, resulting in a hub-and-spoke network, if found beneficial. We, therefore, denote this problem *Feeder Network Design with Optional Transshipment* (FND-OT). In addition to having routes that carry cargo directly between the hub and the feeder ports with mother ships, transshipment can take place at intermediate feeder ports using daughter ships. The possibility of including transshipment and daughter ships is optional and will only be used if it reduces the total costs.

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

- We propose a novel adaptive heuristic for the FND-OT. The heuristic exploits an efficient data structure for representing the solutions in order to reduce the number of calculations at each iteration of the algorithm. With the proposed solution representation, modifying the solutions becomes easier and more efficient. Our proposed algorithm also uses a random restart mechanism to escape from local optima.
- We show in the computational study that the proposed heuristic outperforms the previous algorithm by Msakni et al. (2020) on the real test instances from that paper as well as on a set of adapted test instances from the LINER-LIB benchmark suite.
- Finally, the most important point is that we show that allowing transshipment at any feeder port might potentially result in significant cost reductions for liner shipping companies when designing feeder networks. We also show that this is independent of the geographical structure of the problem instance.

The outline of the remainder of this paper is as follows. Section 2 describes in more detail the FND-OT. The proposed adaptive heuristic is presented in Section 3. Section 4 shows the computational study, while Section 5 concludes the paper.

2. Problem description

In FNDP, all cargoes (i.e., containers) are to be transported between a given hub port and a given set of feeder ports. The problem is to design a network of routes that ensures that the given weekly demand of all ports is transported, as well as to determine the fleet of vessels to service these routes. In traditional feeder networks, there is no transshipment, meaning that the cargo is transported between any two ports using a single vessel. Any route in the feeder network begins and ends at the hub port and can serve all or a subset of the feeder ports. We denote these routes for *mother routes*, and the vessels deployed on these routes are called *mother vessels*. Since the weekly demand is assumed to be known, the amount of cargo transported along any such mother route can be computed so that the vessel capacity required to sail this route can be derived. Also, the number of mother vessels to deploy on a route can be derived from its duration, which consists of sailing and port times along the route. So to ensure a weekly visit of all served ports, the number of deployed vessels must be equal to the number of weeks to sail the route rounded up to the nearest integer. For example, if two different mother routes last for six and 19 days, one and three mother vessels are deployed on the first and second routes, respectively. We assume that it is possible to deploy a heterogeneous fleet of vessels, meaning that two different mother routes can be serviced by two different vessel types. However, when there are two or more vessels sailing the same route, these vessels must have the same capacity because they visit the same ports with the same demand.

In contrast to the traditional FNDP, we allow for the possibility of having a hub-and-spoke structure within the feeder network, where cargo can be transshipped between two routes at any feeder port. We denote the routes that are connected to a mother route through the chosen transshipment (feeder) ports for *daughter routes*, where each daughter route connects a subset of feeder ports to only one mother route. As such, the daughter routes become feeder routes within the feeder liner shipping network. The vessels deployed to these routes are called *daughter vessels*.

The possibility of transshipping cargo is optional and incurs extra costs in the transshipment ports, i.e., cargo handling between mother and daughter vessels, as well as additional port visit fees for the daughter vessels. This means that having transshipment by including daughter routes is optional, i.e., it is done only when found beneficial in terms of operational costs. Therefore, we denote this version of the problem as the Feeder Network Design with Optional Transshipment (FND-OT). In fact, daughter routes are sailed by small vessels (e.g., with a capacity of only a few hundred TEUs) that have lower weekly charter costs and fuel consumption compared to mother vessels. Since it is rarely beneficial to have long daughter routes due to their low capacity, we assume that the maximum total duration of a daughter route is one week to ensure a weekly visit for the serviced ports. This also means that each daughter route can be serviced by one daughter vessel. In addition, we assume that each daughter route has only one transshipment port, i.e., it can only be connected to one mother route.

Fig. 1 illustrates an example of FNDP with one hub port and six feeder ports. While conventional feeder networks consist only of mother routes, as shown in the leftmost part of the figure, the rightmost part illustrates a possible solution to the FND-OT consisting of a hub-and-spoke structure with both mother and daughter routes.

We consider two types of daughter route structures. The first type starts from the transshipment port, services a set of feeder ports, and ends at the transshipment port, resulting in a *simple cycle route*. The second type of daughter route is the so-called *butterfly route*, which is obtained by mixing two simple cycle routes such that their total duration does not exceed one week. For this route

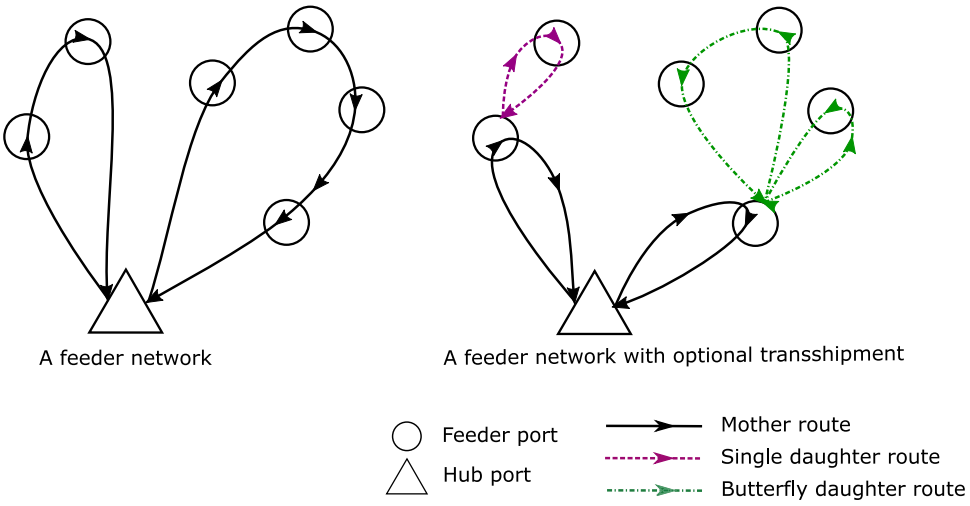

Fig. 1. Illustration of a traditional feeder network with only mother routes (to the left) and a solution with hub-and-spoke structure in the case when transshipment is allowed as for the FND-OT (to the right).

type, the transshipment port is visited twice and may have the advantage of better usage of vessel capacity. In the rightmost part of Fig. 1, an example of both a simple cycle and a butterfly route is shown.

To summarize: the FND-OT consists of determining (1) a set of mother and daughter routes, (2) which feeder port(s) to use as transshipment port(s) (if any at all), as well as (3) the fleet of mother and daughter vessels deployed to these routes, so that all cargo demand between the hub port and the feeder ports is satisfied and the total operational cost is minimized. A mathematical formulation of the problem is provided in Appendix A.

3. Adaptive heuristic

As shown by Msakni et al. (2020), a commercial MIP solver can solve only relatively small instances of the problem using the model presented in Appendix A, mainly due to the large number of possible non-dominated routes (and hence variables). Even the heuristic proposed in Msakni et al. (2020), which is derived from the mathematical model, is limited since it is tailored to problem instances with a specific geographical structure, where ports are located along a coastline.

In this study, we propose an adaptive heuristic to solve the FND-OT for a problem with a general geographical structure. The adaptive heuristic includes various heuristics to diversify the type of solutions explored and to make it more reliable to variations between instances.

3.1. Overview of the adaptive heuristic

The pseudo-code of the adaptive heuristic is given in Algorithm 1. The algorithm starts by generating an initial solution s using the nearest neighbor algorithm explained in Section 3.3. Then, it moves into a loop block (Lines 5 to 21) that aims to improve the current solution by applying different heuristics. For each iteration, Line 12 generates a new solution from the current one. If the objective function value of the solution s' is better than the value of the current best one s_{best}, the best solution is updated, as stated in Line 18. The new generated solution can replace the current solution based on the acceptance criteria described in Section 3.7. The algorithm terminates when the stop condition is reached. To diversify the search space, an *Escape* algorithm is applied at the beginning of the loop block after checking an escape condition (Line 6). The details of this algorithm are described in Section 3.8.

The following sections explain in detail each part of the adaptive heuristic.

3.2. Solution representation

A major feature of the proposed adaptive heuristic is the way to represent a solution. As will be seen in the subsequent sections, this representation makes it easier to generate a new solution while verifying the decisions mentioned above. Mainly, a solution (a complete network with a fleet of vessels) can be coded using two vectors: a *main vector* and a *transshipment vector*.

