A two-phase heuristic for an in-port ship routing problem with tank allocation

Xin Wang[1]
Mari Jevne Arnesen[1]
Kjetil Fagerholt[1,2]
Magnhild Gjestvang[1]
Kristian Thun[1]

[1]Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology, Trondheim, Norway

[2]SINTEF Ocean, Trondheim, Norway

X. Wang, M. J. Arnesen, K. Fagerholt, M. Gjestvang and K. Thun (2017). A two-phase heuristic for an in-port routing problem with tank allocation. *Computers & Operations Research*. doi:10.1016/j.cor.2017.11.005

Abstract

This paper addresses an in-port ship routing problem with tank allocation that arises in the chemical shipping industry. The aim is to optimize a tanker's port call operation that integrates sequencing decisions for visiting terminals and allocating cargo loads to available tanks while taking into account cargo time windows, terminal draft limits and various tank allocation restrictions. We model the problem as a Traveling Salesman Problem with Pickups and Deliveries, Time Windows, Draft Limits and Tank Allocation (TSPPD-TWDLTA), and propose a two-phase heuristic to solve it. Computational studies show that the heuristic is able to provide good solutions to real-sized in-port routing problems with tank allocation in chemical shipping in a reasonable amount of time.

A two-phase heuristic for an in-port ship routing problem with tank allocation

Xin Wang*[1], Mari Jevne Arnesen[1], Kjetil Fagerholt[1,2], Magnhild Gjestvang[1], and Kristian Thun[1]

[1]Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology, Trondheim, Norway
[2]SINTEF Ocean, Trondheim, Norway

November 7, 2017

Abstract

This paper addresses an in-port ship routing problem with tank allocation that arises in the chemical shipping industry. The aim is to optimize a tanker's port call operation that integrates sequencing decisions for visiting terminals and allocating cargo loads to available tanks while taking into account cargo time windows, terminal draft limits and various tank allocation restrictions. We model the problem as a Traveling Salesman Problem with Pickups and Deliveries, Time Windows, Draft Limits and Tank Allocation (TSPPD-TWDLTA), and propose a two-phase heuristic to solve it. Computational studies show that the heuristic is able to provide good solutions to real-sized in-port routing problems with tank allocation in chemical shipping in a reasonable amount of time.

Keywords: maritime transportation; ship routing; tank allocation; traveling salesman problem with pickup and delivery; dynamic programming

1 Introduction

This paper introduces a real-life in-port ship routing problem with tank allocation faced by Odfjell, a Norwegian public listed chemical shipping and tank terminal company based in Bergen, Norway. The problem considers a tanker arriving at a particular port with numerous terminals and the aim is finding an optimized plan with shortest time spent in port that comprises sequencing decisions for visiting terminals, and the corresponding

*Corresponding author. *E-mail address:* xin.wang@iot.ntnu.no

tank allocation plan. Figure 1 shows a small example port call at the Port of Houston. In this example, the ship enters the port via an anchorage point in the Galveston Bay, and services three terminals along the channel before sailing off for a new voyage. In Arnesen et al. (2017), we have studied a simplified version of this problem without tank allocation, by modeling the in-port ship routing problem as a Traveling Salesman Problem with Pickups and Deliveries, Time Windows and Draft Limits (TSPPD-TWDL), and proposed a solution method based on forward dynamic programming (DP). In this paper, we explicitly take into account tank allocation as an essential component of the port call plan, i.e. the decisions with regards to which tank(s) should be used for each cargo, which is an important aspect of reality in the daily operations in chemical shipping.

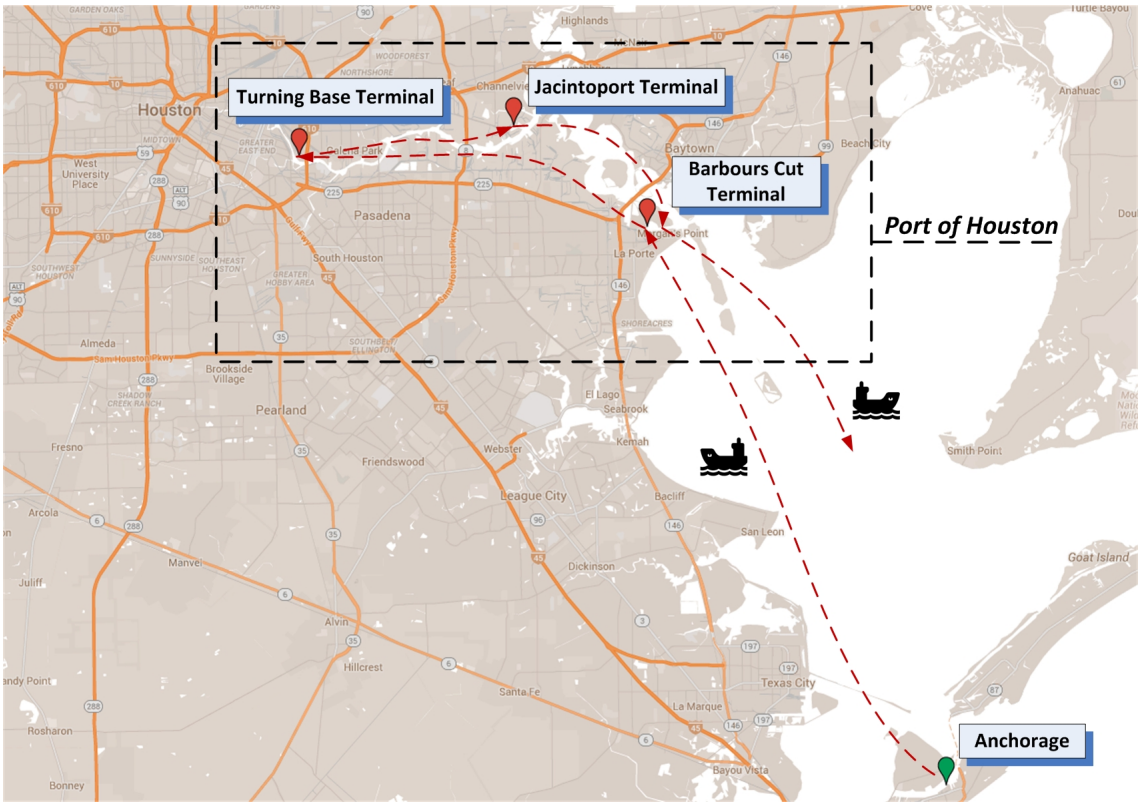
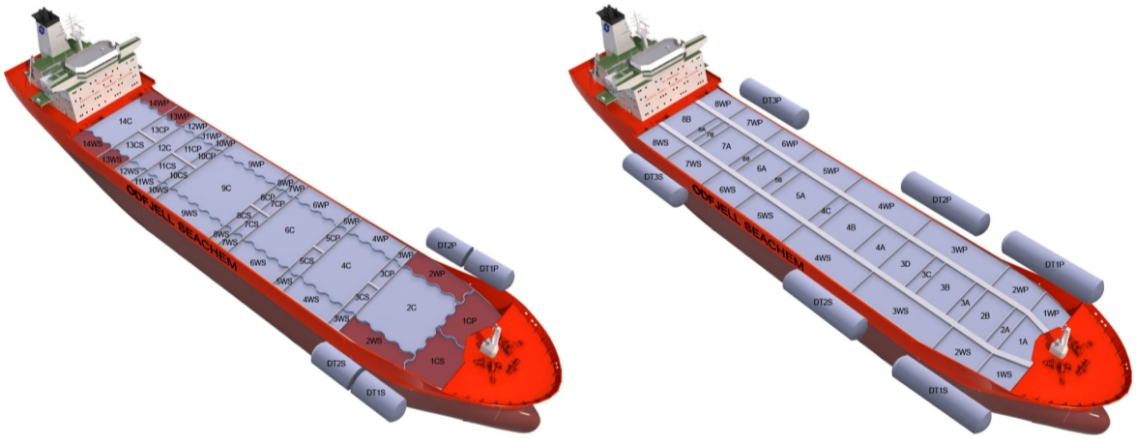


Figure 1: Illustration of an example port call. The ship approaches the Port of Houston from an anchorage point in the Galveston Bay and services three terminals within the port.

