Simultaneous Optimization of Container Ship Sailing Speed and Container Routing with Transit Time Restrictions

Karsten, Christian Vad; Røpke, Stefan; Pisinger, David

Publication date:
2015

Document Version
Publisher's PDF, also known as Version of record

Simultaneous Optimization of Container Ship Sailing Speed and Container Routing with Transit Time Restrictions

Management Science

DTU Management Engineering

Christian Vad Karsten
Stefan Ropke
David Pisinger

10.2015

DTU Management Engineering
Department of Management Engineering

Simultaneous Optimization of Container Ship Sailing Speed and Container Routing with Transit Time Restrictions

Christian Vad Karsten, Stefan Ropke, and David Pisinger
DTU Management Engineering,
The Technical University of Denmark

October 26, 2015

Abstract

We introduce a decision support tool for liner shipping companies to optimally determine the sailing speed and needed fleet for a global network. As a novelty we incorporate cargo routing decisions with tight transit time restrictions on each container such that we get a realistic picture of the utilization of the network. Furthermore, we show that it is possible to extend the model to include optimal time scheduling decisions such that the time associated with transshipments is also reflected accurately. To solve the speed optimization problem we propose an exact algorithm based on Benders decomposition and column generation that exploits the separability of the problem. Computational results show that the method is applicable to liner shipping networks of realistic size and that it is important to incorporate cargo routing decisions when optimizing speed.

1 Introduction

Liner shipping companies operate a set of sailing routes to provide transport for containers so as to maximize their revenue. Once the strategic decisions of which markets to serve have been made by a carrier and the sailing routes have been determined, most companies will adjust the network continuously. This is done due to changes in the global economic environment such as fluctuations in fuel prices, freight rates, and container demand. One way of optimizing the profitability of the network is to minimize the cost related to the operation of the routes, the deployment of vessels, and the handling of cargo. However, Karsten et al. (2015) recently showed that this approach will likely result in prolonged transit times. From a customer perspective not only low cost but the *level of service* offered is of concern. The level of service represents both the transportation cost and the transit time provided for a given cargo. Therefore, among the most influential decisions is the sailing speed between the serviced ports and the deployment of the available fleet. Higher sailing speeds will offer better transit times to the customers but will at the same time be more expensive to operate as there is an inherent trade-off in operating a low cost network versus a competitive network which is optimized in terms of both cost and offered cargo transit times. In the longer perspective, changes in sailing speed will also affect strategic decisions regarding the required fleet size as sailing routes, usually called *rotations*, are cyclic and require a weekly frequency, i.e. for a route the number of vessels deployed will correspond to the number of weeks it takes one vessel to complete a round trip, which will vary greatly depending on the sailing speed. The sailing speed has a significant impact on the operating costs as bunker may constitute more than 75% of the total operating cost of a vessel (Ronen, 2011). Furthermore, the consumption is approximately

cubic in speed. Therefore, it is important to have a model that accurately assesses the impact of changes in sailing speed both from an operational, tactical and strategic perspective.

As a novelty we integrate the problems of sailing speed optimization, fleet deployment, and time constrained cargo routing as the decisions are highly dependent. Our model and solution method is aimed at optimizing the sailing speed for all sailing legs or a subset of these in a global liner shipping transportation network so as to maximize profit. For each rotation the fleet deployment can be adjusted maintaining weekly frequency and the speed between any pair of serviced ports is selected from a discrete set of speeds based on the characteristics of the deployed vessel class. The model is solved using Benders decomposition (constraint generation) where the rows are generated by solving a time constrained multi-commodity flow problem using column generation. That way we select the optimal sailing speed for each sailing leg under consideration of cargo transit time restrictions. This also means that by speeding up, new cargo that is not currently transported may become available to transport. Furthermore, we show in Appendix A that it is possible to extend the model to include optimal time scheduling decisions such that the port arrival and departure times, and the time associated with transshipments is also reflected accurately. The extended model makes it possible to determine an optimal time schedule and corresponding optimal sailing speeds while considering optimal routings of cargo subject to transit time restrictions.

Christiansen et al. (2004, 2013) provide comprehensive reviews of the more recent literature on ship routing and scheduling. Brouer et al. (2014a) and Meng et al. (2014) give an introduction to the domain of liner shipping and an overview of recent literature specific to this area. The literature on optimization of liner shipping networks has been growing significantly during the last decade, and several planning problems at both the strategic, tactical and operational level have been addressed. Notteboom and Vernimmen (2009) and Ronen (2011) give background on speed optimization in liner shipping and show the importance of optimizing speed in liner shipping networks by studying a single rotation. Wang and Meng (2012c) formulate the speed optimization problem in a liner shipping network as a non-linear MIP. Cargo routing is considered for a pre-defined set of container routes where all demand must be met and cost is minimized in the model. Cheaitou and Cariou (2012) propose a model that explore and incorporate the available demands' dependence on transit time. Gelareh and Meng (2010) present a model for fleet deployment in a network where they also determine the sailing speed necessary to meet all demand while minimizing costs. Meng and Wang (2011) study the same problem for a single rotation. Similarly Zacharioudakis et al. (2011) optimize speed in a fleet deployment model, which they solve by assigning ships using a genetic algorithm. Xia et al. (2015) present a heuristic for optimizing fleet deployment and speed in an aggregated network but do not consider transshipments. They report computational results based on an aggregated network of up to 18 nodes. Psaraftis and Kontovas (2013) survey models and taxonomy on speed optimization in maritime transportation and Psaraftis and Kontovas (2015) discuss the practice of slow steaming. A related tactical problem is studied by Wang and Meng (2012a) who consider fleet deployment and transit time in a space time network and Wang and Meng (2012b) who study a tactical schedule model, where cost is minimized while maintaining a required transit time under uncertainty. Karsten et al. (2015) develop an efficient algorithm for the time-constrained cargo routing problem. This problem arise as a sub-problem in many of the tactical and strategic planning problems encountered by liner shipping companies when level of service is considered.

