Benchmark suite for industrial and tramp ship routing and scheduling problems

Ahmad Hemmati[1], Lars Magnus Hvattum[1], Inge Norstad[2], Kjetil Fagerholt[1]

[1] Department of Industrial Economics and Technology, Norwegian University of Science and Technology, Alfred Getz veg 3, NO-7491 Trondheim, Norway
{ahmad.hemmati}{lars.m.hvattum}{kjetil.fagerholt}@iot.ntnu.no

[2] Norwegian Marine Technology Research Institute (MARINTEK), POB 4125 Valentinlyst, NO-7450 Trondheim, Norway
{inge.norstad}@marintek.sintef.no

Abstract

This paper considers the cargo ship routing and scheduling problem that arises in industrial and tramp shipping and presents a wide range of benchmark instances that have been created to represent realistic planning problems for various shipping segments. Initial results for the benchmark instances are provided both through exact and heuristic methods. Optimal solutions to smaller problem instances are provided by a commercial mixed-integer programming solver, and high-quality solutions to larger problem instances are provided by a state-of-the-art adaptive large neighborhood search heuristic. The provided benchmark instances and an instance generator intends to stimulate future development of better solution algorithms for this important planning problem, and to provide a basis for modelling and solving different real-life extensions that go beyond what is included in the basic version of the cargo ship routing and scheduling problem.

Keywords: maritime transportation; instance generation; pick-up and delivery; adaptive large neighborhood search

1 Introduction

International trade heavily depends on maritime transportation with more than nine billion tons of goods carried at sea annually by a world fleet with capacity of more than 1.6 billion deadweight tons (UNCTAD 2013). It is common to distinguish between the following three modes of operation in maritime transportation (Lawrence 1972): *liner*, *industrial* and *tramp* shipping. Liner shipping resembles bus line operations since the ships follow published schedules and itineraries. Container shipping belongs to this mode. An industrial operator owns the cargoes and controls the ships, trying to minimize the cost of transporting its own cargoes, similar to a private fleet. Tramp shipping has similarities with taxi services, where the ships follow the available cargoes, usually with a mix of mandatory contract cargoes and optional spot ones.

In this paper we focus on industrial and tramp shipping, which have several similarities when it comes to operational characteristics. Oil and bulk products, which constitute about 60% of the total amount of goods carried at sea, are almost entirely transported within these two modes of operation. Al-Khayyal and Hwang (2007) distinguish between *cargo routing* and *inventory routing*, which both may appear in industrial and tramp shipping. We focus in this paper on cargo routing, which involves the problem of routing a fleet of ships to service a number of cargoes that are given as input to the planning process, in contrast to inventory routing where the cargoes are determined through the planning process itself.

A shipping company operating in the industrial or tramp market has a set of mandatory contract cargoes that it is committed to carry. Typically in a tramp shipping company, these mandatory contract cargoes come from long-term agreements between the shipping company and the cargo owners, while an industrial operator transports its own cargoes. Each cargo in the given planning period must be picked up at its port of loading, transported and then delivered to its corresponding discharging port. There are time windows given, within which the loading of the cargoes must start, and there may also exist time windows for discharging. The shipping company controls a heterogeneous fleet of ships to transport the cargoes, each ship with a given initial position, possibly at sea, and a time for when it becomes available for new transport assignments. There may be compatibility constraints between ships and cargoes, for example because some ships cannot enter a cargo's loading or discharging port due to draft limitations. Whereas for major bulk commodities a cargo is usually a full shipload, the ship capacity may accommodate several cargoes simultaneously for minor bulk commodities and chemicals where the shipments are smaller. In the full load case, the problem simplifies compared to the case where a ship may carry several cargoes simultaneously. The challenge for the industrial operator is to construct routes and schedules such as to minimize costs. Tramp shipping companies will also try to increase their revenue by transporting optional spot cargoes, so their objective, in addition to constructing routes and schedules, is to select the optional spot cargoes so as to maximize profits.

The planners of an industrial or tramp shipping company basically must solve, on a daily basis, a ship routing and scheduling problem which in structure has many similarities to the *multi-vehicle pickup and delivery problem with time windows* (m-PDPTW) described by Desrosiers et al. (1995). However, there are also a number of differences in the operational environment between cargo routing problems in industrial and tramp shipping and the m-PDPTW that arises for example in land transport, see the discussions by Ronen (1983) and Christiansen et al. (2004). The most important ones are perhaps that the fleet is heterogeneous, that ships operate around the clock and do not start from a common depot, that not all ships and cargoes are compatible, and that the travel distances are based on a different topology.

Christiansen et al. (2013) present a recent general survey on ship routing and scheduling, and show that the research within this topic has blossomed and that its volume has more than doubled compared with the previous decade. A large share of this research is on cargo routing problems in industrial and tramp shipping, see for example (Brønmo et al. 2007; Hwang et al. 2008; Korsvik et al. 2010; Maliappi et al. 2011; Jetlund and Karimi 2004; Lin and Liu 2011). A recent book chapter on industrial and tramp ship routing by Christiansen and Fagerholt (2014) presents mathematical models for several problem variants.

Despite the recent increase in research on ship routing and scheduling, this transportation mode still receives little attention compared to its importance. One of the reasons for this lack of attention is probably the barriers for researchers to engage in such problems due to lack of standardized data. Much of the previous research has been performed on real life problems that cannot be shared due to confidentiality issues, or on ad hoc, randomly generated instances that do not sufficiently capture the particular features of maritime transportation. There is therefore a need to develop standardized benchmark problem instances. Such benchmark instancess have been developed for various vehicle routing problems, such as Solomon's instances (Solomon 1987) for the vehicle routing problem with time windows, and have undoubtedly contributed to an increased research interest which has led to many important contributions.