The main vector uses the symbols “M”(mother), “D”(daughter) and “L”(loop) to separate routes from each other. “M” indicates the start of a new mother route, “D” the start of a new daughter route, and “L” the start of a new loop within a butterfly route. The main vector is composed of two parts. The first part, denoted hereafter by Part 1, is delimited by the first “D”. The second part, denoted by Part 2, is what comes after the first “D”.

Algorithm 1 Adaptive Heuristic

Inputs: a set of heuristics $\mathbb{H}$ ▷ **Section 3.4**
Generate an initial solution, s ▷ **Section 3.3**
$s_{best} \leftarrow s$
$i \leftarrow 0$
repeat
  if Escape_Condition(i, m) **then**
    Escape_Algorithm(s, s_{best}) ▷ **Section 3.8**
    $i \leftarrow 0$
  end if
  $s' \leftarrow s$
  select $h \in \mathbb{H}$ based on selection parameters ▷ **Section 3.5**
  apply heuristic h to s'
  if $f(s') < f(s_{best})$ **then**
    $s_{best} \leftarrow s'$
    $i \leftarrow 0$
  end if
  if $accept(s', s)$ **then**, ▷ **Section 3.7**
    $s \leftarrow s'$
  end if
  update selection parameters and $i \leftarrow i + 1$ ▷ **Section 3.6**
until stop condition
return s_{best}

Part 1 represents the sequence of ports visited by the mother routes (one or many). The port sequence does not include the hub port as any mother route departs and arrives at this port.

Part 2 is the remaining part of the main vector. If it exists, Part 2 represents the daughter routes (separated by "D"s) connected to the mother routes defined in Part 1. The order of ports between two "D"s corresponds to port visits of the corresponding daughter route. The transshipment port from which the daughter route starts and ends is not included in this sub-sequence but is part of the *transshipment vector*. Moreover, the sub-sequence of ports indicates whether the daughter route has a simple or a butterfly structure. If there is a "L" tag in the sub-sequence, the route is a butterfly route, in which the "L" separates between the first and the second loop.

Transshipment vector denotes the transshipment port for each daughter route included in Part 2. The transshipment vector's length corresponds to the number of daughter routes. Each element of this vector indicates the index of the transshipment port in Part 1, based on the order of the daughter routes in Part 2.

An example of the solution representation is given in Appendix B.

3.3. Initial solution

The proposed heuristic starts with an initial feasible solution. It is advantageous for the algorithm if the initial solution contains both mother and daughter routes. To this end, we use the nearest neighbor approach to create a solution that contains mother routes and possibly daughter routes. The closest feeder port to the hub port is placed in a mother route. Then, the port closest to the most recently added port is selected and is inserted in one of four ways:

1. at the end of an existing daughter route
2. as a new daughter route
3. at the end of an existing mother route
4. as a new mother route

The above insertion methods are ranked to their order of execution. First, the initial solution procedure tries Option 1. If it fails to produce a feasible solution, it moves to Option 2, then Option 3, and finally Option 4. For the first three options, the capacities of the concerned routes are updated, and the smallest vessels are assigned to them. If an option yields an infeasible solution, its related solution is discarded. However, the last option is always possible and generates a feasible solution because it is assumed that any port demand can be handled by at least the largest mother ship. In general, it is easier to insert a selected port into an existing or new mother route than into a daughter route, which means that in most cases, an initial solution will not contain any daughter routes.

3.4. Heuristics

In this section, we describe various heuristics that are called in Line 11 of Algorithm 1. Each heuristic operates differently and targets part of the solution representation. When combined, the heuristics ensure a good balance between intensification and diversification during the search. We point out that the list of developed heuristics is not limited to the ones presented in this section. We have implemented other heuristics that we have not included in this paper because, according to preliminary results, they do not significantly improve the performance of the adaptive algorithm.

3.4.1. Mother route to daughter route

In this heuristic, we randomly select a port from the mother route (Part 1) and move it to a daughter route (Part 2). This operation might alter the indices of the transshipment vector. For this step, any entry in the transshipment vector having a value greater than or equal to the index of the selected port in Part 1 must be decremented. For the insertion of the selected port in Part 2, all possible positions are tested to retain the one offering the best operational cost. The possibility of making a new daughter route with only the selected port is also considered in this step. In the case where no feasible solution could be obtained from all possible positions, the heuristic returns the input solution without any change.

3.4.2. Mother route to mother route

This heuristic randomly selects a port from Part 1 and inserts it in a different position in the same part. This results in one of the following: (i) a change in the order of visits of the concerned mother route, (ii) a transfer of the selected port from one to another mother route, or (iii) a creation of a new mother route visiting only the selected port. Indeed, Part 1 can represent more than one mother route. This is made possible by introducing a cell with the "M" tag to indicate that the ports' sequence belongs to separate mother routes. In a case where all possible re-insertions of the selected port produce infeasible solutions, the heuristic keeps the initial solution.

Obviously, the transshipment vector may require an adjustment to make the port re-insertion correct. The adjustment is made in two steps. The first step decrements any value in the transshipment vector that is greater than or equal to the removed port index. The second step increments the values of the transshipment port that are greater than or equal to the new index of the re-inserted port. By doing so, daughter routes keep the same transshipment ports. This step is followed by the "New transshipment" heuristic, described in Section 3.4.5, which aims at optimizing a particular transshipment vector.

3.4.3. Swap in the mother route

This heuristic randomly swaps two ports of Part 1. The transshipment vector is still valid as the number of ports in Part 1 remains the same. The new solution may be infeasible. In this case, the heuristic goes back to the initial solution.

3.4.4. Daughter route to mother route

This heuristic transfers a port from a daughter route to a mother route, i.e., a randomly selected port from Part 2 is removed and inserted in Part 1. Two particular cases can hold. If the daughter route serves only the chosen port, this route is removed, and its corresponding entry in the transshipment vector is also removed. The second case is when the related daughter route is a butterfly route, and one loop contains only the selected port. In this case, this daughter route becomes a simple cycle, i.e., the tag "L" is removed from the corresponding sub-sequence in Part 2.

All possible insertions in Part 1 are tested, including creating a new mother route with only the selected port, and the best feasible insertion is retained. The transshipment vector is updated in a similar way, as described in Section 3.4.2.

3.4.5. New transshipment

This heuristic randomly selects a cell in the transshipment vector and tries to find a better transshipment alternative in terms of operational costs for the corresponding daughter route.

3.4.6. Daughter route to daughter route

This heuristic randomly moves a port from one daughter route to another daughter route. All possible insertions are tested to keep the best option, including having a new daughter route. If the selected port is the only port in the original daughter route, this route is removed, and its corresponding transshipment entry is also removed. If the original daughter route is a butterfly and has only the selected port in one of its loops, this route is converted to a simple daughter route. For the case where a new daughter route is created, the heuristic described in Section 3.4.5 assigns a transshipment port for it.

3.4.7. Several ports from mother route to daughter route

This heuristics is similar to the *mother route to daughter route* heuristic, with the difference that several ports are involved at the same time. The selected ports are inserted one by one following the same procedure explained in Section 3.4.1.

3.4.8. Make butterfly

This heuristic converts a daughter route from a simple to a butterfly structure. More precisely, a daughter route with more than one port visited is split into two loops. i.e., by inserting the tag "L". This heuristic tries all possibilities to consider the best split. It is clear that the transshipment vector does not require any adjustment.

