Transportation Research Part E 123 (2019) 200–220

Contents lists available at ScienceDirect

Transportation Research Part E

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

A rolling horizon heuristic for the stochastic cargo mix problem

Jonas Christensen[a,*], Alan Erera[b], Dario Pacino[a]

[a] Technical University of Denmark, Produktionstorvet 424, 2800 Kgs. Lyngby, Denmark
[b] Georgia Institute of Technology, H. Milton Stewart School of Industrial and Systems Engineering, Atlanta, USA

ARTICLE INFO

Keywords:
Liner shipping
Stochastic optimisation
Rolling horizon
Matheuristic
Cargo composition
Maritime logistic
Block stowage
Stowage planning

ABSTRACT

This paper presents the stochastic cargo-mix problem, which aims at analysing the cargo composition needed for a liner vessel to maximise its revenue on a given service. The unreliability with respect to the demand forecast is included by considering the cargo-flows as being stochastic instead of deterministic. We also take into account accepted bookings, draft, stability and capacity constraints. A compact formulation of the problem is shown to be too complex to solve industrially sized instances. Instead, a rolling horizon matheuristic is presented, and the computational results show that it can achieve high-quality results in reasonable time.

1. Introduction

Aside from a few years of financial crisis, the liner shipping industry has had a continuous growth. The growing demand has resulted in a fierce competition to deliver the best product concerning efficiency, reliability, and most importantly cost. As a result, shipping rates are historically low, making it vital for the carriers to utilise their vessels as efficiently as possible. In recent years, carriers have been building bigger and bigger vessels to follow demand trends, but also to achieve economies of scale.

While academic focus on vessel intake maximisation is relatively new, it is nothing new in the shipping industry. Container vessels are delivered with a theoretical nominal capacity. Only if the weight distribution is perfect, the full nominal capacity can be reached, which hardly ever happens. With the increasing size of the vessels, a small decrease in utilisation results in hundreds of containers having to be dropped. Stowage coordinators are responsible for planning the cargo and finding a load configuration (*stowage plan*) that suits the cargo to be loaded at the current port, while also making sure the vessel can be utilised to its maximum in future ports. The unreliability with respect to the demand forecast in the industry further complicates this problem. For the shippers, there is no fee for booking shipments, and they will only pay for a container transport once it has been undertaken by the liner. Thus, a booking does not mean the containers will ever arrive in time. Therefore, it is complicated for the stowage coordinators to make a good plan as the high unreliability is not likely to change without radically changing the cost structure.

The focus of our work is the analysis of vessels' cargo mix (*the cargo mix problem*) and, in particular finding a cargo composition needed for a vessel to maximise its revenue on a given service. We include the unreliability of the demand and model confirmed bookings. We focus on out-of-region demands (e.g. from Asia to Europe), where we consider load and discharge of cargo in the current region and only discharge in the destination region. This corresponds to optimising the revenue from an inter-regional leg while still considering the revenue obtained by shipping containers within the region. Delgado (2013) shows that a cargo-mix analysis based on simple capacity constraints overestimates the revenue generated by operating a vessel. Therefore, it is important to include additional features and limits to estimate the capacity and revenue correctly. Additionally, we enforce that the stowage plan

[*] Corresponding author.
E-mail addresses: jomc@dtu.dk (J. Christensen), alan.erera@isye.gatech.edu (A. Erera), darpa@dtu.dk (D. Pacino).

https://doi.org/10.1016/j.tre.2018.10.010
Received 15 May 2018; Received in revised form 5 September 2018; Accepted 23 October 2018

adheres to a block stowage strategy used within the industry. This corresponds to a logical partitioning of the vessel into blocks, and by enforcing the block strategy each block is only allowed to stow containers that have the same discharge port. This way of stowing containers is aimed at improving operations at ports since it makes it possible to perform e.g. dual cycling (where load and discharge operations are no longer sequential). Examples of block stowage are found in the stowage planning literature in e.g. Ambrosino et al. (2015a,b).

The proposed model can have multiple applications, e.g., driving rate prices and improving fleet composition. However, we see greater potential in using the model as an analytic tool. We envision its true potential in the analysis of different what-if scenarios. To allow the execution of multiple analyses, fast solution methods are sought. We will thus put a lower priority with respect to optimality and prioritise an adaptable, simple and fast solution method.

In Christensen and Pacino (2017) a deterministic version of the cargo mix problem is described, and multiple matheuristics based on the same idea are compared. The problem studied in Christensen and Pacino (2017) assumes perfect information and is aimed at identifying an optimal cargo-mix for a specific vessel. Such cargo-mix can be used to e.g. evaluate which vessel is better suited for a specific service. In this paper, we present a more operational version of the cargo-mix problem, where the current out-of-region configuration of a vessel is studied. Essentially, we aim at utilising the vessel as best as possible before the long-haul journey.

The contributions of this paper are threefold. First, we present a more operational version of the cargo-mix problem that is able to analyse what-if scenarios and present a novel multi-stage stochastic formulation. Second, among the major container liner shipping problems studied in the literature (e.g. stowage planning, service network design, cargo flow optimisation and empty repositioning), this is, to the best of the authors' knowledge, the first work that includes stochastic cargo forecast elements. Third, an efficient 2-phase framework for the proposed multi-stage stochastic program has been implemented and tested. The method is inspired by the multi-phase approach from Christensen and Pacino (2017), which has been extended and adapted to address this problem. Moreover, a rolling horizon heuristic has been used for the second phase in order to overcome the problem intractability introduced by the stochastic program.

The remainder of the paper is organised as follows. Section 2 presents background knowledge of vessel architecture and the industry as a whole. In Section 3 relevant existing literature is reviewed. Section 4 gives a detailed description of the problem and presents a compact formulation of the problem. Section 5 describes the matheuristic approach. Section 6 describes the data generated and used for this study, and Section 7 presents the results for the matheuristic and compares it with the result of the compact model. At last, Section 8 contains the final remarks and conclusions.

2. Background

Liner shipping is the service of long-haul transportation of goods by using ocean-going vessels. The vessels used are of high capacity and operate on a fixed route with published schedules. The high capacity of the vessels helps to keep costs down, and the liner shipping industry is the cheapest and most energy-efficient form of international transportation.

The cargo to be transported is packed in standardised containers, which are then loaded on vessels. Most containers carried on liner vessels are 8 feet wide, 8′6″ high and *20′*, *40′* or *45′* long. *High-cube* containers also exist, which are 9′6″ high. Perishable goods are packed in so-called *reefer* containers. These are refrigerated containers with an integrated cooling unit to keep the cargo cool. The cooling unit gets power from the vessel, and thus reefer containers can only be stowed where an electrical outlet is available. Additionally, there are dangerous goods containers, which must adhere to a certain safety standard describing the segregation rules for different kind of dangerous material (e.g. toxic, nuclear or flammable).

Fig. 1 shows the layout of a vessel. A vessel is divided into *bays*, and for each bay there is an *on-deck* and *below-deck* part which are physically separated by a *hatch cover*. Hatch covers are flat and leak-proof structures that can be removed when loading and unloading containers. The bays are further divided into a number of *stacks* and *cells*. A cell can hold a single 40' container or two 20' containers, and the stack refers to a cell's longitudinal/transversal position on the vessel. Fig. 2 outlines the design of a bay. First, Fig. 2a shows a transversal section of a bay, where the numbers denote the blocks. Secondly, Fig. 2b shows the layout of a stack within a bay.

A cell consists of two *slots*; an *aft* (towards the stern) and a *fore* (towards the bow) slot. For each stack, there are weight and height limits on the containers stowed. Furthermore, there are specific stacking rules, describing how the containers can be stowed. E.g. 20′ containers cannot be stowed above 40' containers, and generally, the weight of the containers must decrease upwards in the stacks on deck.

The main cost components in liner shipping are fuel costs and costs associated with port stays. The costs at port depend on the

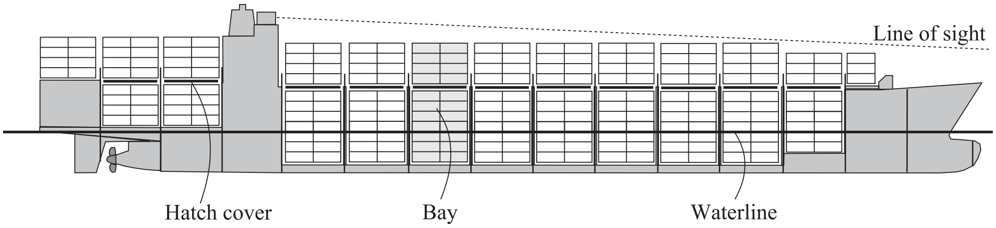
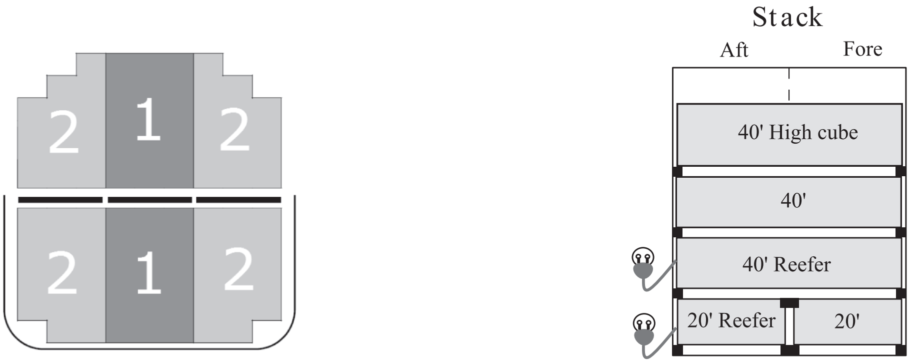


Fig. 1. Example figure of a container vessel.



(a) A bay in the vessel

(b) The layout of a stack

Fig. 2. Example figure showing the design of a bay, and a stack.

time spent at the port, and thus decreasing the time spent at the port will most often decrease the associated costs as well. To reduce the time at port, it is important to avoid *overstowage*, and especially *hatch overstowage*. Overstowage occurs when a container is stowed below a container destined for a later port. Hence, to unload the desired container, you will need to move the overstowing container first. Hatch overstowage happens when a container on-deck is overstowing a container below deck, in which case all containers on-deck will need to be moved in order to remove the hatch cover. These excess crane moves are costly and take time, and the number of these moves should, therefore, be kept at a minimum. The benefit of decreasing the time at port is twofold; the port costs are minimised, and if less time is spent in port, the vessel can sail slower to the next port and thus save fuel.

The profit comes from the transported containers, and naturally, carriers want to maximise the capacity usage to increase revenue. Vessel's capacities are measured in the number of *Twenty-foot Equivalent Units* (TEUs) the vessel can carry under perfect conditions, i.e. the number of slots on the vessel. A 40′ foot standard container and 40′ high-cube container are 2 TEU or 1 *FEU* (*Forty-foot Equivalent Unit*).

To facilitate better planning and ease port handling operations, vessels are divided into *blocks* (a virtual grouping of stacks). In the industry, different companies have different definitions of a block. In accordance with our industry collaborator, we define a block such that stacks below and above the same hatch-cover belong to the same block. When stowing the ship, all containers in a block must have the same discharge port. Therefore, overstowage and hatch-overstowage cannot happen.

Before a vessel leaves port, the captain of the ship must deem the vessel *seaworthy*. Besides ensuring that the vessel is not defective or undermanned, this also entails that the static stability is correct and all stress forces are within limits. The weight of the cargo causes shearing and bending stresses over the vessel structure, which must be within limit at certain calculation points. Ensuring the vessel is stable also requires the three coordinates of the centre of gravity (longitudinal, vertical and transversal) to be within their limits. The *aft draft* is the distance between the waterline and the bottom of the hull at the stern of the ship, and the *fore draft* is at the bow. The trim is defined as the difference between the aft draft and the fore draft. The trim, fore draft, and aft draft must all be within limits.

The outline of a vessel is described in a document called the *vessel profile*, which specifies all the weight and capacity limits along with the *hydrostatic table*. Using the hydrostatic table one can calculate the draft, centre of gravity and metacentre. For a more detailed description of the stability calculation and requirements see Delgado (2013).

3. Literature review

The existing literature on the liner shipping cargo mix problem is limited. The problem was formally introduced in the PhD thesis of Delgado (2013). Here a mixed integer programming model is presented, and the multi-port version is shown not to be scalable. To achieve scalability, the problem is decomposed in a similar way as to what is suggested in earlier stowage planning work (see Pacino et al. (2011)). The work of Christensen and Pacino (2017) extends the liner shipping cargo mix problem by including the concept of block stowage, draft restrictions and restricts the number of containers that can be selected. The inclusion of the block stowage strategy makes the MIP model proposed in Delgado (2013) intractable for industrial-sized instances. Instead, a novel matheuristic is developed. Tested on industrially sized instances, the matheuristic is shown to be scalable and produce high-quality solutions in a matter of seconds. A revenue model for the short-sea shipping service is presented in Feng and Chang (2008). This model disregards most aspects of what it means for a vessel to be seaworthy and only considers the TEU and weight capacity. In Delgado (2013) it is shown that, due to inaccuracies wrt. stability considerations, the model in average overestimates the revenue by 8%. To get an accurate estimate of the revenue we, therefore, believe it is important to consider the seaworthiness of the vessel with care. Our work differs from previous work by also including the unreliability wrt. the demand forecast within the industry. This is done by considering the cargo flow as being stochastic instead of deterministic. Doing so, a list of ports are considered instead of a circular service, and the vessel is assumed to already be loaded with some containers.