The chemical shipping market is defined as the market for transportation of bulk liquid chemicals by sea. Compared to most other bulk shipping markets, the shipments in the chemical shipping industry are usually of smaller sizes and thus allow the carrier to combine and transport different cargoes on one single tanker. Chemical tankers are therefore designed with a large number of compartments or tanks, each being able to carry several product types, which makes chemical tankers very flexible. In the mean

time, more sophisticated allocation plans are required to deal with the large number of tanks and a wide range of chemical products.

The number of tanks on a typical tanker in chemical shipping can range from 15 to over 50, and the tanks often differ in shapes and volumes. Figure 2 shows the tank configuration of two tankers operated by Odfjell that have 52 and 40 tanks, respectively. The tanks may be coated with different materials, e.g. stainless steel, zinc, epoxy or polymer, and cargoes can only be allocated to tanks with compatible coatings. Other constraints, such as tank capacities and stability of the ship, also need to be considered when allocating cargo loads to the tanks on board the ship.



(a) Bow Faith – Kvaerner
52 tanks

(b) Bow Sea – Poland
40 tanks

Figure 2: Illustration of the tank configuration of (a) tanker Bow Faith of class Kvaerner with 52 tanks, and (b) tanker Bow Sea of class Poland with 40 tanks (Odfjell, 2016).

In maritime transportation, there exists a rather limited literature on the *tank allocation problem* (TAP), and usually for a fixed set of cargoes on a given route. Vouros et al. (1996) propose a framework for developing Expert Loading Systems for allocating chemical products to the compartments of a ship and for planning cargo handling operations, but for a fixed set of cargoes only. Hvattum et al. (2009) present several models for the TAP that arises in maritime liquid bulk shipping, considering the loads being moved on and off the ship along a given trading route. Vilhelmsen et al. (2016) further propose a hybrid method that can produce a feasible solution within a short period of time for a TAP on a given route. There also exist in the literature other types of loading and allocation problems in maritime transportation. See for example Wilson and Roach (2000) and Kang and Kim (2002) for stowage problems in the loading of container ships; Øvstebø et al. (2011a) for the optimization of stowage plans for RoRo (Roll-on/Roll-off) ships.

Some research has also been done on combined problems considering both ship routing and stowage planning. Fagerholt and Christiansen (2000) present a combined ship routing

and allocation problem in dry bulk shipping, where each ship is equipped with a flexible cargo hold that can be partitioned (with bulkheads) into several small holds. Øvstebø et al. (2011b) also introduce a problem where routing and scheduling are considered simultaneously with stowage decisions in RoRo shipping. The stowage problems in these studies are, however, different from the TAPs faced in the chemical shipping industry.

The goal of this paper is therefore to provide a first attempt to introduce and solve a combined ship routing problem integrating tank allocation as a planning component. We model the problem as a Traveling Salesman Problem with Pickups and Deliveries, Time Windows, Draft Limits and Tank Allocation (TSPPD-TWDLTA) and propose a two-phase heuristic to solve it. In the first phase a dynamic programming (DP) procedure, which is an adaption of the DP-algorithm proposed in Arnesen et al. (2017), is used to generate a set of candidate servicing routes. Instantiated fixed-route TAPs are then solved as subproblems in the second phase. The computational study presented in this paper using data acquired from the shipping company shows that the proposed heuristic is able to solve real-sized in-port routing problems with tank allocation in chemical shipping efficiently.

The remainder of this paper is organized as follows. Section 2 gives the problem description and mathematical formulation, while a two-phase heuristic is introduced in Section 3. The computational study is presented in Section 4, and we conclude in Section 5.

2 The problem of in-port routing with tank allocation

We consider a chemical tanker arriving at a particular port with a given set of pickup and delivery commitments indicating which cargoes (and their associated terminals) it must service during the port call for loading and unloading operations. The tanker enters the port via an *anchorage* point to the first terminal in its visiting sequence, carrying a number of contractual *delivery cargoes* (that are to be delivered to their respective destination terminals). The tanker then sails a planned route with sequenced terminal visits within the port to unload these cargoes, while also loading a set of *pickup cargoes* from different terminals along the way, prior to finishing the port call and sailing back to the anchorage.

During the course of the port call, certain constraints have to be respected. Apart from the obvious requirement of servicing all cargoes before sailing back to the anchorage, every cargo has a *time window* indicating the earliest and latest service starting times at its corresponding terminal. All terminals in the port have their respective *draft limits* which are represented in terms of maximum weights the ship can safely carry on board, see for example Rakke et al. (2012) for more discussions. Additionally, the *total capacity* (in dead weight tonnes) of the ship must not be exceeded.

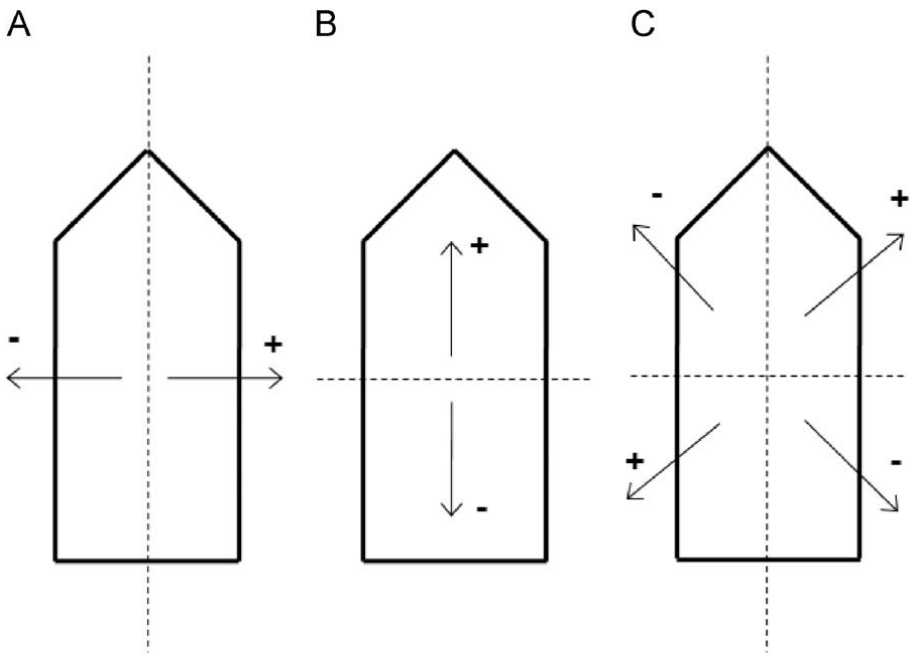


Figure 3: Various dimensions around which stability and strength requirements are formed: (A) roll, (B) trim, and (C) strength (Hvattum et al., 2009).

As regards *tank allocation* considerations, every cargo on board the ship has to be allocated to one or more tanks depending on the tank sizes and types, compatibility between tanks and cargoes, and the volume of the cargo. It is not allowed to mix different cargoes in the same tank, even if the cargoes are of the same product. In some cases, two cargoes may be in conflict with each other and therefore cannot be stored in adjacent tanks due to special rules with chemical products. In addition, there are ship stability and strength requirements that prohibit certain allocation combinations. Some dimensions with regards to ship stability and strength are shown in Figure 3, and the related restrictions regulate the weight distribution and ensure the steady state of a ship. In this paper, we incorporate tank allocation restrictions into our model using constraints inspired by Hvattum et al. (2009).

Let there be n cargoes that require pickup or delivery services. Associate to cargo i a node i and every cargo is, according to the contracts, either a pickup node or a delivery node. Note that different nodes may correspond to the same physical terminal in the port. Also associate to the anchorage, nodes 0 and $n+1$, representing the origin and destination nodes. We then propose in the following the arc-flow formulation of the TSPPD-TWDLTA, defined on a directed graph $G = (N, A)$, where $N = \{0, ..., n + 1\}$ is the set of nodes and A the set of admissible arcs, which is a subset of $\{(i, j) : i, j \in N, i \neq j\}$.