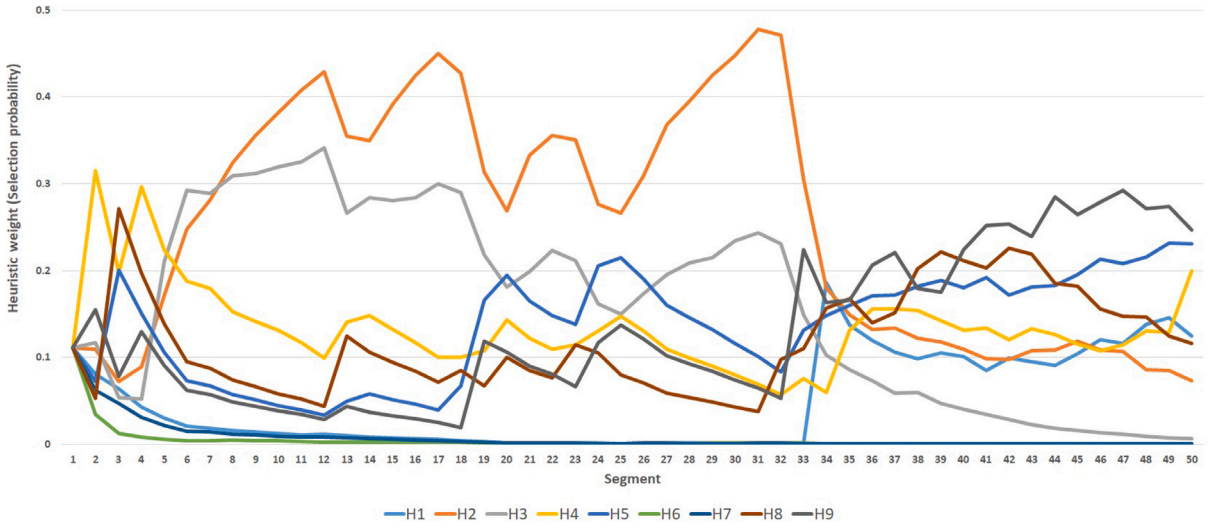

Fig. 2. An illustration of the development of the heuristics weight probability when running the adaptive algorithm. The x-axis represents segments and the y-axis represents the probability of selecting a heuristic.

3.4.9. Remove butterfly

Unlike in the "make butterfly" heuristic, the "remove butterfly" heuristic transforms a butterfly route into a simple daughter route. The heuristic considers this transformation only if it provides better operational costs.

3.5. Choosing a heuristic

Alternating between several types of heuristics allows to have a more robust algorithm (Moshref-Javadi et al., 2020). The selection of a heuristic in each iteration of Algorithm 1 is made according to the *roulette wheel* principle where a heuristic h is selected among the set of all heuristics $\mathbb{H}$, presented in Section 3.4, with a probability of $\frac{w_h}{\sum_{g\in\mathbb{H}} w_g}$, where w_h is the weight of heuristic h.

The weights are adjusted automatically as explained in Section 3.6.

3.6. Adaptive weight adjustment

For each iteration of the main loop in Algorithm 1, a heuristic is selected based on its weight (Line 11) to apply to the current solution (Line 12). The heuristic weights are iteratively adjusted using statistics from earlier iterations (Ropke and Pisinger, 2006). The basic idea is to keep track of each heuristic performance through a scoring system, where a heuristic is given a higher score for better performance and a lower score for worse performance. For this purpose, we divide the search process into segments. At the beginning of the adaptive algorithm, all heuristics are given the same weight, meaning they have an equal probability for selection during the first segment.

For each iteration of the adaptive heuristic, a reward system is adapted according to the following criteria:

- a heuristic that finds a new global best solution is given a high score,
- a heuristic that finds a better solution than the current solution is given a medium score, and
- a heuristic that finds a new solution that has not been found before is rewarded a small score.

At the end of each segment, the sum of the cumulative scores of each heuristic is used to update the weights. Let s be the current segment, $w_{h,s}$ be the weight of heuristic h in s, π_h be the score of the heuristic h, and λ_h be the number of times the heuristic h was run in s, the updated weight of h is computed as follows:

$$w_{h,s} = w_{h,(s-1)}r + (1-r)\frac{\pi_h}{\lambda_h}, \tag{1}$$

where r represents the percentage of the historical weight factor. The weight of the previous segment represents r percent of the current weight, while the remaining $(1-r)$ comes from the computed weight of the current segment.

We provide in Appendix C evidence of the importance of including the adaptive mechanism to find better solutions. Furthermore, we show in Fig. 2 an example of the evolution of the selection probabilities of the nine heuristics proposed for the adaptive algorithm, where H1 to H9 denote the heuristics presented in Sections 3.4.1 to 3.4.9, respectively. We observe that all heuristics start with the same probability selection, and after only a few segments, some heuristics obtain higher probabilities while others obtain reduced

ones. In segment 33, H1 generates better solutions and sees its weight increase. This increases the probability selection of H1 and impacts other probabilities, especially for H2. For this example, we can see that H1, H6, and H7 have lower probabilities due to normalized scores across all heuristics. However, as supported by additional experiments, the inclusion of these heuristics is necessary as they generally improve the performance of the adaptive heuristic.

3.7. Acceptance criteria and stopping condition

We use an acceptance criterion in Line 17 of Algorithm 1 whether to accept a new solution or not. A better solution than the current one is always accepted; however, a worse solution is accepted only if it verifies the Boltzmann probability function, which is defined as $e^{-|f-f_{new}|/T}$, where $T > 0$ is the temperature, f is the objective value of the current solution and f_{new} is the objective value of the new solution.

Besides, we implement the cooling schedule suggested by Crama and Schyns (2003), which is defined as $T_i = \alpha T_{i-1}$, where i is the iteration and α is the cooling factor. The initial temperature, T_0, is calculated as follows:

We run the first 100 iterations with a fixed acceptance rate of p. We calculate the average of all $|f - f_{new}|$ for worse solutions that are accepted over these iterations, f^T_{avg}. Then, we calculate the initial temperature as follows:

$$T_0 = \frac{-f^T_{avg}}{ln(p)} \tag{2}$$

Algorithm 1 stops after n iterations. We use the initial temperature T_0, the final temperature T_f and n to find the cooling factor α. We find a value for α such that $T_f = 1$ is reached in n iterations. α is calculated as follows:

$$\alpha = (\frac{T_f}{T_0})\frac{1}{n} \tag{3}$$

3.8. Escape algorithm

Although we propose different improving heuristics, it is possible not to obtain any new improved solution after several iterations. Even with the implementation of the acceptance criterion, the adaptive algorithm can be caught in a local optimum. To escape from this situation, we have introduced an *Escape Algorithm* (described in Algorithm 2) that is called when there is no improvement number m iterations.

Algorithm 2 Escape algorithm procedure

```
1: procedure ESCAPE_ALGORITHM(s, s_best: solution)
2:     repeat
3:         choose a random heuristic h from ℍ                ▷ Section 3.4
4:         apply h to s
5:         if f(s) < f(s_best) then
6:             s_best ← s
7:         end if
8:     until stop condition met
9:     return the modified s and s_best
10: end procedure
```

The Escape algorithm aims to move as far away from the current solution s. It iteratively applies heuristic h on s without checking the quality of the solution obtained (Lines 3 to 4). The selection of a heuristic is entirely random, without the application of weighted probability. Once a better solution is found, the procedure updates s_{best} that will be later transferred to the adaptive algorithm. The escape algorithm stops after a number of iterations or if a new s_{best} is found. The modified s and s_{best} are returned.

4. Computational experiments

The aim of this section is to assess the performance of the adaptive heuristic for solving the FND-OT. The experiments are based on instances of Msakni et al. (2020) and adapted instances of the benchmark suite LINER-LIB introduced by Brouer et al. (2014). The adaptive heuristic was implemented using Kotlin/JVM and compared with the mathematical model (see Section 3) solved using Cplex 12.9. Since this model is based on a path-flow formulation, the routes are first pre-generated before calling the solver. Due to its exponential nature, the number of generated routes is limited to include only visits to the kth nearest port from a current port. This approach was proposed by Msakni et al. (2020) and constitutes a reference to compare with for the adaptive heuristic. The experiments with the adaptive heuristics were performed on a late 2013 MacBook Pro with 2 GHz Intel Core i7 and 8 GB of RAM.

The adaptive heuristic includes a number of parameters. Preliminary testing showed that the performance of the heuristic was relatively stable with respect to the following parameter values:

p Initial probability of selecting a worse solution. We set this to 0.8, meaning worse solutions are accepted 80% of the time when finding the initial temperature.

Table 2
Class A, B and C instances and the adapted LINER-LIB instances.

Category	# Ports	Description	Total demands (TEUs)
Class A	10–11	Rotterdam - Norwegian coastline	1070–1579
Class B	15	Rotterdam - Norwegian coastline	1074–1725
Class C	20–22	Rotterdam - Norwegian coastline	1850–4126
BALTIC	12	Bremerhaven - Baltic sea	4904
WAF	20	Algeciras - West Africa	8489
MEDIT1	33	Algeciras - Mediterranean - Canarias	2705
MEDIT2	32	Gioia Tauro - Mediterranean - Canarias	599

n Number of iterations for the adaptive heuristic, set to 20,000 iterations.

m Number of iterations without improvement before the Escape algorithm is applied in the adaptive heuristic. This is set to 1000 iterations.

l Number of iterations for the Escape Algorithm, set to 20 iterations.

r Historical weight factor, which we have set to 80%.

s Segment size, set to 200, meaning that the weights are adjusted every 200th iteration.

