The Heterogeneous Fleet Vehicle Routing Problem with Draft Limits

Paolo Fadda

DICAAR, Department of Civil and Environmental Engineering and Architecture, University of Cagliari, Italy

Gianfranco Fancello

DICAAR, Department of Civil and Environmental Engineering and Architecture, University of Cagliari, Italy

Simona Mancini

Department of Operations, Energy, and Environmental Management, University of Klagenfurt, Austria

Department of Science, Innovation and Technology, University of Eastern Piedmont, Alessandria, Italy

Patrizia Serra

DICAAR, Department of Civil and Environmental Engineering and Architecture, University of Cagliari, Italy

Abstract

Over the past two decades, international maritime transport has been characterized by the advent of ever larger ships. This phenomenon is known as naval gigantism. If, on the one hand, naval gigantism allows to reduce transport costs by exploiting the economies of scale achievable by large ships, on the other hand, it implies a series of operational issues. Indeed, due to their large draft, such giant vessels are not allowed to enter small ports when fully or near-fully loaded, and in some cases, they cannot enter such small ports at all. In fact, their draft can strongly vary depending on the load on board. This implies restrictions for vessels in accessing ports, which impact not only at the strategical level on the fleet sizing problem, but also at the tactical/operational level, on the sequence of port visits among each route. In fact, given a set of ports that a ship has to visit, determining the optimal sequence of visits becomes a very challenging issue, as the sequence that gives the shortest travel distance (i.e., the smallest travel cost) may prove infeasible due to draft limit restrictions for accessing ports. Furthermore, the same sequence of ports, which may be infeasible for a large ship, may become viable if operated by a smaller ship. On the other hand, due to the economy of scale, travel costs per load unit are generally much lower for large ships than for small ones. Therefore, the draft restrictions also affect the fleet sizing problem. In this paper, we introduce the Heterogeneous Fleet Vehicle Routing Problem with Draft Limits (HF-VRP-DL). We propose a mixed integer programming formulation and several valid inequalities to strengthen it. Since the mathematical model is able to handle only small-sized instances, to address larger instances we propose a Large Neighborhood Search matheuristic (LNS) and an Iterated Local Search matheuristic (ILS). Computational tests carried out show excellent performances of the proposed approach. Further analysis is provided on the impact of the instance layout on the computation time required to solve the problem to optimality.

Keywords: Routing, Heterogeneous Fleet, Draft Limits, Large Neighborhood Search, Matheuristic, Maritime Transportation

1. Introduction and motivation

In the context of maritime logistics, routing and scheduling decisions are affected by a combination of requirements, both organizational (e.g., contractual obligations, mandatory points of passage, demand requests, etc.) and phisical (e.g., drafts, marine currents, sea and wind conditions, etc.). Particularly, the role of drafts in ship routing, which has traditionally characterized maritime transportation, is becoming more and more important due to the tendency to build ships of ever larger dimensions ([3]). As a consequence of naval gigantism, the draft issue, which has been traditionally related to tankers, bulk ships and GAS carriers ([5],[6]), now involves container vessels as well ([12]). Some data can help to better understand the scale of the phenomenon.

During the last decades, the size of container ships has grown at a faster rate than all other ship types. In the 1996-2015 period, the average increase in the size of container ships (in dead weight tons) was 90%, 55% for bulk carriers and 21% for oil tankers ([17]). Modern container ships have a length of 400 meters and are now able to carry more than 23,000 TEUs ([21]), designs for over 25,000 TEU ships are in the pipeline ([10]). However, it is worth pointing out that although container ships are now the largest ships in the world with regards to length, they still have smaller drafts than tankers and bulk carriers.

In the maritime context, the draft of a ship is defined as the distance between the surface of the water and the lowest point of the ship's hull. Higher drafts require deeper maritime access and port waters, which can be a challenge for many ports. The needed draft of a ship may depend on several factors that determine the minimum depth of water the ship can navigate safely. Among the most influential factors, we can mention the depth of the water, the tide at a particular time (of particular importance for some ports), and the load on board. The latter varies based on the sequence of visited ports in the route and may, thus, impose restrictions on the visiting sequence, other than restrictions about the ports that can be visited within the same route. If the draft of a ship, when approaching a port, is greater than the draft limit allowed by the port, the ship can not enter it. The same ship will be able to access that port only after having unloaded part of its cargo in other ports and having reduced its draft to a limit that allows safe access. This also implies that some ports with relatively small water depths can still be able to accommodate large ships provided that these hold a position within the ports sequence visit that allows for low loads.

This paper deals with the design of an optimal maritime network able to efficiently connect a potential supply port to a set of receiving ports while ensuring minimum transport costs and satisfying operational constraints related to draft limits. Each receiving port is characterized by a demand that must be fulfilled in the planning horizon. Ships operating on the network can be selected among a heterogeneous fleet characterized by different capacities. The problem faced consists into selecting how many ships of each type to involve, and providing the routing plan for each ship, in compliance with the draft-based constraints. The aim is to minimize total network costs comprehensive of both port entering costs and travel costs (different for each ship category). This problem can be modeled as a Heterogeneous Fleet Vehicle Routing Problem with Draft Limits (HF-VRP-DL). To solve the HF-VRP-DL, we propose a Mixed Integer Programming (MIP) model and several Valid Inequalities (VI) to strengthen the formulation and reduce computational times. Such model is effective only on small-sized instances. To address larger instances we have designed a Large Neighborhood Search (LNS) matheuristic and an Iterated Local Search (ILS) matheuristic. Computational results show the efficiency and effectiveness of the proposed approaches. The paper is organized as follows. Section 2 reports a review of the literature on related topics. Section 3 provides a formal description of the problem addressed. The proposed mathematical model and the Valid Inequalities to strengthen it are presented in Section 4. Section 5 presents the solution approaches developed. Section 6 is devoted to the presentation of the experimental campaign while Section 7 reports the computational results. Finally, the conclusions and future developments are discussed in Section 8.

2. Literature Review

Despite its relevance in maritime applications, the issue of draft limits in ports, and the consequent limit imposed on the maximum load of a vessel to enter a port, has received limited attention in the literature. This issue has been explicitly addressed in a routing problem for the first time by [20] who introduce the Traveling Salesman Problem with Draft Limits (TSP-DL), considering only one vehicle. They propose a mathematical model and valid inequalities to strengthen the formulation. The proposed approach performs well on small-sized instances with a small percentage of nodes affected by draft limits, but computational times explode when this percentage grows. In [4], new formulations and a branch and cut approach are proposed for the same problem, sensibly improving the state of the art. [22] propose a Variable Neighbourhood Search (VNS) heuristic approach able to address larger instances. [12] propose a Pick-up and Delivery version of the problem in which each node may request either a pick-up or a delivery service. A similar problem is studied in [2] considering visit time windows at customers. The problem is motivated by a real-world application in chemical shipping. [9] propose an extension of the TSP-DL where it is not mandatory to visit all nodes, each node is associated with a profit and the goal is to maximize the collected profit. The authors propose a Mixed Integer Linear Programming formulation (MILP) able to address only small-sized instances and a VNS heuristic to solve larger instances.

The literature on routing problems with draft limits involving more than one vessel is even more limited. In [7], authors study a very complex maritime routing problem with draft limits and penalties for time windows violation, and vessel speed considered as a decision variable. They propose a mathematical model too complex even to address small-sized instances and use a Particular Swarm Optimization (PSO) to heuristically address the problem.

Therefore, no optimal solutions are provided for this problem. The problem treated in this paper is novel because it considers an heterogeneous fleet of vessels among which to choose, each one characterized by different loading capacity, fixed and unitary costs, and draft.