Sets

N	set of nodes, $N = \{0, ..., n+1\}$.
A	set of admissible arcs.
N^C	set of cargoes, $N^C = \{1, ..., n\}$.
N^+, N^-	set of pickup and delivery cargoes, respectively.
F	set of tanks on board the ship.
S	set of stability and strength dimensions, such as trim and roll.
N_f^F	set of cargoes compatible with tank f.
F_i^N	set of tanks compatible with cargo i.
N_i^N	set of cargoes in conflict with cargo i.
F_{ijf}^P	set of tanks that cannot be used for cargo j if cargo i is in tank f.

Parameters

Q_i	*signed* weight of cargo i, **positive** for pickup cargoes and **negative** for delivery ones.
V_i	volume of cargo i.
E_i	density of cargo i, equal to $\lvert Q_i \rvert / V_i$.
Q^+, Q^-	total tonnage of the pickup and delivery cargoes, respectively.
$\underline{T_i}, \overline{T_i}$	earliest and latest time to start the service for cargo i.
D_i	draft limit of the terminal associated with cargo i.
T_{ij}	movement time from node i to node j. This includes sailing time, cargo loading/unloading time, tank washing time (when node i is a delivery cargo) and berthing time, see Arnesen et al. (2017) for details.
K	total capacity of the ship in dead weight tonnes.
$\underline{K}_f^F, \overline{K}_f^F$	minimum and maximum volume of tank f when used, respectively. Minimum volume is imposed to avoid sloshing inside the tank.
W_{if}^I	initial cargo placements, equal to 1 if tank f contains cargo i when starting from anchorage, and 0 otherwise.
V_{if}^I	initial volume of cargo i in tank f when starting from anchorage.
M_{sf}	moment arm with respect to stability and strength dimension s for tank f.
M_s^+, M_s^-	upper and lower limit on total moment for stability and strength dimension s.

Decision Variables

t_i	time at which the service at node i begins.
x_{ij}	binary flow variables, equal to 1 if the ship sails directly from node i to node j, and 0 otherwise.
y_{ij}	total weight on board the ship when sailing from i to j.
z_{ik}	sequence variables, equal to 1 if node i is visited as number k in the sequence of nodes visited, and 0 otherwise.
w_{ifk}	allocation variables, equal to 1 if cargo i is allocated to tank f in sequence number k, and 0 otherwise.
v_{ifk}	volume of cargo i allocated to tank f in sequence number k.

$$\text{minimize} \qquad t_{n+1} \tag{1}$$

Objective function (1) minimizes the completion time for the ship's port operations, which is when the ship is back to the anchorage after servicing all cargoes.

$$\sum_{j \in N} x_{0j} = 1 \tag{2}$$

$$\sum_{i \in N} x_{i,n+1} = 1 \tag{3}$$

$$\sum_{j \in N | (i,j) \in A} x_{ij} = 1 \qquad i \in N^C \tag{4}$$

$$\sum_{i \in N | (i,j) \in A} x_{ij} = 1 \qquad j \in N^C \tag{5}$$

Constraints (2) and (3) ensure the conservation of flow in the origin and destination nodes, respectively, while constraints (4) and (5) ensure that every cargo within the port is serviced exactly once.

$$\sum_{j \in N^C} y_{0j} = Q^- \tag{6}$$

$$\sum_{(j,i) \in A} y_{ji} - \sum_{(i,j) \in A} y_{ij} = Q_j \qquad j \in N^C \tag{7}$$

Constraint (6) states that the ship leaves from the anchorage carrying the total load that is to be delivered to the terminals within the port. Constraints (7) ensure that the difference between ingoing and outgoing shiploads of each node equals the weight of the cargo serviced at the node.

$$y_{0j} \leq Q^{-} x_{0j} \qquad (0,j) \in A \tag{8}$$

$$y_{i,n+1} \leq Q^{+} x_{i,n+1} \qquad (i,n+1) \in A \tag{9}$$

$$Q_i x_{ij} \leq y_{ij} \leq (K - Q_j) x_{ij} \qquad (i,j) \in A \mid i,j \in N^{+} \tag{10}$$

$$(Q_i - Q_j) x_{ij} \leq y_{ij} \leq K x_{ij} \qquad (i,j) \in A \mid i \in N^{+}, j \in N^{-} \tag{11}$$

$$-Q_j x_{ij} \leq y_{ij} \leq (K + Q_i) x_{ij} \qquad (i,j) \in A \mid i,j \in N^{-} \tag{12}$$

$$y_{ij} \leq min\{K - Q_j, K + Q_i\} x_{ij} \qquad (i,j) \in A \mid i \in N^{-}, j \in N^{+} \tag{13}$$

Constraints (8) and (9) connect the x- and y- variables out from the origin node and into the destination node. Constraints (10)-(13) ensure that the load on board the ship along the route never exceeds the total capacity of the ship.

$$y_{ij} \leq D_j \, x_{ij} \qquad (i,j) \in A \mid j \in N^{-} \tag{14}$$

$$y_{ij} \leq D_i \, x_{ij} \qquad (i,j) \in A \mid i \in N^{+} \tag{15}$$

Constraints (14) and (15) represent the draft limit restrictions when arriving at delivery nodes and departing from pickup nodes, respectively.

$$t_i + T_{ij} - t_j - M_{ij}(1 - x_{ij}) \leq 0 \qquad (i,j) \in A \tag{16}$$

$$t_i - t_0 \geq 0 \qquad i \in N \setminus \{0\} \tag{17}$$

$$t_{n+1} - t_i \geq 0 \qquad i \in N \setminus \{n+1\} \tag{18}$$

$$\underline{T_i} \leq t_i \leq \overline{T_i} \qquad i \in N^{C} \tag{19}$$

The scheduling of the ship is restricted by constraints (16) - (19). Constraints (16) state that the time at which the ship starts to serve node j must be greater than or equal to the start of serving the previous node i, plus the time taken to move from i to j, where $M_{ij} = max(0, \overline{T_i} + T_{ij} - \underline{T_j})$. Constraints (17) enforce node 0 to be the origin node while constraints (18) enforce node $n+1$ to be the destination node. Constraints (19) ensure that the service at each node starts within the given time window.

$$z_{00} = 1 \tag{20}$$

$$z_{j1} = x_{0j} \qquad j \in N \mid (0,j) \in A \tag{21}$$

$$\sum_{k \in N} z_{ik} = 1 \qquad i \in N \tag{22}$$

$$\sum_{i \in N} z_{ik} = 1 \qquad k \in N \tag{23}$$

$$z_{jk} - z_{i,k-1} - x_{ij} \geq -1 \qquad (i,j) \in A, k \in N \setminus \{0\} \tag{24}$$

The sequencing constraints (20) - (24) connect the binary flow variables x and the sequence variables z. Constraint (20) ensures that the start node at anchorage is assigned the first number in the sequence, while constraints (21) impose the start of the sequence. Constraints (22) state that a node can be assigned to only one number in the sequence, while constraints (23) ensure that one sequence number is also assigned to only one particular node. Constraints (24) make sure that if the ship sails from i to j where node i was number $k-1$ in the sequence, then node j is number k in the same sequence.

$$w_{if0} = W^I_{if} \qquad i \in N^-, f \in F \tag{25}$$

$$v_{if0} = V^I_{if} \qquad i \in N^-, f \in F \tag{26}$$

Constraints (25) and (26) state that when the ship leaves anchorage, its tanks contain all cargoes that are going to be delivered according to the initial cargo placements.

$$w_{ifk} - w_{if,k-1} + z_{ik} \geq 0 \qquad i \in N^-, f \in F, k \in N \setminus \{0\} \tag{27}$$

$$w_{ifk} + \sum_{q=0}^{k} z_{iq} \leq 1 \qquad i \in N^-, f \in F, k \in N \setminus \{0\} \tag{28}$$

$$v_{ifk} - v_{if,k-1} + V^I_{if}(1 - w_{ifk}) \geq 0 \qquad i \in N^-, f \in F, k \in N \tag{29}$$

$$v_{ifk} - V^I_{if} w_{ifk} \leq 0 \qquad i \in N^-, f \in F, k \in N \tag{30}$$