Section 4.1 describes the test instances used in the computational tests, while Section 4.2 presents the results. Finally, in Section 4.3, we analyze the value of allowing transshipment in the feeder network. Here, we also randomly generate a number of additional test instances with different sizes and geographical structures to provide some additional insights into which situations it can be beneficial to allow transshipment.

4.1. Test instances

The first set of instances was introduced by Msakni et al. (2020). It is based on realistic data from a Norwegian liner shipping company transporting cargo between a hub port in Rotterdam and 21 ports along the Norwegian coastline. From this case, three classes denoted A, B, and C, were derived, differing in the number of ports included and the total demand. The class A instances include 10 or 11 ports, while the instances of classes B and C include 15 and 20 to 22 ports, respectively. Each class consists of four instances.

The second set of instances is adapted from the publicly available LINER-LIB benchmark suite (Brouer et al., 2014). Originally, this benchmark was proposed for the general case of the liner shipping network design problem. Considering that our problem is defined for the FNDP with one hub port that serves many feeder ports, three out of seven instances of LINER-LIB can be retained for our experiment. The BALTIC instance has Bremerhaven as a hub port and includes 11 ports at the North Sea and Baltic Sea. The hub port of the WAF instance is Algeciras which serves 19 feeder ports on the coast of West Africa. The Mediterranean instance of LINER-LIB originally has a multi-hub structure, where the hubs are Algeciras, Tangier, and Gioia Tauro, and serves feeder ports mainly located in the Mediterranean region. We split this instance up into two instances. Instance MEDIT1 has Gioia Tauro as a hub, while MEDIT2 groups Algeciras and Tangier in one hub as they are geographically close to each other.

The experiments in this study use the same realistic parameter values as in Msakni et al. (2020), which are gathered from a real feeder network shipping company. These parameters include the capacities of mother and daughter vessels, charter costs, sailing speed, bunker costs, cargo handling rates and costs, and port call fees.

Table 2 summarizes the properties of the instances. All instances (including the ones used for the additional tests described in Section 4.3), as well as the best solutions found, are available on the public repository https://github.com/qez008/FND-OT.

4.2. Computational results

For all the experiments, the adaptive heuristic has been run ten times. The average solution time, average objective, and best found objective over the ten runs have been reported.

Table 3 compares results from the adaptive heuristic with the best known results from Msakni et al. (2020), while a similar comparison is made on the adapted LINER-LIB instances in Table 4. Msakni et al. (2020) present two variants of their model for the FND-OT. One of these is an even more general variant of the FND-OT than we study here, which allows split pickups and deliveries, where demand to and from a given feeder port can be shared by more than one mother route visiting that port. The other variant is the same as our version of the FND-OT and does not include this splitting possibility. It should be noted that the results in Msakni et al. (2020) show that one usually obtains the same results for the two variants. In the few cases this does not happen, the difference in objective values is only marginal. Here, we compare with the best objective value for each instance obtained by Msakni et al. (2020). In the cases where both variants reach the same objective, we compare with the one with the least solution time. As explained at the beginning of Section 4, the adaptive heuristic experiments are performed on a late 2013

Table 3
Comparison of results on the class A, B and C instances.

Instance	Best results of Msakni et al. (2020)		Adaptive heuristic		
	Objective	Time (s)	Average improvement (%)	Best improvement (%)	Time (s)
A1	142 160	0.03	0.02	0.02	0.88
A2	161 868	0.03	0.02	0.02	0.91
A3	196 198	3516.70	6.89	6.89	1.83
A4	200 875	3027.50	0.02	0.02	1.07
B1	238 570	9.10	7.21	9.21	1.88
B2	215 098	9.30	12.99	13.08	1.54
B3	158 060	5.60	0.04	0.04	1.16
B4	206 669	10.10	9.44	9.44	1.57
C1	271 451	74.50	0.20	2.76	2.42
C2	294 510	126.30	−1.77	1.07	2.62
C3	344 048	239.70	0.78	0.94	4.08
C4	444 963	1656.90	0.31	1.28	3.08
Average	248 392	788.70	3.28	4.07	2.01

Table 4
Comparison of results on the adapted LINER-LIB instances.

Instance	Best results with (Msakni et al., 2020)'s model (2a and 2b)		Adaptive heuristic		
	Objective	Time (s)	Average improvement (%)	Best improvement (%)	Time (s)
BALTIC	623 167	91.20	4.49	4.56	1.01
MEDIT1	1 081 721	694.61	12.85	15.89	10.70
MEDIT2	754 840	655.00	21.50	23.22	7.60
WAF	2 817 332	59622.34	1.82	2.69	1.97
Average	1 319 265	15265.79	10.17	11.59	5.32

Table 5
Detailed results on the class A, B and C and the adapted LINER-LIB instances.

Instance	Size	Average objective	Best objective	Average time (s)	# MR	# DR	% DR
A1	10	142 130	142 130	0.88	1	0	0.0
A2	10	161 834	161 834	0.91	1	0	0.0
A3	11	182 675	182 672	1.83	1	1	9.1
A4	11	200 840	200 840	1.07	1	1	27.3
B1	15	221 375	216 591	1.88	1	1	33.3
B2	15	187 147	186 964	1.54	1	1	20.0
B3	15	158 000	158 000	1.16	1	0	0.0
B4	15	187 168	187 158	1.57	1	1	13.3
C1	20	270 920	263 948	2.42	2	1	15.0
C2	22	299 717	291 362	2.62	2	1	18.2
C3	22	341 349	340 825	4.08	1	1	18.2
C4	22	443 588	439 255	3.08	2	0	0.0
BALTIC	12	595 216	594 734	1.01	2	0	0.0
MEDIT1	33	942 678	909 865	10.70	3	4	27.3
MEDIT2	32	592 531	579 556	7.60	1	5	43.8
WAF	20	2 766 069	2 741 407	1.97	5	0	0.0

MacBook Pro, while the (Msakni et al., 2020) experiments are performed on a computer cluster. The adaptive heuristic will likely run even faster on a more powerful machine.

Tables 3 and 4 show the comparison of objective values and solutions for the class A, B, and C instances and the adapted LINER-LIB instances, respectively. It can be noted that the average solution times for the adaptive heuristic are very short compared with the algorithm by Msakni et al. (2020), which does not scale well with increased problem size. Also, we can see from the results in Table 3 that the adaptive heuristic performs 3.28% and 4.07% better on average over these instances in terms of objective value using the average and best results of the adaptive heuristic, respectively. This corresponds to significant cost savings for a feeder liner shipping company. The only exception is for instance C2 where the average objective value found is slightly worse than the previous best-known solution. Trying to push the results of the adaptive heuristic by increasing the number of iterations does not help as the heuristic converges after the specified number of iterations n. However, since the solution times for the adaptive heuristic are so low, it is possible to run the heuristic multiple times and pick the best solution. Considering this, we can see that the best solution after ten runs for C2 is better than the solution reported by Msakni et al. (2020).

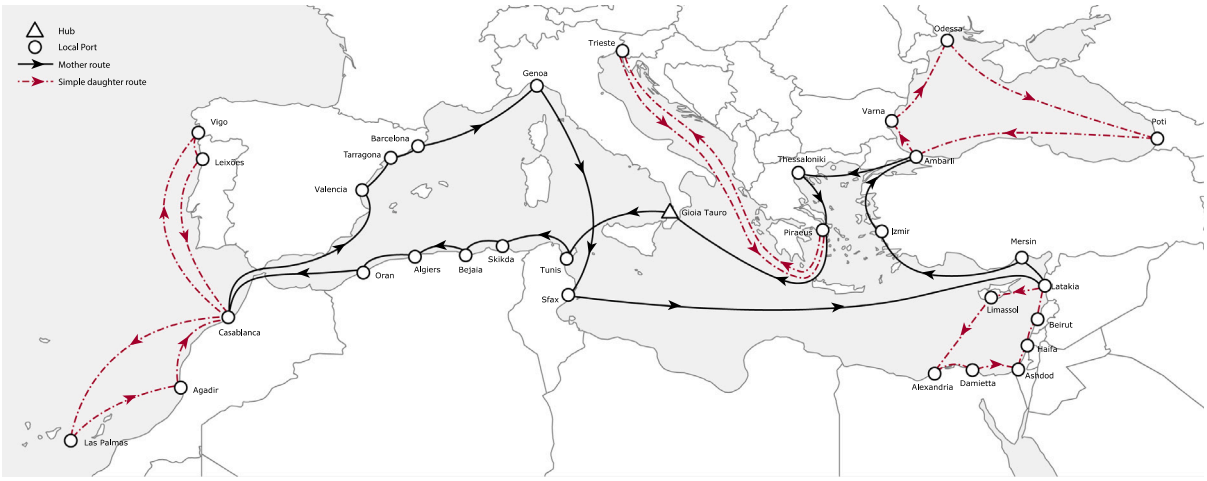

Fig. 3. Illustration of the mother and daughter routes of the best obtained solution for MEDIT2.