Although, in [5] and [6], the authors deal with fleet sizing for a maritime routing problem with draft limits, they consider a fixed draft for each category of vessels, which does not vary with the percentage of load on the ship. Therefore, they model the problem as a standard VRP with Heterogeneous fleet in which some ports can be visited only by a subset of the available category of ships. This only impacts on the fleet sizing but does not affect the visiting sequence within a route. Our paper extends these two documents by considering a more realistic feature whereby the ship's draft varies according to the load on board, and this affects the selection of ships but also the sequence of visits within the routes. Indeed, given a subset of nodes assigned to the same route, the sequence providing the shortest travel distance (and therefore the lowest cost) could turn out to be infeasible due to draft limits restrictions. On the other hand, the available papers explicitly dealing with load-dependent draft variations, address single vehicle problems without possibility of choosing the size of the vehicle. Therefore, to the best of our knowledge, this paper is the first attempt to integrate the fleet sizing problem with draft limit constraints and load-dependent draft variations, filling, hence, a gap in the current literature.

3. Problem Description

This study introduces a Heterogeneous Fleet Vehicle Routing Problem with Draft Limits (HF-VRP-DL) in which a set of ports, I, must be served starting from a depot. Each port i is characterized by a demand to be served, Q_i, and a draft limit representing the maximum draft of a ship allowed to enter the port. Draft limits can prevent ships to enter some ports when they are fully loaded, thus imposing constraints on the sequence of ports visited. The fleet is composed by a set of heterogeneous ships, S, each characterized in terms of load capacity, Q_s, fixed costs to enter each port i, r_{is}, unitary sailing costs, c_s, and empty and full load draft values. The actual draft of a ship in a given time is calculated as the draft of the empty ship plus a linear function of the load on board at the time. Based on these data, we are able to compute, for each ship s and port i, the maximum allowed load, for s, to safely access port i, L_{is}. Sailing times among each pair of ports, t_{ij}, and between the depot and each port, are known. The objective of the problem addressed is to minimize the total network cost given by the sum of fixed costs to access ports and the sailing costs.

4. Mathematical Model

In the following, we provide the mathematical formulation of the newly introduced problem.

Table 1: List of parameters and variables involved in the model

Sets	
$I = [1, Imax]$	set of ports
$I0 = [0, Imax]$	set of ports including the depot
$S = [1, Smax]$	set of ships
Parameters	
Q_s	ship capacity (tons)
q_i	port demand (tons)
L_{is}	maximum loading for ship s to access port i (tons)
t_{ij}	sailing time between port i and port j (h)
c_s	hourly sailing cost for ship s (€/h)
r_{is}	access cost for ship s entering port i (€)
Variables	
X_{ijs}	binary variables taking value 1 if arc ij is traversed by ship s
Y_{is}	binary variables taking value 1 if port i is served by ship s
l_{is}	loading of ship s entering port i
u_i	position of port i in the sequence of visited ports
p_s	total load for ship s

$$\min \sum_{i \in I0} \sum_{j \in I0} \sum_{s \in S} c_s t_{ij} X_{ijs} + \sum_{i \in I} \sum_{s \in S} r_{is} Y_{is} \tag{1}$$

$$\sum_{s \in S} Y_{is} = 1 \quad \forall i \in I \tag{2}$$

$$\sum_{i \in I} q_i Y_{is} \leq Q_s \quad \forall s \in S \tag{3}$$

$$\sum_{i \in I0} X_{ijs} = Y_{js} \quad \forall j \in I \;\; \forall s \in S \tag{4}$$

$$\sum_{i \in I0} X_{ijs} = \sum_{i \in I0} X_{jis} \quad \forall j \in I \;\; \forall s \in S \tag{5}$$

$$X_{0js} \leq \sum_{j \in I} Y_{js} \quad \forall s \in S \tag{6}$$

$$X_{0js} \geq \sum_{j \in I} Y_{js}/Imax \quad \forall s \in S \tag{7}$$

$$u_j \geq u_i + 1 - Imax(1 - \sum_{s \in S} X_{ijs}) \quad \forall i \in I \;\; \forall j \in I0 \tag{8}$$

$$l_{js} \geq l_{is} - q_i - Q_s(1 - X_{ijs}) \quad \forall i \in I \;\; \forall j \in I0 \;\; \forall s \in S \tag{9}$$

$$l_{is} \leq L_{is} \quad \forall i \in I \;\; \forall s \in S \tag{10}$$

$$l_{0s} = \sum_{i \in I} q_i Y_{is} \quad \forall s \in S \tag{11}$$

$$X_{ijs} \in \{0, 1\} \quad \forall i \in I0 \;\; \forall j \in I0 \;\; \forall s \in S \tag{12}$$

$$Y_{is} \in \{0, 1\} \quad \forall i \in I \;\; \forall s \in S \tag{13}$$

$$u_i \in N^+ \quad \forall i \in I \tag{14}$$

The objective function is reported in (1). Constraints (2) imply that each port is assigned to a ship. Constraints (3) ensure that the maximum load capacity of a ship is never exceeded. If a port is assigned to a ship s it must be visited by that ship exactly once, as stated by Constraints (4) and (5). Each ship must enter and exit the depot once if at least one port has been assigned to it (Constraints (6) and (7)). The position of a port j in the sequence of visits in the route is tracked by Constraints (8), while Constraints (9) track the load of the ship when entering port j. This load must always be lower than the maximum allowed load, as implied by Constraints (10). The load of a ship exiting the depot is equal to the sum of the demands of the ports assigned to it (Constraints (11)). Finally, Constraints (12)-(14) specify variables domain.

4.1. Valid Inequalities

In order to strengthen the mathematical formulation presented above, we propose the following valid inequalities (VI).

$$VI1 : X_{0js} \leq 1 - \frac{1}{TOT_q}(\sum_{i \in I} q_i Y_{is} - L_{js}) \quad \forall j \in I \;\; \forall s \in S \tag{15}$$

$$VI2 : X_{ijs} = 0 \quad \forall i \in I \;\; \forall j \in I \;\; \forall s \in S | q_i + q_j > L_{is} \tag{16}$$

$$VI3 : p_s - (u_i - 1)q_{big} \leq L_{is} + Q_{is}(1 - Y_{is}) \quad \forall i \in I \;\; \forall s \in S \tag{17}$$

where p_s and q_{big} represent the total load of a ship and the largest demand, respectively, and can be computed as:

$$p_s = \sum_{i \in I} q_i Y_{is} \quad \forall s \in S \tag{18}$$

$$q_{big} = \max_{i \in I} q_i \tag{19}$$

$$VI4 : u_i \leq I^* \quad \forall i \in I \tag{20}$$

The first set of valid inequalities, $VI1$, implies that, for each port j, if the total load of the ship s, to which it has been assigned, is greater than the maximum allowed load for s to enter j, then j cannot be the first port visited in the route. The second set, $VI2$, states that, for each ship s and each pair of ports i and j, if the sum of their demand, q_i and q_j, is greater than the maximum allowed load for s to enter i, then j cannot be served *immediately* after i by ship s. Please note that this condition could have been further strengthened, imposing that j cannot be served at anytime after i by ship s but this would have implied the generation of an exponential number of constraints, that would have made the model intractable from a computational point of view, therefore we decided to keep the initial version of $VI2$. The third set, $VI3$, allows to identify the earliest position a port i can occupy in the visiting sequence, without violating draft limit constraints, given the ship s to which it has been assigned and the set of ports assigned to it. Finally, $VI4$ states that the latest position a port can assume in the visiting sequence is equal to the maximum number of ports that can be assigned simultaneously to the same ship, I^*. This value can be easily computed ordering the ports by demand in a non-decreasing order, and counting how many ports, scrolling the ordered list, can be assigned to the largest ship in the fleet, before to fill up its capacity.