Recently, benchmark problem instances have been developed also for specific maritime routing problems. Brouer et al. (2013) present an integer programming model and a benchmark suite for liner shipping network design. Their benchmark suite, which is made to reflect the business structure of a global liner shipping network, is presented and discussed in relation to industry standards, business rules, and mathematical programming. The benchmark instances are developed based on real-life data from a global liner-shipping company and supplemented by data from several industry and public stakeholders. Similarly, Papageorgiou et al. (2014) present a library of test instances for deep-sea single-product maritime inventory routing problems. They also propose a mixed-integer programming model for solving the problem, which is used to provide best known results for the benchmark instances.

Our contributions in this paper are to 1) present and make available a wide range of realistic benchmark instances for industrial and tramp ship cargo routing and scheduling problems, 2) make available an instance generator to generate additional instances, 3) present a state-of-the-art adaptive large neighborhood search (ALNS) implementation to solve the problem, and 4) to present a computational study illustrating the performance of the ALNS heuristic and a commercial MIP solver directly solving a mathematical formulation of the cargo routing problem.

The main goal of this work is that other researchers will use our benchmark instances for developing better solution algorithms for this important planning problem. Even more importantly, we hope the paper will stimulate researchers to model and solve various real life extensions to this problem that go beyond what is included in our benchmark instances, such as speed optimization (Norstad et al. 2011; Psaraftis and Kontovas 2013), inclusion of bunkering decisions (Bebes and Savin 2009; Vilhelmsen et al. 2014) and environmental regulations (Kontovas and Psaraftis 2011).

The rest of this paper is organized as follows. A formulation of the ship routing and scheduling problem considered in this paper is given in Section 2. The ALNS heuristic proposed for solving the problem is presented in Section 3. The generation of the benchmark instances is described in Section 4. Section 5 presents the computational study, while concluding remarks are given in Section 6.

2 Problem formulation

We consider a shipping company, operating either within tramp shipping or industrial shipping, which operates a heterogeneous fleet of ships. At a given point in time we consider a static and deterministic planning problem, consisting of determining how the fleet of ships should service a set of known cargoes. The ships may have different capacities, service speeds and cost structures, and due to previous assignments, they become available for new assignments at different times and different locations. In addition, some ships may be prevented from carrying certain cargoes due to incompatibilities

If a cargo is assigned to a ship, the ship must load the cargo in its corresponding loading port and later unload the cargo in its unloading port. All loading and unloading operations must be performed within a time interval that is specific to that operation for a given cargo. Ship capacities must not be violated. The shipping company may choose to use spot charter to transport a given cargo, instead of carrying it using one of its own ships.

Below is a mathematical formulation of the industrial and tramp routing and scheduling problem, where the set of ships is denoted by V, and the capacity of ship $v \in V$ is K_v. If we let i denote a cargo, there is a node i corresponding to the loading port and a node $i + n$ corresponding to the unloading port, with n being the number of cargoes. The set of loading nodes is denoted by N^P, and the set of unloading nodes is denoted by N^D. The set of nodes that can be visited by ship v is N_v, and this set includes an origin node $o(v)$ and an artificial destination node $d(v)$. The set of arcs that ship v can traverse is A_v. We also introduce the shorthand $N_v^P = N^P \cap N_v$ for loading nodes that can be visited by ship v, and $N_v^D = N^D \cap N_v$ for unloading nodes that can be visited by ship v.

Each node has a time window $[\underline{T}_i, \overline{T}_i]$. The cost of sailing from i to j using ship v is C_{ijv}, and the associated travel time is T_{ijv}. The time at which service begins at node i using ship v is t_{iv}, and l_{iv} is the total load onboard after completing service at node i using ship v. The variables x_{ijv} are binary flow variables, indicating whether ship v moves directly from node i to node j. Binary variables y_i indicate whether cargo i is transported by the available vessel fleet. If the cargo is not transported with the available vessel fleet, a cost C_i^S is incurred. For industrial shipping, this corresponds to the cost of using spot charter to transport the cargo. For tramp shipping the cost represents either the loss of revenue from not being able to carry an optional cargo or the cost of using spot charter to transport a mandatory contract cargo. The mathematical formulation of the problem becomes as follows.

$$\min \sum_{v \in V} \sum_{(i,j) \in A_v} C_{ijv} x_{ijv} + \sum_{i \in N^P} C_i^S y_i \tag{1}$$

subject to

$$\sum_{v\in V}\sum_{j\in N_v} x_{ijv} + y_i = 1, \qquad i \in N^P, \tag{2}$$

$$\sum_{j\in N_v} x_{o(v)jv} = 1, \qquad v \in V, \tag{3}$$

$$\sum_{j\in N_v} x_{ijv} - \sum_{j\in N_v} x_{jiv} = 0, \qquad v \in V, i \in N_v \backslash \{o(v), d(v)\}, \tag{4}$$

$$\sum_{j\in N_v} x_{jd(v)v} = 1, \qquad v \in V, \tag{5}$$

$$l_{iv} + Q_j - l_{jv} \leq K_v(1 - x_{ijv}), \qquad v \in V, j \in N_v^P, (i,j) \in A_v, \tag{6}$$

$$l_{iv} - Q_j - l_{(n+j)v} \leq K_v(1 - x_{i(j+n)v}), \qquad v \in V, j \in N_v^P, (i,n+j) \in A_v, \tag{7}$$