The results for the adapted LINER-LIB instances in Table 4 show even larger differences between the adaptive heuristic and the algorithm by Msakni et al. (2020), both in terms of solution quality and solution time. Although the algorithm by Msakni et al. (2020) was run on a computer cluster for this experiment, there is a large difference between the two tested solution approaches in terms of solution time. There are two major reasons for this difference. First, most of these instances are larger than the class A, B, and C instances, and second, the adaptive heuristic scales significantly better with problem size. Before considering the solution quality criterion, we recall that the algorithm by Msakni et al. (2020) was specially developed to utilize the geographical structure of these particular instances where the feeder ports are located more or less along a single coastline. Since most of the adapted LINER-LIB instances do not have this structure, we see that the improvements obtained for the adaptive heuristic are even larger than for the instances in Table 3. One notable exception is the WAF instance, which also has a similar geographical structure as the class A, B, and C instances, where the feeder ports are located along the coast of West Africa.

Table 5 shows more details for the results obtained by the adaptive heuristic for the class A, B and C, as well as the adapted LINER-LIB instances. The average and best columns are the average and best objectives found over ten runs. The last three columns show the structure of the best found solutions, where # MR is the number of mother routes in the solutions, # DR is the number of daughter routes, and % DR is the percentage of ports served by daughter routes.

As shown in Table 5, the possibility of allowing transshipment by introducing a hub-and-spoke structure seems to be particularly relevant for the Mediterranean instances. The ports in MEDIT1 and MEDIT2 are somewhat clustered, and it is typical for the solutions obtained for these instances that the mother routes include one or two ports from each cluster, while daughter routes serve the remaining ports within each cluster. This can be seen in Fig. 3 showing the best-found solution for the instance MEDIT2. Only one mother route is necessary to serve all ports to and from the hub port as there are only 599 containers to be transported in MEDIT2. Hence, the smallest mother vessel with a capacity of 700-TEU can carry out this shipping while the total operational costs are minimized. This mother route covers almost all ports located in the Mediterranean sea. The daughter routes serve clustered ports, e.g., in the Black Sea or in the Canary Islands region.

4.3. The effect of allowing transshipment in the feeder network

The last set of experiments studies the economic benefits of including optional transshipment, i.e., treating the problem as the FND-OT instead of as a conventional feeder network without this possibility. For this purpose, we ran the adaptive heuristic without including daughter routes — the initial solution generates only mother routes, and all heuristics involving daughter routes are disabled. The experiment was performed on the A, B and C class instances, as well as the adapted LINER-LIB instances. Each instance was run ten times, and the best objective value is reported. The obtained results for the adaptive heuristic with and without allowing transshipment are given in Table 6. The results show that for most test instances, the gain from allowing transshipment is significant. For six out of the 16 instances, the cost reduction of the feeder network is more than 10%.

A number of additional instances with different sizes and geographical structures were randomly generated to provide more insight about in which situations it can be beneficial to allow transshipment or not. Four parameters were used when generating these instances: pattern, clustered, hub location and size. Three different patterns were used for the placement of ports: (1) *circle*, (2) *field* and (3) *line*. Circle means that the ports are located more or less along a circle, field means that the ports are randomly located in a given square, while line means the ports are located along a (shore-)line. Fig. 4 provides one example of each of these. The hub ports were placed in three different locations: (a) in the middle of the other ports, (b) randomly among the other ports, or (c) isolated away from the other ports. We refer to the different hub port placements as central, local and remote, respectively. Additionally, the ports were placed clustered or not clustered. This adds up to 18 different combinations. For each combination, an

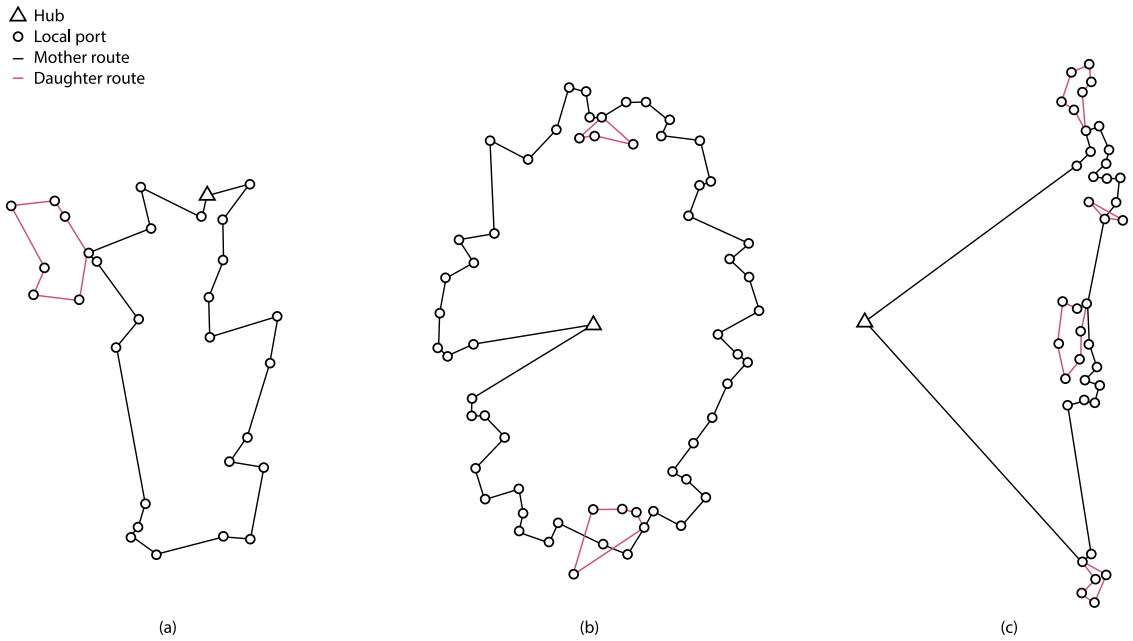

Fig. 4. Examples of instances with the obtained solutions for (a) Field-30-LocalHub, (b) Circle-60-CentralHub, (c) Line-Cluster-40-RemoteHub. The light blue solid lines illustrate mother routes, and the red dotted lines represent daughter routes. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 6
Results with and without the possibility of transshipment.

Instance	Objective w/o transshipment	Objective with optional transshipment	Gain from allowing transshipment (%)
A1	142 130	142 130	0.00
A2	161 834	161 834	0.00
A3	207 060	182 672	11.78
A4	228 380	200 840	12.06
B1	243 968	216 591	11.22
B2	211 080	186 964	11.43
B3	158 000	158 000	0.00
B4	212 210	187 158	11.81
C1	276 633	263 948	4.59
C2	296 670	291 362	1.79
C3	365 288	340 825	6.70
C4	439 930	439 255	0.15
BALTIC	594 734	594 734	0.00
MEDIT1	925 438	909 865	1.68
MEDIT2	645 730	579 556	10.25
WAF	2 741 407	2 741 407	0.00

instance with 10, 20, 30, 40, 50, 60, 100 and 150 ports was generated. This results in 144 randomly generated instances in total. The naming scheme used for these instances is the following: Pattern-Cluster(Optional)-Size-Hub, such that for example the instance "Circle-Cluster-50-CentralHub" refers to an instance where 50 ports are located in clusters along a circle with the hub located in the center of the circle.