1.1 Industry Practice

The current practice used by major liner shipping companies is to vary speed across each of the operated rotations. Figure 1 shows the speed profile for three rotations recently operated by one of the leading global carriers, Maersk Line. It is clearly seen that speed is varied along the rotation and that it is rarely operated at or near the average speed for the entire rotation. The main driving

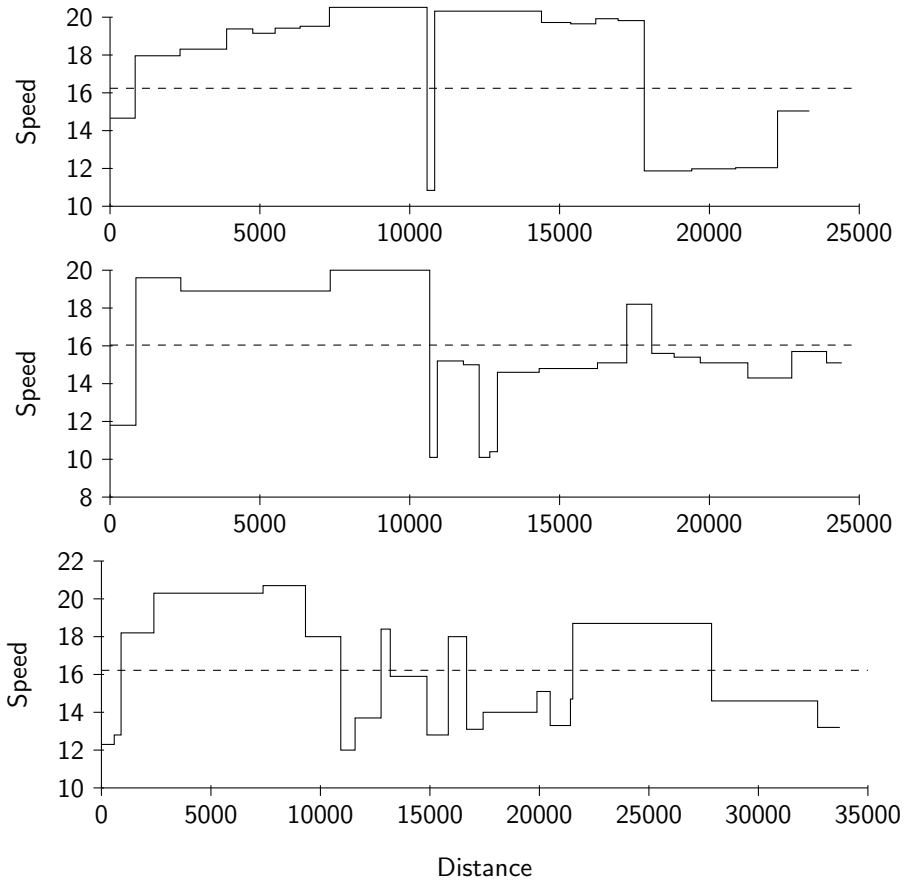

Figure 1: Speed profiles for three different rotations (AsiaEurope1, AsiaEurope10, AsiaEurope6TP6) operated by Maersk Line. The distance is in nautical miles and speed in knots. The steps corresponds to the average speed between waypoints (ports, canals etc.) on the rotation. Hence, some parts of the rotations might be operated at a lower or higher speed than showed. The dashed line is the average speed for the entire rotation.

factors in determining the sailing speed is the fuel price and whether the vessel is on its head or back haul, i.e. sailing in the cargo intensive direction or not. Most empirical findings as well as hydrodynamics suggest that the fuel consumption per time unit for container vessels is proportional to the third power of the sailing speed. In other words the fuel consumption per unit distance is proportional to the second power of the sailing speed. However, it is vessel dependent and the relationship can best be derived empirically. There is some evidence that for certain weather and hull conditions the bunker consumption can be greater than cubic in the speed, (Kontovas and Psaraftis, 2011) and, for large container vessels sailing at high speed, the power requirement may even be proportional to the fourth power of sailing speed (Man, 2013). For a fleet of vessels Wang and Meng (2012c) found the exponent to be between 2.7 and 3.3 empirically. In accordance with this, and following the benchmarks in Brouer et al. (2014a), we assume a third power relationship in the rest of this paper. For this relationship reducing speed by 20% can give up to a 50% reduction of fuel consumption and corresponding emissions for a vessel, or up to a 35% reduction in fuel consumption for a rotation since it requires operating additional vessels in order to meet

demand. However, some time critical transportation requests may not be available if operating at reduced speeds. As a consequence, lowering transportation cost while offering competitive cargo transit times (and a low number of transshipments) presents an inherent trade-off as fuel cost is the most important factor contributing to the operational cost of a network. This means that it may be worth selecting a more "expensive" rotation configuration which offers better connections. However, this may also save one vessel on the rotation. To address this we introduce a model to optimize the sailing speed and fleet deployment in a liner shipping network while considering a tight transit time restriction on each individual container. To solve the model we propose a decomposition based algorithm based on simultaneous column and row generation.

The rest of the paper is organized as follows. Section 2 introduces the needed transportation network. Section 3 describes the optimization problem and Section 4 shows the decomposition, derives stronger Benders cuts, and discusses how additional Benders cuts can be generated. Section 5 describes the solution algorithm. Computational results are presented in Section 6 for the speed optimization problem before finally discussing possible extensions and concluding in Section 7. Appendix A shows how the model and solution method can be extended to include a time schedule. Appendix B discusses additional model improvements.

2 Transportation Network

Figure 2 illustrates a basic container shipping network. In this example the network is composed of two rotations R_1 and R_2 visiting various ports (nodes) and the solid black arcs correspond to sailing arcs. Containers can be transported between any pairs of ports and if the origin and destination port is not serviced by the same vessel, it can be transshipped between rotations at intermediate ports where rotations meet. In Figure 2 containers can be transshipped between rotation R_1 and R_2 in node w by using a transshipment arc which has an associated cost and time. Depending on the time schedule of the two rotations, the delay associated with the transshipment can be determined. In the following an exact time schedule is not known so an estimated transshipment time is used. Optimization with an actual time schedule is further addressed in Appendix A. The capacity of each arc is determined by the size of the vessel deployed and the time it takes to traverse an arc by the sailing speed. As there are two vessels assigned to each rotation the total round trip time for each rotation must be two weeks to satisfy the weekly frequency requirement used by most liner shipping companies (Brouer et al., 2014a), but the speed on each sailing leg can vary greatly as discussed in the previous section.