5. Solution Approaches

The mathematical model proposed for the HF-VRP-DL is able to efficiently handle only small-sized instances. To overcome this issue and address larger instances, we propose an LNS matheuristic and an ILS matheuristic. Matheuristics (MH) have become very popular in the last decade. They cover different types of methods base on the common idea of hybridizing mathematical programming and (meta)heuristics ([8, 1]). Within this large family, two main groups can be identified. The first group contains sequential methods composed by two main phases. In general, in the first phase a metaheuristic is used to exploit the solution space, while, in the second one, a mathematical model is used to refine the obtained solution. This type of matheuristics has been broadly applied to VRPs. A common procedure is to generate promising solutions with a randomized heuristic and to pass all obtained routes to a set partitioning formulation which selects the best combination of them ([14, 18]). A slightly different framework has been used to address the Electric VRPs in which a metaheuristic is used to build up the routing plan, while a mathematical model is used to insert recharging stops along the route ([19, 11]).

The second group of MH contains approaches in which the mathematical model is directly used to explore the search neighborhood, thus allowing to exhaustively explore very large neighborhoods in relatively small computational times. Large Neighborhood Search based MH have been successfully applied to several routing problems ([13, 15, 16]). The MH proposed in this paper belongs to this category of solution approaches. Differently from [13, 15, 16], which operate on the customers assignment variables (to a vehicle, a depot, a period, etc.) fixing only the assignment of most of the customers and letting the model choose the assignment of the remaining customers and building the routes, our approach directly works on arc variables, destroy part of the current solution by removing some arcs. In this way, at each iteration, we do not need to completely reoptimize the routing of the whole solution, but only of those parts of the solution affected by the perturbation. Clearly, the size of the explored neighborhood is still large but smaller respect to the approaches proposed in the literature, though this allows to explore them much faster. This kind of approach is specifically useful when dealing with routing problems in which the number of customers that can be served in a route is large, such as in maritime applications where the capacity of the ships is very large respect to the port demand. On this kind of problem, neighborhoods based on assignment variables would require too much time to be exhaustively explored, while arcs variables based ones would be more suitable.

5.1. An LNS Matheuristic for the HF-VRP-DL

We have designed an LNS Matheuristic in which we use a randomized operator to partially destroy the solution, and we exploit the mathematical model, presented in Section 4, to optimally rebuild a feasible solution starting from the partial solution obtained and then inserting the removed nodes in the most convenient position. A pure random arc removal operator would provide certainly some improvements at the beginning of the search, especially when starting from a poor quality solution, but once we have reached good quality solutions, it would frequently happen to remain trapped into local minima. If, instead, we use a randomized arcs removal procedure aiming at removing arcs which are spatially near to each others, the probability to obtain an improving solution sensibly increases. The procedure we propose works as follows. We run the mathematical model with a short time-limit (15 s) and we keep the best solution obtained so far, Ω^0, as the current best solution, Ω^{best}. Then, at each iteration, we randomly select m ports and add them to the set of ports to be removed from the solution, I^R. Then, for each $i \in I^R$, we remove from the solution (destroying entering and exiting arcs) all the ports within a certain radius, ρ_i, from i. The value of ρ_i varies among ports and is computed as $\rho_i = \alpha\nu_i$ where α is a parameter and ν_i is the distance between i and its nearest port. Using a variable radius we avoid situations in which the circle of radius ρ does not contain any ports (for peripheral ports) or contains too many ports (for most central ones). The partially destroyed solution obtained after the application of the destroy operator, Ω^R, is then passed to the model, fixing to 1 the variables corresponding to the arcs selected in Ω^R. Then, this over-constrained version of the model is run to optimally complete the routes, inserting in the best position the ports that have been previously removed from the solution (i.e., ports belonging to $I - I^R$). If the newly obtained solution, Ω^{new}, is better than Ω^{best}, it is kept as current best, otherwise it is discarded. This procedure is repeated until the maximum number of iterations, $ITERMAX$ is reached, or after $MAXNOIMPROVE$ iterations without any improvement.

A pseudocode of the above described LNS matheuristic is reported in Algorithm 1.

The main advantage of this procedure respect to classical metaheuristics is that a very large neighborhood can be efficiently explored at each iteration, allowing to quickly move toward strongly better solutions, as reported in the example in Fig. 1.

5.2. An ILS Matheuristic for the HF-VRP-DL

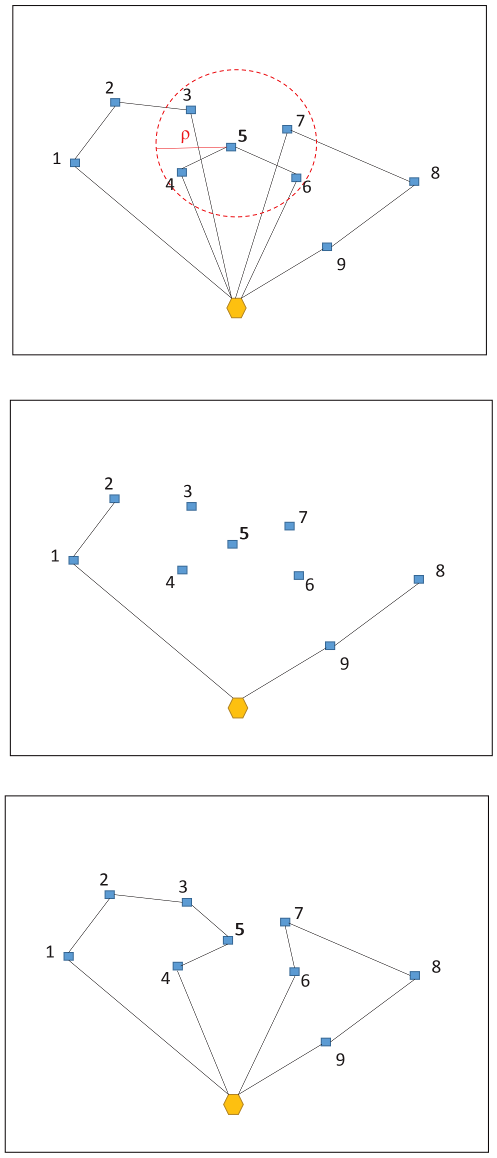
The ILS we propose shares the common basic idea of the LNS, but uses a different arcs removal operator. It is composed of two phases: a deterministic local search (LS) and a randomized diversification phase (DIV). Initially, we run the mathematical model with a short timelimit (15 s) and we keep the best solution obtained so far, Ω^0, as the current best solution Ω^{best}. Then, we start applying the LS and every time we remain trapped into a local minimum, we apply DIV and restart the LS. The procedure terminates after $ITERMAX$ iterations or after $MAXNOIMPROVE$ iterations without improvements. The LS works as follows: named λ_i the length of the longest between the arc entering and the arc exiting port i, we define a score for each port i, $\sigma_i = \frac{\lambda_i}{\nu_i}$. Then, we remove from the current solution the m ports with the highest scores, and let the model optimally reconstruct the solution starting from the partial destroyed one. The newly obtained solution is kept as current best. The procedure terminates as soon as no further improvements are reached, i.e., when the reconstructed solution coincides with the current best. The diversification phase, DIV, is performed randomly choosing m ports to be removed from the solution together with their *neighborhoods* within a radius ρ_i, where $\rho_i = \alpha\nu_i$. The best feasible solution obtained starting by the partial destroy solution generated by DIV is kept as a current solution even if it is worse than the current best, as usual in diversification procedures.