Table 7 shows the average results for the different geographical structures of the randomly generated test instances. The "X" in the names of the instance types indicates that these are aggregated over all eight instance sizes. The results show that the gains from allowing optional transshipment are significant for all these different instance types. However, it is not clear from these results whether there are certain geographical structures which results in larger gains than others.

Table 8 presents the same results for the different sizes of the randomly generated instances, where each line in this table shows the average results for all 18 different geographical structures for each instance size. From this table, we can see a trend that the gains from allowing optional transshipment increase with increased size of the instances. This means that it becomes more beneficial to consider this possibility when designing large feeder networks than for small ones.

All these results, i.e., both for the real and the randomly generated instances, show that it is worthwhile to include optional transshipment, even though it makes the feeder network design problem harder to solve. It is also more complex to operate a

Table 7
Aggregated results differing in geographical structure.

Instance type	#MR	#DR	% DR	Gain from OT (%)
Circle-Cluster-X-LocalHub	1.88	2.50	11.50	4.98
Circle-Cluster-X-RemoteHub	1.63	1.88	8.13	6.37
Circle-Cluster-X-CentralHub	1.86	1.86	9.98	6.00
Avg. Circle-Cluster instances	1.79	2.08	9.87	5.78
Circle-X-LocalHub	2.00	0.75	2.08	4.76
Circle-X-RemoteHub	1.50	2.13	9.08	7.39
Circle-X-CentralHub	1.75	1.75	7.13	7.19
Avg. Circle instances	1.75	1.54	6.10	6.45
Line-Cluster-X-LocalHub	2.25	1.38	10.38	7.12
Line-Cluster-X-RemoteHub	1.50	2.88	21.21	6.08
Line-Cluster-X-CentralHub	2.00	1.38	9.33	7.82
Avg. Line-Cluster instances	1.92	1.88	13.64	7.01
Line-X-LocalHub	1.75	1.13	8.08	7.89
Line-X-RemoteHub	1.38	2.00	15.46	6.38
Line-X-CentralHub	1.88	1.13	8.25	10.98
Avg. Line instances	1.67	1.42	10.60	8.42
Field-Cluster-X-LocalHub	2.25	0.50	5.96	8.47
Field-Cluster-X-RemoteHub	1.50	2.88	10.79	6.81
Field-Cluster-X-CentralHub	2.25	2.38	13.98	6.80
Avg. Field-Cluster instances	2.00	1.92	10.24	7.36
Field-X-LocalHub	2.38	1.00	11.25	6.91
Field-X-RemoteHub	1.63	2.50	9.96	3.19
Field-X-CentralHub	2.38	1.25	11.29	4.53
Avg. Field instances	2.13	1.58	10.83	4.88

Table 8
Aggregated results differing by size.

Instance size (#ports)	#MR	#DR	% DR	Gain from OT (%)
10	1.00	0.78	16.11	3.79
20	1.00	1.00	14.72	3.20
30	1.33	0.50	4.45	3.86
40	1.83	1.00	8.61	5.31
50	2.22	0.39	3.44	7.25
60	1.94	1.22	8.61	9.78
100	2.67	4.72	15.94	11.28
150	2.94	4.17	9.26	8.39

hub-and-spoke network than a simple feeder network without transshipment. Another issue is that introducing transshipment will affect the transit times for the cargo. Whether transshipment will increase or decrease the transit times depend on the instance. When inspecting the solutions with and without transshipment for the C1–C4 instances and the adjusted LINER-LIB instances in Table 6, we found that the average transit time increased by 1.7% (averaged over all these instances) when allowing transshipment. However, this varies from a decrease of the average transit time of 32.6% for instance C1 to an increase of 52.0% for instance C3. If we look at the worst-case transit times (i.e., the port pairs with the longest transit time), we find that these increase by 7.2% on average over all these instances when allowing transshipment, ranging from a decrease of 38.5% (MEDIT2) to an increase of 36.6% (WAF).

5. Conclusion

We studied the Feeder Network Design Problem (FNDP), which considers the design of a minimum cost liner shipping network for the transportation of cargo (containers) between a given hub port and a set of feeder ports. In addition to determining the services to operate (i.e., the routes), the FNDP deals with deciding the fleet of vessels to deploy on these services. In contrast to most FNDPs previously studied in the literature, we include the possibility of transshipping cargo at any feeder ports if found beneficial in terms of costs. We denote the derived problem by the Feeder Network Design with Optional Transshipment (FND-OT), for which we proposed an adaptive heuristic that benefits from a novel data structure for solution representation. The proposed representation makes it easier to move from one solution to another one. It also reduces the number of calculations and, therefore, significantly shortens the computational times.

The adaptive heuristic was tested on different sets of instances. The first set includes the same instances as in Msakni et al. (2020), which are related to a real case study for a Norwegian liner shipping company transporting cargo between a hub port

in Rotterdam and 21 ports located along the Norwegian coastline. The instances in the second set have been adapted from the publicly available LINER-LIB benchmark suite (Brouer et al., 2014). We showed that the adaptive heuristic outperforms the algorithm proposed by Msakni et al. (2020) both in terms of both quality and solution time.

We also did an experiment to analyze the economic benefits of allowing transshipments in the feeder networks. In addition to the test instances mentioned above, we also generated a set of random instances (not based on real port data) with different geographical structures and sizes for this purpose. These tests showed that including the possibility of cargo transshipment in the FNDP can provide significant cost benefits and that these seem to increase with problem size.

We have in this study, based on the real case which motivated this research, assumed a maximum duration of one week for the daughter routes. This is natural in many feeder networks since the sailing times are usually rather limited, making it impractical to have longer daughter routes. However, in some applications with longer sailing times, such as MEDIT1 and MEDIT2 from the LINER-LIB instances, it could be beneficial to relax this assumption and consider also longer daughter routes. We leave this as an interesting topic for future research. Another relevant topic for future research would be to include transit time requirements for the cargo.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Mathematical model

The mathematical model for the FND-OT is based on the work of Msakni et al. (2020).

Sets

$\mathcal{K}$	set of available mother vessel types (in TEUs),
$\mathcal{R}^{\mathcal{M}}$	set of candidate mother routes,
$\mathcal{R}^{\mathcal{M}}_{p}$	set of candidate mother routes that include feeder port p,
$\mathcal{R}^{\mathcal{D}}$	set of candidate daughter routes,
$\mathcal{R}^{\mathcal{D}}_{rp}$	subset of daughter routes that can do transshipment at port p with mother route r,
$\mathcal{P}$	set of served feeder ports,
$\mathcal{P}_{r}$	set of served feeder ports by mother route r,
$\mathcal{P}^{-}_{rp}$	set of ports that precede the visit of port p in mother route r,
$\mathcal{D}^{\mathcal{R}}$	set of route pairs, where a pair indicates which daughter route can be connected to which mother route, i.e., $\mathcal{D}^{\mathcal{R}} = \{(r,d), r \in \mathcal{R}^{\mathcal{M}}, p \in \mathcal{P}_{r}, d \in \mathcal{R}^{D}_{rp}\}$,
$\mathcal{D}^{\mathcal{R}}_{p}$	subset of $\mathcal{D}^{\mathcal{R}}$ for mother and daughter routes pairs that are connected using port p.

Parameters

C^{MH}_{k}	weekly time charter cost of a mother vessel of type k,
C^{RM}_{rk}	sailing cost of mother route r using vessel type k, composed of port costs, bunker and cargo handling costs,
C^{D}_{d}	total operational cost of daughter route d, including weekly time charter cost, sailing costs, port costs, and cargo handling and transshipment costs,
U_{k}	capacity in TEUs of a mother vessel of type k,
D_{p}	demand of cargo (number of containers) to deliver to feeder port p from the hub port,
P_{p}	demand of cargo (number of containers) to pick up from feeder port p for transportation to the hub port,
L^{-}_{d}	total number of containers to deliver to feeder ports visited by daughter route d,
L^{+}_{d}	total number of containers to pick up from feeder ports visited by daughter route d,
H_{E}	cargo handling rate at the hub port (in TEU per hour),
H_{p}	cargo handling rate at feeder port p (in TEU per hour),
S_{r}	sailing time of mother route r (in hours),
N	the maximum number of daughter vessels that can do the transshipment at any feeder port.