The cargo-mix problem can be seen as a generalisation of a stowage planning problem. Given the limited amount of existing literature on the cargo mix problem, it is, therefore, relevant to introduce relevant literature related to stowage planning. The main difference between the cargo mix problem and stowage planning is that the cargo mix problem considers a set (or in this case; distribution) of expected containers that is possible to be loaded in the visited ports. Whereas in stowage planning a list of containers has already been selected, and the main decision is where these containers should be stowed on the vessel. The stowage planning literature can be divided into two main groups; single model approaches and multi-phase methods. The first fullly detailed mathematical model is presented in Botter and Brinati (1992). Additional approaches based on Mixed Integer programming formulations are found in the work of Ambrosino et al. (2004) and Li et al. (2008). These models are unsuitable for practical use due to scalability issues from the large number of variables and constraints. Heuristic single model approaches include the placement heuristic by Aslidis (1984) and the *Suspensory Heuristic* by Avriel et al. (1998). The suspensory heuristic only focuses on container stacking and does not include stability. Dubrovsky et al. (2002) obtain results similar to the suspensory heuristic, but has the added flexibility of incorporating simple stability constraints. Ding and Chou (2015) describes a state-of-the-art heuristic that improves the results from the suspensory heuristic. In Zhang and Lee (2016) and Imai et al. (2006) the stowage planning problem is considered as a multi-objective optimisation problem, with both stability objectives and objectives on the number of container rehandles. The placement heuristic in Low et al. (2009) is based on the work practice of stowage coordinators, and the similarities to packing problems are studied in Sciomachen and Tanfani (2003), where a 3D-bin packing heuristic is used. Constraint programming approaches for stowage planning includes the work of Ambrosino and Sciomachen (1998).

The first multi-phase method was proposed in Botter and Brinati (1992). However, it was too computationally expensive. Instead, the problem was solved by a branch-and-bound search. The first multi-phase method presenting promising results was Wilson and Roach (2000). In this work, first containers are assigned to general areas on the ship following high-level capacity constraints, known as the master bay planning problem. Hereafter the specific containers are assigned a specific slot (slot planning problem). Kang and Kim (2002), Ambrosino et al. (2010), Pacino et al. (2011) all rely on a similar approach. A three-step heuristic is introduced in Ambrosino et al. (2009) in which the left and right, and bow and stern weight differences are kept within a tolerance. The slot planning problem has also received attention. The GRASP approach presented in Parreno et al. (2016) is the current state-of-the-art. The constraint programming methods of Delgado et al. (2012) and Pacino and Jensen (2012) must also be mentioned in this context.

The problem studied in this paper resembles the deterministic cargo-mix problem considered in Christensen and Pacino (2017), but some key features make this work more than just a stochastic extension. The stochastic cargo-mix, as presented in this paper, is a more operational problem, where the existing bookings and future bookings are taken into account. Therefore, previous solution methods cannot be used directly. However, some of the challenges faced when solving the deterministic cargo-mix problem will also carry over when solving this stochastic cargo-mix problem. In Christensen and Pacino (2017) it is shown that the deterministic cargo mix problem is too complex to be solved using standard mixed integer programming methods. Based on these results we do not expect to be able to solve the stochastic version to optimality within a time that makes it applicable to the industry. It is therefore relevant to look at specialised heuristic methods for stochastic problems.

Large stochastic MIPs are computationally hard to solve due to a large number of scenarios. One way to handle this is to use a rolling horizon heuristic (RHH). Such a planning procedure only considers a portion of the entire planning horizon at a time, solves this reduced problem and fixes parts of the solution. Rolling horizons schemes have been applied to several problems within operations management (see Chand et al. (2002) for an extensive review). Due to the myopic nature of this procedure, it is highly appropriate when only limited information is available. However, it can also be used to reduce the problem size and thereby lessen the computational effort to solve the problem.

The literature on rolling horizon heuristics within a maritime setting is limited. Rakke et al. (2011) describes a rolling horizon heuristic for a maritime routing and inventory management problem. A reduced version of the mathematical model is used as an improvement heuristic where the feasible solution found by the RHH is used to reduce the number of variables. The results show that the RHH outperforms the MIP model and produces quality solutions in a relatively short amount of time. Additionally, the improvement heuristic is shown to improve the result by 2–10% for the datasets tested. Bredstrom and Rönnqvist (2006) apply a rolling horizon heuristic to a combined supply chain and ship routing problem. Real-world industrial-sized data is used, and the heuristic is shown to outperform the MIP model.

4. The Stochastic Cargo Mix Problem with Block Stowage (SCMPBS)

Given a vessel, a string of ports, an initial configuration of the vessel, a list of accepted bookings and a distribution of the cargo flow (i.e. origin-destination demand matrix), the Stochastic Cargo Mix Problem with Block Stowage (SCMPBS) aims at optimising the expected revenue of the cargo loaded. A high degree of accuracy is imposed wrt. stability constraints to ensure the vessel is seaworthy. Furthermore, the loading must comply with the block stowage requirement, ensuring that all containers in a block have the same discharge port. We will not assume the vessel is initially empty, and the initial configuration of the vessel will describe the cargo already loaded, as well as their destination.

The string of ports is divided in two; a set of visited ports, for which unload and loading operations will be planned, and a set of demand ports for which only unload operations will be scheduled. This separation corresponds to a situation where a vessel is in the Far East, and the carrier wants to optimise the revenue of the containers to be brought to Europe. When the vessel arrives in Europe, the demand is known more accurately, and the problem can be solved once again taking the current vessel configuration into consideration.

The aim is to determine the optimal cargo-mix and not to make a fully feasible stowage plan. In Pacino et al. (2011) it is shown that working on an aggregated level gives a good approximation to the full problem. Therefore, the problem is decomposed similarly

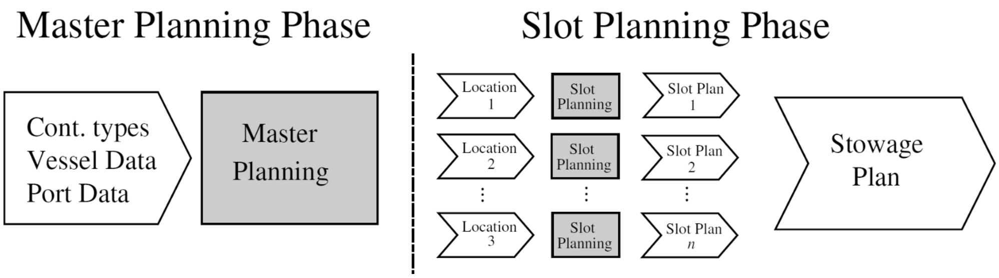


Fig. 3. The two-phase heuristic decomposition.

to earlier work (Wilson and Roach, 2000; Kang and Kim, 2002; Ambrosino et al., 2010; Pacino et al., 2011; Delgado, 2013), which also helps to ease the computational effort needed to solve the problem. Instead of assigning specific containers to specific slots on the vessel, containers are grouped in *container types* and assigned to *blocks*. Each container type represents a number of containers with the same properties wrt. weight, height, length and reefer capabilities. The blocks correspond to a logical partitioning of the vessel into non-overlapping sections. In the decomposition, the planning is split into two; first, a master planning phase where container types are assigned to blocks while satisfying high-level capacity constraints. Hereafter, a slot planning phase, consider each block sequentially and assign a specific slot for each of the containers assigned in the master planning phase. This decomposition is illustrated in Fig. 3. Delgado (2013) showed that the master planning phase gives an accurate estimation of the achievable capacity utilisation. As this is the goal of this study, we disregard the slot planning phase.

For the container types, we consider the most standard containers: 20′ dry and reefer containers, 40′ dry and reefer containers both normal height and high-cube. All with a set of weight classes, ranging from 3 tonnes (empty) to 27 tonnes (full).

The described problem can be modelled as a multi-stage stochastic programming problem. The decision process is thus as follows;

Observe demand at port 1 → Decide block assignment for port 1 → Find containers to load at port 1→
Observe demand at port 2 → Decide block assignment for port 2 → Find containers to load at port 2 → ...→
Observe demand at port n → Decide block assignment for port n → Find containers to load at port n

When determining the block assignment at port *p*, only the blocks being emptied at port *p* will need a new destination port assigned. The rest of the blocks will have to keep the destination that has already been assigned in order not to move the containers. The decision process can be illustrated by a scenario tree, as shown in Fig. 4, where Low, Medium and High describe three possible realisations of the demand at port *a*. A scenario is defined as a path from the root of the tree to one of the leaves. The nodes visited by each path corresponds to a realisation of the random parameters in the model. Let $\mathcal{S}$ be the set of scenarios, and $\mathcal{N}$ the set of nodes in the scenario tree, each one of these nodes $n \in \mathcal{N}$ corresponds to a vector of random parameters with a particular history up to that node.

Let $\pi_{nn'}$ be the transitional conditional probability of going from node *n* to node *n*′, and furthermore let Π_n be the unconditional probability of node *n*. Π_n is computed by multiplying the conditional probabilities of nodes positioned on the path starting at the root node and ending at node *n*.

$$\Pi_n = \prod_{\langle n_1,n_2 \rangle \in path(n)} \pi_{n_1,n_2}$$

Here $path(n)$ describes the path from the root to node *n*. The scenario probabilities Π_s can be computed similarly.

Along with the sets $\mathcal{S}$ and $\mathcal{N}$, define C as the set of container types. A container type, $c \in C$, is defined by the dimensions and

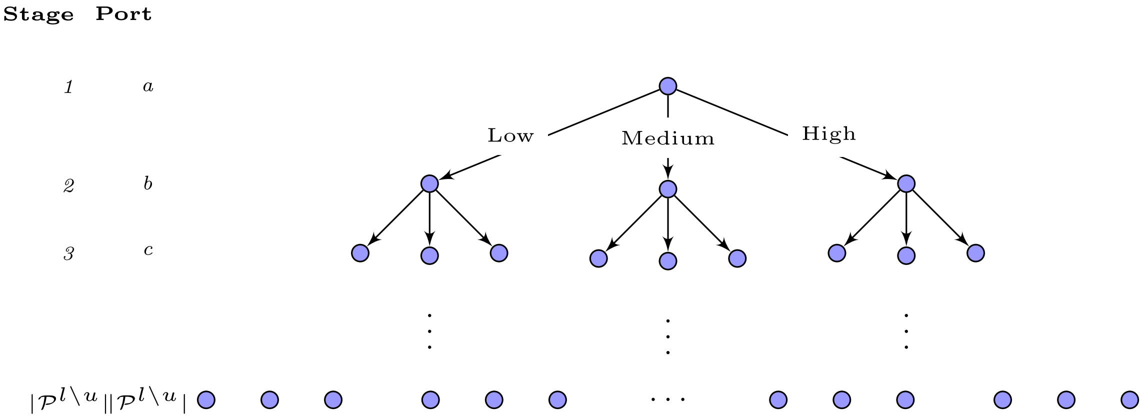


Fig. 4. Scenario tree illustrating the multi-stage decision process.

properties of the containers. The parameters $\Gamma^c,\ \Phi^c,\ \nu^c$ describe the TEU coefficient (1 for a 20′ container, 2 for a 40′ container), volume and weight of container type $c \in C$, respectively. The volume is needed for the modelling of the high-cube containers, which need further restrictions than a simple capacity constraint. Also, let the set $\mathcal{B}$ be the set of blocks. Let $\mathcal{P}$ be the full set of ports, $\mathcal{P} = \mathcal{P}^{l\setminus u} \cup \mathcal{P}^u$ where $\mathcal{P}^{l\setminus u}$ is the set for which both loading and unloading operations will be planned, and $\mathcal{P}^u$ the set of ports where only unloading operations are planned. The container demand is described by the set $\mathcal{T}$ and the parameter a_n^{dc}. The set $\mathcal{T}$ is the set of transports. A transport describes an origin–destination pair and a_n^{dc} is the number of containers available of type $c \in C$ during transport $t = \langle p^n, d\rangle$ in node $n \in \mathcal{N}$. Here p^n is the port associated with node $n \in \mathcal{N}$. The set $\mathcal{T}_p^{ON}$ is the set of transports where port p is visited between the origin and destination of the transport, defined as follows.

$$\mathcal{T}_p^{ON} = \{t \in \mathcal{T} | o^t \leqslant p < d^t\} \qquad \forall\, p \in \mathcal{P}$$

Here o^t and d^t are the origin and destination ports of transport t.

The main decision variable is $y_{bs}^{tc} \in \mathbb{R}^+$, defining the (possibly fractional) number of containers of type $c \in C$ of transport $t \in \mathcal{T}$ to be stowed in block $b \in \mathcal{B}$ in scenario $s \in \mathcal{S}$. Naturally, the container load variables must be an integer, however, as earlier mentioned this work mostly concerns revenue estimation, and the aim is thus not to generate a fully feasible stowage plan. Christensen and Pacino (2017) show that relaxing the integrality requirement of the container load variables gives an accurate estimation of the revenue, and furthermore describes how the resulting solution can be converted into a solution with integer container load variables.