Figure 3 a) shows an example of the transportation network we use in the model before speeds have been selected. We duplicate each sailing leg (dashed black lines in the figure) and assign different possible speeds to the duplicates such that cost and transit time is known a priori. E.g. between port k and l it is possible for the model to choose between three different speeds, V_1, V_2, and V_3. Figure 3 b) shows an example flow from k to t where the speeds for each leg have been selected. The sailing speed between e.g. k and l is selected at V_1. This way it is possible to calculate the transit time from k to s as the length of each sailing leg divided by the selected speed for each of the legs plus the average transshipment time. If the total transit time from k to s exceeds the requirement for a commodity going from k to s, there is no feasible path and it may be worth adjusting the speeds. If the speed is increased significantly on all legs in a rotation, it may be possible to reduce the number of vessels and still maintain weekly frequency. Likewise it may be necessary to use an extra vessel if speed is reduced significantly on all legs.

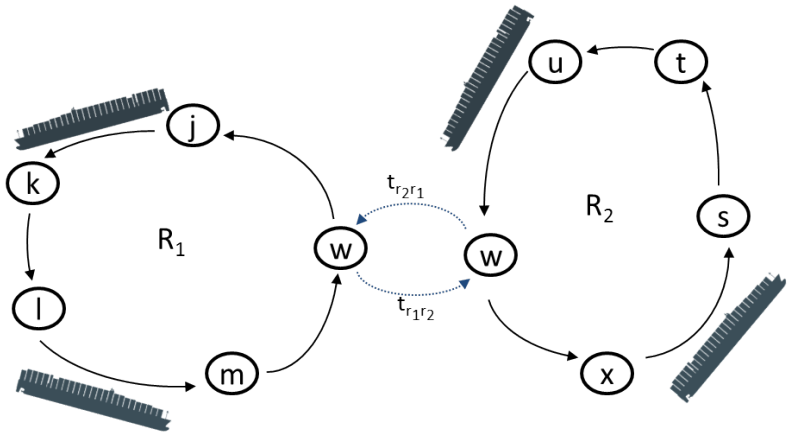

Figure 2: Simple representation of a liner shipping transportations network. The nodes correspond to ports and the arcs to sailing legs. In port w it is allowed to transship containers between rotation R_1 and R_2.

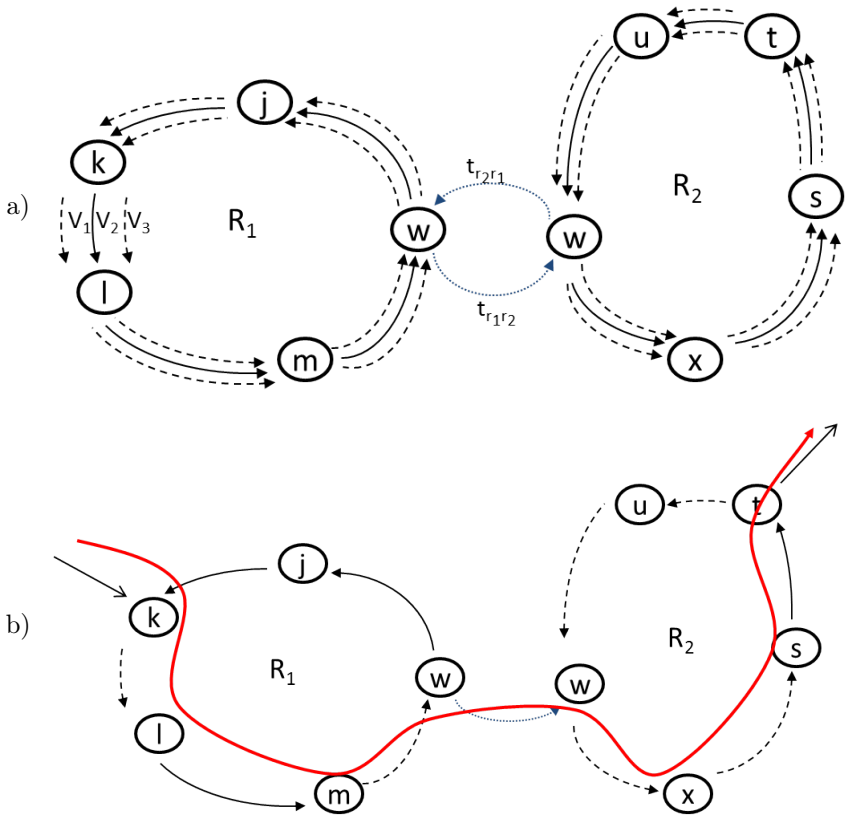

Figure 3: Figure a) a transportation network used for speed optimization. Figure b) the flow through a network at a spcific speed where load and unload arcs are included to correctly account for cost and time. The nodes correspond to ports and port w allows transshipments between rotation R_1 and R_2.