A pseudocode of the above described ILS matheuristic is reported in Algorithm 1, while procedure LS and DIV are enlightened in Algorithms 3 and 4.

Algorithm 1 A pseudocode for the LNS matheuristic

```
1: $iter \leftarrow 1$
2: $NOIMPROVE \leftarrow 0$
3: $\Omega^{best} \leftarrow \Omega^0$
4: while $iter \leq ITERMAX$ and $NOIMPROVE \leq MAXNOIMPROVE$ do
5:     $I^R = \emptyset$
6:     $\Omega^R \leftarrow \Omega^{best}$
7:     for $pert \in 1..m$ do
8:         randomly select a port $j$ in $I - I^R$
9:         $I^R \leftarrow I^R + \{j\}$
10:    end for
11:    for $i \in I^R, j \in I - I^R$, do
12:        if $t_{ij} \leq \rho_i$ then
13:            $I^R \leftarrow I^R + \{j\}$
14:        end if
15:    end for
16:    for $i \in I^R$ do
17:        remove from $\Omega^R$ the arcs entering and exiting from $i$
18:    end for
19:    fix all the arc variables active in $\Omega^R$ equal to 1
20:    run the over-constrained version of the model to optimize the remaining
       variables obtaining a solution $\Omega^{new}$
21:    if $\Omega^{new}$ is better than $\Omega^{best}$ then
22:        $\Omega^{best} \leftarrow \Omega^{new}$
23:        $NOIMPROVE \leftarrow 0$
24:    else
25:        $NOIMPROVE \leftarrow NOIMPROVE + 1$
26:    end if
27: end while
```

Figure 1: An iteration of LNS Matheuristic



Algorithm 2 A pseudocode for the ILS matheuristic

```
 1: $iter \leftarrow 1$
 2: $NOIMPROVE \leftarrow 0$
 3: $\Omega^{best} \leftarrow \Omega^0$
 4: while $iter \leq ITERMAX$ and $NOIMPROVE \leq MAXNOIMPROVE$ do
 5:     if $NOIMPROVE = 0$ then
 6:         $\Omega^{new} \leftarrow LS(\Omega^{best})$
 7:         if $\Omega^{new}$ is better than $\Omega^{best}$ then
 8:             $\Omega^{best} \leftarrow \Omega^{new}$
 9:             $NOIMPROVE \leftarrow 0$
10:         else
11:             $NOIMPROVE \leftarrow NOIMPROVE + 1$
12:         end if
13:     else
14:         $\Omega^{new} \leftarrow DIV(\Omega^{best})$
15:         if $\Omega^{new}$ is better than $\Omega^{best}$ then
16:             $NOIMPROVE \leftarrow 0$
17:         else
18:             $NOIMPROVE \leftarrow NOIMPROVE + 1$
19:         end if
20:     end if
21: end while
```

Algorithm 3 A pseudocode for the local search procedure LS

```
Require: $\Omega^{best}$
 1: $I^R = \emptyset$
 2: $\Omega^R \leftarrow \Omega^{best}$
 3: for $i \in I$ do
 4:     $\lambda_i \leftarrow max(max_j(t_{ij}), max_j(t_{ji}))$
 5: end for
 6: Define $I^W$ the set containing the $m$ ports with the largest value of $lambda_i$
 7: $I^R \leftarrow I^R + I^W$
 8: for $i \in I^R, j \in I - I^R$, do
 9:     if $t_{ij} \leq \rho_i$ then
10:         $I^R \leftarrow I^R + \{j\}$
11:     end if
12: end for
13: for $i \in I^R$ do
14:     remove from $\Omega^R$ the arcs entering and exiting from $i$
15: end for
16: fix all the arc variables active in $\Omega^R$ equal to 1
17: run the over-constrained version of the model to optimize the remaining
    variables obtaining a solution $\Omega^{new}$
18: return $\Omega^{new}$
```

Algorithm 4 A pseudocode for the diversification procedure DIV

```
Require: $\Omega^{best}$
1:  $I^R = \emptyset$
2:  $\Omega^R \leftarrow \Omega^{best}$
3:  for $pert \in 1..m$ do
4:      randomly select a port $j$ in $I - I^R$
5:      $I^R \leftarrow I^R + \{j\}$
6:  end for
7:  for $i \in I^R, j \in I - I^R$, do
8:      if $t_{ij} \leq \rho_i$ then
9:          $I^R \leftarrow I^R + \{j\}$
10:     end if
11: end for
12: for $i \in I^R$ do
13:     remove from $\Omega^R$ the arcs entering and exiting from $i$
14: end for
15: fix all the arc variables active in $\Omega^R$ equal to 1
16: run the over-constrained version of the model to optimize the remaining
    variables obtaining a solution $\Omega^{new}$
17: return $\Omega^{new}$
```

6. Experimental Campaign

We conducted an experimental campaign on a set of realistic instances developed ad-hoc to investigate whether the layout of the instances could influence the performance of the proposed approaches. The impact of the layout of the instances on the computational performance was analyzed in relation to the following features:

- number of ports involved in the network
- percentage of ports affect by draft-based restrictions
- tightness of total ship capacity with respect to the total demand of the ports.

We have analyzed small instances with 15 ports and large instances with 25 ports. While in some contexts, like last-mile delivery, it is very common to face instances with hundreds or even thousands of customers, in maritime routing problems the number of nodes involved is much smaller (dozens), since the number of ports in a specific geographical area is somehow limited. In particular, in maritime routing problems involving very large ships, the number of nodes is even smaller, since only large ports, with a deep seabed can be entered with such ships, due to draft limits. It is worth noting that maritime VRPs that are not affected by draft limits can be treated as standard VRP, since no limitations are imposed on the maximum load allowed on the ship to enter a port, and are out of the scope of this paper. Maritime VRPs in which access restrictions hold depending on the on-board load, as the one studied in this paper, generally consider a very limited number of nodes. In this application, we assume that small networks (15 ports) are served by 3 ships (1 small, 1 medium, 1 large), medium networks (25 ports) by 6 ships (2 small, 2 medium, 2 large) and large networks (50 ports) by 15 ships (5 small, 5 medium, 5 large) For what concerns the percentage of ports affected by draft restrictions (DR), we generated two scenarios: LOW_{DR} where only 30% of ports are affected by draft limits and $HIGH_{DR}$ with 70% of ports with draft limit restrictions. We made a similar choice for what concerns capacity tightness (CT). In the LOW_{CT} case the total demand corresponds to 30% of the total ships capacity, while, in the $HIGH_{CT}$ case, it corresponds to 70%. Resuming, we have created four set of instances with 15 ports, one for each combination of the parameters DR and CT.

- Set 1: $LOW_{DR} - LOW_{CT}$
- Set 2: $LOW_{DR} - HIGH_{CT}$
- Set 3: $HIGH_{DR} - LOW_{CT}$
- Set 4: $HIGH_{DR} - HIGH_{CT}$

Each set contains 10 instances with randomly generated coordinates for the ports. In order to evaluate the contribution of the valid inequalities from a computational point of view, we have tested the model without VIs and with different combinations of VIs. More in detail, the following combinations have been tested:

- without VIs
- considering each VI separately
- combining the VI that has been shown to work best with each of the other VIs
- considering the four VIs together

In a second phase, we have detected the combination of layout parameters that makes the problem more difficult to solve and we have generated one set of 10 instances with 25 nodes sharing this layout. On this fifth set, we compared the performance of the model with the best combination of valid inequalities with the LNS and the ILS.