Decision variables

x_{rk}	takes value 1 if mother route r is sailed with vessel of type k, and 0 otherwise,
y_{rk}	number of mother vessels of type k used on mother route r,
q_{r}	maximum number of containers carried on mother route r including containers of daughter routes connected to r,
t_{rk}	total time (sailing time and cargo handling time) of mother route r with vessel type k,
z_{rd}	takes value 1 if daughter route d is selected and connected to an active mother route r, and 0 otherwise.

The mathematical model of the underlying problem is as follows:

$$(F2): \ \min \sum_{r \in \mathcal{R}^{\mathcal{M}}} \sum_{k \in \mathcal{K}} \left(C^{MH}_{k} y_{rk} + C^{RM}_{rk} x_{rk} \right) + \sum_{(r,d) \in \mathcal{D}^{\mathcal{R}}} C^{D}_{d} z_{rd} \tag{A.1}$$

subject to:

$$\sum_{k\in\mathcal{K}} x_{rk} \le 1, \qquad r \in \mathcal{R}^{\mathcal{M}}, \quad \text{(A.2)}$$

$$\sum_{r\in\mathcal{R}_p^{\mathcal{M}}} \sum_{k\in\mathcal{K}} x_{rk} + \sum_{(r,d)\in D_p^{\mathcal{R}}} z_{rd} \ge 1, \qquad p \in \mathcal{P}, \quad \text{(A.3)}$$

$$N \sum_{k\in\mathcal{K}} x_{rk} \ge \sum_{(r,d)\in D^{\mathcal{R}}} z_{rd}, \qquad r \in \mathcal{R}^{\mathcal{M}}, \quad \text{(A.4)}$$

$$q_r \ge \sum_{p'\in\mathcal{P}_r} D_{p'} \sum_{k\in\mathcal{K}} x_{rk} + \sum_{(r,d)\in D^{\mathcal{R}}} L_d^- z_{rd} + \sum_{p'\in\mathcal{P}_{rp}^-} \Big((P_{p'} - D_{p'}) \sum_{k\in\mathcal{K}} x_{rk} + \sum_{(r,d)\in D_{p'}^{\mathcal{R}}} (L_d^+ - L_d^-) z_{rd} \Big), \qquad r \in \mathcal{R}^{\mathcal{M}}, p \in \mathcal{P}_r, \quad \text{(A.5)}$$

$$\sum_{k\in\mathcal{K}} U_k x_{rk} \ge q_r, \qquad r \in \mathcal{R}^{\mathcal{M}}, \quad \text{(A.6)}$$

$$t_{rk} \ge S_r x_{rk} + \sum_{p\in\mathcal{P}_r} \Big(\frac{P_p + D_p}{H_E} + \frac{P_p + D_p}{H_p} \Big) x_{rk} + \sum_{p\in\mathcal{P}_r} \sum_{(r,d)\in D_p^{\mathcal{R}}} \Big(\frac{L_d^+ + L_d^-}{H_E} + \frac{L_d^+ + L_d^-}{H_p} \Big) z_{rd}, \qquad \in \mathcal{R}^{\mathcal{M}}, k \in \mathcal{K}, \quad \text{(A.7)}$$

$$y_{rk} \ge \frac{t_{rk}}{168}, \qquad r \in \mathcal{R}^{\mathcal{M}}, k \in \mathcal{K}, \quad \text{(A.8)}$$

$$x_{rk} \in \{0,1\}, \qquad r \in \mathcal{R}^{\mathcal{M}}, k \in \mathcal{K}, \quad \text{(A.9)}$$

$$y_{rk} \in \mathbb{N}, \qquad r \in \mathcal{R}^{\mathcal{M}}, k \in \mathcal{K}, \text{(A.10)}$$

$$z_{rd} \in \{0,1\}, \qquad (r,d) \in D^{\mathcal{R}}, \text{(A.11)}$$

$$q_r \ge 0, \qquad r \in \mathcal{R}^{\mathcal{M}}, \text{(A.12)}$$

$$t_{rk} \ge 0, \qquad r \in \mathcal{R}^{\mathcal{M}}, k \in \mathcal{K}. \text{(A.13)}$$

The objective function (A.1) minimizes the weekly operational costs. The first term computes the weekly cost of selected mother routes, which is composed of the weekly time charter cost of used mother vessels and the operational costs (port costs, fuel costs, and cargo handling costs) of the mother route. The second term is related to the total weekly operational cost of the selected daughter routes, which implicitly includes the transshipment costs. The total cargo to be picked up from the daughter route and delivered to the transshipment port is known in advance. By adding the port cost, the transshipment cost is part of the total operational cost of daughter routes.

Constraints (A.2) state that a mother route can be sailed by only one vessel type. Constraints (A.3) ensure that all ports are served, either by a mother route or a daughter route. A daughter route can only be used when it can be connected to a mother route, as set by Constraints (A.4). The maximum number of containers transported by a mother route can be computed by Constraints (A.5). This computation considers the initial containers loaded at the hub port, which is composed of the containers delivered to the different feeder ports visited by the mother route (the first term of the right-hand side of (A.5)), and the containers delivered to feeder ports served by daughter routes connected to that mother route (the second term of the right-hand side of (A.5)). Through the visited ports of mother route r, $p \in \mathcal{P}_r$, the third term of the right-hand side of (A.5) computes the picked up and delivered number of containers, including those of the connected daughter routes. By doing so, the maximum number of containers q_r carried on mother route r is correctly computed. This variable is then used in Constraints (A.6) to determine the appropriate mother vessel size, and hence the vessel type. Constraints (A.7) compute the total duration of mother routes, which is composed of the pre-computed sailing time and the cargo handling time to deliver and pick up the containers to and from the visited ports. As stated by Constraints (A.8), the total duration determines the necessary number of vessels to sail that mother route. Finally, Constraints (A.9)–(A.13) define the domain of the decision variables.

Appendix B. Example of the solution representation

Fig. B.5 shows an example of a hub and spoke network composed of a mother route and two daughter routes, where the first daughter route has a simple cycle, and the second route is a butterfly route. With the solution representation proposed in Section 3.2, these mother and daughter routes are represented with the main and transshipment vectors illustrated in Fig. B.6.

Part 1 of Fig. B.6 corresponds to the sequence of ports visited by the mother route, while Part 2 represents the two daughter routes. The first daughter route has a butterfly structure, where the first loop serves only port 1, and the second loop visits ports 4 and 3. The second daughter route is simple and serves ports 7 and 8. The corresponding transshipment vector of the two daughter routes includes two cells. Each one refers to the cell index of Part 1. For the transshipment vector of Fig. B.6, the first cell of the transshipment vector contains 3, which means that the transshipment port for the first daughter route is the third cell of Part 1, which is port 2.

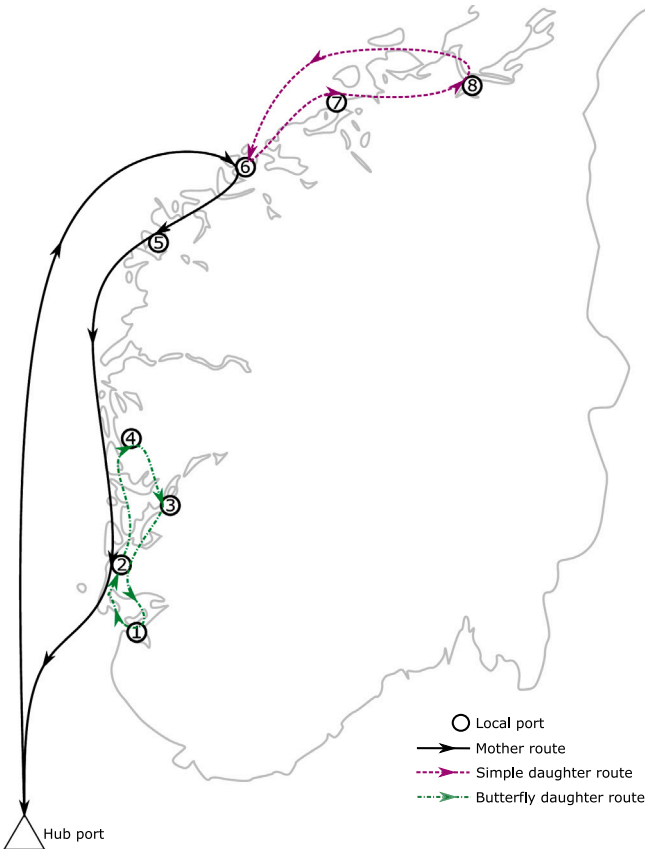

Fig. B.5. Illustration of a solution composed of a mother route, a simple (cycle) daughter route, and a butterfly daughter route.