3 Mathematical Modeling

To formulate the sailing speed optimization problem we formally define the graph described in the previous section $G = (N, A)$ with nodes N and directed arcs A. The arcs in A are based on the original sailing arcs in an existing network, $\bar{A}$, at different speeds and hence A contains multiarcs. This is illustrated in Figure 3 a). Here multiple arcs are shown which are based on the original sailing arcs, $\bar{A}$, shown in Figure 2. Therefore, if, for example, every original sailing arc is considered at three different speeds, then $|A| = 3|\bar{A}|$, as illustrated by Figure 3 a). For important sailing legs the optimal sailing speed can be determined in greater detail (by considering more arcs) than at other sailing legs which are less flexible, e.g. because of fixed berthing times at both the departure and arrival port. To earn a revenue there is a set of commodities K that can be transported through the network between various origin-destination pairs. The amount of commodity $k \in K$ that is available to be transported is d^k. W.l.o.g. we assume that each commodity has a single origin node and a single destination node. Furthermore, let q_a be the capacity of arc $a \in A$ and t_a be the travel time for arc $a \in A$ measured in days, including port time in the destination port. The decision variables x_a specify whether arc $a \in A$ is used. The amount of commodity $k \in K$ that is routed through path p is determined by y_p^k. The set of possible paths for commodity k is denoted P^k and the set of all paths is denoted P. Only paths that satisfy the given transit time restriction for commodity k are included. The integer decision variable L_r specifies the number of vessels used for rotation r. The set of rotations is denoted R. The set of rotations using vessel class $v \in V$ is given by $R(v)$, and N^v specifies the number of available vessels of class v. The set of arcs that can be used by a rotation is denoted $E(r) \subseteq A$. The set $P(a, k)$ contains the set of paths for commodity $k \in K$ using arc $a \in A$. The multiarcs corresponding to a given sailing arc, $\bar{a}$, at different speeds is denoted $A(\bar{a})$. The cost of using arc $a \in A$ is c_a and it includes the portion of the vessels fuel used at this arc sailing at the corresponding speed. Hence, the model easily allows different bunker consumption rates for different vessel classes and the non-linearity of the consumption as a function of speed is handled through the multiarcs. The cost of using a vessel at rotation r is C^r and the cost of sending commodity k through path p is r_p^k. A negative cost corresponds to a profitable path where a revenue can be obtained. The revenue includes loading, unloading, and transshipment costs and additionally there is a service penalty for not meeting demand. The costs are handled by introducing additional load, unload, and transshipment arcs as described in Karsten et al. (2015). Additionally, these arcs make sure we obtain the correct travel time for cargo through the network, including loading, unloading and transshipment time. We use the same objective (costs, revenues and penalties) as described in the reference model by Brouer et al. (2014a) and a negative objective value indicates a profitable network. As we wish to maximize profit, this corresponds to minimizing the following objective in the integrated sailing speed optimization and cargo routing model, which is given by

$$\min \sum_{a \in A} c_a x_a + \sum_{r \in R} C^r L^r + \sum_{k \in K} \sum_{p \in P^k} r_p^k y_p^k \tag{1}$$

subject to

$$\sum_{a \in A(\bar{a})} x_a = 1 \qquad \bar{a} \in \bar{A} \tag{2}$$

$$\sum_{p \in P^k} y_p^k \leq d^k \qquad k \in K \tag{3}$$

$$\sum_{k \in K} \sum_{p \in P(a,k)} y_p^k \leq x_a q_a \qquad a \in A \tag{4}$$

$$\sum_{a \in E(r)} t_a x_a \leq 7 L_r \qquad r \in R \tag{5}$$

$$\sum_{r \in R(v)} L_r \leq N^v \qquad v \in V \tag{6}$$

$$y_p^k \in \mathbb{R}_+ \qquad k \in K,\ p \in P^k \tag{7}$$

$$x_a \in \{0,1\} \quad a \in A \tag{8}$$

$$L^r \in \mathbb{Z}_+ \qquad r \in R \tag{9}$$

The objective (1) maximizes the total profit by minimizing the variable and fixed cost as well as the transportation cost (a negative transportation cost corresponds to a profitable path). Constraints (2) ensure that only one of the multiarcs at different speeds is selected such that a vessel is assigned exactly one speed at arc a. Constraints (3) assign cargo to paths to meet the demand or reject it if not profitable. Constraints (4) make sure that flow is only permitted on the selected arcs and the capacity of the arc is not violated. Finally, Constraints (5) and (6) ensure enough vessels are assigned to all rotations to meet the weekly frequency requirement without violating the fleet availability for each vessel class. Here the time t_a is measured in days and includes the port time in the origin port. If t_a is measured in hours the right-hand-side of (4) should be multiplied by the number of hours per week (168) rather than the days per week.

4 Decomposition of the Speed Optimization Problem

The model (1)-(9) is difficult to solve directly since it contains a large number of variables. The number of y_p^k variables can grow exponentially in the size of the graph. One solution approach would be to solve the LP relaxation of the model using column generation and obtain integer solutions using a branch-and-price algorithm (see e.g. Barnhart et al. (1998) for more information about branch-and-price).

Here we suggest a different approach. We notice that the y_p^k variables all are continuous. This means that we can apply Benders decomposition to model (1)-(9) and place the constraints related to the y_p^k variables in the sub-problem. The sub-problem in Benders decomposition has to be solved by column generation but the master problem can be solved either using a standard integer programming solver or using a branch and cut framework that allows the user to add cut callbacks. Both approaches are typically simpler to implement compared to a full branch-and-price algorithm. A Benders decomposition algorithm furthermore has the advantage that it continuously produces feasible solutions such that the method works as a heuristic when it is stopped before optimality is reached. A potential drawback is that algorithms based on Benders decomposition have a reputation of converging slowly.

In the following we are going to review the parts of Benders decomposition algorithm that are necessary for our application. The presentation is largely based on Costa (2005). In general we

have a mixed integer problem (MIP1)

$$\min cx + dy$$

subject to

$$\begin{aligned} Ax + By &\geq b \\ Dx &\geq e \\ x &\in \mathbb{Z}^{n_1} \\ y &\in \mathbb{R}^{n_2} \end{aligned}$$

Let $X = \{x \in \mathbb{Z}^{n_1} : Dx \geq e\}$ then MIP1 can be reformulated as:

$$\min_{\bar{x} \in X} \Big\{ c\bar{x} + \min\{dy \,:\, By \geq b - A\bar{x}, y \in \mathbb{R}^{n_2}\} \Big\} \tag{10}$$

The inner minimization is a linear program ($\bar{x}$ are merely constants in this problem), which we denote the *primal Benders sub-problem* (PBSP). To ease the following we will assume that the inner minimization problem is feasible and bounded for all choices of $\bar{x} \in X$ since this is the case for our decomposition (as will be explained in the sequel). We note that in general Benders decomposition also applies when these assumptions do not hold, but is slightly more complex to handle, see for example Benders (1962) or Costa (2005) for details.