7. Computational Results

7.1. Small-sized instances

All the results on small-sized instances are reported in Tables 2-5, which are organized as follows. In the first column is reported the instance ID, then columns 2-4 report instance layout characteristics: number of ports, capacity tightness and percentage of ports affected by draft limits. The objective of the optimal solution (OF) is indicated in column 5, while the remaining columns report the computational time (s) requested by each version of the model. The best results in terms of computational times are enhanced in bold. We compare the mathematical model without VIs with four versions of the model, each with only one VI. As can be evinced from the results, all

Table 2: Results on small-sized instances with LOW capacity tightness and LOW percentage of ports affected by draft limits ($LOW_{DR} - LOW_{CT}$)

					COMPUTATIONAL TIME								
INSTANCE	PORTS	TOTq/TOTQ	% draft limits	OF	NO VI	1	2	3	4	1+4	2+4	3+4	1+2+3+4
1	15	0.3	30	328.1	9.80	10.73	9.73	6.74	3.73	4.74	3.67	**2.98**	3.00
2	15	0.3	30	280.5	17.07	**2.84**	16.94	17.86	5.63	6.21	5.65	5.63	4.34
3	15	0.3	30	341.4	6.27	8.46	6.10	7.71	3.18	3.73	**3.14**	12.41	17.19
4	15	0.3	30	275	2.93	4.18	3.00	4.38	**0.83**	1.42	**0.83**	1.69	1.42
5	15	0.3	30	357.5	2.37	2.78	2.36	1.83	2.82	1.17	2.89	1.08	**1.00**
6	15	0.3	30	329.6	6.58	3.81	6.79	4.67	1.83	**0.98**	1.86	4.56	5.98
7	15	0.3	30	320.7	9.02	10.22	9.19	8.46	6.24	5.87	6.19	**3.06**	3.12
8	15	0.3	30	310.6	11.64	9.92	11.89	7.97	6.90	**5.10**	6.85	5.43	8.81
9	15	0.3	30	367.6	9.53	4.24	9.55	2.68	6.62	**0.84**	6.58	3.81	2.68
10	15	0.3	30	272.7	1.64	2.82	1.58	1.45	**1.05**	1.14	**1.05**	1.26	1.25
				318.37	7.6847	6.0001	7.7115	6.3751	3.8816	**3.1201**	3.8687	4.1898	4.8782

Table 3: Results on small-sized instances with LOW capacity tightness and HIGH percentage of ports affected by draft limits ($LOW_{DR} - HIGH_{CT}$)

					COMPUTATIONAL TIME								
INSTANCE	PORTS	TOTq/TOTQ	% draft limits	OF	NO VI	1	2	3	4	1+4	2+4	3+4	1+2+3+4
1	15	0.3	70	328.8	10.65	5.93	9.73	9.87	4.21	6.02	4.21	2.81	**2.75**
2	15	0.3	70	280.5	27.11	19.90	16.94	11.60	2.24	2.39	**2.15**	7.06	11.23
3	15	0.3	70	341.9	11.32	7.34	6.10	8.21	4.69	4.83	4.69	7.08	**3.60**
4	15	0.3	70	275	2.99	3.98	3.00	7.22	2.44	3.48	2.23	**0.89**	1.45
5	15	0.3	70	357.5	1.91	1.87	2.36	4.50	1.06	1.32	**1.05**	1.99	2.82
6	15	0.3	70	329.6	7.96	9.38	6.79	9.26	12.83	18.88	12.71	**4.88**	7.54
7	15	0.3	70	320.7	7.61	6.80	9.19	8.80	4.77	7.47	4.57	**3.35**	9.72
8	15	0.3	70	310.6	10.87	8.99	11.89	14.18	7.00	7.30	6.99	**5.44**	6.78
9	15	0.3	70	367.6	2.90	5.91	9.55	4.10	5.67	1.08	5.60	1.08	**0.93**
10	15	0.3	70	272.7	6.65	6.23	**1.58**	3.96	4.91	1.62	4.98	3.54	2.12
				318.49	8.9985	7.6322	7.7115	8.1705	4.9828	5.4401	4.9183	**3.8131**	4.8947

the VIs are useful to reduce computational times but the best performance is obtained with VI 4. For this reason, we have tested the combination of VI 4 with each one of the other VIs. The best VIs configuration turned out to be the combination of VIs 3 and 4. Finally, we have tested all the VIs together. This set-up performs better than the model without VIs but it is outperformed by the combination 3-4, which results to be the global best set-up.

Regarding the impact of the layout on the computational times, we can deduct from the results that the first three sets ($LOW_{DR} - LOW_{CT}$, $LOW_{DR} - HIGH_{CT}$ and $HIGH_{DR} - LOW_{CT}$) show a similar behavior and all the instances belonging to them can be solved in very few seconds. The $HIGH_{DR} - HIGH_{CT}$ combination, instead, turned out to be more difficult to be solved, requiring, on average 96 seconds to be solved to the optimality.

Table 4: Results on small-sized instances with HIGH capacity tightness and LOW percentage of ports affected by draft limits ($HIGH_{DR} - LOW_{CT}$)

					COMPUTATIONAL TIME								
INSTANCE	PORTS	TOTq/TOTQ	% draft limits	OF	NO VI	1	2	3	4	1+4	2+4	3+4	1+2+3+4
1	15	0.7	30	400.8	1.50	1.41	1.48	3.61	1.38	**0.89**	1.35	2.98	1.72
2	15	0.7	30	385.4	33.72	17.32	31.88	15.45	22.51	20.17	20.38	**5.63**	9.98
3	15	0.7	30	437.9	5.84	8.18	5.71	24.78	8.18	9.08	7.90	12.41	**4.82**
4	15	0.7	30	383.5	34.89	25.26	33.08	28.69	16.70	7.89	16.35	**1.69**	15.72
5	15	0.7	30	472	7.92	5.88	7.31	8.09	2.51	2.93	2.47	**1.08**	3.34
6	15	0.7	30	433.3	5.63	3.81	5.15	6.39	5.33	**2.84**	4.93	4.56	5.71
7	15	0.7	30	451.9	52.91	31.81	48.32	49.71	30.01	29.53	29.79	**3.06**	27.63
8	15	0.7	30	421.6	14.40	20.56	14.21	27.34	11.86	9.35	11.90	**5.43**	10.18
9	15	0.7	30	569.5	72.32	48.03	68.97	69.77	34.68	67.71	33.18	**3.81**	44.60
10	15	0.7	30	373.8	17.85	13.93	17.37	19.26	10.44	9.81	10.42	**1.26**	11.51
				432.97	24.6974	17.618	23.3477	25.3077	14.3575	16.0199	13.8657	**4.1898**	13.5205

Table 5: Results on small-sized instances with HIGH capacity tightness and HIGH percentage of ports affected by draft limits ($HIGH_{DR} - HIGH_{CT}$)