Let $w_{bps} \in \mathbb{R}^+$ be a variable denoting the weight stowed in block $b \in \mathcal{B}$ at port $p \in \mathcal{P}^{l\setminus u}$ in scenario $s \in \mathcal{S}$. To enforce the block stowage requirement, we introduce the variable σ_{bn}^d, defined as follows

$$\sigma_{bn}^d = \begin{cases} 1 & \text{If block } b \in \mathcal{B} \text{ is assigned destination } d \in \mathcal{P} \text{ in node } n \in \mathcal{N} \\ 0 & \text{Otherwise.} \end{cases}$$

Let u^{tc} describe the accepted bookings i.e. the number of accepted containers of type $c \in C$ for transport $t \in \mathcal{T}$. From the initial configuration of the vessel we define two parameters, Θ_{bp}^c and Ω_{bp}^d. Θ_{bp}^c describe the cargo loaded, i.e number of containers of type $c \in C$ initially loaded that still occupy block $b \in \mathcal{B}$ at port $p \in \mathcal{P}$, whereas Ω_{bp}^d describe the partial block assignment imposed by the initial configuration. That is;

$$\Omega_{bp}^d = \begin{cases} 1 & \text{If the initial configuration imposes that block } b \in \mathcal{B} \text{ is assigned destination } d \in \mathcal{P} \text{ at port } p \in \mathcal{P}^{l\setminus u} \\ 0 & \text{Otherwise.} \end{cases}$$

Let $\mathbf{y}_s \in \mathbb{N}^{|\mathcal{B}|\cdot|\mathcal{T}|\cdot|C|}$ be the vector of y_{bs}^{tc} variables for scenario $s \in \mathcal{S}$. Then define the polyhedron $\mathcal{Y}$ as the loading configurations that satisfy the capacity constraints for the variables $\mathbf{y}_s$.

$$\mathcal{Y} = \{s \in \mathcal{S}, \mathbf{y}_s \in \mathbb{R}^{|\mathcal{B}|\cdot|\mathcal{T}|\cdot|C|}\ s.\ t\ \ (A.1)\text{–}(A.5)\}$$

In essence, the vector $\mathbf{y}_s$ describe a full solution for a scenario, and the polyhedron $\mathcal{Y}$ is a set of constraints ensuring the solution is feasible wrt. the capacity constraints. Similarly, define $\mathbf{w}_s \in \mathbb{R}^{|\mathcal{B}|\cdot|\mathcal{P}^{l\setminus u}|}$ as the vector of w_{bps} variables for scenario $s \in \mathcal{S}$. Then define the polyhedron $\mathcal{W}$ as follows

$$\mathcal{W} = \{s \in \mathcal{S}, \mathbf{w}_s \in \mathbb{R}^{|\mathcal{B}|\cdot|\mathcal{P}^{l\setminus u}|}\ s.\ t\ \ (B.1)\text{–}(B.20)\}$$

Here (B.1)–(B.20) describe the feasible weight allocations that ensure the seaworthiness of the vessel. Here the polyhedron $\mathcal{Y}$ describes the feasibility regarding capacity, and the polyhedron $\mathcal{W}$ describes the feasibility concerning the seaworthiness of the vessel. We only ensure the feasibility wrt. seaworthiness for the load/unload ports (the set $\mathcal{P}^{l\setminus u}$); for the remaining ports we assume containers can be loaded ensuring a seaworthy vessel. The polyhedron $\mathcal{Y}$ is formally described in Appendix A, and the stability constraints are described in Christensen and Pacino (2017). For the reader's convenience, a description of the stability constraints can also be found in Section B.

Below, all the sets, variables and parameters are summarised, and additional sets and parameters are introduced.

Sets:

$\mathcal{S}$	Set of scenarios
$\mathcal{N}$	Set of nodes
$\mathcal{N}(s, o, d)$	Set of nodes for scenario s between port o and d (including o, excluding d) $\mathcal{N}(s, o, d) = \{n \in \mathcal{N} \vert n \in s, o \leq p^n \wedge p^n < d\}$
$\mathcal{S}(n)$	Set of scenarios having node $n \in \mathcal{N}$ on the path. ($\mathcal{S}(n) = \{s \in \mathcal{S} \vert n \in s\}$)
$\mathcal{P}$	Set of ports
$\mathcal{P}^{l\setminus u} \subset \mathcal{P}$	Set of load/unload ports
$\mathcal{P}^u \subset \mathcal{P}$	Set of demand ports (only unload operations will be planned)
$\mathcal{T}$	Set of transports
$\mathcal{T}_p^{ON}$	Set of transports that visits port $p \in \mathcal{P}$
C	Set of container types
$\mathcal{B}$	Set of blocks.

Variables:

$y_{bs}^{tc} \in \mathbb{R}^+$ Number of containers of type $c \in \mathcal{C}$ of transport $t \in \mathcal{T}$ to be stowed in block $b \in \mathcal{B}$ in scenario $s \in \mathcal{S}$.

$w_{bps} \in \mathbb{R}^+$ Weight of the containers stowed in block $b \in \mathcal{B}$ at port $p \in \mathcal{P}^{l \backslash u}$. in scenario $s \in \mathcal{S}$.

$\sigma_{bn}^{d} \in \{0, 1\}$ Variable denoting whether or not block $b \in \mathcal{B}$ can only contain containers with destination $d \in \mathcal{P}$ in node $n \in \mathcal{N}$

Parameters:

$\Pi_s \in [0;1]$ Probability for scenario $s \in \mathcal{S}$.

$f^{tc} \in \mathbb{R}^+$ The value of container type $c \in \mathcal{C}$ of transport $t \in \mathcal{T}$.

$p^n \in \mathcal{P}$ The port associated with node $n \in \mathcal{N}$.

$v^c \in \mathbb{R}^+$ Weight of container type $c \in \mathcal{C}$

$\theta_{bp}^{c} \in \mathbb{N}$ Number of containers of type $c \in \mathcal{C}$ initially loaded that still occypy block $b \in \mathcal{B}$ at port $p \in \mathcal{P}$

$\Omega_{bp}^{d} \in \{0, 1\}$ Parameter denoting if block $b \in \mathcal{B}$ is enforced to only contain containers with destination $d \in \mathcal{P}$ at port $p \in \mathcal{P}^{l \backslash u}$.

$a_n^{dc} \in \mathbb{N}$ Number of available containers of type $c \in \mathcal{C}$ for transport $t = \langle p^n, d \rangle$ at node $n \in \mathcal{N}$.

$u^{tc} \in \mathbb{N}$ Number of already accepted containers of type $c \in \mathcal{C}$ for transport $t \in \mathcal{T}$.

$q_b \in \mathbb{R}^+$ Weight capacity of block $b \in \mathcal{B}$

With this we can model the problem as follows

$$\text{Max Z} = \sum_{s \in \mathcal{S}} \sum_{b \in \mathcal{B}} \sum_{t \in \mathcal{T}} \sum_{c \in \mathcal{C}} \Pi_s f^{tc} y_{bs}^{tc} \tag{1}$$

Subject to:

$$\mathbf{w}_s \in \mathcal{W} \qquad \forall\, s \in \mathcal{S} \tag{2}$$

$$\mathbf{y}_s \in \mathcal{Y} \qquad \forall\, s \in \mathcal{S} \tag{3}$$

$$y_{bs}^{tc} = y_{bs'}^{tc} \qquad \forall\, s \in \mathcal{S},\, n \in s,\, s' \in \mathcal{S}(n),\, b \in \mathcal{B},\, d \in \mathcal{P},\, t = \langle p^n, d \rangle,\, c \in \mathcal{C} \tag{4}$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{C}} v^c y_{bs}^{tc} + \sum_{c \in \mathcal{C}} v^c \theta_{bp}^{c} \leqslant w_{bps} \qquad \forall\, s \in \mathcal{S},\, b \in \mathcal{B},\, p \in \mathcal{P}^{l \backslash u} \tag{5}$$

$$\sigma_{bn}^{d} \geqslant \Omega_{bp^n}^{d} \qquad \forall\, n \in \mathcal{N},\, b \in \mathcal{B},\, d \in \mathcal{P} \tag{6}$$

$$\sum_{d \in \mathcal{P}} \sigma_{bn}^{d} = 1 \qquad \forall\, n \in \mathcal{N},\, b \in \mathcal{B} \tag{7}$$

$$y_{bs}^{tc} \leqslant \max(a_n^{dc}, u^{tc}) \sigma_{bn'}^{d} \qquad \forall\, n \in \mathcal{N},\, s \in \mathcal{S}(n),\, b \in \mathcal{B},\, c \in \mathcal{C},\, d \in \mathcal{P},\, t = \langle p^n, d \rangle,\, n' \in \mathcal{N}(s, p^n, d) \tag{8}$$

$$\sum_{b \in \mathcal{B}} y_{bs}^{tc} \leqslant \max(a_n^{dc}, u^{tc}) \qquad \forall\, n \in \mathcal{N},\, s \in \mathcal{S}(n),\, d \in \mathcal{P},\, c \in \mathcal{C},\, t = \langle p^n, d \rangle \tag{9}$$

$$\sum_{b \in \mathcal{B}} y_{bs}^{tc} \geqslant u^{tc} \qquad \forall\, s \in \mathcal{S},\, t \in \mathcal{T},\, c \in \mathcal{C} \tag{10}$$

$$0 \leqslant w_{bps} \leqslant q_b \qquad \forall\, s \in \mathcal{S},\, b \in \mathcal{B},\, p \in \mathcal{P}^{l \backslash u} \tag{11}$$

$$y_{bs}^{tc} \in \mathbb{R}^+ \qquad \forall\, s \in \mathcal{S},\, b \in \mathcal{B},\, t \in \mathcal{T},\, c \in \mathcal{C} \tag{12}$$

$$\sigma_{bn}^{d} \in \{0, 1\} \qquad \forall\, n \in \mathcal{N},\, b \in \mathcal{B},\, d \in \mathcal{P} \tag{13}$$

Objective (1) maximises the expected value of the containers loaded, where f^{tc} is a general function describing the value of each container. This parameter can be changed to accommodate different objectives, e.g. intake maximisation as in Christensen and Pacino (2017). Here it is the revenue.

Constraint (2) and (3) ensure that the stowage plans for all the scenarios are feasible with respect to stability and capacity respectively. The parameter θ_{bp}^{c} is used in the capacity constraints to reduce the capacity of the blocks, according to the initially loaded containers.

Constraint (4) are the non-anticipativity constraints, making sure that the decisions for two scenarios with a common history are the same up until that point. In other words; no information revealed at a later stage is utilised at the current stage. Constraint (5) links the weight variables w_{bps} with the weight of the containers stowed in the scenario plus the weight of the containers initially stowed on the vessel. Intuitively. the left-hand side and right-hand side of (5) should be equal. However, the work of Christensen and Pacino (2017) relies on relaxing this equality constraint and later undoing the relaxation, ensuring a strict correspondence between the weight variable and the weight of the containers. It is shown that the relaxed model gives an accurate revenue estimation in considerably less time. Our main focus is to get an accurate revenue estimation fast, and therefore (5) does not impose a strict correspondence for the weight variables.

Constraints (6)–(8) are the block assignment constraints. Constraint (6) makes sure the block assignment follows what is imposed by the initial configuration. Together (7) and (8) enforce that no overstowage can occur. Specifically, constraint (7) ensures that exactly one port is assigned as discharge port for every node. Constraint (8) makes sure that containers to a destination d can only be stowed in a block if the assigned destination matches d during the full journey. Constraint (7) does not enforce that if $\sigma_{bn}^{d'} = 1$ for a block b, then all nodes with $p^n < d'$ must have d' as discharge port. This will, however, be the case since constraint (8) is posted for every node, which effectively disallows the stowage of containers with another port of destination.

Constraint (9) ensures that no more containers than the ones available in the given scenario are loaded, and constraint (10) makes sure we load the containers already accepted. Finally, constraints (11)–(13) define the variables domain, and (11) enforce the block weight capacity limit. The model includes ballast water within the set $\mathcal{W}$, but only as a way to fix stability issues. Thus, the ballast water is not minimised as part of the objective as in Delgado (2013).

5. Solution method

The number of scenarios in the stochastic model in Section 4 grows exponentially with the number of ports considered. As the number of scenarios quickly increases, even more so does the number of variables. Therefore, it is not expected that the model can solve more than the smallest toy example.

Besides the number of variables, the main contributor to the intractability of the stochastic model is the binary indicator variables enforcing the block stowage. To overcome this, we will describe a multiphase hierarchical heuristic where the block assignment is heuristically determined in Phase I. Hereafter a rolling horizon heuristic is used in Phase II to deal with the stowing of containers.

5.1. Phase I

Phase I heuristically determines the block assignment. In Section 4 the block assignment is allowed to be different from scenario to scenario. We will disregard this in this matheuristic and instead set one full block assignment a priori. The block assignment is determined for all ports $p \in \mathcal{P}^{l \setminus u}$. For all blocks, we want to find a block assignment that fits the attributes of the specific block. First, we will describe how the block assignment can be determined if no bookings have been accepted ($u^{tc} = 0$ for all transports and container types). Hereafter, this method will be extended to handle accepted bookings.

5.1.1. No accepted containers

When no containers have been accepted, the block assignment will be determined heuristically by solving a longest path problem in a directed acyclic graph. Fig. 5 is an example of such a graph with $|\mathcal{P}^{l \setminus u}| = 4$ and $|\mathcal{P}^u| = 4$. The ports in $\mathcal{P}^{l \setminus u}$ are labelled a, b, c and d, whereas the ports labelled 1, 2, 3 and 4 belong to the set $\mathcal{P}^u$. An edge in the graph describes an assignment of a destination port to an origin port, and a s-t path in the graph G describes a block assignment in which all ports in $\mathcal{P}^{l \setminus u}$ are assigned a destination port $d \in \mathcal{P}$. If a path uses an edge (p', d'), port d' will be assigned as the destination port for port p'. At all ports $p \in \mathcal{P}^{l \setminus u}$ in between p' and d', d' will have to be assigned as the destination port as well.