Constraints (27) state that if a tank is used for a given cargo at sequence number $k-1$, then it must also be used at sequence number k unless the ship unloads that cargo at that sequence number. Constraints (28) ensure that a tank is used for a delivery cargo only, as long as its corresponding delivery node has not yet been visited. Constraints (29) make sure that the volume in a tank remains unchanged unless the corresponding cargo is unloaded, otherwise constraints (30) ensure that the volume in a tank is emptied.

$$\sum_{q=0}^{k-1} w_{ifq} + z_{ik} \leq 1 \qquad i \in N^+, f \in F^N_i, k \in N \setminus \{0\} \tag{31}$$

$$w_{ifk} - w_{if,k-1} \geq 0 \qquad i \in N^+, f \in F, k \in N \setminus \{0\} \tag{32}$$

$$v_{ifk} - v_{if,k-1} \geq 0 \qquad i \in N^+, f \in F, k \in N \setminus \{0\} \tag{33}$$

Constraints (31) force w_{ifq} to be zero until the pickup node i is visited and the corresponding cargo is picked up. Constraints (32) and (33) make sure that if a cargo is picked up, it stays in the tank for the rest of the route.

$$\sum_{f \in F_i^N} \overline{K}_f^F w_{ifk} - V_i z_{ik} \geq 0 \qquad i \in N^+, k \in N \tag{34}$$

$$\sum_{f \in F_i^N} v_{ifk} - V_i z_{ik} = 0 \qquad i \in N^+, k \in N \tag{35}$$

$$\underline{K}_f^F w_{ifk} \leq v_{ifk} \leq \overline{K}_f^F w_{ifk} \qquad i \in N^+, f \in F, k \in N \tag{36}$$

Constraints (34) and (35) ensure that if a cargo is picked up, all of the cargo volume is allocated to one or more tank(s). Constraints (36) restrict the quantity (in volume) of cargo i in tank f to be between the tank's minimum and maximum volume capacity.

$$\sum_{f \in F} v_{if,n+1} = V_i \qquad i \in N^+ \tag{37}$$

$$\sum_{f \in F} w_{ifn} = \sum_{f \in F} w_{if,n+1} \qquad i \in N^+ \tag{38}$$

Constraints (37) and (38) ensure that the allocation on board when the ship reaches anchorage in the last sequence number is the same as in the second last sequence number.

$$\sum_{i \in N_f^F} w_{ifk} \leq 1 \qquad f \in F, k \in N \tag{39}$$

$$\sum_{j \in N_i^N} \sum_{e \in F_{ijf}^P} w_{jek} - |F|(1 - w_{ifk}) \leq 0 \qquad i \in N^C, f \in F_i^N, k \in N \tag{40}$$

$$M_s^- \leq \sum_{f \in F} \sum_{i \in N_f^F} M_{sf} E_i v_{ifk} \leq M_s^+ \qquad k \in N, s \in S \tag{41}$$

Constraints (39) state that maximum one cargo can be allocated to a specific tank in every given sequence. Constraints (40) enforce that if cargo i is in tank f, cargo j is not placed in prohibited tanks. Constraints (41) make sure that the ship does not become too unbalanced with regards to stability and strength constraints.

$$x_{ij} \in \{0, 1\} \qquad (i, j) \in A \tag{42}$$

$$y_{ij} \geq 0 \qquad (i, j) \in A \tag{43}$$

$$t_i \geq 0 \qquad i \in N \tag{44}$$

$$z_{ik} \in \{0, 1\} \qquad i \in N, k \in N \tag{45}$$

$$w_{ifk} \in \{0, 1\} \qquad i \in N, f \in F, k \in N \tag{46}$$

$$v_{ifk} \geq 0 \qquad i \in N, f \in F, k \in N. \tag{47}$$

Finally, the variable domains are given by constraints (42) - (47).

$$x_{ij} + x_{ji} \leq 1 \qquad (i,j) \in A. \tag{48}$$

Additionally, we also propose the two-node subtour elimination constraints (48), with the intention to reinforce the formulation. Though essential to classic TSPs, subtour elimination constraints are not necessary for our TSPPD-TWDLTA formulation. This is because the time variables t_i will keep track of the service start times of all cargo nodes, and thus forbid the creation of any subtour in a feasible solution. However, constraints (48) are found to be able to reduce run times in our experiments and are therefore added as valid inequalities.

Note that in the above TSPPD-TWDLTA formulation, constraints (20) - (24) serve to connect the binary flow variables x_{ij} and the sequence variables z_{ik}, and the latter (sequencing) decisions may be seen as the input to the subsequent tank allocation constraints (25) - (41). Therefore, by assuming a given route, i.e., fixing the flow variables x_{ij} and hence the sequence variables z_{ik}, we may define a fixed-route TAP through the tank allocation constraints (25) - (41). Without requiring a particular objective function, the fixed-route TAP takes the values of z_{ik} (the given route) as input and aims to find a feasible allocation plan along the given route with loads moving on and off the ship while respecting all allocation constraints such as the stability restrictions.

3 Solution method

In this section we introduce a two-phase heuristic for the TSPPD-TWDLTA, consisting of (1) a dynamic programming procedure used to generate a set of candidate servicing routes, and (2) solving individual fixed-route TAPs. We first describe a full DP-formulation for the TSPPD-TWDLTA in Section 3.1, and a reduced version without tank allocation in Section 3.2. We then introduce a relaxed DP procedure implementing an acceptance level in Section 3.3. The complete heuristic is presented in Section 3.4.

3.1 Dynamic programming formulation

Let us define function $t(S, j, U)$ as the shortest ending time of a path visiting every node of $S \subseteq N$ exactly once and ending at node $j \in S$, where U is the accumulated tank allocation plan when leaving node j. Let $R_j(U')$ be the set of all feasible accumulated tank allocations that can follow U' when the next node to visit is j. Further let l^S be the load onboard the ship after servicing all nodes in S, so $l^S = Q^- + \sum_{i \in S} Q_i$. The function $t(S, j, U)$ can be computed by solving the recursive equations:

$$t(S,j,U) = \min_{(i,j)\in A}\Big\{\ max\{\underline{T_j}, t(S\backslash\{j\},i,U') + T_{ij}\}\ \mid \\ \Big(t(S\backslash\{j\},i,U') + T_{ij} \le \overline{T_j}\ ,\ max\{l^{S\backslash\{j\}}, l^{S\backslash\{j\}} + Q_j\} \le min\{K, D_j\}\Big)\ \Big\}, \tag{49}$$

for all $S \subseteq N$, $j \in S$ and $U \in R_j(U')$. For the transition between states $(S\backslash\{j\}, i, U')$ and (S, j, U), the constraints regarding the latest service time $\overline{T_j}$ of node j, the ship's total capacity K and the draft limit D_j at the terminal associated to node j are to be respected. The recursion formula is initialized by $t(\{0\}, 0, U_0) = 0$, indicating that the ship starts from node 0 (anchorage) at time 0 with an initial tank allocation, described by input parameters W^I_{if} and V^I_{if}. The optimal solution will be given by the shortest route with a state $(N, n+1, U^*_{n+1})$, with $n+1$ being the destination node (also anchorage) and U^*_{n+1} being a feasible accumulated tank allocation plan for the corresponding route.

In the above DP formulation, the accumulated tank allocation U of a state (S, j, U) keeps track of the status (loaded/empty) of all tanks as well as the allocation plan along the path reaching this state. This, however, could result in significant increase in state space compared to the case where tank allocation considerations are not taken into account. This is because that there may be several alternative ways of allocating the cargoes to the tanks for any feasible (partial) path, and we do not know if the path with one of the alternative allocations can be extended to a feasible route at all, until the DP-algorithm is terminated with a feasible accumulated allocation plan. Therefore most dominance tests used for similar time-constrained routing problems, see for instance Desrosiers et al. (1986) and Dumas et al. (1991), usually cannot be applied to this problem since we may have to keep track of several paths ending at the same node j covering the same set of visited nodes S, and also maintain several states even for the same path. For example, consider two states (S, j, U_a) and (S, j, U_b), where $t(S, j, U_a) < t(S, j, U_b)$ which would normally suggest that State A "dominates" State B. In this problem, however, State B may still be more preferable as its allocation plan may turn out to be better when adding the remaining nodes to the path (and there might not exist a feasible allocation plan at all after State A). Therefore, due to the difficulty to reduce the state space, directly implementing this DP-algorithm can be very time consuming.