If we let π be the dual variables corresponding to $By \geq b - A\bar{x}$ then we can write the dual of the inner minimization as:

$$\max\{\pi(b - A\bar{x}) \,:\, \pi B \leq d, \pi \geq 0\}.$$

This problem is denoted the *dual Benders sub-problem* (DBSP)**.** Since we assumed $\min\{dy \,:\, By \geq b - A\bar{x}, y \in \mathbb{R}^{n_2}\}$ to be feasible and bounded the PBSP will be feasible and bounded as well. Using the DBSP and strong duality we can rewrite (10) to:

$$\min_{\bar{x} \in X} \Big\{ c\bar{x} + \max\{\pi(b - A\bar{x}) \,:\, \pi B \leq d, \pi \geq 0\} \Big\} \tag{11}$$

Here we notice that the constraints of the inner maximization problem are independent on the choice of $\bar{x} \in X$. Furthermore, $F = \{\pi B \leq d, \pi \geq 0\}$ is bounded and non-empty due to our assumptions and we can use Minkowski-Weyl's Theorem to express F using a set of extreme points $\Pi = \{\pi_1, \ldots, \pi_q\}$. DBSP will have an optimal solution at one of the extreme points in Π and we can reformulate (11) to:

$$\min_{\bar{x} \in X} \Big\{ c\bar{x} + \max\{\pi(b - A\bar{x}) \,:\, \pi \in \Pi\} \Big\}$$

using an auxiliary variable $z \in \mathbb{R}$ this problem can be written as:

$$\min c\bar{x} + z$$

subject to

$$z \geq \pi(b - A\bar{x}) \qquad \pi \in \Pi \tag{12}$$

$$\bar{x} \in X \tag{13}$$

$$z \in \mathbb{R} \tag{14}$$

This problem is denoted the *Benders master problem* (BMP). The constraints (12) are known as *optimality cuts*. This BMP is usually solved in an iterative fashion since the cardinality of Π is such that enumerating all extreme points is out of question.

The lower bound on the optimal objective value is always a monotonic increasing function as it is obtained from the relaxed master problem where more and more constraints, (extreme points), are added and the formulation continuously gets tighter. However, the upper bound or objective value is not guaranteed to be a monotonic decreasing function since it is just produced by a sequence of feasible solutions but by maintaining the best found solution the algorithm converges to the optimal solution.

We apply Benders decomposition to the speed optimization problem (1)-(9) such that (x_a, L_r) are found in the Benders master problem, BMP. This means that constraints (2), (5), (6), (8) and (9) are moved to the master problem while constraints (3), (4), (7) are moved to the primal sub-problem, which is given by:

$$\min \sum_{k \in K} \sum_{p \in P^k} r_p^k y_p^k \tag{15}$$

subject to

$$\sum_{p \in P^k} y_p^k \leq d^k \qquad k \in K \tag{16}$$

$$\sum_{k \in K} \sum_{p \in P(a,k)} y_{p_k}^k \leq \bar{x}_a q_a \quad a \in A \tag{17}$$

$$y_p^k \in \mathbb{R}_+ \qquad k \in K,\ p \in P^k \tag{18}$$

where $\bar{x}_a$ is the value of the x_a variables chosen in the master problem. It is intuitive to view constraint (17) as the two constraints (19) and (20)

$$\sum_{k \in K} \sum_{p \in P(a,k)} y_{p_k}^k \leq q_a \quad a \in O(\bar{x}) \tag{19}$$

$$\sum_{k \in K} \sum_{p \in P(a,k)} y_{p_k}^k \leq 0 \quad a \in C(\bar{x}) \tag{20}$$

where $O(\bar{x})$ and $C(\bar{x})$ denote "open" and "closed" arcs. The open arcs are the arcs with $x_a = 1$ in the BMP and the closed arcs have $x_a = 0$. The PBSP is always feasible since setting all $y_{p_k}^k$ equal to 0 produces a feasible solution. It is also bounded since constraints (16) and (18) ensure that $0 \leq y_{p_k}^k \leq d^k$ for all $a \in A$, $k \in K, p_k \in P^k$.

The PBSP can be identified as the cargo routing multi-commodity flow problem, MCF, which can be solved efficiently using column generation. When iterating through solutions from the BMP, this method allows that some columns can be reused. We introduce side constraints on the transit time such that we solve a time-constrained multi-commodity flow problem and hence determine an optimal speed selection taking transit time restrictions into consideration as done in Karsten et al. (2015).

To derive the optimality cuts for the BMP we associate with (16)-(20) the non-positive dual variables α_k^i, δ_a^i, and λ_a^i. Then an extreme point solution gives a new Benders cut, which can be added to the BMP

$$z_0 \geq \sum_{k \in K} \alpha_k^i d_k + \sum_{(a) \in O(\bar{x})} \delta_a^i q_a x_a + \sum_{(a) \in C(\bar{x})} \lambda_a^i q_a x_a \tag{21}$$

With the set of all Benders cuts, BC, the BMP can be written as:

$$\min \sum_{a \in A} c_a x_a + \sum_{r \in R} C^r L^r + z_0 \tag{22}$$

subject to

$$z_0 \geq \sum_{k \in K} \alpha_k^i d_k + \sum_{(a) \in O(\bar{x})} \delta_a^i q_a x_a + \sum_{(a) \in C(\bar{x})} \lambda_a^i q_a x_a \quad i \in BC \tag{23}$$

$$\sum_{a \in A(\bar{a})} x_a = 1 \quad \bar{a} \in \bar{A} \tag{24}$$

$$\sum_{a \in E(r)} t_a x_a \leq 7 L_r \quad r \in R \tag{25}$$

$$\sum_{r \in R(v)} L_r \leq N^v \quad v \in V \tag{26}$$

$$x_a \in \{0, 1\} \quad a \in A \tag{27}$$

$$L^r \in \mathbb{Z}_+ \quad r \in R \tag{28}$$

Where the Benders cuts (23) are added iteratively.