$$0 \leq l_{iv} \leq K_v, \qquad v \in V, i \in N_v^P, \tag{8}$$

$$t_{iv} + T_{ijv} - t_{jv} \leq (\bar{T}_i + T_{ijv})(1 - x_{ijv}), \qquad v \in V, (i,j) \in A_v, \tag{9}$$

$$\sum_{j\in N_v} x_{ijv} - \sum_{j\in N_v} x_{(n+i)jv} = 0, \qquad v \in V, i \in N_v^P, \tag{10}$$

$$t_{iv} + T_{i(n+i)v} - t_{(n+i)v} \leq 0, \qquad v \in V, i \in N_v^P, \tag{11}$$

$$\underline{T}_i \leq t_{iv} \leq \bar{T}_i, \qquad v \in V, i \in N_v, \tag{12}$$

$$y_i \in \{0,1\}, \qquad i \in N^C, \tag{13}$$

$$x_{ijv} \in \{0,1\}, \qquad v \in V, (i,j) \in A_v. \tag{14}$$

The goal is to minimize the value of the objective function (1), which sums up the costs from operating the fleet plus the cost of spot charters. Constraints (2) state that all cargoes must either be picked up by a ship or transported using spot charter. Constraints (3)-(5) describe the flow on the sailing route used by ship v. Constraints (6) and (7) keep track of the load onboard at loading and unloading nodes respectively. Constraints (8) make sure that the load does not exceed the ship capacity. Constraints (9) ensure that the time at which service starts is possible with respect to travel times. Constraints (10) make sure that if a cargo is loaded, its unloading port is also visited by the same ship. Precedence requirements are imposed through constraints (11). Time windows are stated by constraints (12), while constraints (13) and (14) impose binary requirements on the spot charter and flow variables.

The industrial and tramp routing and scheduling problem is NP-hard, being more general than the traveling salesman problem with time windows. By the same argument, the problem of determining whether any feasible solution to the problem exists where no spot charters are used is NP-complete (Savelsbergh, 1985).

3 Adaptive large neighborhood search

The ALNS heuristic was introduced by Ropke and Pisinger (2006) extending the large neighborhood search heuristic of Shaw (1997) by allowing the use of multiple destroy and repair methods within the same search process (Ribeiro and Laporte 2012). Given below as Algorithm 1, is a general pseudo-code for our ALNS implementation. The algorithm picks an initial solution and then, at each iteration, removes q cargoes from the current solution and then reinserts them in the solution differently.

Algorithm 1: ALNS heuristic for cargo routing problems

```
1   Function ALNS
2     generate initial solution s
3     s_best = s
4     repeat
5       s' = s
6       select removal and insertion heuristics based on search parameters
7       select the number of cargoes to remove and reinsert, q
8       remove q cargoes from s'
9       reinsert removed cargoes into s'
10      if ( f ( s' ) < f ( s_best ) ) then
11        s_best = s'
12      If accept ( s' , s ) then
13        s = s'
14      update search parameters
15    until stop-criterion met
16   return s_best;
```

We now describe our implementation in more detail.

3.1 Solution representation and initial solution

A solution consists of a set of visiting sequences, one for each of the actual ships and one for an artificial ship that represents cargoes transported by spot voyages. In each sequence some cargoes are transported by a specific ship. The sequence of loading and unloading is represented by using the cargo number twice, first for the loading and then for the unloading of the cargo.

The proposed ALNS starts with an initial feasible solution. To be sure of feasibility, all cargoes are put in the artificial ship representing the use of spot voyages. Thus, there is no sailing cost for the initial solution and the only cost is the cost of not transporting the cargoes.

3.2 Removal heuristics

In step 8 of Algorithm 1, some cargoes are removed from the current solution, using a removal heuristic. In these heuristics we remove q cargoes from the current solution. In each iteration, q is chosen randomly in $[4, \min(100, \xi n)]$ where n is the number of cargoes and ξ is a constant parameter. Thus, the number of removed requests, q, varies during the search to provide different neighborhood size. Three different removal heuristics are used: the Shaw removal heuristic, random removal, and worst removal.

In the Shaw removal heuristic the main idea is to remove a set of similar cargoes. First, a random cargo is selected, and then $q - 1$ additional similar cargoes are selected for removal. We define the similarity of two cargoes by a measure called relatedness. The relatedness is defined with respect to a distance measure, a time measure, a size measure, and a measure that consider the ships that can be used to serve the cargoes:

$$R(i,j) = \varphi\left(d_{A(i),A(j)} + d_{B(i),B(j)}\right) + \chi\left(\left|T_{A(i)} - T_{A(j)}\right| + \left|T_{B(i)} - T_{B(j)}\right|\right)$$

$$+\psi(Q_i - Q_j) + \omega\left(1 - \frac{|K_i \cap K_j|}{\min\{|K_i|,|K_j|\}}\right).$$

We define $d_{i,j}$ as the minimum travailing cost from i to j. The pickup and delivery locations of cargo i is denoted by $A(i)$ and $B(i)$, T_i indicates the time when location i is visited and Q_i denotes the size of request i. The set of vehicles that are able to transport cargo i is K_i.

The term weighted by φ measures distance, the term weighted by χ measures temporal connectedness, the term weighted by ψ measures size of the requests and the term weighted by ω ensures that two requests get a high relatedness measure if only a few or no vehicles are able to serve both requests.

The random removal heuristic is the simplest of the removal heuristics, and basically selects q cargoes at random and removes them from the solution.

The last removal heuristic is worst removal. The main idea of this heuristic is to remove cargoes that are placed in high cost positions. In worst removal heuristic we calculate and sort the cost of the cargoes C_i and then remove cargoes with high costs C_i. For more detail, we refer the readers to (Ropke and Pisinger 2006).