					COMPUTATIONAL TIME								
INSTANCE	PORTS	TOTq/TOTQ	% draft limits	OF	NO VI	1	2	3	4	1+4	2+4	3+4	1+2+3+4
1	15	0.7	70	458.1	29.867	16.526	28.769	38.949	10.132	**5.385**	10.051	12.729	9.279
2	15	0.7	70	523.4	545.952	509.138	521.994	522.317	636.593	595.288	605.364	**301.013**	512.538
3	15	0.7	70	545.3	183.2	205.241	**170.693**	235.057	180.362	175.471	171.197	224.129	230.837
4	15	0.7	70	432.4	97.237	116.912	95.457	94.755	60.708	63.009	**57.558**	92.514	95.376
5	15	0.7	70	579	86.379	78.861	82.992	93.02	57.511	53.148	54.845	**45.784**	99.356
6	15	0.7	70	504.6	80.064	79.874	79.471	118.052	87.021	89.709	84.763	42.67	**35.907**
7	15	0.7	70	479.2	95.124	123.372	92.235	152.319	79.106	78.861	75.19	83.194	**61.708**
8	15	0.7	70	474	63.645	65.677	61.16	59.047	18.526	27.29	**17.665**	62.551	57.45
9	15	0.7	70	584.6	105.442	128.975	96.178	114.139	62.318	**53.418**	62.147	90.584	115.462
10	15	0.7	70	363.7	11.99	10.841	11.167	5.948	**3.995**	4.331	4.086	6.911	7.723
				494.43	129.89	133.5417	124.0116	143.3603	119.6272	114.591	114.2866	**96.2079**	122.5636

7.2. Medium-sized instances

Starting from the results obtained on small-sized instances, we have decided to analyze, for large instances, only the combination of parameters $HIGH_{DR} - HIGH_{CT}$ because it turned out to be the most challenging one. The goal of this analysis was to compare the performance of the best set-up of the mathematical model (with VIs 3 and 4) to the LNS and the ILS, both in terms of efficiency and effectiveness.

7.2.1. Algorithms' parameters tuning

Since both proposed algorithms are parametric, we performed preliminary tests to tune the parameters involved. The algorithm depends on two parameters: 1) m, which is the number of nodes involved by the destroy operator and 2) α, which is the proximity threshold, used to identify the nodes to involve in the diversification phase in the ILS. We first analyze how the variation of a single parameter influences the performance of the algorithms, and then study the combined effect of the two parameters. In Figure 2, we report a graphic of the average objective function found by the ILS depending on the parameter m. We have considered 3 values of m: $\{3, 5 \text{ and } 7\}$. All the tests have been carried out considering a fixed value of $\alpha = 1.5$. In Figure 3 we report average computational times variation depending on m. As can be evinced from the graphic, m=3 provides a too small neighborhood and the algorithm immediately remains trapped in local minima. Since the neighborhood size is very small, computational times are very short, but the quality of the solutions provided is very low. The value m=5 provides the best results, outperforming m=7, both from efficiency and from effectiveness. In fact, when the size of the neighborhood is very large, the mathematical model requires longer times to exhaustively explore them, which justify the larger computational times reported in Figure 3. Moreover, a larger perturbation in the diversification phase, can yield to poor quality solutions. We do not explicitly reports tuning parameters results for the LNS since the graphics was almost completely overlapping with those related to ILS, both for what concerns solution quality and computational times, showing exactly the same trend. Finally, from this analysis, we can conclude that the best choice is to fix m=5.

We also analyzed the influence of the proximity parameter, α, testing the values $\{1, 1.25, 1.5, 2\}$. We recall to the reader that α is used to determine the set of nodes which, basing on the nodes removed from the solution by the destroy operator, are candidate to be reallocated by the model. The larger

Figure 2: Impact of m on the solutions' quality for a fixed value of $\alpha = 1.5$.

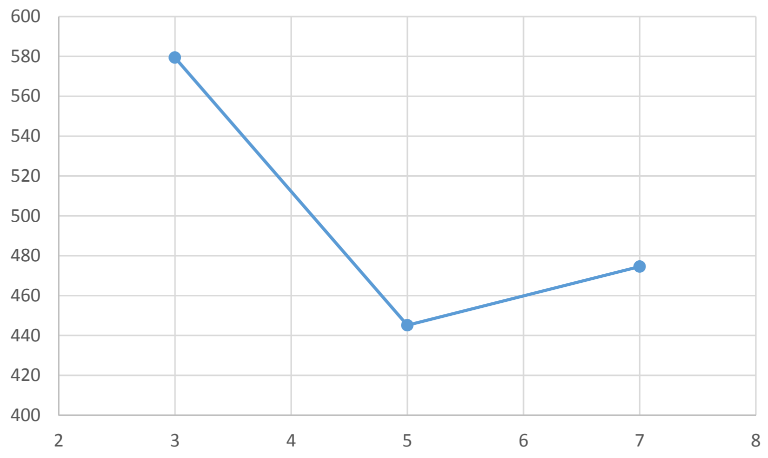


Figure 3: Impact of m on the computational time for a fixed value of $\alpha = 5$.

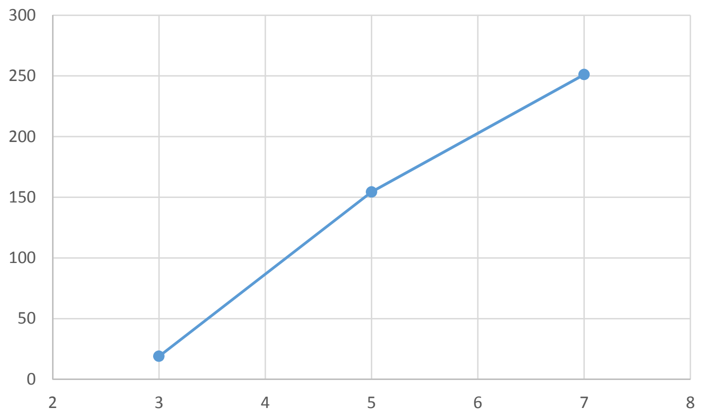
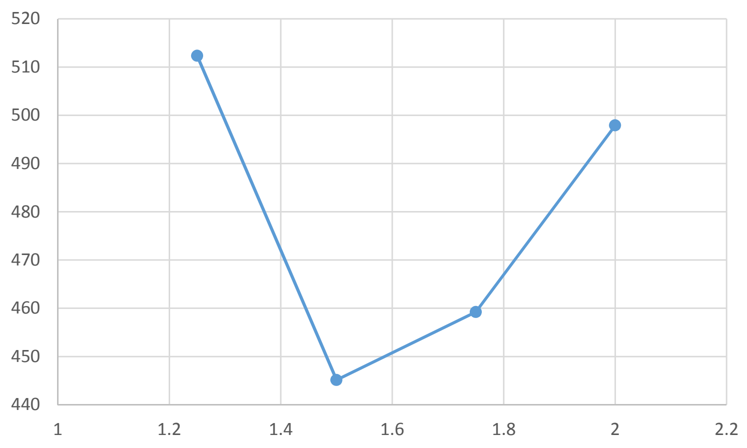


Figure 4: Impact of α on the solutions' quality for a fixed value of $m = 5$.



the value of α the larger the number of candidate nodes. The value of m has been fixed equal to 5, which has been proved to be the best value in the above described analysis. As shown in Figure 4, the best configuration is to set $\alpha = 1.5$. Smaller values will include too few nodes as candidate and yield the algorithm to quickly converge to local minima, as proved by the very short computational times (see Figure 5). Larger values of α involve a large number of candidate solutions, strongly slowing the algorithm.

A third analysis has been conducted to determine the effect of the combination of m and α on the algorithm's performance. Results are reported in Table 6. The best global results are obtained fixing $m = 5$ and $\alpha = 1.5$. In Figure 6, we plot for each value of m the variation of performance depending on α. We can easily note that the trend does not depend on the value of α. It is worth notice that the parameter that mostly influences the performance is m, while the effect of the variation of α is much more limited. Whichever the value of α, the best configuration for m is 5, by a large amount. Similarly, whichever the value of m, the best configuration for α is 1.5. This proves that the two parameters' impacts are uncorrelated and that there is no mutual interaction. Last, but not least, it is worth notice that all the configurations with $m = 5$ strongly outperform the others, while this does not hold for α. This proves that the most influencing parameter is m.