4.1 Decomposition and Solution of the MCF Sub-problem

The PBSP, (15)-(18), is the path-flow formulation of a multi-commodity flow problem. It has |A|+|K| constraints, but the number of variables (paths) grows exponentially with the size of the graph in the worst case. The necessary variables can be generated dynamically using another decomposition technique, namely column generation, and in practice the path-flow model can be solved efficiently even for very large scale instances, see Karsten et al. (2015). Column generation works with a reduced version of the LP (15)-(18) defined by a reduced set of columns $\bar{P}^k$ for each commodity k such that a feasible solution can be found using variables from $\cup_{k \in K} \bar{P}^k$. Solving this LP gives rise to dual variables α_k and δ_a corresponding to constraint (16) and (17), respectively. For a variable $j \in P$ let $\kappa(j)$ denote the commodity that a variable serves, $p(j)$ the path (set of arcs) corresponding to the variable j, and $c_a^{\kappa(j)}$ the cost of sending one unit through arc a. The *reduced cost* $\bar{c}_j$ of each path variable $j \in P$ is $\bar{c}_j = \sum_{a \in p(j)} (c_a^{\kappa(j)} - \delta_a) - \alpha_{\kappa(j)}$ and we wish to find variables such that $\bar{c}_j < 0$, as this variable can potentially improve the current LP solution and give new dual variables. To find a variable with negative reduced cost or prove that no such variable exists, we solve a shortest path problem for each commodity from the source to the destination on the reduced cost graph. As we want to accommodate the transit time restrictions for each commodity, we use a resource constrained shortest path algorithm with time as the resource to ensure that the transit time of each generated path is less than or equal to the maximum transit time for the given commodity as described in Karsten et al. (2015). Transit time is in addition to the sailing legs calculated by considering the multi-commodity flow problem on a graph including transshipment, loading, and unloading arcs.

We can add Benders cuts based on the LP-relaxation of the BMP as the right hand side of (17) is multiplied by the capacity q_a of each arc, $a \in A$, such that this will correspond to solving the same time constrained MCF problem but on a multi graph where the "fractional" capacity of parallel arcs will sum to the original capacity.

In both cases we can warm start the column generation procedure by using the columns from previous configurations.

4.2 Strengthening the Benders Cuts

From duality we can gain some additional insights on the dual values associated with the "closed" arcs. Let $p(O(\bar{x}))$ and $p(C(\bar{x}))$ be the set of "open" and "closed" arcs used by path p. The DBSP is

$$\max \sum_{k \in K} \alpha_k + \sum_{a \in O(\bar{x})} q_a \delta_a + \sum_{a \in C(\bar{x})} 0\lambda_a \tag{29}$$

subject to

$$\alpha_k + \sum_{a \in p(O(\bar{x}))} \delta_a + \sum_{a \in p(C(\bar{x}))} \lambda_a \leq r_p^k \quad k \in K,\ p \in P^k \tag{30}$$

$$\alpha_k, \delta_a, \lambda_a \leq 0 \quad k \in K,\ a \in A \tag{31}$$

Since we want the Benders cut to be as strong as possible, we can optimize λ_a in the dual as it does not contribute to the objective value. For a given solution where we obtain the dual values α_k^* and δ_a^* we want to find an alternative solution where α_k and δ_a take the values of α_k^* and δ_a^* but where $\sum_{a \in C} \lambda_a \geq \sum_{a \in C} \lambda_a^*$. This can be done by solving the following problem

$$\max \sum_{a \in C(\bar{x})} \lambda_a \tag{32}$$

subject to

$$\sum_{a \in p(C(\bar{x}))} \lambda_a \leq r_p^k - \alpha_k^* - \sum_{a \in p(O(\bar{x}))} \delta_a^* \quad k \in K,\ p \in P^k \tag{33}$$

$$\lambda_a \leq 0 \qquad a \in C \tag{34}$$

Let y_p^k be the dual corresponding to constraint (33), then we get the dual problem (corresponding to the PBSP)

$$\min \sum_{k \in K} \sum_{p \in P^k} \left((r_p^k - \sum_{a \in p(O(\bar{x}))} \delta_a^*) - \alpha_k^* \right) y_p^k \tag{35}$$

subject to

$$\sum_{k \in K} \sum_{p \in P(a,k)} y_{p_k}^k \leq 1 \quad a \in C(\bar{x}) \tag{36}$$

$$y_p^k \geq 0 \quad k \in K,\ p \in P^k \tag{37}$$

The problem has a similar structure to the original problem and can be solved using column generation as well. For the set of columns, Δ^k, the reduced cost for a path variable $l \in \cup_{k \in K} \Delta^k$ with original revenue/cost $r_p^k - \alpha_k^* - \sum_{a \in p(O(\bar{x}))} \delta_a^*$ is given by the following resource constrained shortest path problem

$$\bar{c_l} = \sum_{(a) \in p(l)} (c_a - \delta_a^* - \lambda_a) - \alpha_{k(l)}^* \tag{38}$$

The columns for a given commodity are added to the master problem when the reduced cost is less than the revenue associated with the commodity. When the solution to (32)-(34) is different from the initially found duals we can add an additional Benders cut. However, this cut does not necessarily dominate the original cut. Again we can reuse all columns corresponding to paths using at least one closed arc to warm start the column generation.

4.3 Generating Additional Benders Cuts

Generally, the Benders decomposition approach is more successful for standard multi commodity network design problems when the BSP decomposes into even smaller sub-problems. It can decompose e.g. by commodity (Gendron, 2011) or by equipment type (Cordeau et al., 2000) and especially if these can be solved by special purpose algorithms (Magnanti and Wong, 1981) such that more cuts can be added very effectively in each iteration. In the present problem the multi-commodity flow problem is not separable by commodity or equipment type but it is still possible to generate several alternative cuts in each iteration. In Appendix B we describe a method for generating cuts in other areas of the solution space based on the solution found to the multi-commodity flow problem.