3.3 Insertion heuristics

In these heuristics we reinsert the removed cargoes into the solution. We have implemented two different insertion heuristics: a basic greedy heuristic and a regret-k heuristic.

The basic greedy heuristic is an iterative heuristic that inserts one cargo in each iteration. We calculate the cost of inserting a cargo at its best position overall and then the cargo with minimum cost is selected to be inserted at its best position. This process continues until all cargoes have been inserted, possibly by using the artificial ship representing spot voyages.

Regret heuristics improve the myopic behavior of the basic greedy heuristic. A regret-k heuristic calculates the regret value which is equal to the sum of the differences in the cost of solutions when a cargo is inserted in its best position and when it is inserted in its 2nd best, 3rd best, ..., and k-th best position. In each iteration the regret-k heuristic chooses to insert the cargo that maximizes the regret value. The selected request is inserted in its best position.

3.4 Adaptive weight adjustment

Adaptive weight adjustment is an iterative process to define and adjust (automatically) the weight of each removal and insertion heuristic using statistics from earlier iterations. For this purpose we divide the entire search into segments, with 100 iterations in each segment. Then we calculate a score for each of the heuristics during the search in the current segment and at the end of segment new weights will be calculated.

The main point here is how to score insertion and removal heuristics and update their weights. The score of a removal or insertion heuristic is increased based on the quality of the new solution obtained. A heuristic operation which results in a new global best solution gets a high increase in its score. Solutions that have not been accepted before and which improves the cost of the current solution also gets an increased score. Finally, solutions that are accepted but does not improve on the current solution get a small increase in their score.

For updating the scores between segments, we first normalize the scores (π_i) by dividing them by the number of times they have been used (θ_i) and then the new weights will be calculated by using a predefined parameter (r) to balance the earlier weights and the new normalized scores as follows. Let $w_{i,s}$ be the weight of heuristic i used in segment s. The new weights become:

$$w_{i,s+1} = w_{i,s}(1-r) + r\frac{\pi_i}{\theta_i}$$

In the first segment all heuristics have the same weight, and π_i and θ_i are re-set between segments.

3.5 Selection of heuristics

After weighting the heuristics, we have k heuristics with weights w_i, $i \in \{1,2,\dots,k\}$, then we select heuristic j with probability $\frac{w_j}{\sum_{i=1}^{k} w_i}$ by using a roulette wheel selection principle. The removal and insertion heuristic are selected independently.

3.6 Acceptance criterion and stopping criterion

We have implemented the acceptance criterion of simulated annealing which accepts solutions that are better than the current solution and also accepts solutions that are worse than the current solution with probability $e^{-|f-f_{new}|/T}$, where $T > 0$ is the temperature. The cooling schedule implemented by Crama and Schyns (2003) is used, and the algorithm stops when a specified number of iterations have been completed.

4 Benchmark suite

This section describes how an instance generator has been created, and how this generator has been used to create a basic set of benchmark instances for industrial and tramp ship routing and scheduling. The instance generator and the benchmark instances are currently available at the web page http://iot.ntnu.no/users/larsmahv/benchmarks/.

4.1 The instance generator

To generate instances, certain inputs must be specified. Initially, a set of ports and a set of ship types must be defined. Each port has a given location and belongs to a corresponding geographical region. For each ship type, the cost of entering each port is given as well as the time spent in port as a function of the quantity loaded or unloaded in the port. In addition, the travel time and travel cost, assuming a fixed sailing speed, is given between each pair of ports for each ship type.

The above inputs are static, in the sense that a relevant set of ports and a relevant set of ship types will not change frequently. Therefore, two different sets of static inputs have been prepared and included in the instance generator: One set defines ports and ship types that are relevant for deep sea shipping, and another set defines ports and ship types for short sea shipping. Figure 1 below shows a world map where most of the ports in the deep sea data set are marked. The short sea data set comprises a subset of these ports, namely those to be found in Europe. For the deep sea data set, the vessel set contains six types, including general cargo ships and oil tankers of varying sizes, whereas the short sea data set contains seven different ship types used in dry bulk shipping.

The ports are geographically divided into regions. For the deep sea data set there are five regions, namely Europe, South America, the Far East, the US East Coast, and the US West Coast. The short sea data set consists of Europe, but it is subdivided into 14 regions, corresponding to different countries. Travel times and travel costs between each pair of ports are based on real distances (Fagerholt et al. 2000) and current day fuel prices.

Remaining inputs to the instance generator are used to further specify the type of instances that should be generated. First, the user must specify which ship types to include in the instances, as well as how many ships to include from each type. As an example, when generating deep sea instances, it is appropriate to generate instances using either only oil tankers or only general cargo ships, but not a mix thereof.

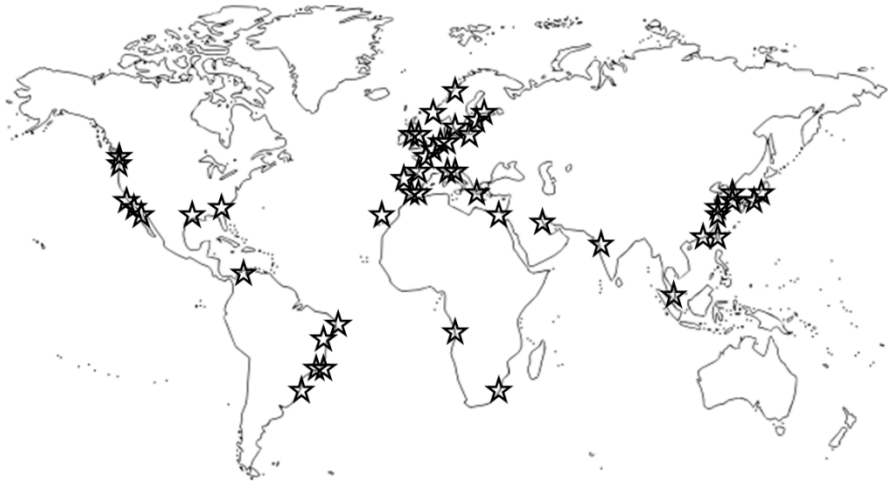

Figure 1: World map showing a selection of the ports included in the deep sea data set.