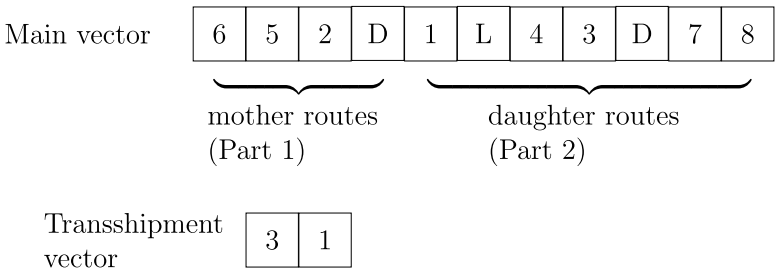

Fig. B.6. The representation of the solution of Fig. B.6 using main and transshipment vectors.

To illustrate how the heuristics described in Section 3.4 operate. Consider the heuristic "mother route to daughter route" operates, and assume that port 6 is randomly selected for the above example. It is possible to insert this port in eight positions in the two daughter routes. The ninth possibility is related to having a third daughter route visiting only port 6. Assume that $\{5, 2, D, 6, 1, L, 4, 3, 0, 7, 8\}$ is the selected option, i.e. port 6 will be part of the first daughter route. The transshipment vector requires to be updated. Since the previous index of port 6 in Part 1 is 1, any cell in the transshipment vector having a value greater or equal to 1 must be decremented by one. The resulting vector is $\{2, 0\}$. As "0" is here meaningless and does not refer to any port in the mother route, it is replaced with "1" to obtain vector $\{2, 1\}$. If, however, the re-insertion results in a new daughter route, so that the main vector is $\{5, 2, D, 1, L, 4, 3, D, 7, 8, D, 6\}$, a new cell is created in the transshipment vector and assigned to a value computed by the heuristic described in Section 3.4.5. For this case, the resulting transshipment vector is $\{2, 1, 1\}$.

Appendix C. The effect of integrating the adaptive mechanism

A recent meta-analysis by Turkeš et al. (2021) suggests that the adaptive layer of the ALNS framework has little impact on solution quality in studies that have used this approach. However, we contest this statement and present evidence from our experiments that

Table C.9
Cost increase for the results of the adaptive heuristic without adaptive mechanism. The average cost increase is over ten runs for each instance.

Class	Size	Avg. cost increase (%)	Cost increase for the best-found solution (%)
A	10–11	0.00	0.00
B	15	0.10	0.00
C	20–22	1.72	0.27
LINER-LIB	12–33	1.00	0.28
Random	20	0.30	0.00
	20	2.16	0.74
	30	3.96	5.50
	40	4.68	3.72
	50	5.46	6.81
	60	4.76	7.47
	100–150	7.33	4.73

shows the benefits of the adaptive layer in improving solution quality for the FND-OT problem. We emphasize herein that the effectiveness of the adaptive layer depends strongly on the nature of the problem being addressed and the design of the operators to solve it.

In our experiment, we compare the results of the adaptive heuristic with and without the adaptive layer. When run without the adaptive layer, we use a uniform selection method to choose heuristics, whereby all heuristics discussed in Section 3.4 have an equal chance of being chosen. To make this experiment significant, we carry out tests on a large set of instances that includes the A, B, C, and the adapted LINER-LIB instances, as well as the randomly generated instances described in Section 4.3. Table C.9 presents a comparison of the results obtained for the adaptive heuristic with and without the adaptive mechanism. The column "Avg. cost increase (%)" reports the increase in objective value (i.e., costs) over the ten runs without the adaptive layer compared to the results with. The last column "Cost increase for the best-found solutions (%)" shows the same for the best-obtained solutions over ten runs.

It is evident from the results that the adaptive mechanism improves the general performance of the proposed adaptive heuristic, and this improvement is more noticeable for larger instances. When each instance is considered separately, there is no case where a better solution could be found by a uniform random selection. This demonstrates the importance of including an adaptive selection of heuristics that rewards those heuristics that perform well in finding good solutions.

References

Agarwal, R., Ergun, O., 2008. Ship scheduling and network design for Cargo routing in liner shipping. Transp. Sci. 42 (2), 175–196.

Akbar, A., Aasen, A.K.A., Msakni, M.K., Fagerholt, K., Lindstad, E., Meisel, F., 2020. An economic analysis of introducing autonomous ships in a short-sea liner shipping network. Int. Trans. Oper. Res..

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

Ameln, M., Sand Fuglum, J., Thun, K., Andersson, H., Stålhane, M., 2021. A new formulation for the liner shipping network design problem. Int. Trans. Oper. Res. 28 (2), 638–659.

Balakrishnan, A., Karsten, C.V., 2017. Container shipping service selection and cargo routing with transshipment limits. European J. Oper. Res. 263 (2), 652–663.

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

Crama, Y., Schyns, M., 2003. Simulated annealing for complex portfolio selection problems. European J. Oper. Res. 150 (3), 546–571.

Fagerholt, K., 1999. Optimal fleet design in a ship routing problem. Int. Trans. Oper. Res. 6 (5), 453–464.

Fagerholt, K., 2004. Designing optimal routes in a liner shipping problem. Marit. Policy Manag. 31 (4), 259–268.

Hellsten, E.O., Sacramento, D., Pisinger, D., 2021. A branch-and-price algorithm for solving the single-hub feeder network design problem. European J. Oper. Res..

Holm, M.B., Medbøen, C.A.B., Fagerholt, K., Schütz, P., 2018. Shortsea liner network design with transhipments at sea: a case study from western Norway. Flexible Serv. Manuf. J..

Jin, J.G., Meng, Q., Wang, H., 2021. Feeder vessel routing and transshipment coordination at a congested hub port. Transp. Res. B 151, 1–21.

Karlaftis, M.G., Kepaptsoglou, K., Sambracos, E., 2009. Containership routing with time deadlines and simultaneous deliveries and pick-ups. Transportation Research Part E: Logistics and Transportation Review 45 (1), 210–221.

Karsten, C.V., Brouer, B.D., Pisinger, D., 2017. Competitive liner shipping network design. Comput. Oper. Res. 87, 125–136.

Koza, D.F., Desaulniers, G., Ropke, S., 2020. Integrated liner shipping network design and scheduling, vol. 54. (ISSN: 15265447) http://dx.doi.org/10.1287/TRSC.2018.0888.

Medbøen, C.A.B., Holm, M.B., Msakni, M.K., Fagerholt, K., Schütz, 2020. Combining optimization and simulation for designing a robust short-sea feeder network. Algorithms 13 (11), 22.

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), 265–280.

Moshref-Javadi, M., Hemmati, A., Winkenbach, M., 2020. A truck and drones model for last-mile delivery: A mathematical model and heuristic approach. Appl. Math. Model. 80, 290–318.

Msakni, M.K., Fagerholt, K., Meisel, F., Lindstad, E., 2020. Analyzing different designs of liner shipping feeder networks: A case study. Transp. Res. E Logist. Transport. Rev. 134, 101839.

Ropke, S., Pisinger, D., 2006. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transp. Sci. 40 (4), 455–472.

Sambracos, E., Paravantis, J., Tarantilis, C., Kiranoudis, C., 2004. Dispatching of small containers via coastal freight liners: The case of the aegean sea. European J. Oper. Res. 152 (2), 365–381.
Santini, A., Plum, C.E., Ropke, S., 2018. A branch-and-price approach to the feeder network design problem. European J. Oper. Res. 264 (2), 607–622.
Thun, K., Andersson, H., Christiansen, M., 2017. Analyzing complex service structures in liner shipping network design. Flex. Serv. Manuf. J. 29 (3–4), 535–552.
Turkeš, R., Sörensen, K., Hvattum, L.M., 2021. Meta-analysis of metaheuristics: Quantifying the effect of adaptiveness in adaptive large neighborhood search. European J. Oper. Res. 292 (2), 423–442.
Wang, S., Meng, Q., 2014. Liner shipping network design with deadlines. Comput. Oper. Res. 41, 140–149.
World Shipping Council, 2020. About the industry. http://www.worldshipping.org/about-the-industry.
Zheng, J., Meng, Q., Sun, Z., 2015. Liner hub-and-spoke shipping network design. Transp. Res. E Logist. Transp. Rev. 75, 32–48.