Figure 5: Impact of α on the computational time for a fixed value of $m = 5$.

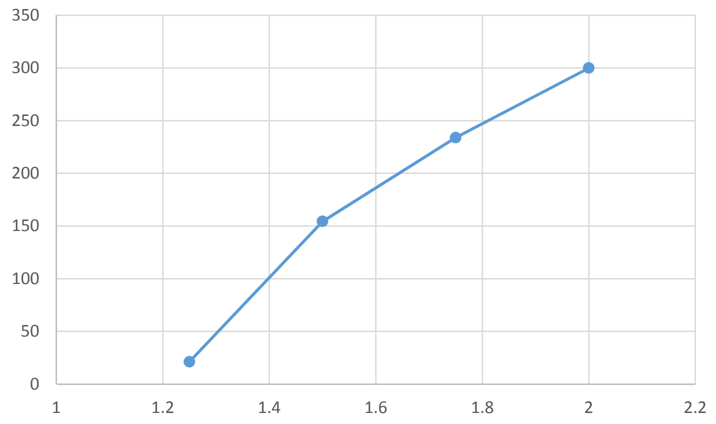
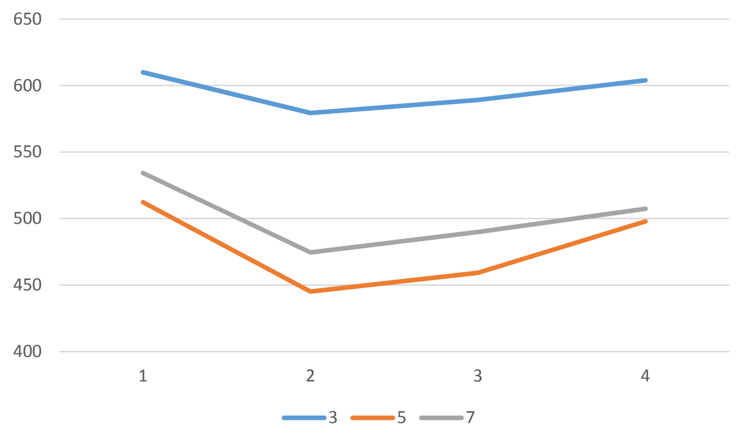


Table 6: Average results for different combinations of m and α

		m		
		3	5	7
α	1.25	609.89	512.33	534.22
	1.5	579.36	445.13	474.55
	1.75	589.16	459.23	489.91
	2	603.94	497.89	507.33

Figure 6: Impact of α on the computational time for different values of m.



7.2.2. Comparison with the model

Results are resumed in Table 7, which is organized as follows. We report the objective function (OF) of the best solution found by the mathematical model within a time limit of 3600 seconds, the best lower bound (LB) obtained so far, the time frame (TF) on which the best solution has been found and the total computational time elapsed (TIME). For both LNS and ILS, we report the gap with the best solution obtained by the model (BEST GAP), the average gap (AVG GAP) obtained over 10 runs, the average time on which the best solution was found (AVGTF) and the average computational time elapsed (AVG TIME). The value of the parameters used in ILS and LNS are $m = 5$ and $\alpha = 1.5$. The initial solution is computed running the model for 15 seconds and keeping the best solution obtained so far. The mathematical model was unable to optimally solve any of the ten medium-sized instances within the set time limit; an average residual optimality gap of 11% emerged. Both LNS and ILS matheuristics showed very good performances, obtaining the same solution of the mathematical model in nine out of ten cases, and further improving the solution obtained by the model in the remaining one. Both matheuristics turned out to be robust with a very small gap (0.97% for LNS, 0.19% for ILS) between the average objective function they obtained over 10 runs and the best solution of the mathematical model. From the point of view of computational efficiency, both matheuristics are

Table 7: Comparison of model, LNS and ILS on medium-sized instances

	MODEL +3 4				LNS				ILS			
	OF	LB	TF	TIME	BEST GAP	AVG GAP	AVG TF	AVG TIME	BEST GAP	AVG GAP	AVG TF	AVG TIME
I1	455	431.289	375.788	3600	0.00%	0.92%	138.03	171.85	0.00%	0.23%	76.89	160.89
I2	435.4	396.444	1447.36	3600	0.00%	1.54%	102.78	160.19	0.00%	0.09%	88.97	161.89
I3	427.5	398.983	1088.9	3600	0.00%	1.07%	119.22	169.36	0.00%	0.89%	110.92	159.33
I4	389	313.762	2594.18	3600	0.00%	1.89%	128.32	183.64	0.00%	0.23%	88.67	180.91
I5	444.6	417.267	1060.88	3600	0.00%	0.61%	121.74	161.62	0.00%	0.00%	101.43	154.33
I6	483.4	439.5	1653.43	3600	0.00%	0.85%	72.62	112.89	0.00%	0.00%	70.11	120.23
I7	409.9	354.847	198.696	3600	0.00%	0.00%	49.67	114.8	0.00%	0.00%	39.78	117.78
I8	464.9	414.255	1396.61	3600	0.00%	0.23%	83.17	135.77	0.00%	0.11%	77.12	144.44
I9	447.8	408.557	1436.58	3600	0.00%	1.10%	103.08	182.92	0.00%	0.16%	89.9	190.87
I10	485.4	426.428	1789.56	3600	-1.34%	1.46%	94.68	150.84	-1.34%	0.17%	90.8	155.55
	444.29	400.1332	1304.198	3600	-0.13%	0.97%	101.331	154.388	-0.13%	0.19%	83.459	154.622

more efficient than the mathematical model. The latter took on average 1304 seconds to find the best solution, while LNS and ILS achieved it on average in 101 and 83 seconds, respectively (see Table 7). In conclusion, both methods performed very well and showed strong robustness. Although ILS slightly outperforms LNS, there is no statistically significant difference in their performance, which means that both can be considered valid and high-performing tools to solve the problem.

Since the size of the perturbation addressed in both LNS and ILS is a parameter of the algorithm and does not vary depending on the size of the instances, the computational times required by both approaches do not strongly increase with the size of the instance. Therefore, even very large instances can be addressed by these methods. Conversely, the mathematical model becomes intractable, from a computational point of view, as the size of the instances increases.

7.3. *Large-sized instances*

This section analyzes the results on a set of 10 large-sized instances, with 50 ports and 15 ships (5 small, 5 medium and 5 large).

In Table 8, we compare the results obtained by the model with those obtained by LNS and ILS. Since both LNS and ILS contain a random component, we performed 10 runs for each instance and plotted the best and the average value. The model is not able to solve any instance to optimality within the imposed time-limit of 3600 seconds, and the reached optimality gap is always very high (in some cases higher than 50%). Both LNS and ILS show very good performances, providing solutions with an average cost