Second, the user specifies the number of cargoes to generate. This number is the total number of cargoes, which may consist of both optional and mandatory cargoes, and the instance generator will decide the mix of these depending on the other inputs described below. It is assumed that mandatory cargoes may be handled by using spot voyages, so the difference between mandatory cargoes and optional cargoes is mainly in the interpretation of the cost of not transporting them.

Third, the size of the cargoes is given as input. To simplify the input, the user may select to generate small cargoes, large cargoes, cargoes of mixed size, or full load cargoes. For each of these categories, the size of a cargo is generated randomly from a given interval which is relative to the average ship capacity. Only one dimension of capacity is considered.

Fourth, the size of the time windows is determined. There are four alternatives for the time windows namely tight, normal, loose, or none. Time windows are also relative to the travel times, and for loading ports the time windows will have a width of seven days for deep sea shipping and three days for short sea shipping. The time windows for unloading ports begin at the same time as the corresponding loading time windows, but are wider, typically allowing a ship enough time to load and unload some additional cargoes before making the corresponding delivery.

Fifth, there is an option to adjust the flow of goods between regions. Normally the flow of goods is not balanced, as some regions consume more goods than they produce or vice versa. The instance generator may either generate cargoes purely at random, or first establish trade patterns that influence the flow of goods between regions. The latter is more realistic, and ensures that certain regions are considered mainly as export regions and others mainly as import regions.

Finally, the last input regards the market condition, which may be specified as either poor, normal, or good. In a good market, there will be more optional cargoes available, with higher revenues, but the cost of using spot voyages is higher. In a poor market, there are fewer cargoes to transport, and the cost of using spot voyages is relatively low.

Once inputs have been given, the instance generator proceeds as follows: First, two subsets of ports are generated, with one subset consisting of ports where cargoes are often loaded and the other consisting of ports where cargoes are often unloaded. If the requested instance should have balanced flow of goods, these subsets are equal to the full set of ports. Second, a set of cargoes is generated, where for each cargo the loading port, the unloading port, and the size are all specified.

Third, the generator makes sure that all cargoes can be serviced by the selected fleet, and otherwise some of the cargoes are generated anew. Time windows are also calculated at this point, trying to ensure that all the ships in the fleet can be assigned an even work load and that at least one ship can reach any cargo in time for the loading time window.

Fourth, time windows and spot charter costs are adjusted according to the selected market situation. In a good market there will be more cargoes available within a given time frame, whereas in a poor market there will be less cargoes available per time unit. Finally, if some ships cannot reach any cargo in time for the loading time window, the ship is repositioned so that at least one cargo can be reached.

4.2 Benchmark instances

The instance generator may be used to generate a wide variety of instances. In the following we will describe a set of 240 instances that represent realistic combinations of settings, divided into four groups. For all four groups, the instances have normal time windows, unbalanced flow between regions, and a heterogeneous fleet of vessels, and are all based on a normal market condition.

Table 1 gives a summary of the four groups of test instances. The number of cargoes range from 7 to 130 for instances with mixed cargo sizes, and from 8 to 100 for instances with full load cargoes. The number of ships vary between three and 50, in general with fewer ships for instances with mixed cargo sizes. There are between two and four different ship types in each instance. For each group, there are 12 different combinations of the number of cargoes and the number of ships, and five instances are generated for each combination, yielding a total of 60 instances in each of the four groups.

Group	Geography	Cargo sizes	#Cargoes	#Ships	#Ship types
1	Short sea	Mixed	7-130	3-40	2-4
2	Short sea	Full load	8-100	3-50	2
3	Deep sea	Mixed	7-130	3-40	2-3
4	Deep sea	Full load	8-100	3-50	2

Table 1: summary of test instances generated.

While the instance generator is able to also generate instances with other characteristics, these instances represent typical basic instances in various segments of the shipping industry. The instance generator could be used to generate further instances to analyze the impact of different market conditions, different regional flow patterns, additional types of cargo sizes, and additional types of time windows.

5 Computational study

The benchmark instances presented in the previous section have been solved using both the ALNS heuristic described in Section 3 as well as by XPRESS Optimizer Version 25.01.05 using the model described in Section 2. This will give an indication about how difficult industrial and tramp ship routing and scheduling problems are to solve to optimality, although it should be possible to solve larger instances by using specially tailored algorithms. By comparing with optimal solutions on small instances and with lower bounds from XPRESS on larger instances, we also get an indication of the performance of the ALNS heuristic. In the following tests, the standard parameters of XPRESS was used, expect that MIPRELSTOP was set to 0 and that the search was stopped after one hour. The ALNS was executed ten times for each instance, each run including 25,000 iterations. Both XPRESS and the ALNS was executed on a single core, using a 3.4 GHz CPU with 8 GB RAM and a 64-bit operating system. Results for the short sea instances with mixed cargo sizes are reported in Tables 2 and 3. Average values are presented in Table 2 for each instance size. For XPRESS we report the gap between the lower and upper bounds found during the one hour run, the gap between the upper bound and the best known solution, and the total running time in seconds. We run the ALNS algorithm 10 times and calculate both the minimum and the average gap between the best solution found and the best known solution, as well as the average running time. The numbers in the table are average values over the five instances of each size. The heuristics included in XPRESS are able to locate feasible solutions for all instances, however for larger instances, these solutions typically involve servicing most of the cargoes by spot charters, leading to high costs.