Let G be a weighted directed acyclic graph with V as the vertex set, and E as the edge set. The graph consists of a source s, and a sink node t. Besides these two nodes, each port $p \in \mathcal{P}$ is represented as a node in the graph, such that

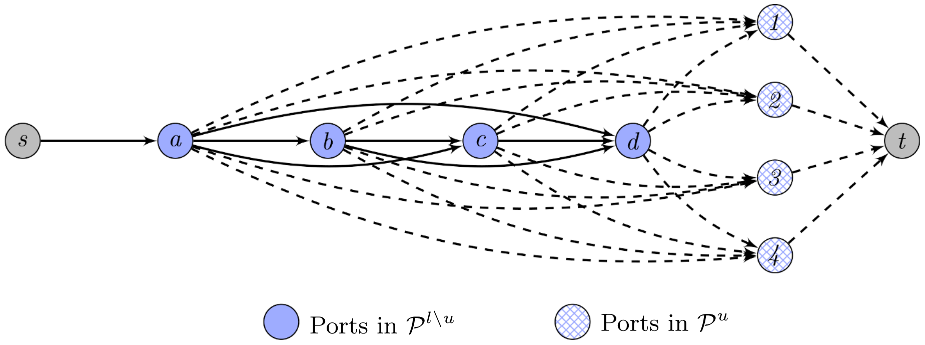

s
a
b
c
d
1
2
3
4
t
● Ports in $\mathcal{P}^{l \setminus u}$
▨ Ports in $\mathcal{P}^u$

Fig. 5. An example graph G with $|\mathcal{P}^{l \setminus u}| = |\mathcal{P}^u| = 4$.

$$V = \{s, t\} \cup \mathcal{P}$$

To define the edge set, let E_{base} be the set of *base edges*, i.e. edges that connect the sink and source node to the rest of the graph.

$$E_{base} = (s, p_0) \cup \{(i, t) | i \in \mathcal{P}^u\}$$

Where p_0 is the first port in the service (a in Fig. 5). Define $\delta^+(i)$ as the set of ports visited after port i in the service. With this define the full set of edges as

$$E = E_{base} \cup \{(i, j) | i \in \mathcal{P}^{l \setminus u}, j \in \delta^+(i)\}$$

The initial configuration implies a partial block assignment for some blocks, which will need to be satisfied when generating the full block assignment. We control this by setting the edge weight, such that the longest path in the graph ensures that the partial block assignment is fulfilled. If for a block b and port p_0 there exists a destination $d \in \mathcal{P}$ such that $\Omega^d_{bp_0} = 1$, then the destination at the first port is imposed by the initial configuration of the vessel. In that case, we want to make sure that port d is assigned as the destination for all ports p visited before port d. Thus, we want to ensure that the longest path in the graph G contains the edge (p_0, d). This also excludes a set of edges we know cannot be used. The set of *invalid* edges for a block b is denoted as $E_I(b)$, and defined as follows.

$$E_I(b) = \{(i, j) \in E,\ \ d \in P \mid i < d, j \in \delta^+(i),\ \Omega^d_{bp_0} = 1\} \setminus (p_0, d)$$

To ensure the set of invalid edges cannot be part of the longest path we set the weight for all these edges to $-M$

$$w_{ij} = -M \qquad \forall\ (i, j) \in E_I(b)$$

For all valid, non-base edges ($(i, j) \in E \setminus \{E_{base} \cup E_I(b)\}$) the weight represents the maximum revenue that can be transported along this edge. The calculation considers the blocks TEU capacity, reefer capacity as well as the demand. Algorithm 1 describes how this is calculated.

Algorithm 1. `Calculate` w_{ij}

```
Input: $G, \overline{D}, F, S, b$
 1: $E_I(b) \leftarrow$ makeInvalidEdges$(G, S, b)$
 2: for all $(i, j) \in E \setminus \{E_{base} \cup E_I(b)\}$ do
 3:   $w_{ij} \leftarrow 0$
 4:   $\Delta\tau \leftarrow \max(k_b^{20}, k_b^{TEU})$
 5:   $\Delta\tau^r \leftarrow \max(r_b^{Slot}, 2r_b^{Cell})$
 6:   $\mathcal{T}_{ij} = \{t \in \mathcal{T} | o^t \geqslant i \wedge d^t = j\}$
 7:   $\mathcal{K} \leftarrow$ All tuples $\langle t, c, f^{tc} \rangle$ where $t \in \mathcal{T}_{ij}$, $c \in C$, and $f^{tc}$ being the value of container type $c \in C$ of transport $t \in \mathcal{T}_{ij}$.
 8:   $\mathcal{K} \leftarrow$ Sort the set $\mathcal{K}$ after descending order of $f^{tc}$.
 9:   for all $\langle t, c, f^{tc} \rangle \in \mathcal{K}$ do
10:     if $c \in \mathcal{R}$ then
11:       capacity $\leftarrow \min(\Delta\tau,\ \Delta\tau^r)$
12:     else
13:       capacity $\leftarrow \Delta\tau$
14:     end if
15:     load $\leftarrow \min\left(d^{tc}, \left\lfloor \frac{\text{capacity}}{\Gamma^c} \right\rfloor\right)$
16:     $w_{ij} \leftarrow w_{ij} + f^{tc}$load
17:     $\Delta\tau \leftarrow \Delta\tau - \Gamma^c$load
18:     if $c \in \mathcal{R}$ then
19:       $\Delta\tau^r \leftarrow \Delta\tau^r - \Gamma^c$load
20:     end if
21:   end for
22: end for
23: $w_{ij} = 0$     $\forall\ (i, j) \in E_{base}$
24: $w_{ij} = -M$    $\forall\ (i, j) \in E_I(b)$
25: return $W$
```

The input for Algorithm 1 is the following: The graph G, average demand- and value matrix $\overline{D}$ and F, the initial configuration S and the block b. The algorithm returns the edge weight matrix W to be used when calculating the block assignment for the given block. Line 1 makes the set of invalid edges, and lines 2–22 calculate the edge weight for all valid, non-base edges. Lines 3–5 initialize the weight and the remaining TEU ($\Delta\tau$) and remaining reefer capacity ($\Delta\tau^r$). The remaining capacities are initialised using the block capacities defined in Appendix A. In line 6, the set $\mathcal{T}_{ij}$ is created. This set consists of the transports that can have cargo transported by the block if the considered edge is part of the longest path. The set $\mathcal{K}$ is a tuple with all the relevant transports and container types together with the value f^{tc}. We want to assign the most profitable containers first, and therefore the set $\mathcal{K}$ is sorted according to descending order of f^{tc} in line 8. In lines 9–21 elements from the set $\mathcal{K}$ are processed in order. Lines 10–14 calculate the capacity of the specific type depending on whether it is a reefer or not. In line 15 we determine the actual number of containers of the considered type that can be assigned. Here we consider the demand (d^{tc}) as well as the capacity. We divide the capacity by the TEU coefficient (Γ^c) to have the actual number of containers. Line 16 updates the edge weight and lines 17–20 update the remaining TEU capacity, $\Delta\tau$, and the remaining reefer capacity $\Delta\tau^r$. Lastly lines 23 and 24 respectively set the weight for the base and invalid edges.

The edge weights are set in such a way that the longest path in the graph corresponds to a block assignment where the structure of the demand is taken into account as well as the attributes of the specific blocks. The idea here is similar to what Christensen and Pacino (2017) proposed for the deterministic version of the cargo-mix problem. In Christensen and Pacino (2017) a full service is considered, and therefore the graph structure is different. Results showed that this outperformed a random method.

The longest path problem is solved for each block. The assigned cargo is then subtracted from the demand matrix to account for the cargo which the generated schedule can carry. The edge weight calculations are based on the demand, and thus the edge weights are updated from block to block. Therefore, the longest path differs from block to block.

The graph G is a directed acyclic graph, and the longest path problem can be solved efficiently by topologically sorting the vertices and using a slightly modified version of Dijkstra's algorithm, such that the longest path is found instead of the shortest.

5.1.2. Including accepted bookings

The previously described procedure does not ensure that there is enough capacity assigned to a transport t' to load the cargo that has been accepted for that transport. Assume some cargo has been accepted for the transport t' (i.e. $\sum_{c\in C} u^{t'c} > 0$), then there must exists at least one block where port $d^{t'}$ is assigned as the discharge port for port $o^{t'}$. If this is not the case, the accepted cargo cannot be loaded, making it impossible to keep the promise to the customer.

Aside from the capacity considerations, the stability of the vessel will also need to be taken into account when ensuring that the accepted cargo can be loaded. With accepted cargo assigned to blocks, it must to be possible to load the rest of the ship in a way that makes the vessel stable and the overall problem feasible.

Thus, before making the full block assignment (using the graph-based method), we want to partially fix the block assignment, ensuring that the accepted cargo can be transported. We want to do this alongside minimising the capacity usage which the partially fixed block assignment takes up. The partially fixed block assignment can be satisfied in the graph-based method by extending the set of invalid edges ($E_I(b)$).

The problem is formulated as a mixed integer programming problem. First, let the variable $y_b^{tc} \in \mathbb{R}^+$ be the number of containers of type $c \in C$ of transport $t \in \mathcal{T}$ to be stowed in block $b \in \mathcal{B}$. This is similar to the variable y_{bs}^{tc} (from Section 4) but without the scenario index. Similar, define w_{bp} as the weight stowed in block $b \in \mathcal{B}$ at port $p \in \mathcal{P}^{l\backslash u}$. The main decision variable is σ_{bp}^d defining the block assignment

$$\sigma_{bp}^d = \begin{cases} 1 & \text{If block } b \in \mathcal{B} \text{ is assigned destination } d \in \mathcal{P} \text{ at port } p \in \mathcal{P}^{l\backslash u} \\ 0 & \text{Otherwise.} \end{cases}$$

Along with the sets and parameters defined in Section 4, define $\mathcal{T}(d)$ as the set of transports with d as destination for which containers has been accepted.

$$\mathcal{T}(d) = \left\{ t \in \mathcal{T} \middle| d^t = d, \sum_{c\in C} u^{tc} > 0 \right\}$$

and define $st(d)$ as the first port for which containers destined to d has been accepted

$$st(d) = \min_{t\in\mathcal{T}(d)} (o^t)$$

With this, the problem can be formulated as follows

$$\text{Min} \sum_{d\in\mathcal{P}\backslash p_0} \sum_{b\in\mathcal{B}} k_b^{TEU} \sigma_{bd-1}^d \tag{14}$$

Subject to:

$$\sum_{d\in\mathcal{P}} \sigma_{bp}^d \leqslant 1 \qquad \forall\, b \in \mathcal{B},\, p \in \mathcal{P}^{l\backslash u} \tag{15}$$

$$\sigma_{bp}^{d^t} \geqslant \sigma_{bo^t}^{d^t} \qquad \forall\, b \in \mathcal{B},\, t \in \mathcal{T},\; o^t \leqslant p < \min(|\mathcal{P}^{l\backslash u}|, d^t) \tag{16}$$

$$\sigma_{bp}^d \geqslant \Omega_{bp}^d \qquad \forall\, b \in \mathcal{B},\, p \in \mathcal{P}^{l\backslash u},\, d \in \mathcal{P} \tag{17}$$

$$\mathbf{w} \in \mathcal{W} \tag{18}$$

$$\sum_{t\in\mathcal{T}_p^{ON}} \sum_{c\in C} v^c y_b^{tc} \leqslant w_{bp} - \sum_{c\in C} v^c \theta_b^{tc} \qquad \forall\, b \in \mathcal{B},\, p \in \mathcal{P}^{l\backslash u} \tag{19}$$

$$\sum_{b\in\mathcal{B}} y_b^{tc} \leqslant \max(\min_{n\in\mathcal{N}}(a_n^{tc}), u^{tc}) \qquad \forall\, t\in\mathcal{T}, c\in C \tag{20}$$

$$\sum_{b\in\mathcal{B}} y_b^{tc} \geqslant u^{tc} \qquad \forall\, t\in\mathcal{T}, c\in C \tag{21}$$

$$\sum_{t\in\mathcal{T}(d)} \sum_{c\in C} v^c y_b^{tc} \leqslant \left(q_b - \sum_{c\in C} v^c \theta_{bst(d)}^c\right)\sigma_{bst(d)}^d \qquad \forall\, d\in\mathcal{P}, b\in\mathcal{B} \tag{22}$$

$$\sum_{t\in\mathcal{T}(d)} \sum_{c\in C^{20}} y_b^{tc} \leqslant \left(k_b^{20} - \sum_{c\in C^{20}} \theta_{bst(d)}^c\right)\sigma_{bst(d)}^d \qquad \forall\, d\in\mathcal{P}, b\in\mathcal{B} \tag{23}$$

$$\sum_{t\in\mathcal{T}(d)} \sum_{c\in C} \Gamma^c y_b^{tc} \leqslant \left(k_b^{TEU} - \sum_{c\in C} \Gamma^c \theta_{bst(d)}^c\right)\sigma_{bst(d)}^d \qquad \forall\, d\in\mathcal{P}, b\in\mathcal{B} \tag{24}$$

$$\sum_{t\in\mathcal{T}(d)} \sum_{c\in\mathcal{R}^{20}} y_b^{tc} \leqslant \left(r_b^{Slot} - \sum_{c\in\mathcal{R}^{20}} \theta_{bst(d)}^c\right)\sigma_{bst(d)}^d \qquad \forall\, d\in\mathcal{P}, b\in\mathcal{B} \tag{25}$$

$$\sum_{t\in\mathcal{T}(d)} \sum_{c\in\mathcal{R}} \frac{1}{2}\Gamma^c y_b^{tc} \leqslant \left(r_b^{Cell} - \sum_{c\in C} \frac{1}{2}\Gamma^c \theta_{bst(d)}^c\right)\sigma_{bst(d)}^d \qquad \forall\, d\in\mathcal{P}, b\in\mathcal{B} \tag{26}$$

$$\sum_{t\in\mathcal{T}(d)} \sum_{c\in C} \Phi^c y_b^{tc} \leqslant \left(h_b - \sum_{c\in C} \Phi^c \theta_{bst(d)}^c\right)\sigma_{bst(d)}^d \qquad \forall\, d\in\mathcal{P}, b\in\mathcal{B} \tag{27}$$

$$0 \leqslant w_{bp} \leqslant q_b \qquad \forall\, b\in\mathcal{B}, p\in\mathcal{P}^{l\backslash u} \tag{28}$$

$$y_{bs}^{tc} \in \mathbb{R}^+ \qquad \forall\, b\in\mathcal{B}, t\in\mathcal{T}, c\in C \tag{29}$$

$$\sigma_{bp}^d \in \{0, 1\} \qquad \forall\, b\in\mathcal{B}, p\in\mathcal{P}^{l\backslash u}, d\in\mathcal{P} \tag{30}$$

The objective function (14) minimises the TEU capacity occupied by the partial block assignment. Here, we consider σ_{bd-1}^d instead of σ_{bp}^d to only account for the capacity once for a particular destination. Constraints (15)–(17) define what should be satisfied for the block assignment, i.e at the most one destination can be assigned a port (15), if d is chosen as the destination at port o, then all ports between o and d must have d as discharge port (16), and lastly constraint (17) ensures that the block assignment follows what is imposed by the initial configuration. Constraints (18)–(27) ensure that the partially fixed block assignment results in an overall feasible problem. Constraints (18) and (19) ensure the stability of the vessel similar to how it is described in Section 4. In (20) we consider the number of containers available in the most conservative scenario, ensuring that in every scenario there are enough containers to make the vessel stable. Constraint (21) ensures that all the accepted cargo is assigned to a block on the vessel. Constraints (22)–(27) are capacity constraints, ensuring that enough blocks are allocated to the transports for which containers have been accepted. These are reformulated capacity constraints, respectively the weight capacity, 20-foot capacity, TEU capacity, reefer slot capacity, reefer cell capacity and volume capacity (see Appendix A for a description of the standard capacity constraints). The left-hand side of these constraints is the capacity usage of containers with a specific destination loaded in the considered block, and the right-hand side is the capacity. Here, the capacity is the original capacity minus the capacity used by the initially loaded containers in this block. For a specific block and container type, the parameter θ_{bp}^c is non-increasing as p increases. Therefore, we consider the port $st(d)$ when calculating the capacity used by the initially loaded containers, as this ensures enough capacity during the full journey.