Table 8: Comparison of model, LNS and ILS on large-sized instances

	MODEL +3 4				LNS				ILS			
	OF	LB	TF	TIME	BEST GAP	AVG GAP	AVG TF	AVG TIME	BEST GAP	AVG GAP	AVG TF	AVG TIME
I1	972.5	545.839	1097.85	3600	-35.70%	-34.87%	694.056	768.174	-35.70%	-35.49%	306.418	438.464
I2	1092.8	521.827	998.9	3600	-43.14%	-42.76%	279.779	387.079	-42.96%	-42.63%	57.183	107.561
I3	960.7	553.589	988.89	3600	-29.34%	-29.19%	216.071	253.732	-29.11%	-28.90%	382.291	540.294
I4	842.2	506.996	972.37	3600	-20.08%	-19.36%	288.343	404.379	-20.08%	-19.48%	451.003	511.953
I5	1137.6	591.865	1005.12	3600	-25.76%	-25.18%	224.257	250.341	-25.71%	-25.54%	541.729	681.074
I6	1005.7	561.827	1015.21	3600	-12.86%	-12.61%	67.219	132.301	-12.95%	-12.61%	92.708	162.224
I7	933.7	534.435	1989.09	3600	-28.71%	-28.54%	267.37	386.056	-28.71%	-28.57%	290.946	327.932
I8	1050.9	573.624	2989.9	3600	-34.10%	-33.40%	163.81	256.337	-34.10%	-33.77%	309.112	407.554
I9	891.5	606.729	3006.76	3600	-26.69%	-26.58%	364.743	498.057	-26.34%	-25.78%	136.869	231.131
I10	927.5	556.19	3330.17	3600	-25.56%	-25.20%	254.297	346.152	-25.56%	-24.96%	483.064	556.239
	981.51	555.29	1739.43	3600.00	-28.19%	-27.77%	281.99	368.26	-28.1%	-27.77%	305.13	396.44

that is 28% lower than the model. Both matheuristics are very robust and strongly independent by the random seed. In fact, average results, over 10 runs, only slightly differ from best results. Computational times are very similar for the two approaches, since both of them are capable to find the best solution in around 300 seconds on average.

We can conclude that, while the mathematical model becomes intractable as the size of the instances increases, both matheuristics scale very well both in terms of efficiency and effectiveness. Since there is no statistically significant difference between the performances of LNS and ILS, both can be seen as powerful tools for solving the problem.

8. Conclusions and Future Developments

This paper introduced the Heterogeneous Fleet Vehicle Routing Problem with Draft Limits (HF-VRP-DL). The problem treated presents several novelties with respect to the existing literature: (i) it extends the maritime routing problem with heterogeneous fleet by considering draft limit restrictions to access ports and (ii) it extends the Traveling Salesman Problem with Draft Limits by introducing an heterogeneous fleet. To address the HF-VRP-DL, this paper proposed a mixed-integer programming model and several Valid Inequalities to strengthen it. Since this formulation allowed to efficiently handle only small-sized instances (up to 15 nodes), a Large Neighborhood Search (LNS) matheuristic and an Iterated Local Search (ILS) matheuristic were designed to address larger instances (25 nodes). Computational tests, carried out on realistic instances, showed the utility of the valid inequalities, and excellent performances of both LNS and ILS, either in terms of efficiency and of effectiveness. The proposed heuristic approaches, belonging to the family of model-based matheuristics, are innovative because, differently from previous works in the literature, they directly work with routing variables based neighborhood, instead of assignment variables based neighborhoods. From a methodological point of view, the proposed approaches can be adapted to solve several Vehicle Routing Problems, especially those in which routes contain a large number of customers, for which the visit sequencing problem becomes more difficult, and for which assignment variables based matheuristic generally fail. The performed computational tests also investigated the impact of the layout of the instances on the performance of the different solution approaches. The instances characterized by a high percentage of ports affected by draft restrictions, and high tightness of the total ship capacity with respect to the total demand of the ports, turned to be the most challenging. However, both matheuristics have been proved to be effective and robust also for the solution of these instances. Further extensions of the research could include the analysis of a multi-depot version of the problem, and the adaption of the developed approaches to different Vehicle Routing Problems, in particular those in which the routes contain a large number of nodes, for which the sequencing of the visits becomes more difficult, and assignment variables based matheuristic generally fail.

References

[1] Archetti, C., Speranza, M.G., 2014. A survey on matheuristics for routing problems. EURO Journal on Computational Optimization 2, 223–246.

[2] Arnesen, M., Gjestvanga, M., Wanga, X., Fagerholt, K., Thuna, K., Rakkeb, J.G., 2017. A traveling salesman problem with pickups and deliveries, time windows and draft limits: Case study from chemical shipping. Computers & Operations Research 77, 20–31.

[3] Baccelli, O., Senn, L., 2014. Italian Maritime Economy. New routes for growth. Giannini Editore. chapter Naval gigantism and new alliances in the container sector. Current status and future outlook. pp. 149–164.

[4] Battarra, M., Alves Pessoa, A., Subramanian, A., Uchoa, E., 2014. Exact algorithms for the traveling salesman problem with draft limits. European Journal of Operational Research 235, 115–128.

[5] Bittante, A., Jokinen, R., Pettersson, F., Saxen, H., 2015. Optimization of lng supply chain. Computer aided chemical engineering 37, 779–784.

[6] Bittante, A., Pettersson, F., Saxen, H., 2018. Optimization of a small-scale lng supply chain. Energy 148, 78–89.

[7] De, A., Choudhary, A., Tiwari, M., 2019. Multiobjective approach for sustainable ship routing and scheduling with draft restrictions. IEEE Transactions on Engineering Management 66, 35–51.

[8] Fischetti, M., Fischetti, M., 2016. Matheuristics, in: Martí, R., Panos, P., Resende, M.G. (Eds.), Handbook of Heuristics. Springer International Publishing, Cham, pp. 1–33.

[9] Gelareh, S., Gendron, B., Hanafi, Monemi, R.N., Todsijevic, R., 2019. The selective traveling salesman problem with draft limits. Journal of Heuristics Https://doi.org/10.1007/s10732-019-09406-z.

[10] Haralambides, H., 2019. Gigantism in container shipping, ports and global logistics: A time-lapse into the future. Maritime Economics & Logistics 21.

[11] Keskin, M., Catay, B., 2018. A matheuristic method for the electric vehicle routing problem with time windows and fast chargers. Computers & Operations Research 100, 172–188.

[12] Malaguti, E., Martello, S., Santini, A., 2018. The traveling salesman problem with pickups, deliveries, and draft limits. Omega 74, 50–58.

[13] Mancini, S., 2016. A real-life multi depot multi period vehicle routing problem with a heterogeneous fleet: Formulation and adaptive large neighborhood search based matheuristic. Transportation Research Part C: Emerging Technologies 70, 100–112.

[14] Mancini, S., 2017a. A combined multistart random constructive heuristic and set partitioning based formulation for the vehicle routing problem with time dependent travel times. Computers & Operations Research 88, 290–296.

[15] Mancini, S., 2017b. The hybrid vehicle routing problem. Transportation Research Part C: Emerging Technologies 78, 1–12.

[16] Mancini, S., Stecca, G., 2018. A large neighborhood search based matheuristic for the tourist cruises itinerary planning. Computers & Industrial Engineering 122, 140–148.

[17] Merk, O., Busquet, B., Aronieti, R., 2015. The impact of mega-ships, in: Int Transp Forum OECD.

[18] Montoya, A., Gueret, C., Mendoza, J.E., Villegas, J.G., 2016. A multi-space sampling heuristic for the green vehicle routing problem. Transportation Research Part C: Emerging Technologies 70, 113–128.

[19] Montoya, A., Gueret, C., Mendoza, J.E., Villegas, J.G., 2017. The electric vehicle routing problem with nonlinear charging function. Transportation Research Part B: Methodological 103, 87–110.

[20] Rakke, J., Christiansen, M., Fagerholt, K., G., L., 2012. The traveling salesman problem with draft limits. Computers & Operations Research 39, 2161–2167.

[21] Serra, P., Fancello, G., 2020. Towards the imos ghg goals: A critical overview of the perspectives and challenges of the main options for decarbonizing international shipping. Sustainability 12, 3220.

[22] Todosijevic, M., Mladenovic, N. abd Hanafi, S., Gendron, B., 2017. A general variable neighborhood search variants for the travelling salesman problem with draft limits. Optimization Letters 11, 1047–1056.