Table 3 gives the best known values for each instance individually, with known optimal solutions highlighted in bold. Within one hour computation time, the formulation given in Section 2 can only be solved to optimality by XPRESS for instances with up to around 18 cargoes. The ALNS identifies solutions that are at least as good as those found by XPRESS on all instances in this set.

		XPRESS			ALNS		
#Cargoes	#Ships	Optimality Gap (%)	Gap to Best Known (%)	Seconds	Minimum Gap to Best Known (%)	Average Gap to Best Known (%)	Seconds
7	3	0.00	0.00	0.26	0.00	0.00	1.59
10	3	0.00	0.00	0.76	0.00	0.00	2.54
15	4	0.00	0.00	85.36	0.00	0.58	5.28
18	5	3.70	1.00	2549.74	0.00	0.51	7.54
22	6	19.32	3.09	3038.10	0.00	1.82	11.50
23	13	43.17	17.67	3599.58	0.00	0.57	14.95
30	6	391.85	244.64	3599.60	0.00	1.48	22.61
35	7	480.86	276.47	3599.58	0.00	1.67	32.34
60	13	590.88	325.89	3600.56	0.00	0.91	120.89
80	20	652.46	317.22	3602.80	0.00	0.91	248.08
100	30	95344.74	345.49	3614.92	0.00	0.95	466.46
130	40	119034.11	344.60	3691.34	0.00	0.70	1016.78

Table 2: average results for short sea shipping instances with mixed cargo sizes.

#Cargoes	#Ships	Instance #1	Instance #2	Instance #3	Instance #4	Instance #5
7	3	**1476444**	**1134176**	**1196466**	**1256139**	**1160394**
10	3	**2083965**	**2012364**	**1986779**	**2125461**	**2162453**
15	4	**1959153**	**2560004**	**2582912**	**2265396**	**2230861**
18	5	**2374420**	**2987358**	**2301308**	2400016	**2813167**
22	6	**3928483**	3683436	3264770	3228262	3770560
23	13	2276832	2255870	2362503	2250110	2325941
30	6	4958542	4568421	4106293	4449812	4528514
35	7	4942430	4543650	4433847	4580935	5511661
60	13	8202138	8055970	7651685	8593410	8950046
80	20	10376392	10387253	9763401	11598420	10983117
100	30	12845591	13057536	12088444	13791917	13502730
130	40	16524192	16713067	15862154	17305841	18533293

Table 3: individual best known results for short sea shipping instances with mixed cargo sizes, with bold face font indicating known optimal values.

Tables 4 and 5 provide similar information about the short sea instances with full load cargoes. The difference is that full load instances are easier for XPRESS to solve, finding optimal solutions of instances with up to 25 cargoes. Among the five instances with 17 cargoes, there is one instance for which the ALNS does not find the optimal solution. However, for that instance ALNS finds, in all 10 runs, a solution whose value is only 0.05 % higher than the optimal solution.

		XPRESS			ALNS		
#Cargoes	#Ships	Optimality Gap (%)	Gap to Best Known (%)	Seconds	Minimum Gap to Best Known (%)	Average Gap to Best Known (%)	Seconds
8	3	0.00	0.00	0.12	0.00	0.00	1.84
11	4	0.00	0.00	0.46	0.00	0.13	2.85
13	5	0.00	0.00	0.26	0.00	0.07	4.10
16	6	0.00	0.00	0.70	0.00	0.05	6.14
17	13	0.00	0.00	1.00	0.01	0.01	8.59
20	6	0.00	0.00	47.78	0.00	0.14	10.71
25	7	0.85	0.00	2428.02	0.00	0.22	16.47
35	13	9.83	0.60	3600.04	0.00	0.29	35.91
50	20	16.41	1.90	3600.52	0.00	0.35	83.08
70	30	204.65	156.51	3603.32	0.00	0.70	210.66
90	40	364.12	276.11	3617.60	0.00	0.47	418.57
100	50	381.14	269.85	3651.22	0.00	0.35	587.52

Table 4: average results for short sea shipping instances with full load cargoes.

#Cargoes	#Ships	Instance #1	Instance #2	Instance #3	Instance #4	Instance #5
8	3	**1391997**	**1246273**	**1698102**	**1777637**	**1636788**
11	4	**1052463**	**1067139**	**1212388**	**1185465**	**1310285**
13	5	**2034184**	**2043253**	**2378283**	**2707215**	**3011648**
16	6	**3577005**	**3560203**	**4081013**	**3667080**	**3438493**
17	13	**2265731**	**3154165**	**2699378**	**2806231**	**2910814**
20	6	**2973381**	**3206514**	**3197445**	**3342130**	**3156378**
25	7	**3833588**	**3673666**	**4238213**	4260762	4069693
35	13	2986667	3002974	3084339	3952461	3293086
50	20	7265169	7470529	6938306	8947342	7330386
70	30	10088768	10503191	10314521	10910832	10908679
90	40	13468846	13981808	12767716	14506983	13720466
100	50	13893237	14718351	13206559	14936198	14106741

Table 5: individual best known results for short sea shipping instances with full load cargoes, with bold face font indicating known optimal values.