3.2 The reduced problem without tank allocation

In order to circumvent the difficulty of dealing with the quickly increasing state space in the DP-algorithm brought by the consideration of tank allocation as well as the difficulty to reduce the state space efficiently, we first start with removing the constraints with regards to tank allocation from our model. The problem then becomes a general in-port routing problem without tank allocation, and can be addressed with a simpler version of the DP formulation proposed in Section 3.1 with the states reduced to (S, j). However,

by solving this reduced DP formulation, only one solution (optimal route disregarding tank allocation) is produced when the algorithm terminates due to the label extension and dominance rules performed (Arnesen et al., 2017). The label extension rules check the *post-feasibility* of a partial path (Desrosiers et al., 1986) when attempting to extend the path, and eliminate inappropriate extensions. The dominance rule consists of a simple dominance check, by comparing paths visiting the same set of nodes S and ending at the same node j, and always only keeping the shortest path for every state (S, j) during the DP algorithm. For example, if both partial paths (0, 1, 2, 3, 4, 5) and (0, 1, 2, 4, 3, 5) are feasible, the dominance rule would compare the length of the two paths, and immediately discard the longer one and hence remove all of its potential extensions.

Recall that by assuming a given route the TSPPD-TWDLTA becomes a fixed-route TAP, defined through constraints (25) to (41), which seeks a feasible allocation plan along the given route. Since only one *candidate route* (whose allocation-feasibility is ready to be checked) is produced using the above-mentioned reduced DP method, we obviously cannot guarantee the feasibility of such route with tank allocation taken back into account, and the likelihood of it being feasible can be quite limited. The actual optimal route with a feasible allocation plan may already be discarded as its partial path may be dominated at some stage of the reduced DP-algorithm. One way to get out of this is to turn off dominance checks, and hence keep track of all feasible routes in the reduced problem. We may then check if there exists a feasible allocation for each of the routes by solving individual fixed-route TAPs, and the shortest route thus found (together with its tank allocation) is the optimal solution. This is, however, not a viable approach in all but trivial cases due to the high computational requirements even for the DP-algorithm alone, not to mention the additional computational efforts needed to solve the TAPs.

Another straightforward way to increase the number of candidate routes using the reduced DP method is to turn off dominance checks only for the *completed* routes, which have finished the port call and serviced every node exactly once. Instead of only keeping the shortest completed route as the optimal route as in Arnesen et al. (2017), we can obtain several completed candidate routes, which may increase the likelihood of at least one of them having a feasible allocation plan. Since this approach requires almost no additional computational effort, we call it the ***Original (reduced) DP*** in this paper, which will be further discussed in Section 4.

3.3 A relaxed dynamic programming procedure

We now propose a relaxed dynamic programming procedure in order to generate a reasonable number of potentially good candidate routes without allocation constraints. This procedure is based on relaxing the simple dominance rule used in the *Original DP*, by replacing it with a new dominance rule with an associated acceptance level.

Recall that the reduced DP formulation removes all tank allocation constraints from the full DP formulation (Section 3.1) and obtains states that are independent of U, i.e. reducing the state to (S, j). To be able to maintain multiple paths (in contrast to only keeping the one "dominating" path) for each state, we define label $\mathcal{L}(\hat{S}, j)$ to represent the path following the sequence recorded in $\hat{S}$ that ends at j, where $\hat{S}$ is an ordered set of visited nodes (in contrast to S which contains an unordered set of visited nodes). Let $t^{\mathcal{L}}$ represent the completion time for label $\mathcal{L}$, i.e. the time spent following the path indicated by the label. We further define a non-zero integer number m^{AL} as the *acceptance level* based on which we accept labels at every state in order for further extension in the DP-algorithm. For example, for $m^{AL} = 2$, we accept all labels up to the second shortest completion time for every state; and for $m^{AL} = 3$, we also accept the label(s) with the third shortest completion time in addition to when $m^{AL} = 2$. The idea behind the acceptance level is that, it determines how detailed one would like to sample from the set of all feasible routes without tank allocation; moreover, it favours the routes with less time spent in port as only the labels with shorter completion times are accepted. This is in contrast to the simple dominance rule adopted in the *Original DP*, which only accept the single shortest label for every state.

The higher value we choose for m^{AL}, the more labels are kept for every state and eventually a longer list of candidate routes is produced when the DP procedure is terminated. It is also important to notice that all candidate routes generated by a certain m^{AL} will also be found by using a larger m^{AL} value. When m^{AL} is large enough, we can have an extreme case where *all* feasible routes can be found by the DP procedure, which is equivalent to turning off dominance checks all together. Another extreme case is when m^{AL} is set to 1, where only the label(s) with the shortest completion time for every state (if any) are kept for further extension. This, however, does not necessarily mean that we only accept exactly one label in a particular state if $m^{AL} = 1$ (as the simple dominance rule does in the *Original DP*), as there may be several labels having the same completion time. On the other hand, we also distinguish between *symmetrical* and *asymmetrical* labels when they have the same completion time, so as to avoid certain symmetry in the relaxed DP procedure. We illustrate with two examples in Figure 4. In each example we compare two labels, $\mathcal{L}_X$ and $\mathcal{L}_Y$ where $t^{\mathcal{L}_X} = t^{\mathcal{L}_Y}$, both having visited the same set of nodes (nodes no. $0 \sim 9$) and ending at the same node (node no. 9), and show whether we eliminate one of the two labels or keep both. Note that in both examples, node 0 corresponds to the anchorage; nodes 1,2,3 are located at terminal A; nodes 4,5,6 are located at terminal B; nodes 7,8 are located at terminal C; and node 9 is located at terminal D. In **Example 1** the only difference between the two labels is the order in which the nodes at terminal B are serviced. In this case we consider the two labels to be *symmetrical* and consider it more beneficial to deliver before picking up if there are both pickup and delivery cargoes

at terminal B (since by doing so it is easier to find a feasible tank allocation), and thus only keep one of the two labels; otherwise if the nodes at terminal B are all of the same type, we keep one of the two labels randomly. In **Example 2** however, $\mathcal{L}_X$ and $\mathcal{L}_Y$ differ also in the order of the *terminal* visits and not just the node visiting sequence, in which case we do not consider the two labels symmetrical and therefore keep both labels.

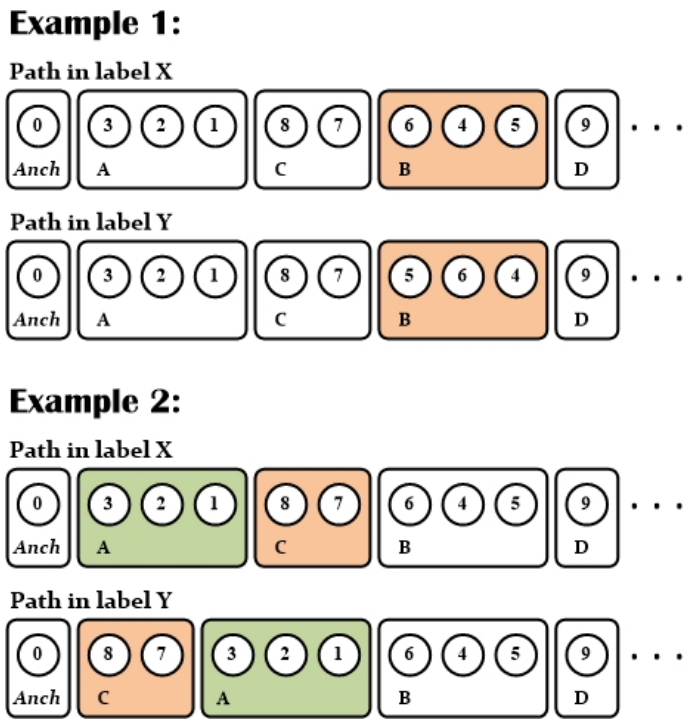


Figure 4: Illustration of two examples comparing $\mathcal{L}_X$ and $\mathcal{L}_Y$, where $t^{\mathcal{L}_X} = t^{\mathcal{L}_Y}$ in both examples. Nodes and terminals are represented by circles and squares, respectively.