In the above model, only containers from transports with accepted cargo must meet the block stowage requirement. Additionally, containers from transports with no accepted cargo are not accounted for in the capacity constraints. This is intentional and helps to simplify the model. The main decision variables in the model are the block stowage variables, and the capacity constraints should only enforce that we have enough capacity for the accepted cargo. We assume that if this model is feasible, then we can achieve a similar weight distribution from loading cargo satisfying the block stowage constraint and capacity constraints.

From the solution we extract the partial block assignment which is used to make edges invalid in the graph-based method. Let $\Lambda(b)$ describe the partial block assignment for the block b.

$$\Lambda(b) = \{\langle p, d\rangle \in \mathcal{P}^{i\backslash u} \times \mathcal{P} | d\in\mathcal{P}, t\in\mathcal{T}(d), p = o^t, \sigma_{bp}^d = 1\} \qquad \forall\, b\in\mathcal{B}$$

Here, $\langle p, d\rangle$ describes an assignment of d as discharge port for port p. From this we add the following set of edges to the set of invalid edges

$$\{(i, j)\in E | i < p, p > j, j \neq d\} \qquad \forall\, b\in\mathcal{B}, \langle p, d\rangle \in \Lambda(b) \tag{31}$$

$$\{(i, j)\in E | i = p, j \neq d\} \qquad \forall\, b\in\mathcal{B}, \langle p, d\rangle \in \Lambda(b) \tag{32}$$

Consider a block b, and a block assignment $\langle p, d\rangle$, Eq. (31) makes the edges originating from a port before p and terminating at a port after p invalid, except the edges terminating at port d. This ensures that either d is chosen as discharge port for port p, or the block is emptied at port p, making sure we can assign a new discharge port. If the block is emptied at port p, Eq. (32) ensures port d is chosen as the discharge port, as it makes all other edges originating from p invalid.

5.2. Phase II

In Phase II the block assignment is fixed. Let $\hat{\sigma}_{bp}^{d}$ be the block assignment found in Phase I, this is treated as an input parameter in the model and will replace the σ_{bn}^{d} variables. Note that $\hat{\sigma}_{bp}^{d}$ does not depend on the node, but only on the port, as discussed in Section 5.1. Phase II aims at finding a high-quality, feasible solution to the following problem.

$$\text{Max } (1) \tag{33}$$

Subject to:

$$(2) - (5)$$
$$(9) - (12)$$
$$y_{bs}^{tc} \leqslant \max(a_n^{dc}, u^{tc})\hat{\sigma}_{bp^n}^{d} \qquad \forall\, n \in \mathcal{N},\, s \in \mathcal{S}(n),\, b \in \mathcal{B},\, c \in C,\, d \in \mathcal{P},\, t = \langle p^n, d\rangle \tag{34}$$

Constraint (34) ensures that the block assignment found in Phase I is satisfied, and it is thus equivalent to constraint (8). Here, we only need to check if the container can be loaded in its port of origin. If this is the case, due to the construction of the block assignment, we know the block assignment will be ensured until its destination. This is not the case with constraint (8), as the block assignment is part of the decision.

By solving this model as is, we will still have the problem that the number of scenarios grows exponentially. The problem with a large number of scenarios is twofold; first, the computational effort needed to solve the model will increase with the number of scenarios, second, memory problems might occur making it intractable to even build the model. To alleviate this, we will use a Rolling Horizon Heuristic (RHH).

The RHH solves the problem by decomposing the full problem into subproblems with shorter planning horizons. Doing so, the uncertainty in demand is only considered for the next few ports ahead. The solution to the subproblems is used to fix part of the solution for the full problem. There are multiple benefits derived from only considering a shorter planning horizon, e.g. the problems to be solved become smaller and can thus be solved more efficiently, the subproblems can be solved in parallel and thus take advantage of the power of modern computers. The main downside is that we will have more problems to solve.

A subproblem considers a subtree (T') of the full scenario tree and fixes the solution for the nodes in the subtree. Let $root(T') \in \mathcal{N}$ be the root of a subtree and let ρ be a parameter describing the number of forthcoming ports for which the stochasticity has to be considered for, i.e. the horizon. For a subproblem, SCMPBS(T') the subtree T' contains the following set of nodes.

$$\mathcal{N}(T') = \{n \in \mathcal{N} | stage(root(T')) \leqslant stage(n) \leqslant stage(root(T')) + \rho,\ root(T') \in path(n)\}$$

where $stage(n)$ is the stage associated with node $n \in \mathcal{N}$. Thus, for the subproblem SCMPBS(T') we consider the nodes in the next ρ stages that have $root(T')$ on their path. Let $\mathcal{S}(T')$ be the scenarios considered in the problem SCMPBS(T'). Along with the nodes $\mathcal{N}(T')$, ***artificial nodes*** are considered as well. The artificial nodes make sure that solution does not only optimise over the set of ports considered in T', but that the forthcoming ports are also taken into account. For the artificial nodes, the uncertainty in demand is not considered, and the average is used as an estimate for the demand. For all forthcoming ports not considered in T' (all ports $p > p^{root(T')} + \rho$) a single artificial node is added for every scenario $s \in \mathcal{S}(T')$. After solving the subproblem SCMPBS(T') the solution is fixed for all nodes $n \in \mathcal{N}(T')$, and we continue with the next subproblem.

Fig. 6 shows an example of a subtree T' with $\rho = 1$. In Fig. 6 the solution for the diamond-shaped node has already been fixed by another subproblem. The solid circle nodes make up the set $\mathcal{N}(T')$, the striped nodes are the artificial nodes, and the square nodes are considered in another subproblem and are thus disregarded in the current.

The parameter ρ describes the subproblems we need to solve to get a full solution. For all the subtrees T' having a root node with the stage within the set $\{1, 1 + (\rho + 1), 1 + 2(\rho + 1), \ldots, 1 + |\mathcal{P}^{l \setminus u}|(\rho + 1)\}$ a subproblem SCMPBS(T') will be solved. As an example, if $\rho = 2$ and $|\mathcal{P}^{l \setminus u}| = 7$ we need to solve a subproblem for all nodes in the following stages: 1, 4, and 7. For the rest of the nodes, the solution is fixed in a previously solved subproblem. The subproblems are solved in order, starting with the subproblem with the lowest stage. Fig. 7 shows an example of the RHH with $\rho = 2$. Fig. 7a shows the scenario tree for the full problem, and Fig. 7b shows how this is decomposed into a set of subproblems. In Fig. 7b the solid circle nodes are the nodes $\mathcal{N}(T')$, the striped nodes are the artificial nodes, and the diamond-shaped nodes are the nodes for which the solution already has been fixed. The numbers within the nodes are used to identify the nodes from each other. The number in the top left corner is the subproblem number and shows the order in which the subproblems are solved.

The problem solved for subproblem SCMPBS(T') is the same as the problem (33)-(34), but on a redefined set of scenarios, nodes and ports. The parameter θ_{bp}^{c} is updated for every subproblem to account for the part of the problem for which the solution has already been fixed.

The parameter ρ describes a tradeoff between the complexity of the subproblems and the number of subproblems to solve. The higher a value of ρ, the more complex the subproblems are. A lower value means simpler subproblems, but more will have to be solved. However, a higher value of ρ should also imply a better solution.

With the rolling horizon heuristic we can effectively handle the exponential growth in the number of scenarios. Instead, the

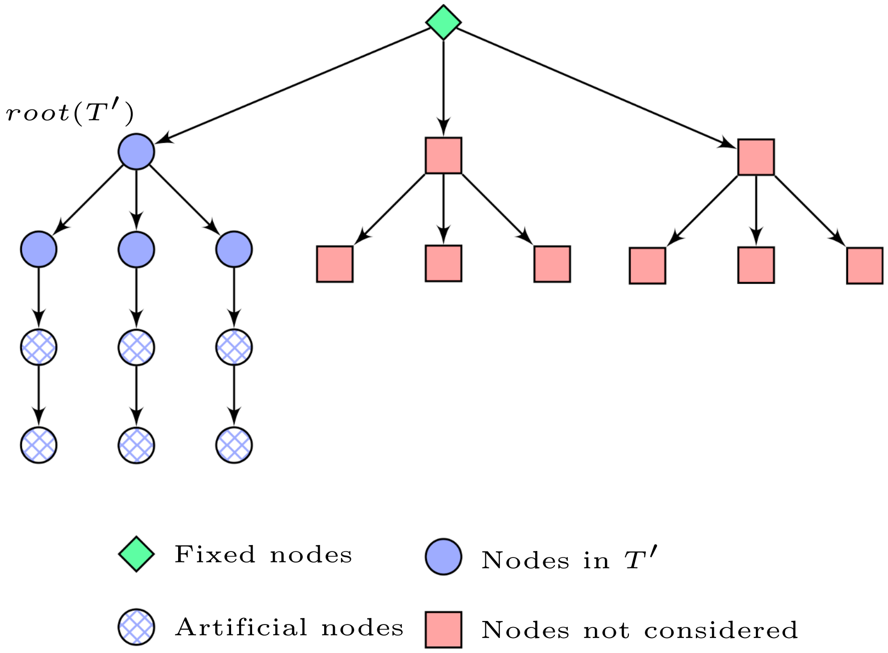


Fig. 6. An example subtree with $\rho = 1$.

number of subproblems to solve will grow exponentially, but it is very unlikely to experience memory problems, and the RHH allows for the subproblems to be solved in parallel as the subproblems for a specific stage are independent of each other.

6. Data

Most liner shipping routes reflect the nature of an international trade, e.g. the Europe to Far East services. Here, containers are exported from the Far East and imported to Europe. Vessels seldom dock at ports between the Suez canal and the Singapore Strait. Due to the length of this leg, it is the most important as regards revenue optimisation. To reflect this, we consider 5 Europe-Far East/ Far East-Europe services operated by our industry collaborator. These services will be the basis of our data. All of the services are being considered from both directions. For the Europe-Far East direction the ports in Europe are considered to be unloading/loading ports, whereas only unload operations are planned for the ports in the Far East, and vice versa for the Far East-Europe direction. With this separation, we want to maximise the revenue from the most significant leg, while still considering the in-region revenue. To test how the solution method scales with respect to the number of loading ports, several instances are made out of each of the services. We control this by selecting different ports for what we will consider being the start. We restrict the number of loading ports to be considered to be greater than or equal to 3. The maximum number of loading ports to consider is determined by the services and this number is 9. In total 82 instances have been generated.