Deep sea instances differ from short sea instances not only in the geography and in the ratio between traveling times and time spent in port, but also through different ship types being included. It turns out that XPRESS is able to solve to optimality larger instances for the deep sea case than for the short sea case. Tables 6 and 7 give results for deep sea instances with mixed cargo sizes, and Tables 8 and 9 give results for deep sea instances with full load cargoes.

		XPRESS			ALNS		
#Cargoes	#Ships	Optimality Gap (%)	Gap to Best Known (%)	Seconds	Minimum Gap to Best Known (%)	Average Gap to Best Known (%)	Seconds
7	3	0.00	0.00	0.40	0.00	0.00	1.64
10	3	0.00	0.00	0.58	0.00	0.01	2.40
15	4	0.00	0.00	21.26	0.00	1.26	5.07
18	5	2.45	0.00	979.46	0.00	0.47	7.58
22	6	3.92	0.10	1569.82	0.00	2.18	11.29
23	13	5.33	0.11	1583.08	0.00	0.12	14.59
30	6	239.06	107.93	3600.28	0.00	0.79	22.06
35	7	236.45	114.91	3600.58	0.00	0.98	30.89
60	13	845.58	327.13	3601.52	0.00	2.41	115.32
80	20	1124.49	349.94	3603.58	0.00	1.83	255.76
100	30	9380.99	324.07	3616.30	0.00	1.50	479.91
130	40	810851.39	359.54	3677.62	0.00	1.44	1048.17

Table 6: average results for deep sea shipping instances with mixed cargo sizes.

#Cargoes	#Ships	Instance #1	Instance #2	Instance #3	Instance #4	Instance #5
7	3	**5233464**	**6053699**	**5888949**	**6510656**	**7220458**
10	3	**7986248**	**7754484**	**9499357**	**8617192**	**8653992**
15	4	**13467090**	**12457251**	**12567396**	**11764241**	**10833640**
18	5	**43054055**	25068287	**29211238**	**32281904**	**40718028**
22	6	**41176718**	**37236363**	**38215238**	34129809	**46379332**
23	13	**41002992**	28014147	29090422	**33685274**	**38664843**
30	6	19227093	16896623	21298546	21076728	24490671
35	7	65082675	54839181	56258180	61489997	63943961
60	13	81709586	75878996	92585918	91076203	89721702
80	20	72019446	74476462	78918099	76459495	75378149
100	30	153747613	154533570	152338747	157017186	162084771
130	40	239877031	234909061	244239529	228680203	246883618

Table 7: individual best known results for deep sea shipping instances with mixed cargo sizes, with bold face font indicating known optimal values.

For deep sea instances with mixed cargo sizes, XPRESS finds optimal solutions to three out of five instances with 23 cargoes, although it fails to prove optimality for one of the instances with 18 cargoes. While there appears to be a serious degradation of the solution quality obtained by using XPRESS when going from 23 to 30 cargoes, the ALNS seems to scale well.

For the instances with full load cargoes, XPRESS finds optimal solutions to all instances with 25 cargoes or less, and is even able to prove optimality for one out of five instances with 50 cargoes. However, for one of the other full load deep sea instances with 50 cargoes XPRESS terminates prematurely when using standard settings, and had to be run without cut generation by setting CUTSTRATEGY = 0. There is again only one instance where ALNS does not succeed in finding the best known solution, and for this instance the gap to the best known varies between 0.21 % and 1.45 %.

		XPRESS			ALNS		
#Cargoes	#Ships	Optimality Gap (%)	Gap to Best Known (%)	Seconds	Minimum Gap to Best Known (%)	Average Gap to Best Known (%)	Seconds
8	3	0.00	0.00	0.32	0.00	0.00	1.75
11	4	0.00	0.00	0.48	0.00	0.00	2.74
13	5	0.00	0.00	0.52	0.00	0.00	3.79
16	6	0.00	0.00	0.86	0.00	0.03	5.91
17	13	0.00	0.00	1.34	0.00	0.00	8.00
20	6	0.00	0.00	1.42	0.00	0.01	9.82
25	7	0.00	0.00	8.02	0.00	0.41	15.87
35	13	4.84	0.26	2733.18	0.00	1.03	35.29
50	20	10.14	0.79	3375.20	0.04	0.48	84.81
70	30	13.12	1.21	3604.16	0.00	0.28	212.74
90	40	119.03	91.67	3614.92	0.00	0.60	420.68
100	50	163.01	119.11	3630.36	0.00	0.60	591.11

Table 8: average results for deep sea shipping instances with full load cargoes.

#Cargoes	#Ships	Instance #1	Instance #2	Instance #3	Instance #4	Instance #5
8	3	**9584863**	**9369654**	**4596681**	**6899730**	**6815253**
11	4	**34854819**	**25454434**	**29627143**	**33111680**	**28175914**
13	5	**11629005**	**11820655**	**9992593**	**12819619**	**10534892**
16	6	**51127590**	**44342796**	**45391842**	**39687114**	**42855603**
17	13	**17316720**	**12194861**	**12091554**	**12847653**	**13213406**
20	6	**16406738**	**16079401**	**17342200**	**16529748**	**17449378**
25	7	**22773158**	**20206329**	**19108952**	**22668675**	**23036603**
35	13	86951609	**83422071**	83898591	**91970481**	91130154
50	20	41398100	37872273	39916853	**43941098**	41971890
70	30	142923793	135766719	162903901	156541043	157037323
90	40	191675120	191143649	212152967	211046180	198625224
100	50	207105715	208540820	218438412	221248187	224430601

Table 9: individual best known results for deep sea shipping instances with full load cargoes, with bold face font indicating known optimal values.