After the relaxed DP procedure is carried out, we can obtain a set of candidate routes. It is important to note that, we do not apply the acceptance level on the final state $(N, n+1)$, as every feasible label that has "survived" the DP algorithm and made its way to this state represents a complete route and is therefore added to the final set of candidate routes.

3.4 The two-phase heuristic

The two-phase heuristic can simply be written as follows:

Phase 1:

Generate a set of candidate routes using the relaxed DP procedure with a given m^{AL} value. Sort the routes in ascending order of their completion times and denote by $\mathbf{R}$ the sorted list, with the shortest route $\hat{\mathbf{r}}$ put at the first place of the list.

Phase 2:

Select a route in $\mathbf{R}$ in order and solve the corresponding fixed-route TAP, until a feasible allocation plan is found.

In Phase 1, regardless of the value that m^{AL} is set to, the optimal route $\hat{\mathbf{r}}$ for the problem without tank allocation can always be found by the relaxed DP procedure, which determines the shortest possible completion time of a complete port call. It thus represents the ***lower bound*** of the solutions to the full TSPPD-TWDLTA, as there might not exist a feasible allocation plan for $\hat{\mathbf{r}}$. Also, when selecting an acceptance level, one cannot be sure if any feasible routes exist at all (after taking tank allocation back into account) in $\mathbf{R}$ before actually trying to solve the individual fixed-route TAPs. Therefore, a reasonable acceptance level m^{AL} is to be found by testing on the real problem data, and the results will be shown in Section 4.2.

4 Computational study

The two-phase heuristic is tested on instances based on real data of port calls performed by Odfjell in the Port of Houston. We introduce the test instances in Section 4.1, and present the numerical results in Section 4.2.

The DP procedure (Phase 1 of the heuristic) is coded in Java, while the individual TAPs (Phase 2) are solved using the commercial solver Xpress. As a comparison to the heuristic approach, the arc-flow formulation presented in Section 2 is also implemented and solved with Xpress. All tests have been run on an Intel 3.4 GHz processor with 16 GB of installed memory.

4.1 Test instances

In our experiments we use one fictitious and three real chemical tankers: Bow Fictitious (`FCT` in the following), Bow Fuji - Usuki 19-20 (`FJU` in the following), Bow Faith - Kvaerner (`KVN` in the following) as shown in Figure 2(a), Bow Sea - Poland (`POL` in the following) as shown in Figure 2(b). The Kvaerner and Poland classes are two of the largest ship classes operated by Odfjell, on which they encounter tank allocation problems of probably the highest level of complexity. The `FCT` ship is assumed to carry 7 tanks and have a total capacity of 11,000 deadweight tonnes (*dwt*); the three real ships have a total capacity of 18,047, 33,642 and 36,844 *dwt*, and carry 22, 52 and 40 tanks, respectively.

Based on the instances used in Arnesen et al. (2017), we construct in total 25 test instances for our computational study, in which the largest instances consider 30 cargoes and 11 terminals which are at the high end of the numbers of cargoes and terminals that chemical tankers encounter in real-operation cases. The 25 instances are categorized into six groups varying in problem size (a complete list of all instances can be found

in Table 1). Each instance is named after the associated ship type, number of cargoes, number of terminals and the variant number, e.g. instance `KVN_20_8_2` considers a `KVN` ship with 20 cargoes, 8 terminals and is variant 2. In all instances, the associated tanker enters the Port of Houston carrying a number of delivery cargoes on board (and no other cargoes) and is committed to unload all of them to their respective destination terminals within the port. Before finishing the port call, the tanker also needs to load a number of pickup cargoes from various terminals in the port.

For every test instance, the information regarding the ship and its commitments within the port is given and used as input, such as the moment arms with respect to stability restrictions for all tanks on board, the weights and volumes of the delivery and pickup cargoes, and the associated terminals (with their respective draft limits) and time windows of all relevant cargoes inside the port. In addition, the initial placements of the delivery cargoes when the ship first enters the port are also input to our model (described by parameters W^I_{if} and V^I_{if} in the formulation), as their allocation decisions were made during the ship's last port call. For each instance used in our experiments, given the information of the delivery cargoes (volume V_i and density E_i, etc.) and of the tanks (minimum and maximum volumes $\underline{K}^F_f, \overline{K}^F_f$, moment arms M_{sf}, etc.), we use the following small optimization model to help decide the initial cargo placements on board the ship. Let $w_{if}, i \in N^-, f \in F$ be the binary decision variables that equal to one if *delivery* cargo i is allocated to tank f, and zero otherwise; and let $v_{if}, i \in N^-, f \in F$ be the volume of cargo i allocated to tank f.

$$\min \sum_{f \in F} \sum_{i \in N^- \cap N^F_f} \overline{K}^F_f w_{if} \tag{50}$$

$$\underline{K}^F_f w_{if} \leq v_{if} \leq \overline{K}^F_f w_{if}, \qquad i \in N^-, f \in F, \tag{51}$$

$$\sum_{f \in F^N_i} v_{if} = V_i, \qquad i \in N^-, \tag{52}$$

$$\sum_{i \in N^- \cap N^F_f} w_{if} \leq 1, \qquad f \in F, \tag{53}$$

$$\sum_{j \in N^N_i} \sum_{e \in F^P_{ijf}} w_{je} - |F|(1 - w_{if}) \leq 0, \qquad i \in N^-, f \in F^N_i, \tag{54}$$

$$M^-_s \leq \sum_{f \in F} \sum_{i \in N^- \cap N^F_f} M_{sf} E_i v_{if} \leq M^+_s, \qquad s \in S, \tag{55}$$

$$w_{if} \in \{0, 1\}, \qquad i \in N^-, f \in F, \tag{56}$$

$$v_{if} \geq 0, \qquad i \in N^-, f \in F. \tag{57}$$

Note that the above model may be seen as a *single instant TAP* presented in Hvattum et al. (2009), that only deals with a fixed set of cargoes. Objective function (50) minimizes

the total capacity of the occupied tanks. Constraints (51) to (55) are the single instant versions of Constraints (36),(37), (39), (40) and (41) in the TSPPD-TWDLTA formulation proposed in Section 2. Constraints (51) make sure that the maximum and minimum volume allowed in a tank is not violated if the tank is used. Constraints (52) and (53) ensure that all delivery cargoes are placed on the tanker, and only one cargo can be allocated to any specific tank. Constraints (54) and (55) enforce the incompatibility and stability restrictions, respectively. The variable domains are given by Constraints (56) and (57). The solution obtained after solving the single instant TAP for each test instance is then used as input for initial placement parameters W^I_{if} and V^I_{if} in our problem.

4.2 Numerical results

Recall that in the proposed two-phase heuristic, Phase 1 is to generate a set of candidate routes using a relaxed DP procedure, and Phase 2 solves the individual fixed-route TAPs based on the candidate routes and finds the shortest possible one with a feasible tank allocation plan. To test for the effect of using different values for the acceptance level m^{AL}, we run the relaxed DP algorithm (*R-DP* in short) with $m^{AL} = 1$, $m^{AL} = 2$ and $m^{AL} = 5$, respectively, in order to observe how the number of candidate routes produced changes according to the acceptance level chosen. As a comparison, we also test for the *Original DP* (with the simple dominance rule, see Section 3.2) as the first phase of the heuristic instead of the relaxed version. We report in Table 1 the results of solving all 25 instances under these four settings, "*Original DP*", "*R-DP*, $m^{AL} = 1$", "*R-DP*, $m^{AL} = 2$", and "*R-DP*, $m^{AL} = 5$".