4.4 Valid Inequalities

In this section we consider valid inequalities for the Benders master problem. We will solely focus on inequalities defined on the x_a and L_r variables, thus omitting the z variable.

We first introduce a lower limit on the number of needed vessels at a rotation by looking at the maximum speed for each arc in a rotation.

$$L_r \geq y^r_{min} = \left\lceil \sum_{\bar{a} \in \bar{A}(r)} \min_{a \in A(\bar{a})} t_a/7 \right\rceil \quad r \in R. \tag{39}$$

We add (39) to the BMP to strengthen the model. We can also define

$$L_r \leq y^r_{max} = \left\lceil \sum_{\bar{a} \in \bar{A}(r)} \max_{a \in A(\bar{a})} t_a/7 \right\rceil \quad r \in R \tag{40}$$

which will not cut away any optimal solutions, but is strictly speaking not a valid inequality.

Next, we consider valid inequalities that can be constructed based on the frequency constraints (25) along with the domain definitions for the variables and the arc selection constraint (24). In other words, for each $r \in R$ we are interested in the set

$$\mathcal{B}_r = \Big\{ x_a \in \{0,1\} \quad \forall a \in E(r), L_r \in \mathbb{Z}_+, \\ \sum_{a \in E(r)} t_a x_a \leq 7L_r, \sum_{a \in A(\bar{a})} x_a = 1, \quad y_{min} \leq L_r \leq y_{max} \Big\}$$

and valid inequalities for the polyhedron

$$F_r = conv\{\mathcal{B}_r\}.$$

Some families of valid inequality for a similar polyhedron (without the arc selection constraint) have been proposed in Atamtürk and Rajan (2002). For the instances we are considering it is, in practice, relatively easy to optimize over F_r, and we have observed that the arc selection constraint improves the performance compared to not including it. Therefore, we will attempt to find any possible valid inequality for the polyhedron using a cut-finding LP. The use of cut-finding LPs to find all violated valid inequalities for a given polyhedron has, for example, been used by Boyd (1994), Boccia et al. (2008) and Kaparis and Letchford (2010).

4.4.1 Separation

For a given solution to the relaxation of the master problem, (x_a^*, L_r^*), we wish to determine a valid inequality $\sum_{a\in E(r)} \pi_a x_a + \pi_r L_r \leq \beta_r$ for F_r that is violated by the solution. We are to determine the values of π_a, π_r and β_r such that the inequality is valid and violated by (x_a^*, L_r^*). We say that $\sum_{a\in E(r)} \pi_a x_a^* + \pi_r L_r^* - \beta_r$ is the *violation* of the inequality. This can be done by solving the following LP

$$\max \sum_{a\in E(r)} \pi_a x_a^* + \pi_r L_r^* - \beta_r \tag{41}$$

subject to

$$\pi_a \tilde{x}_a + \pi_r \tilde{L}_r \leq \beta_r \quad (\tilde{x}_a, \tilde{L}_r) \in \mathcal{B}_r \tag{42}$$
$$-1 \leq \pi_a \leq 1 \quad a \in E(r) \tag{43}$$
$$-1 \leq \pi_r \leq 1 \tag{44}$$
$$\pi_a, \pi_r, \beta_r \in \mathbb{R} \quad a \in E(r) \tag{45}$$

The objective function (41) maximizes the violation of the inequality. The first constraints (42) ensure that the inequality is satisfied by all solutions from $\mathcal{B}_r$ and therefore is a valid inequality and constraints (43)-(44) normalizes the inequality. Without these constraints, the LP would be unbounded whenever a violated inequality exists (since such a constraint can be scaled to yield any violation). Given that the π_a and π_r are now bounded we can further limit the range of β_r:

$$-|E(r)| - y_{max}^r \leq \beta_r \leq |E(r)| + y_{max}^r \quad a \in E(r)$$

Since the set $\mathcal{B}_r$ can be prohibitively large, we initially remove the constraints (42) and add them dynamically when violated. Given a solution $(\pi_a^*, \pi_r^*, \beta_r^*)$ to the partial cut-finding LP (41), (43)-(45) and a subset of constraints (42) the separation problem for constraints (42) is

$$\max \left\{ \sum_{a\in E(r)} \pi_a^* x_a + \pi_r^* L_r - \beta_r^* : (x_a, \beta_r) \in \mathcal{B}_r \right\}$$

which written in full is:

$$\max \sum_{a\in E(r)} \pi_a^* x_a + \pi_r^* L_r - \beta_r^* \tag{46}$$

subject to

$$\sum_{a\in E(r)} t_a x_a \leq 7 L_r \qquad a \in E(r) \tag{47}$$
$$\sum_{a\in A(\bar{a})} x_a = 1 \qquad \bar{a} \in \bar{A} \tag{48}$$
$$y_{min}^r \leq L_r \leq y_{max}^r \tag{49}$$
$$x_a \in \{0, 1\} \qquad a \in E(r) \tag{50}$$
$$L_r \in \mathbb{Z}_+ \tag{51}$$

If this IP has a positive value function then we have detected a solution $(\tilde{x}_a, \tilde{L}_r)$ in $\mathcal{B}_r$ that is violated by the inequality given by $(\pi_a^*, \pi_r^*, \beta_r^*)$ and we add

$$\pi_a \tilde{x}_a + \pi_r \tilde{L}_r \leq \beta_r$$

to the cut finding LP and resolve.