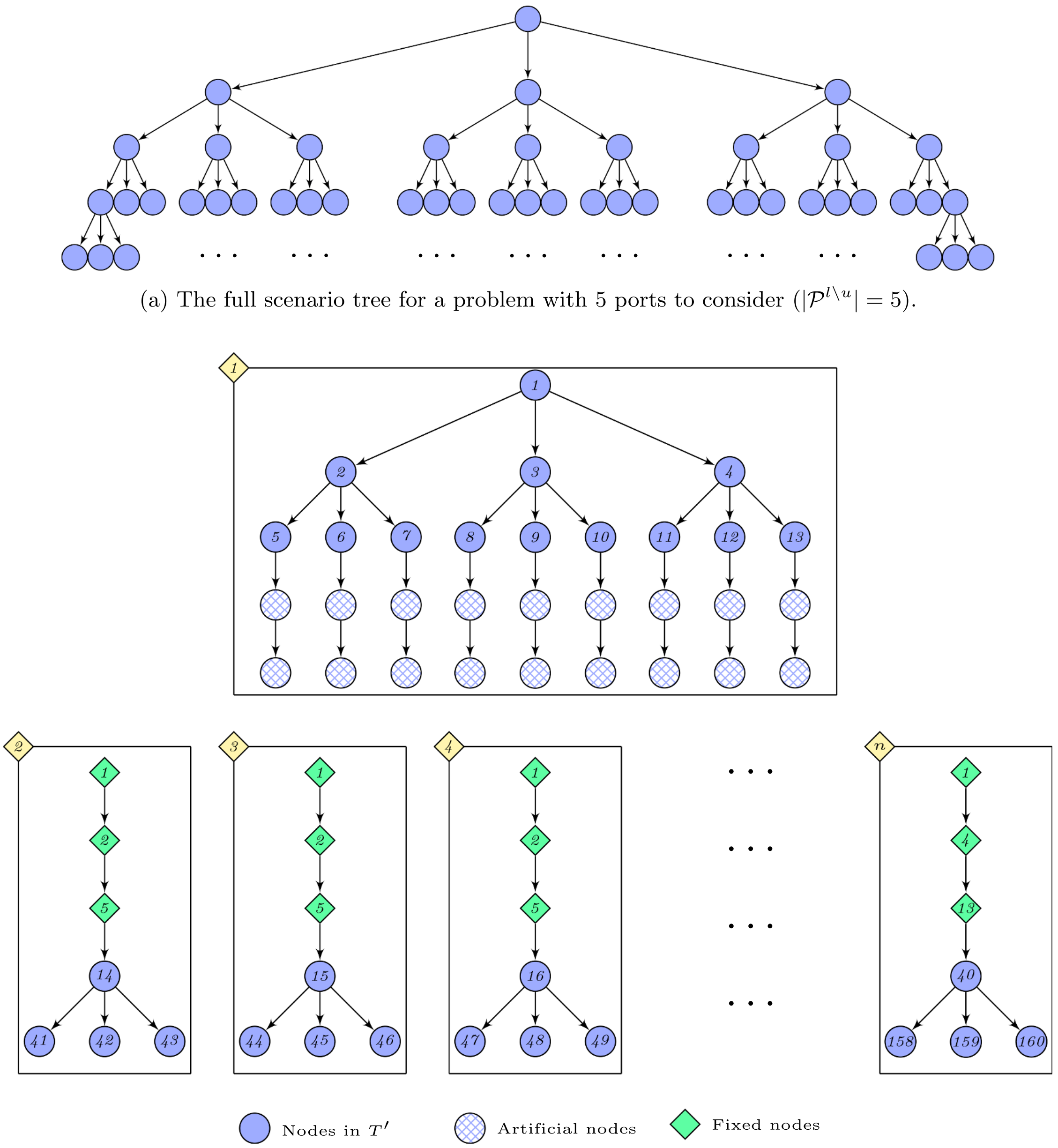
To generate the initial configuration of the vessel, a deterministic version of the problem is considered. Using the method described in Christensen and Pacino (2017) a solution to the deterministic problem is found. From the solution to the deterministic problem, 50% of the containers loaded on the vessel when it docks at the first loading port is removed, and with this an initial discharge port for some blocks is imposed.

The base port-to-port demand and port drafts are based on data from LinerLib (Brouer et al., 2014) and multiplied with a scalar to be more suitable for the SCMPBS. Successively, the total demand of each leg is decomposed into a demand for each of the specific container types by using the same predefined probabilities as in Christensen and Pacino (2017). The revenue of a dry 40-foot container is based on data from LinerLib, and from this, we calculate the revenue of each container type similar to what is described in Delgado (2013). In a given scenario, all the demand is multiplied with a scenario dependent scalar, such that the demand for all containers is ‘high’ in a ‘high’ scenario.

Table 1 describes how the scenario trees are generated. The scenarios for a port depends on the distance (in number of ports) from the start port. The farther away, the bigger the uncertainties. The second column is the number of scenarios generated for the port in question. The third and fourth column describe the demand scenarios with the demand factor and the probabilities. The numbers in the third column describe the difference in percentage with respect to the expected demand.

7. Computational results

Three different solution methods have been tested and compared.



(a) The full scenario tree for a problem with 5 ports to consider ($|\mathcal{P}^{l\setminus u}| = 5$).

(b) An example of the Rolling Horizon Heuristic with $\rho = 2$

Fig. 7. Illustration of the Rolling Horizon Heuristic.

Table 1
Scenario Tree generation description.

Port distance	Number of scenarios	Demand scenarios	Probabilities
1	3	0, 2.5%, −2.5%	70%, 15%, 15%
2	3	0, 4%, −4%	60%, 20%, 20%
3	3	0, 5%, −5%	50%, 25%, 25%
4	3	0, 7.5%, −7.5%	40%, 30%, 30%
5	4	2.5%, −2.5%, 5%, −5%	35%, 35%, 15%, 15%
6	4	4%, −4%, 7.5, −7.5%	30%, 30%, 20%, 20%
7	5	0, 2.5%, −2.5%, 7.5%, −7.5%	10%, 30%, 30%, 15%, 15%
8	5	0, 5%, −5%, 10%, −10%	10%, 25%, 25%, 20%, 20%
9	5	0, 5%, −5%, 12.5%, −12.5%	10%, 22.5%, 22.5%, 22.5%, 22.5%

Table 2
Overall Results. Here † means that the model terminated due to insufficient memory.

		UB		Compact					RHH, $\rho = 1$				ModelHeu				
$\|P^l \backslash ^u\|$	n	$\bar{x}\,(10^6\$)$	$\bar{t}$	#Sol	#Opt	$\bar{x}\,(10^6\$)$	*Gap*	$\bar{t}$	#Sol	$\bar{x}\,(10^6\$)$	*Gap*	$\bar{t}$	#Sol	#Opt	$\bar{x}\,(10^6\$)$	*Gap*	$\bar{t}$
3	18	15.06	15	18	9	13.74	11.88%	10451.6	18	14.83	1.73%	6.7	18	18	14.83	1.72%	91.1
4	17	17.89	83	17	3	9.85	45.38%	15888.5	17	17.64	1.50%	20.6	17	16	17.64	1.50%	4286.8
5	17	20.92	451	15	0	4.24	74.35%	18000.0	17	20.61	1.54%	58.4	17	4	19.99	7.34%	15375.6
6	12	23.42	3267	0	0	†	†	†	12	23.10	1.61%	270.0	3	0	22.99	0.81%	18000.0
7	9	24.07	22131	0	0	†	†	†	9	23.75	1.49%	463.5	0	0	†	†	†
8	6	23.76	198521	0	0	†	†	†	6	23.41	1.77%	3278.8	0	0	†	†	†
9	3	20.03	1613222	0	0	†	†	†	3	19.64	2.46%	9239.1	0	0	†	†	†
Average		19.89	76568			9.56	42.01%	14569.1		19.60	1.63%	686.2			17.74	3.34%	7089.7

- **Compact - SCMPBS Mixed Integer Programming model**
  The mixed integer programming model as presented in Section 4, given a time limit of 5 h.
- **RHH - SCMPBS Rolling Horizon Heuristic**
  The rolling horizon heuristic as described in Section 5. All the subproblems are solved sequentially and not in parallel.
- **ModelHeu - SCMPBS Mathematical modelling based heuristic**
  This is a version of the heuristic where the model (33)-(34) is used instead of the rolling horizon heuristic. This model is given a time limit of 5 h. Phase I remains as described in Section 5.1.

All methods have been implemented in JAVA 1.8 and are tested using a 2.30 GHz Intel Xeon E5 processor and 128 GB memory. CPLEX v. 12.7.0 is used as the mixed-integer-programming solver. For all methods, an optimality tolerance of 0.05% was used.

An upper bound is calculated by relaxing the non-anticipativity constraints (4) in the compact model. Doing so, there are no constraints linking decisions in one scenario to decisions in another scenario. Thus, the problem can be decomposed by scenario. Each scenario-decomposed problem corresponds to a deterministic problem and can be solved independently from the rest.

Table 2 compares the upper bound with the compact model and the two heuristics. For the RHH the horizon parameter ρ is 1. The two first columns show the number of ports and the number of instances in the instance class. For the four methods, #Sol is the number of instances where a feasible solution is found, the $\bar{x}$ columns are the average revenue in millions of dollars, *Gap* is the average relative difference between the solution and the calculated upper bound. Lastly, $\bar{t}$ is the average execution time in seconds. For the compact model, the #Opt column shows the number of instances where the optimal solution is found. For the ModelHeu, the #Opt column shows the number of instances where the optimal solution is found for the fixed block assignment (model (33)-(34)). This does not mean that the overall solution is optimal. In the table † means that the model terminated due to insufficient memory.

With respect to the compact model, the results show what we would have expected; only the smallest of the instances can be solved to optimality but uses considerably more time than the two heuristics. The quality of the solutions from the compact model quickly drops when increasing the instance size, and memory problems occur for the medium to large sized instances. Overall, the compact method finds the optimal solution for 12 of the instances. For these instances, the average relative difference to the calculated upper bound is 0.15%. This suggests that the upper bound method finds upper bounds of high quality.

As it can also be seen from Table 2, the rolling horizon heuristic finds high-quality solutions within a reasonable amount of time. The solution quality is stable around 1.60% with only a few deviations. For the medium to large scale instances only the RHH finds feasible solutions. As no other method exists to compare with, the RHH is only compared to the computed upper bounds. For two of the smallest instances (3 ports), the RHH finds an optimal solution.

For the 2-phased model based heuristic (ModelHeu) we see a behaviour similar to the one for the compact model; the smallest of the instances can be solved but the largest ones become too big. Looking at the smallest of the instances (three and four ports), the solution quality is comparable with the rolling horizon heuristic, but the RHH is much faster. For the instances with 5 ports, the second phase model finds the optimal solution (given the fixed block assignment) four times. For the rest of the instances the model is terminated due to the time limit. Over the 17 instances with 5 ports the RHH on average finds better solutions.

Table 2 also shows that the computational time to get the upper bounds is higher than the times used for the heuristic. The only purpose of the upper bound is to evaluate the effectiveness of the solution methods, and for this purpose the time is not vital.

Table 3 analyses the impact of the horizon parameter ρ in the rolling horizon heuristic. Here $\rho = 0$ means that no stochasticity in the demand is considered in the subproblems. Each subproblem corresponds to a deterministic problem, which is solved to fix the solution in the root node. The results show that a change in the parameter ρ does not have an impact on the quality of solution. The fact that the solution quality is not affected by the parameter ρ points to the fact that the vessels stowage is flexible enough to handle cargo quantity fluctuations in the subproblems. We believe, however, that for instances with changes in a specific cargo type (say heavy containers) the value of ρ will have a larger impact. This parameter should then be tuned to the specific what-if analysis we wish to perform.

In the rolling horizon heuristic the block assignment is fixed in Phase I, and in Phase II the containers are stowed respecting the decisions from Phase I. Due to the splitting of the decisions, both phases have an impact on the final solution quality. Table 4 analyses

Table 3
Rolling horizon heuristic results.

		UB	RHH, $\rho = 0$			RHH, $\rho = 1$			RHH, $\rho = 2$		
$\lvert P^{l\setminus u}\rvert$	n	$\bar{x}\,(10^6\$)$	$\bar{x}\,(10^6\$)$	*Gap*	$\bar{t}$	$\bar{x}\,(10^6\$)$	*Gap*	$\bar{t}$	$\bar{x}\,(10^6\$)$	*Gap*	$\bar{t}$
3	18	15.06	14.83	1.73%	3.0	14.83	1.73%	6.7	14.83	1.72%	65.8
4	17	17.89	17.64	1.51%	7.3	17.64	1.50%	20.6	17.64	1.50%	121.9
5	17	20.92	20.61	1.54%	21.8	20.61	1.54%	58.4	20.61	1.54%	1027.1
6	12	23.42	23.09	1.61%	75.2	23.10	1.61%	270.0	23.10	1.60%	3858.1
7	9	24.07	23.75	1.49%	260.2	23.75	1.49%	463.5	23.75	1.48%	7595.6
8	6	23.76	23.40	1.78%	1218.3	23.41	1.77%	3278.8	23.41	1.77%	10738.6
9	3	20.03	19.64	2.47%	9435.2	19.64	2.46%	9239.1	19.59	2.86%	31324.6
Average		19.89	19.60	1.63%	480.6	19.60	1.63%	686.2	19.60	1.64%	3582.7

Table 4
Rolling horizon heuristic Phase I and II Results.

		UB		RHH, $\rho = 1$				
$\lvert P^{l\setminus u}\rvert$	n	$\bar{x}\,(10^6\$)$	$\bar{x}_{II}\,(10^6\$)$	$\bar{x}\,(10^6\$)$	*Gap*	Gap_{II}	$\bar{t}_I$	$\bar{t}_{II}$
3	18	15.06	14.83	14.83	1.73%	0.01%	0.6	6.1
4	17	17.89	17.64	17.64	1.51%	0.02%	0.7	19.9
5	17	20.92	20.62	20.61	1.54%	0.04%	1.4	56.9
6	12	23.42	23.10	23.09	1.61%	0.04%	1.3	268.8
7	9	24.07	23.76	23.75	1.49%	0.05%	1.5	462.0
8	6	23.76	23.41	23.40	1.78%	0.05%	2.0	3276.8
9	3	20.03	19.67	19.64	2.47%	0.18%	3.4	9235.6
Average		19.89	19.61	19.60	1.63%	0.04%	1.2	685.0

the two phases of the rolling horizon heuristic with respect to the solution quality. The table aims to explain how the final gap is split between the two phases. The UB columns is for the upper bound, and RHH is for the Rolling Horizon Heuristic. For the upper bound, $\bar{x}$ is an overall upper bound, and $\bar{x}_{II}$ is an upper bound given the fixed block assignment. The RHH result and $\bar{x}_{II}$ bound are based on the same block assignment, and thus $\bar{x}_{II}$ is a bound on the solution found in Phase II of the RHH. After fixing the block assignment, the $\bar{x}_{II}$ bound is calculated by decomposing the problem similarly to what was done when calculating the $\bar{x}$ bound, as previously described. For the RHH, the first column is the value of the average solution. The next two columns are the gap to the two different upper bounds, and the last columns show the average time spent in the two phases.

As also shown in Tables 2 and 3, Table 4 shows that the average gap for the RHH is 1.63%. When comparing with the upper bound for the problem with the fixed block assignment, the average gap is only 0.04%, meaning a major part of the 1.63% of the overall gap stems from the solution to Phase I. Looking at the average time spent for each of the phases, we can see that Phase II is the most computational expensive phase, and the model for the accepted bookings in Phase I can easily be solved.