In Table 1 we record, for every instance, the number of candidate routes found (the "#" columns) using each of the four settings in Phase 1 of the heuristic, with a time limit of 50 h; and whether or not there exists at least one route with a feasible tank allocation plan (the "feas.?" columns) in Phase 2, if yes we indicate with a "$\surd$" symbol, and "$\times$" if not. We also record the run times (in seconds if under 1 h, in hours if otherwise) for running Phase 1 and Phase 2, respectively, in each case. Note that, the run time recorded for Phase 2 is the time spent to find the first allocation-feasible route when checking the sorted candidate list, and such route (with the corresponding tank allocation plan), if found, is therefore the best found solution by the heuristic. Also note that for instance `POL_30_11_3`, the *R-DP*, $m^{AL} = 5$ is not completed within 50 h, but is guaranteed to be able to find a feasible route at least as good as the $m^{AL} = 2$ case. This is due to the fact that all candidate routes generated with $m^{AL} = 2$ will also be found with $m^{AL} = 5$. From this table we may clearly see that the proposed relaxed DP procedure (with $m^{AL} = 1$, $m^{AL} = 2$ and $m^{AL} = 5$) is able to generate candidate route sets of larger sizes, rather than the *Original DP* method, which in turn allows the second phase of the heuristic to find a feasible route (considering tank allocation) for more test instances.

Table 1: Results of solving 25 instances using the two-phase heuristic under four different settings, showing the number of candidate routes, and whether or not a feasible route is found therein. Run times are indicated in seconds if under 1 h (3600 s); in hours otherwise.

	Original DP				*R-DP*, $m^{AL} = 1$				*R-DP*, $m^{AL} = 2$				*R-DP*, $m^{AL} = 5$			
	Phase 1		Phase 2		Phase 1		Phase 2		Phase 1		Phase 2		Phase 1		Phase 2	
Instance	#	time	feas.?	time	#	time	feas.?	time	#	time	feas.?	time	#	time	feas.?	time
FCT_8_3_1	3	0.02	√	0.1	5	0.08	√	0.1	11	0.11	√	0.1	44	0.22	√	0.1
FCT_8_3_2	1	0.01	×	0.1	1	0.02	×	0.1	4	0.03	×	0.2	8	0.05	√	0.4
FCT_8_3_3	3	0.02	×	0.2	4	0.04	×	0.2	8	0.08	√	0.3	37	0.13	√	0.4
FCT_8_3_4	1	0.02	√	0.1	1	0.02	√	0.1	2	0.03	√	0.1	8	0.05	√	0.1
FJU_12_4_1	2	0.1	×	0.4	6	0.1	√	0.9	17	0.5	√	1.5	26	0.9	√	1.8
FJU_12_4_2	2	0.1	√	0.3	4	0.2	√	0.3	9	0.4	√	0.3	15	1.2	√	0.3
FJU_12_4_3	2	0.1	√	0.3	6	0.1	√	0.3	11	0.2	√	0.3	20	0.3	√	0.3
FJU_12_4_4	3	0.1	×	0.4	10	0.2	×	1.3	29	0.9	√	2.0	51	5.2	√	2.4
FJU_12_4_5	3	0.1	×	0.5	5	0.1	√	0.8	8	0.1	√	0.9	24	0.2	√	0.9
FJU_12_4_6	3	0.2	×	0.4	7	0.3	×	1.1	15	0.7	√	1.6	60	1.9	√	3.0
FJU_12_4_7	2	0.1	×	0.4	2	0.2	×	0.4	6	0.7	×	0.8	16	2.6	×	2.0
FJU_15_5_1	2	0.5	√	0.8	3	2	√	0.8	6	7.4	√	0.8	16	41.1	√	0.8
FJU_15_5_2	5	0.8	×	4.3	9	5.3	×	8.2	22	24.2	×	20.6	90	183.1	√	78.7
FJU_15_5_3	5	0.6	×	4.0	21	6.5	√	10.2	35	20.1	√	13.4	124	150.2	√	22.9
FJU_15_5_4	5	0.5	√	2.4	7	2.4	√	2.4	13	10.5	√	2.4	40	76.4	√	2.4
FJU_15_5_5	4	0.5	×	4.1	14	2.4	√	9.7	25	10.4	√	12.3	98	60.1	√	19.7
KVN_20_8_1	2	0.3	×	5.7	2	0.3	×	5.7	8	0.4	√	10.3	23	26.2	√	18.9
KVN_20_8_2	1	0.6	√	3.2	3	2.9	√	3.2	6	11.1	√	3.2	6	72.5	√	3.2
KVN_20_8_3	3	7.3	×	7.6	6	55.6	×	15.4	19	107.1	√	36.1	68	1793.7	√	55.1
KVN_25_11_1	1	1.3	×	4.5	13	7.8	×	59.6	19	25.5	√	65.7	42	105.1	√	86.8
KVN_25_11_2	2	3.7	×	8.4	8	13.3	√	19.1	24	35.6	√	27.8	52	338.9	√	43.2
KVN_25_11_3	1	96.2	×	3.7	2	1034.2	×	7.9	4	3190.4	×	14.6	12	9.3 **h**	√	39.3
POL_30_11_1	1	489.6	√	5.2	3	2934.6	√	5.2	6	2.2 **h**	√	5.2	12	26.4 **h**	√	5.2
POL_30_11_2	1	1252.8	×	6.0	3	1.8 **h**	×	19.3	14	4.1 **h**	√	57.5	20	42.9 **h**	√	63.3
POL_30_11_3	3	5.9 **h**	√	6.9	4	21.1 **h**	√	6.9	9	47.4 **h**	√	6.9	≥ 9	>50 **h**	√	-
No. of instances with feasible solution found:																
	9				**14**				**21**				**24**			
No. of instances in total:																
	25				**25**				**25**				**25**			
Success:	**36%**				**56%**				**84%**				**96%**			

Take test instance FJU_12_4_1 for example. After running the *Original DP* algorithm, only two candidate routes are produced which both turn out to be infeasible with tank allocation taken into account in Phase 2. However, the relaxed DP procedure is able to generate 6, 17 and 26 candidate routes with $m^{AL} = 1$, $m^{AL} = 2$ and $m^{AL} = 5$, respectively, and feasible route exists in all three cases. At the bottom of Table 1, we count the total number of instances for which feasible routes are successfully found when using each of the four settings. Out of 25 instances, the four approaches find feasible solutions for 9, 14, 21 and 24 instances, respectively. Observe that, the "success" rate of *Original DP* eventually yielding a feasible solution is quite poor (36%), mainly due to the fact that the number of candidate routes generated is too small. In contrast, the success rate increases to 56%, 84% and 96% using the relaxed DP with $m^{AL} = 1$, $m^{AL} = 2$ and $m^{AL} = 5$, respectively.

It is also clearly visible from Table 1 that the computational time needed to complete the relaxed DP procedure increases with a higher acceptance level m^{AL} chosen. For example, in our largest problem group, i.e. POL_30_11_$*$, the *Original DP* is finished within 6 h for all three instances; and it takes the *R-DP*, $m^{AL} = 2$ up to around 50 h to complete. However, compared to the commercial solver Xpress, the two-phase heuristic is much superior overall by a significant margin. The comparison of solving all test instances using Xpress (time limit set to 15 hours) and using the two-phase heuristic with $m^{AL} = 5$ is shown in Table 2. Note that we do not include the results of solving the problems using the heuristic with $m^{AL} = 1$ or $m^{AL} = 2$ in this table. Although their computation times are much shorter compared with $m^{AL} = 5$ (for those instances to which they do find feasible solutions), their efficiency is less important due to low success rates since finding a feasible solution is essential for the problem.

In Table 2 we record, in the Xpress column, the best found solution, relative gap (between best found solution and lower bound, both found by Xpress, computed as $[BestFoundSolution - LowerBound]/LowerBound$) and run time for every instance. For the heuristic, we record the best found solution, lower bound (corresponds to the shortest route $\hat{\mathbf{r}}$ produced by the relaxed DP procedure, see Section 3.4), relative gap, and run time for every instance. Note that in the Xpress case, "-" indicates that Xpress fails to produce a feasible solution within the 15-h time limit (and runs out of memory for instances of the last three groups); in the heuristic case (i.e. FJU_12_4_7), it indicates that the algorithm is finished, but without producing any feasible solution because none of the route in the candidate list has a feasible tank allocation.