6 Concluding remarks

This paper has considered the cargo ship routing and scheduling problem that arises in industrial and tramp shipping. We have presented and made available a wide range of benchmark instances that were made to represent realistic planning problems for various segments of the shipping industry. An instance generator that can be used to produce additional instances representing other shipping segments has also been provided. To solve the cargo ship routing and scheduling problem, a state-of-the-art adaptive large neighborhood search (ALNS) heuristic has been implemented. The benchmark instances have been solved both by the ALNS and a commercial MIP solver based on a mathematical formulation of the problem. The commercial MIP solver is able to find optimal solutions to instances up to around 18-25 cargoes, depending on the structure of the instances. The ALNS finds solutions that are at least as good as those identified by the MIP solver on 238 out of 240 instances, and seems to scale well also for instances up to 130 cargoes.

We hope our work will stimulate other researchers to develop better solution algorithms for this important planning problem, and, even more importantly, to model and solve various real-life extensions that go beyond what is included in our benchmark instances.

Acknowledgements

This research was carried out with financial support from the Research Council of Norway through the DOMinant II and GREENSHIPRISK projects.

References

Al-Khayyal F, Hwang SJ (2007) Inventory constrained maritime routing and scheduling for multi-commodity liquid bulk, Part I: Applications and model. European Journal of Operational Research 176(1):106-130

Brouer BD, Fernando Alvarez J, Plum CEM, Pisinger D, Sigurd MM (2013) A base integer programming model and benchmark suite for liner-shipping network design. Transportation Science 48(2):281-312

Brønmo G, Christiansen M, Fagerholt K, Nygreen B (2007) A multi-start local search heuristic for ship scheduling – a computational study. Computers and Operations Research 34(3):900-917

Christiansen M, Fagerholt K (2014) Ship routing and scheduling in industrial and tramp shipping. Accepted for publication in Vehicle Routing: Problems, Methods, and Applications 2nd ed. (eds. Toth and Vigo), SIAM

Christiansen M, Fagerholt K, Nygreen B, Ronen D (2013) Ship routing and scheduling in the new millennium. European Journal of Operational Research 228(3):467-483

Christiansen M, Fagerholt K, Ronen D (2004) Ship routing and scheduling: status and perspectives. Transportation Science 38(1):1-18

Crama Y, Schyns M (2003) Simulated annealing for complex portfolio selection problems, European Journal of Operational Research 150(3): 546–571

Desrosiers J, Dumas Y, Solomon MM, Soumis F (1995) Time constrained routing and scheduling. In: Ball MO, Magnanti TL, Monma CL, Nemhauser GL (eds) Handbooks in Operations Research and Management Science, 8th vol. North-Holland, Amsterdam, pp 35-139

Fagerholt K, Heimdal SI, Loktu A (2000) Shortest path in the presence of obstacles: An application to ocean shipping. Journal of the Operational Research Society 51:683-688

Hwang HS, Visoldilokpun S, Rosenberger JM (2008) A branch-and-price-and cut method for ship scheduling with limited risk. Transportation Science 42 (3):336–351

Jetlund AS, Karimi IA (2004) Improving the logistics of multi-compartment chemical tankers. Computers and Chemical Engineering 28(8):1267-1283

Kontovas K, Psaraftis HN (2011) Reduction of emissions along the maritime intermodal container chain: operational models and policies. Maritime Policy and Management 38(4) 451-469

Korsvik JE, Fagerholt K, Laporte G (2010) A tabu search heuristic for ship routing and scheduling. Journal of the Operational Research Society 61(4):594-603

Lawrence SA (1972) International sea transport: The years ahead. Lexington Books, Lexington

Lin DY, Liu HY (2011) Combined ship allocation, routing and freight assignment in tramp shipping. Transportation Research Part E 47(4):414-431

Malliappi F, Bennell JA, Potts CN (2011) A variable neighborhood search heuristic for tramp ship scheduling. Computational Logistics, Lecture Notes in Computer Science 6971, 273–285

Norstad I, Fagerholt K, Laporte G (2011) Tramp ship routing and scheduling with speed optimization. Transportation Research Part C 19(5):853-865

Papageorgiou DJ, Nemhauser GL, Sokol J, Cheon MS, Keha AB (2014) MIRPLib - A library of maritime inventory routing problem instances: Survey, core model, and benchmark results. European Journal of Operational Research 235(2):350-366

Psaraftis HN, Kontovas CA (2013) Speed models for enegy-efficient maritime transportation: A taxonomy and survey. Transportation Research Part C 26: 331-351

Ribeiro GM, Laporte G (2012) An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem. Computers and Operations Research 39(3):728-735

Ronen D (1983) Cargo ships routing and scheduling: survey of models and problems. European Journal of Operational Research 12(2):119-126

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

Savelsbergh MWP (1985) Local search in routing problems with time windows. Annals of Operations Research 4:285-305

Shaw P (1997) A new local search algorithm providing high quality solutions to vehicle routing problems. Technical report, University of Strathclyde, Glasgow

Solomon MM (1987) Algorithms for the vehicle routing and scheduling problem with time window constraints. Operations Research 35(2):254–265

UNCTAD (2013) Review of Maritime Transport. United Nations, New York and Geneva

Vilhelmsen C, Lusby R, Larsen J (2014) Tramp ship routing and scheduling with integrated bunker optimization. EURO Journal on Transportation and Logistics. doi:10.1007s13676-013-0039-8