As explained in Section 5.1, the RHH uses the same block assignment in all scenarios, whereas it is allowed to differ from scenario to scenario in the Compact Model. To analyse the impact of this simplification, a new Compact Model has been formulated, ensuring one block assignment for all scenarios. These two models are compared in 10 small instances where both models find the optimal solution. This comparison shows that the simplification does not have a big impact. We expect the impact to be bigger when the size of the instances increase, however, we still believe it is within acceptable limits (as indicated by the results of the RHH).

8. Conclusion

In this paper, the Stochastic Cargo Mix Problem is studied. The problem aims to find the optimal cargo composition needed for a vessel to maximise its revenue on a given service. A solver for this problem can be a valuable analysis tool for the industry and can be used to perform various kinds of what-if analyses. For this sort of analysis, a fast solver is needed.

The results show that the mathematical model can only solve the smallest of the considered instances. Instead, a rolling horizon based matheuristic is developed. The matheuristic is shown to find high-quality solutions and fulfilling the need for a quick response time. Moreover, the matheuristic is scalable and can solve industrial size instances. The matheuristic is based on mathematical modelling techniques, and it is thus simple to add additional constraints, or remove existing constraints, making it easy to perform various what-if analysis.

The presented solution method is a two-phased matheuristic, and in the computational study we experimentally prove that any improvement of the results should come from improving Phase I. Thus, an idea for further research is to improve the way the block assignment is computed in Phase I, e.g. by making the block assignment scenario dependent. Phase II is the most computationally expensive part of the method. However, the subproblems solved in Phase II can be solved in parallel, and the execution of the method can thus be speeded up by utilising this.

Additional research ideas include a study on how this work can be combined with other essential problems in the liner shipping industry, for example, cargo flow optimisation, service network design or empty re-positioning.

Acknowledgements

This work has been funded by the Danish Innovationsfonden under the GREENSHIP Project (1313-00005B-GREENSHIP).

Appendix A. Capacity constraints

To model the capacity constraints we must introduce additional sets. First let $\mathcal{R} \subset C$ be the set of reefer containers, additionally let $C^{20} \subset C$ be the set of containers with 20 feet length. The sets and additional parameters are summarised below.

Sets:

$C^{20} \subset C$	Set of container types with length of 20 feet.
$\mathcal{R} \subset C$	Set of reefer container types
$\mathcal{R}^{20} \subset \mathcal{R}$	Set of reefer container types, with length of 20 feet
$\mathcal{T}_p^{ON}$	Set of transports that visits port $p \in \mathcal{P}$

Parameters:

$\theta_{bp}^{c} \in \mathbb{N}$	Number of containers of type $c \in C$ initially loaded that still occupy block $b \in \mathcal{B}$ at port $p \in \mathcal{P}$
$k_b^{TEU} \in \mathbb{N}$	Teu capacity for block $b \in \mathcal{B}$
$k_b^{20} \in \mathbb{N}$	Capacity for 20′ containers for block $b \in \mathcal{B}$
$r_b^{Cell} \in \mathbb{N}$	Reefer cell capacity of block $b \in \mathcal{B}$
$r_b^{Slot} \in \mathbb{N}$	Reefer slot capacity of block $b \in \mathcal{B}$
$\Gamma^c \in \{1, 2\}$	TEU coefficient of container type $c \in C$
$h_b \in \mathbb{R}^+$	Volume capacity for block $b \in \mathcal{B}$
$\Phi^c \in \mathbb{R}^+$	Volume coefficient of container type $c \in C$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in C^{20}} y_b^{tc} \leqslant k_b^{20} - \sum_{c \in C^{20}} \theta_{bp}^{c} \qquad \forall\, b \in \mathcal{B},\, p \in \mathcal{P}^{l} \backslash u \tag{A.1}$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in C} \Gamma^c y_b^{tc} \leqslant k_b^{TEU} - \sum_{c \in C} \Gamma^c \theta_{bp}^{c} \qquad \forall\, b \in \mathcal{B},\, p \in \mathcal{P}^{l} \backslash u \tag{A.2}$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{R}^{20}} y_b^{tc} \leqslant r_b^{Slot} - \sum_{c \in \mathcal{R}^{20}} \theta_{bp}^{c} \qquad \forall\, b \in \mathcal{B},\, p \in \mathcal{P}^{l} \backslash u \tag{A.3}$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{R}} \frac{1}{2}\Gamma^c y_b^{tc} \leqslant r_b^{Cell} - \sum_{c \in \mathcal{R}} \frac{1}{2}\Gamma^c \theta_{bp}^{c} \qquad \forall\, b \in \mathcal{B},\, p \in \mathcal{P}^{l} \backslash u \tag{A.4}$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in C} \Phi^c y_b^{tc} \leqslant h_b - \sum_{c \in C} \Phi^c \theta_{bp}^{c} \qquad \forall\, b \in \mathcal{B},\, p \in \mathcal{P}^{l} \backslash u \tag{A.5}$$

Eqs. (A.1)–(A.5) are the block capacity constraints. For each block of the vessel, we distinguish between a TEU and a 20-foot capacity. This is due to the layout of the vessel since not all the blocks can stow 20-foot containers. In most of the cases k_b^{20} is, however, equal to k_b^{TEU}. Constraints (A.1) and (A.2) ensure that both of these capacities are satisfied. Similar to the TEU capacity constraints, the reefer constraints are split into a slot capacity constraint and a cell capacity constraint. Each reefer container needs to be plugged into an electrical output to get power for the cooling unit and r_b^{Slot} describes the number of such slots in a block b. A reefer cell is a cell where one of the slots has a reefer plug, and this number is denoted by r_b^{Cell}, if both slots in a reefer cell have a reefer plug, then $r_b^{Slot} = 2r_b^{Cell}$. In more general terms $r_b^{Cell} \leqslant r_b^{Slot} \leqslant 2r_b^{Cell}$ will be satisfied for all blocks. Constraint (A.3) restricts the number of reefer slots that can be used, and (A.4) is the reefer cell capacity constraint. Here we multiply with $\frac{1}{2}\Gamma^c$ as a 40-foot container occupies a full cell, whereas a 20-foot container occupies half a cell, and due to (A.3) the slot capacity will be met. Constraint (A.5) limits the total volume of the containers in a block. This is required to account for the fact that high-cube containers are higher and thus use more than a simple TEU capacity.

The set of constraints (A.1)–(A.5) defines a polyhedron with the feasible stowage plans. For the variables in the vector $\mathbf{y}_s$, this polyhedron is denoted by $\mathcal{Y}$ in the model in Section 4.

Appendix B. Stability Constraints[1]

The stability constraints ensure that the vessel does not capsize or break, even in tough weather conditions. The constraint here uses similar notations as the one introduced in Delgado (2013), and the interested reader is invited to look at the source for a more elaborate description.

The main difference between the set of constraints presented below and (6.25)–(6.43) of Delgado (2013), is the draft. Delgado (2013) does not consider a draft limit for each port, thus making it possible for the vessel to have a greater displacement. The introduction of the LinerLib (Brouer et al., 2014) makes this data easily available, and the draft limit is therefore included in this work. Also, Delgado considers locations, whereas this work considers blocks.

The functions describing the metacentre, trim, draft and buoyancy force all depend non-linearly on the displacement. However, Delgado (2013) shows that these functions can be approximated by linear planes by splitting the full displacement range into *displacement intervals*. A displacement interval is thus defined as a minimum, maximum (W_i^- and W_i^+) and an average (W_i) weight for the interval.

The sets, variables and parameters used in the model are introduced and explained below.

Sets:

$\mathcal{P}$	Set of ports.
$\mathcal{T}$	Set of ballast tanks.
$\mathcal{B}$	Set of blocks.
$\mathcal{I}$	Set of displacement intervals.
$\mathcal{BS}$	Set of bonjean stations.
$\mathcal{F}$	Set of frames.

Decision Variables:

w_{bp}	$\in \mathbb{R}^+$	Weight stowed in block $b \in \mathcal{B}$ at port $p \in \mathcal{P}$.
x_{tp}	$\in \mathbb{R}^+$	Weight of the tank $t \in \mathcal{T}$ at port $p \in \mathcal{P}$.

Auxiliary Variables:

v_p^W	$\in \mathbb{R}^+$	Vessel Displacement at port $p \in \mathcal{P}$.
v_{ip}^W	$\in \mathbb{R}^+$	Vessel Displacement in interval $i \in \mathcal{I}$ at port $p \in \mathcal{P}$.
v_{ip}^L	$\in \mathbb{R}$	Longitudinal centre of gravity at displacement interval $i \in \mathcal{I}$ at port $p \in \mathcal{P}$.
ψ_{ip}	$\in \{0, 1\}$	Indicator variable for displacement interval $i \in \mathcal{I}$ at port $p \in \mathcal{P}$.
v_p^V	$\in \mathbb{R}^+$	Vertical centre of gravity at port $p \in \mathcal{P}$.
v_p^{VM}	$\in \mathbb{R}^+$	Vertical moment at port $p \in \mathcal{P}$.
v_p^M	$\in \mathbb{R}^+$	Metacentre at port $p \in \mathcal{P}$.
v_{bsp}^{Bs}	$\in \mathbb{R}^+$	Buoyancy force of section between bonjean station bs and $bs + 1$ at port p.
$v_{f\alpha p}^S$	$\in \mathbb{R}^+$	Shear force fore or aft of frame $f \in \mathcal{F}$ at port $p \in \mathcal{P}$.
$v_{f\alpha p}^B$	$\in \mathbb{R}^+$	Bending moment fore or aft of frame $f \in \mathcal{F}$ at port $p \in \mathcal{P}$.

Parameters:

Various parameters:

W^O	Weight of the empty vessel.
Min^{GM}	Lower bound for metacentric height
Max_p^D	Maximum draft allowed at port p

Displacement intervals parameters:

[1] The stability constraints and explanation thereof in this section is identical to Appendix A of Christensen and Pacino (2017) and is only included here for the reader's convenience

$W_i^{\{-,+\}}$ Lower $(-)$ and upper $(+)$ bound of displacement interval $i \in \mathcal{I}$

W_i Average weight of displacement interval $i \in \mathcal{I}$

$A^W_{\{M,T,D,Bs\}}(W_i)$ Weight coefficient of displacement interval $i \in \mathcal{I}$ for the linearization of metacentre (M), trim (T), draft (D), and bonjean at station bs (Bs).

$A^L_{\{M,T,Bs\}}(W_i)$ Lcg coefficient of displacement interval $i \in \mathcal{I}$ for the linearization of metacentre (M), trim (T), and bonjean at station bs (Bs).

$A_{\{M,T,D,Bs\}}(W_i)$ Constant of displacement interval $i \in \mathcal{I}$ for the linearization of metacentre (M), trim (T), draft (D), and bonjean at station bs (Bs).

Centre of gravity parameters:

$\text{Min}_i^L/\text{Max}_i^L$ Min/maximum longitudinal centre of gravity at displacement interval $i \in \mathcal{I}$

$D_b^{\{L,V\}}$ Longitudinal (L), and Vertical (V) centre of gravity of block $b \in \mathcal{B}$

$D_t^{\{L,V,T\}}$ Longitudinal (L), Vertical (V), and transversal (T) centre of gravity of ballast tank $t \in \mathcal{T}$

Max^V Maximum vertical moment possible for the vessel.

LM^O Longitudinal moment of the empty vessel including constant weights.

VM^O Vertical moment of the empty vessel including constant weights.

TM^O Transversal moment of the empty vessel including constant weights.

Bending/Shearing parameters:

$W^S_{f\alpha}$ Constant weights fore or aft of frame $f \in \mathcal{F}$

$G^\alpha_{\{b,t,bs\}f}$ Fraction of block b, ballast tank t, and buoyancy section between bonjean stations bs and $bs + 1$ that lies fore or aft frame f

$W^B_{f\alpha}$ Bending components of the constant weight fore or aft of frame $f \in \mathcal{F}$

D_b^{Bs} Distance in meters between bonjean stations bs and $bs + 1$ multiplied by the density of water

$A^\alpha_{\{b,\, t,\, bs\}f}$ Fore or aft distance from frame f to the longitudinal centre of gravity of block b, ballast tank t, buoyancy section between bonjean stations bs and $bs + 1$

G_f Fore-based fraction of frame $f \in \mathcal{F}$, where $G_f \in [0;1]$. $G_f = 1$ when f is is the first frame at the bow, and $G_f = 0$ when f is the first frame at the stern.