By comparing the performance of Xpress and the two-phase heuristic (with $m^{AL} = 5$) in Table 2, we may observe that the heuristic clearly outperforms Xpress, which by default employs a branch-and-bound algorithm, in almost all test instances (with one exception, FJU_12_4_7). For the instances of the smallest size, FCT_8_3_$*$, Xpress is able to find their optimal solutions using up to around 800 seconds, while the heuristic only takes less than

Table 2: Results of solving all instances using Xpress (with 15-h time limit) and the relaxed DP with $m^{AL} = 5$.

	Xpress				Heuristic (*R-DP*, $m^{AL} = 5$)			
Instance	Best Sol.	L.B.	Gap	Time	Best Sol.	L.B.	Gap	Time
FCT_8_3_1	77.5	77.5	0%	219.8	77.5	77.5	0.0%	0.3
FCT_8_3_2	108.8	108.8	0%	176.4	108.8	103.4	5.2%	0.5
FCT_8_3_3	87.3	87.3	0%	431.2	87.3	80.1	9.0%	0.5
FCT_8_3_4	100.5	100.5	0%	809.7	100.5	100.5	0.0%	0.2
FJU_12_4_1	-	-	-	15 **h**	129.1	125.6	2.8%	2.7
FJU_12_4_2	-	-	-	15 **h**	121.4	121.4	0.0%	1.5
FJU_12_4_3	-	-	-	15 **h**	164.6	164.6	0.0%	0.6
FJU_12_4_4	183.4	110.5	66.0%	15 **h**	140.3	122.1	14.9%	7.6
FJU_12_4_5	-	-	-	15 **h**	158.5	154.6	2.5%	1.1
FJU_12_4_6	181.3	108.4	67.3%	15 **h**	148.3	131.9	12.4%	4.9
FJU_12_4_7	164.8	114.4	44.1%	15 **h**	-	123.4	-	4.6
FJU_15_5_1	-	-	-	15 **h**	157.4	157.4	0.0%	41.9
FJU_15_5_2	-	-	-	15 **h**	169.4	151.9	11.5%	261.8
FJU_15_5_3	-	-	-	15 **h**	154.6	149.4	3.5%	173.1
FJU_15_5_4	-	-	-	15 **h**	157.9	157.9	0.0%	78.8
FJU_15_5_5	-	-	-	15 **h**	163.8	161.1	1.7%	79.8
KVN_20_8_1	-	-	-	-	287.1	284.9	0.8%	45.1
KVN_20_8_2	-	-	-	-	333.3	333.3	0.0%	75.7
KVN_20_8_3	-	-	-	-	294.2	292.9	0.4%	1848.8
KVN_25_11_1	-	-	-	-	404.9	403.4	0.4%	191.9
KVN_25_11_2	-	-	-	-	408.2	406.3	0.5%	382.1
KVN_25_11_3	-	-	-	-	410.8	400.9	2.5%	9.3 **h**
POL_30_11_1	-	-	-	-	464.7	464.7	0.0%	26.4 **h**
POL_30_11_2	-	-	-	-	466.8	464.4	0.5%	42.9 **h**
POL_30_11_3	-	-	-	-	469.9	469.9	0.0%	>50 **h**

one second to find the same solutions. As the size of the problem grows, Xpress starts to give up quite quickly. In the second instance group (with 12 cargoes), Xpress can only find feasible solution for three instances out of seven (and with large gaps). On instances that belong to the next four groups, Xpress fails to produce any feasible solution within 15 h.

In contrast to Xpress, the heuristic (with $m^{AL} = 5$) has managed to find good (optimal in some cases) feasible solutions with relatively small (or zero) gaps for most instances with up to 20 cargoes in a matter of seconds or minutes. When solving the largest test instances with 25 or 30 cargoes (the bottom two groups), which are simply much too large for Xpress, the heuristic needs longer run times but still manages to find a feasible solution in each case. It is important to notice that the heuristic fails to generate a candidate route list that contains a feasible route for instance `FJU_12_4_7`, even with the acceptance level m^{AL} set to 5, which is sufficiently high for all other instances; whereas Xpress finds a feasible solution in this case. However, observe that the heuristic only takes 4.6 seconds to finish, which is a strong indication that we may increase m^{AL} for this instance in particular. In fact, by using $m^{AL} = 8$ and in total 7.2 seconds, the heuristic is able to produce 34 candidate routes and in which finds a feasible solution (135.9), which is a much better solution than the one found by Xpress (164.8) after 15 h.

In real-life operations, the number of cargoes that needs to be considered within a port rarely goes over 25. Therefore we may conclude that the proposed heuristic can be used to deal with most real sized in-port routing problems with tank allocation within quite a short time. We recommend setting the acceptance level m^{AL} to 2 or 5 for the heuristic since they both give a satisfactory success rate in finding (good) feasible solutions (see Table 1). In cases where the heuristic fails to find a feasible solution, we may also try increasing m^{AL} to higher values at the expense of some extra computational efforts.

5 Conclusion

This paper has introduced an in-port routing problem with tank allocation faced by Odfjell, a Norwegian chemical shipping company. This is a combined maritime transportation problem addressing both ship routing and tank allocation simultaneously as a planning component. We model the problem as a Traveling Salesman Problem with Pickups and Deliveries, Time Windows, Draft Limits and Tank Allocation (TSPPD-TWDLTA) and propose a two-phase heuristic to solve it. In the first phase a relaxed dynamic programming (DP) procedure is used to generate a set of candidate routes. Individual fixed-route TAPs are then solved as subproblems in the second phase. A computational study using real data from the shipping company is presented, showing that the proposed heuristic provides good solutions for all test instances in a reasonable amount of time. In real applications,

the method may be used in a decision support system (DSS) in the chemical shipping industry for solving real problems and providing feasible routes before entering a port.

Acknowledgments

The authors acknowledge financial support from the project "Green shipping under uncertainty (GREENSHIPRISK)" partly funded by the Research Council of Norway under grant number 233985. The authors are also grateful to Odfjell that provided real data for the test instances used in the computational study.

References

Arnesen, M. J., Gjestvang, M., Wang, X., Fagerholt, K., Thun, K., and Rakke, 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.

Desrosiers, J., Dumas, Y., and Soumis, F. (1986). A dynamic programming solution of the large-scale single-vehicle dial-a-ride problem with time windows. *American Journal of Mathematical and Management Sciences*, 6(3-4):301–325.

Dumas, Y., Desrosiers, J., and Soumis, F. (1991). The pickup and delivery problem with time windows. *European Journal of Operational Research*, 54(1):7–22.

Fagerholt, K. and Christiansen, M. (2000). A combined ship scheduling and allocation problem. *Journal of the Operational Research Society*, pages 834–842.

Hvattum, L. M., Fagerholt, K., and Armentano, V. A. (2009). Tank allocation problems in maritime bulk shipping. *Computers & Operations Research*, 36(11):3051–3060.

Kang, J.-G. and Kim, Y.-D. (2002). Stowage planning in maritime container transportation. *Journal of the Operational Research Society*, pages 415–426.

Odfjell (2016). Odfjell tankers fleet-list. [http://www.odfjell.com/Tankers/Pages/Odfjell-Fleet-List.aspx; accessed 1-August-2016].

Øvstebø, B. O., Hvattum, L. M., and Fagerholt, K. (2011a). Optimization of stowage plans for RoRo ships. *Computers & Operations Research*, 38(10):1425–1434.

Øvstebø, B. O., Hvattum, L. M., and Fagerholt, K. (2011b). Routing and scheduling of RoRo ships with stowage constraints. *Transportation Research Part C: Emerging Technologies*, 19(6):1225–1242.

Rakke, J. G., Christiansen, M., Fagerholt, K., and Laporte, G. (2012). The traveling salesman problem with draft limits. *Computers & Operations Research*, 39(9):2161–2167.

Vilhelmsen, C., Larsen, J., and Lusby, R. (2016). A heuristic and hybrid method for the tank allocation problem in maritime bulk shipping. *4OR*, 14(4):417–444.

Vouros, G., Panayiotopoulos, T., and Spyropoulos, C. (1996). A framework for developing expert loading systems for product carriers. *Expert Systems with Applications*, 10(1):113–126.

Wilson, I. and Roach, P. (2000). Container stowage planning: a methodology for generating computerised solutions. *Journal of the Operational Research Society*, 51(11):1248–1255.