$\text{Min}_f^{\{S,B\}}$ Lower bound for shear force (S) and bending moment (B) at frame $f \in \mathcal{F}$

$\text{Max}_f^{\{S,B\}}$ Upper bound for shear force (S) and bending moment(B) at frame $f \in \mathcal{F}$

With these the stability constraints can be modelled as seen below.

$$\sum_{t \in \mathcal{T}} x_{tp} + \sum_{b \in \mathcal{B}} w_{bp} + W^O = v_p^W \qquad \forall\, p \in \mathcal{P} \tag{B.1}$$

$$\sum_{i \in \mathcal{I}} W_i^- \psi_{ip} \leqslant v_p^W \leqslant \sum_{i \in \mathcal{I}} W_i^+ \psi_{ip} \qquad \forall\, p \in \mathcal{P} \tag{B.2}$$

$$\sum_{i \in \mathcal{I}} \psi_{ip} = 1 \qquad \forall\, p \in \mathcal{P} \tag{B.3}$$

$$\sum_{i \in \mathcal{I}} v_{ip}^W = v_p^W \qquad \forall\, p \in \mathcal{P} \tag{B.4}$$

$$W_i^- \psi_{ip} \leqslant v_{ip}^W \leqslant W_i^+ \psi_{ip} \qquad \forall\, i \in \mathcal{I},\, p \in \mathcal{P} \tag{B.5}$$

$$\text{Min}_i^L \psi_{ip} \leqslant v_{ip}^L \leqslant \text{Max}_i^L \psi_{ip} \qquad \forall\, i \in \mathcal{I},\, p \in \mathcal{P} \tag{B.6}$$

$$\sum_{b \in \mathcal{B}} D_b^L w_{bp} + \sum_{t \in \mathcal{T}} D_t^L x_{tp} + LM^O = \sum_{i \in \mathcal{I}} W_i v_{ip}^L \qquad \forall\, p \in \mathcal{P} \tag{B.7}$$

$$v_p^V W_i + (1 - \psi_{ip})\text{Max}^V \geqslant v_p^{VM} \qquad \forall\, i \in \mathcal{I},\, p \in \mathcal{P} \tag{B.8}$$

$$v_p^V W_i - (1 - \psi_{ip})\text{Max}^V \leqslant v_p^{VM} \qquad \forall\, i \in \mathcal{I},\, p \in \mathcal{P} \tag{B.9}$$

$$\sum_{b\in\mathcal{B}} D_b^V w_{bp} + \sum_{t\in\mathcal{T}} D_t^V x_{tp} + VM^O = v_p^{VM} \qquad \forall\, p\in\mathcal{P} \tag{B.10}$$

$$\sum_{t\in\mathcal{T}} D_t^T x_{tp} + TM^O = 0 \qquad \forall\, p\in\mathcal{P} \tag{B.11}$$

$$\sum_{i\in\mathcal{I}} A_M^W(W_i)v_{ip}^W + A_M^L(W_i)v_{ip}^L + A_M(W_i)\psi_{ip} = v_p^M \qquad \forall\, p\in\mathcal{P} \tag{B.12}$$

$$v_p^M - v_p^V \geqslant \mathrm{Min}^{GM} \qquad \forall\, p\in\mathcal{P} \tag{B.13}$$

$$\sum_{i\in\mathcal{I}} A_T^W(W_i)v_{ip}^W + A_T^L(W_i)v_{ip}^L + A_T(W_i)\psi_{ip} = 0 \qquad \forall\, p\in\mathcal{P} \tag{B.14}$$

$$\sum_{i\in\mathcal{I}} A_D^W(W_i)v_{ip}^W + A_D(W_i)\psi_{ip} \leqslant \mathrm{Max}_p^D \qquad \forall\, p\in\mathcal{P} \tag{B.15}$$

$$\sum_{s\in\{bs,bs+1\}} \sum_{i\in\mathcal{I}} A_{Bs}^W(W_i)v_{ip}^W + A_{Bs}^L(W_i)v_{ip}^L + A_{Bs}(W_i)\psi_{ip} \ = 2D_d^{Bs}v_{bsp}^{Bs} \qquad \forall\, bs\in\mathcal{BS},\, p\in\mathcal{P} \tag{B.16}$$

$$W_{f\alpha}^S + \sum_{b\in\mathcal{B}} G_{bf}^\alpha w_{bp} + \sum_{t\in\mathcal{T}} G_{tf}^\alpha x_{tp} - \sum_{bs\in\mathcal{BS}} G_{bsf}^\alpha v_{bsp}^{Bs} = v_{f\alpha p}^S \qquad \forall\, f\in\mathcal{F},\, \alpha\in\{A,\, F\}, p\in\mathcal{P} \tag{B.17}$$

$$W_{f\alpha}^B + \sum_{b\in\mathcal{B}} A_{bf}^\alpha G_{bf}^\alpha w_{bp} + \sum_{t\in\mathcal{T}} A_{tf}^\alpha G_{tf}^\alpha x_{tp} - \sum_{bs\in\mathcal{BS}} A_{bsf}^\alpha G_{bsf}^\alpha v_{bsp}^B = v_{f\alpha p}^B \qquad \forall\, f\in\mathcal{F},\, \alpha\in\{A,\, F\}, p\in\mathcal{P} \tag{B.18}$$

$$\mathrm{Min}_f^S \leqslant G_f v_{f,Fore,p}^S + (1-G_f)v_{f,Aft,p}^S \leqslant \mathrm{Max}_f^S \qquad \forall\, f\in\mathcal{F},\, p\in\mathcal{P} \tag{B.19}$$

$$\mathrm{Min}_f^B \leqslant G_f v_{f,Fore,p}^B + (1-G_f)v_{f,Aft,p}^B \leqslant \mathrm{Max}_f^B \qquad \forall\, f\in\mathcal{F},\, p\in\mathcal{P} \tag{B.20}$$

Constraint (B.1) calculates the displacement of the vessel for every port. The next constraint, (B.2), sets the displacement interval variables together with (B.3), which ensures that exactly one displacement interval is active at each port. Constraint (B.4) and (B.5) define v_{ip}^W to be equal to the displacement for the active displacement interval, and 0 for the rest. In a similar fashion (B.6) and (B.7) calculate the longitudinal centre of gravity (LCG). The centre of gravity is calculated as the sum of moments divided by the total displacement. The left-hand side of (B.7) calculates the sum of moments. This is done by considering the longitudinal centre of gravity for the tanks and blocks and multiplying with the weight stowed in these. The right-hand side uses the average weight of the displacement interval, instead of the actual displacement. Similar to v_{ip}^W, v_{ip}^L is zero for the displacement intervals for which $\psi_i = 0$, and for the active displacement interval it lies within the bounds defined by constraint (B.6). Constraints (B.8)–(B.10) approximate the vertical centre of gravity (VCG). Constraints (B.8) and (B.9) define the bound for the vertical moment for each displacement interval, and in the case where $\psi_i = 1$ the two inequalities turn into an equality $v_p^V W_i = v_p^{VM}$. For the non-active displacement intervals, these two constraints have no effect. Constraint (B.10) calculates the vertical moment similar to how the LCG is calculated, but without multiplying the right-hand side by the weight. (B.11) ensures that the transversal centre of gravity is 0, meaning the middle of the vessel. Due to the construction of the blocks, the transversal centre of gravity is 0 for all blocks, and thus only the tanks and the moment of the empty vessel are considered in the calculation. Constraints (B.12)–(B.16) calculate the metacentre, trim, draft and buoyancy force by using the linearization of the non-linear functions. Each of the planes for the functions are described using three factors, $A^W(W_i)$, $A^L(W_i)$ and $A(W_i)$. $A^W(W_i)$ is the displacement factor, $A^L(W_i)$ is the LCG factor, and $A(W_i)$ is the constant factor. (B.12) calculates the metacentre, and (B.13) defines the metacentric height to be greater than the minimum metacentric height allowed. In (B.14) the trim is required to be zero, and (B.15) enforces that the draft be less than or equal to Max_p^D. As the trim is required to be zero, the draft does not depend on the LCG, but only the displacement of the vessel. Max_p^D is the minimum draft allowed when leaving port p, and will thus be the minimum of the draft at port p and port $p+1$. Constraint (B.16) calculates the buoyancy force (*bonjean*) between station bs and $bs+1$. The last four constraints (B.17)–(B.20) are related to the stress forces. The first two calculate the shearing and bending, and (B.19) and (B.20) define the upper and lower bounds. The shear force on a vessel, at a given frame, is the integral of forces on either side of the frame, and the bending moment is the integral of moments on either side of the frame. The buoyancy forces are only approximated, and thus there is an accumulation of error when calculating the shear force and the bending moment. To reduce the impact of this error, constraint (B.17) and (B.18) respectively calculate the shear forces and bending moment with respect to the resulting forces acting fore and aft of the frame. Hence, there are two shear variables for every frame at each port. Constraint (B.19) and (B.20) respectively set the limits for the shear force and bending moment at each frame. The shear force and bending moment at a frame are estimated by using a proportional calculation based on the position of the frame. This reduces the impact of the error accumulation as the fore-based computation is accurate in the bow and the aft-based computation is accurate in the stern. All these constraints ensure that the vessel is stable and can be declared seaworthy if the stacking rules are obeyed.

The set of constraints (B.1)–(B.20) defines a polyhedron with the feasible weight allocations. For the variables in the vector $\mathbf{w}_s$, this polyhedron is denoted by $\mathcal{W}$ in the model in Section 4.

References

Ambrosino, D., Anghinolfi, D., Paolucci, M., Sciomachen, A., 2009. A new three-step heuristic for the master bay plan problem. Mar. Econ Logist. 11, 98–120.

Ambrosino, D., Anghinolfi, D., Paolucci, M., Sciomachen, A., 2010. An experimental comparison of different heuristics for the master bay plan problem. In: Festa, P. (Ed.), Experimental Algorithms. Lecture Notes in Computer Science, vol. 6049. Springer, Berlin Heidelberg, pp. 314–325.

Ambrosino, D., Paolucci, M., Sciomachen, A., 2015a. Computational evaluation of a MIP model for multi-port stowage planning problems. Soft Comput. 1–11.

Ambrosino, D., Paolucci, M., Sciomachen, A., 2015b. A MIP heuristic for multi port stowage planning. Transport. Res. Proc. 10, 725–734 18th Euro Working Group on Transportation, EWGT 2015, 14-16 July 2015, Delft, The Netherlands.

Ambrosino, D., Sciomachen, A., 1998. A constraint satisfaction approach for master bay plans. Water Stud. Ser. 36.

Ambrosino, D., Sciomachen, A., Tanfani, E., 2004. Stowing a containership: the master bay plan problem. Transport. Res. Part A: Policy Pract. 38, 81–99.

Aslidis, A.H., 1984. Optimal Container Loading. Master Thesis.

Avriel, M., Penn, M., Shpirer, N., Witteboon, S., 1998. Stowage planning for container ships to reduce the number of shifts. Ann. Oper. Res. 76, 55–71.

Botter, R.C., Brinati, M.A., 1992. Stowage container planning: a model for getting an optimal solution. In: Proceedings of the IFIP TC5/WG5.6 Seventh International Conference on Computer Applications in the Automation of Shipyard Operation and Ship Design, VII. North-Holland Publishing Co., Amsterdam, The Netherlands, pp. 217–229.

Bredstrom, D., Rönnqvist, M., 2006. Supply chain optimization in pulp distribution using a rolling horizon solution approach. Norwegian School of Economics Dept. of Finance & Management Science Discussion Paper No. 2006/17.

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

Chand, S., Hsu, V.N., Sethi, S., 2002. Forecast, solution, and rolling horizons in operations management problems: a classified bibliography. Manuf. Serv. Oper. Manage. 4, 25–43.

Christensen, J., Pacino, D., 2017. A matheuristic for the cargo mix problem with block stowage. Transport. Res. Part E: Logist. Transport. Rev. 97, 151–171.

Delgado, A., 2013. Models and Algorithms for Container Vessel Stowage Optimization.

Delgado, A., Jensen, R., Janstrup, K., 2012. A constraint programming model for fast optimal stowage of container vessel bays. Euro. J. Oper. Res. 220, 251–261.

Ding, D., Chou, M.C., 2015. Stowage planning for container ships: a heuristic algorithm to reduce the number of shifts. Euro. J. Oper. Res.

Dubrovsky, O., Levitin, G., Penn, M., 2002. A genetic algorithm with a compact solution encoding for the container ship stowage problem. J. Heurist. 585–599.

Feng, C.M., Chang, C.H., 2008. Optimal slot allocation in intra-asia service for liner shipping companies. Mar. Econ. Logist. 10, 295–309.

Imai, A., Sasaki, K., Nishimura, E., Papadimitriou, S., 2006. Multi-objective simultaneous stowage and load planning for a container ship with container rehandle in yard stacks. Euro. J. Oper. Res. 171, 373–389. https://doi.org/10.1016/j.ejor.2004.07.066.

Kang, J.G., Kim, Y.D., 2002. Stowage planning in maritime container transportation. J. Oper. Res. Soc. 53, 415–426.

Li, F., Tian, C., Cao, R., Ding, W., 2008. An integer linear programming for container stowage problem. Comput. Sci. - ICCS 2008, 853–862.

Low, M., Xiao, X., Liu, F., Huang, S., Hsu, W.J., Li, Z., 2009. An automated stowage planning system for large containerships. In: Proceedings of the 4th Virtual Int. Conference on Intelligent Production Machines and Systems.

Pacino, D., Delgado, A., Jensen, R.M., Bebbington, T., 2011. Fast generation of near-optimal plans for eco-efficient stowage of large container vessels. In: Böse, J.W., Hu, H., Jahn, C., Shi, X., Stahlbock, R., Voß, S. (Eds.), Computational Logistics. Lecture Notes in Computer Science, vol. 6971. Springer, Berlin Heidelberg, pp. 286–301.

Pacino, D., Jensen, R., 2012. Constraint-based local search for container stowage slot planning. Lect. Notes Eng. Comput. Sci. 2, 1467–1472.

Parreno, F., Pacino, D., Alvarez-Valdes, R., 2016. A grasp algorithm for the container stowage slot planning problem. Transport. Res. Part E: Logist. Transport. Rev. 94, 141–157.

Rakke, J.G., Stlhane, M., Moe, C.R., Christiansen, M., Andersson, H., Fagerholt, K., Norstad, I., 2011. A rolling horizon heuristic for creating a liquefied natural gas annual delivery program. Transport. Res. Part C: Emerg. Technol. 19, 896–911.

Sciomachen, A., Tanfani, E., 2003. The master bay plan problem: a solution method based on its connection to the three-dimensional bin packing problem. IMA J. Manage. Math. 14, 251–269.

Wilson, I., Roach, P., 2000. Container stowage planning: a methodology for generating computerised solutions. J. Oper. Res. Soc. 51, 1248–1255.

Zhang, Z., Lee, C.Y., 2016. Multiobjective approaches for the ship stowage planning problem considering ship stability and container rehandles. IEEE Trans. Syst. Man Cybernet.-Syst. 46, 1374–1389. https://doi.org/10.1109/TSMC.2015.2504104.