5 Algorithm

Traditional implementations of the Benders decomposition algorithm follow a cutting plane approach where the reduced master problem is solved iteratively to optimality, and a constraint of type (23) (Benders cut) is added in each iteration based on the sub-problem. This procedure is followed iteratively until optimality is reached (or the bounds are within an acceptable tolerance). This has the downside that too much time may be spent on proving optimality and re-processing nodes of the branch-and-bound tree that has already been cut off every time a new Benders cut is added. However, most modern branch-and-bound solvers make it possible to effectively make branch-and-cut algorithms where cuts are added using a callback routine as described by Bai and Rubin (2009) and Fortz and Poss (2009). Using callbacks makes it possible to add cuts efficiently both at integer and LP solutions whenever one is found. This comes at the cost of potentially adding too many cuts, but a node will never have to be revisited. Both implementations have advantages for different types of problems, which we will discuss in the computational section. To improve the implementation of the Benders algorithm, we also test the effect of solving the LP-relaxation of the BMP (we relax the variables related to the number of vessels on a rotation (28) and arc selection variables (27)). After solving the LP-relaxation of the BMP, we add the Benders cuts obtained from this to an initial pool of cuts before eventually solving the integral version of the problem. This procedure has been shown to be very effective by e.g. Cordeau et al. (2001) and Fortz and Poss (2009). Additionally, we add one warm starting cut a priori based on a known initial configuration of the network, which is is usually quite good and hence can be expected to improve performance. We terminate the algorithm when a relative gap of 1 % between the best found solution and the lower bound is achieved, and set the tolerance of the mixed integer programming solver to 1 % as well. The valid inequalities described in Section 4.4 based on the frequency constraints are added dynamically as cuts using callbacks in both the traditional and branch-and-cut approach. They are added to the LP-relaxation as well as the BMP, but only at the root node. The inequalities (39) are always added a priori. The column generation procedure for solving the multi-commodity flow problem is re-using previously generated columns to warm start the algorithm for each new configuration of the network, but to manage the number of columns, unused columns are deleted every 100^{th} iteration of the overall algorithm. Additionally, we keep columns generated for the initial configuration. In the column generation procedure used to generate the strengthened Benders cuts described in Section 4.2, we only reuse columns containing closed arcs to warm start the procedure.

The model is implemented in C++. We use the Boost Graph Library to handle the graph construction. The BMP is solved using Gurobi 6.0 and the PBSP using the COIN-OR linear programming solver. All tests were performed using a single thread on a computer with an Intel Xeon CPU X5550 2.67GHz. We allow the algorithm to run for up to three hours and if the LP-relaxation is solved initially, up to one of the three hours is dedicated to this.

6 Computational Results

We test the algorithm as a post-processing tool on networks created based on realistic data from *Linerlib* (Brouer et al., 2014a) using the matheuristic described in Brouer et al. (2014b, 2015). A summary of the considered networks can be seen in Table 1. We use the average speed configuration for generating the warm starting cut, whereas in a real world network we would have an already optimized configuration which often could give a good quality cut. If there are no transit time sensitive demands being transported by a given rotation, the most cost effective configuration will be sailing all legs with average speed while maintaining a weekly frequency.

We create multigraphs by considering up to five possible sailing arcs for each original sailing arc

Name	$\|R\|$	$\|K\|$	$\|\bar{A}\|$	$\|A\|$
Baltic	3	22	14	46
WAF	10	37	40	130
Mediterranean	5	365	58	189
Pacific	14	722	141	464
WorldSmall	26	1764	287	951

Table 1: Characteristics of the considered networks. $|R|$ is the number of rotations, $|K|$ is the number of commodities, $|\bar{A}|$ is the number of original sailing arcs, and $|A|$ is the number of potential sailing arcs. In addition to the sailing arcs our formulation adds (un)load and transshipment arcs.

in the rotation(s) being optimized such that the input speed, as well as duplicates corresponding to $\pm 10\%$ and $\pm 25\%$ speed change, are considered and added if the resulting speed is feasible for the vessel class. For the Panamax 2400 vessel class sailing at 17 nm/h on a specific arc this corresponds to arcs at 12.75, 15.3, 17, 18.7, and 21.25 since these are all within the speed limits of 12 and 22 nm/h. We set the loading and unloading time to one day and the transshipment time to two days as in Brouer et al. (2015).

We consider optimization at three levels. Each rotation can be optimized separately (while still considering the cargo routing in the entire network), all rotations using the same vessel class can be optimized jointly, and finally all the rotations in a network can be optimized simultaneously. Additionally, rotations can be optimized such that the number of vessels used by each rotation is maintained i.e. L_r is $fixed$ at the current deployment or the number of vessels used by each rotation can be optimized as well such that the fleet deployment for each rotation, L_r, is $flexible$ within the bounds given by the total available fleet. Optimizing for a fixed fleet will be more relevant under operational planning whereas the flexible fleet optimization is more targeted at tactical planning. In both cases the cargo routing in the entire network is considered. In the following we consider optimization at the three levels for a fixed and flexible fleet. Additionally, we will discus the impact of the different proposed algorithmic improvements for the vessel class and network optimization. In the vessel class optimization we present averaged results over several classes, but when we find it relevant we also refer to the underlying detailed results, which are not shown to keep the size of the tables manageable.

6.1 Single Rotation Optimization

Figure 4 shows the improvement for each rotation in the WorldSmall instance where the fleet is flexible. The average improvement over all rotations is 2 % and the maximum improvement is 8 % of the total profit. For most rotations the total cargo routed is unchanged or slightly reduced (less than 0.2 % change in total volume) and for three of the rotations the speed optimization leads to a slight increase in volume of the cargo routed. For all rotations the deployment is either the same or increased i.e. the overall average speed is decreased. This illustrates that in this case the improvements in profit are mainly driven by an overall speed decrease (some sailing legs maintain or increase speed) and change in deployment, but in a few cases also by an increase in the volume of cargo routed. Looking at the container paths used for the cargo routing in the optimized solution (not just different speed but new itinerary or different rotation between same ports) reveals that it is essential to consider cargo routing as part of the speed optimization. On average 9 % of the container paths used in the best found solution are not used by the initial solution. The variation is between 0 % and 20 %. Some of the difference may be accounted for by parallel paths having the same cost and itinerary but using different rotations. Hence, from an objective function point of view they are equally good, but lead to different routings. In practice it may be desired to have as much of the routing unchanged as possible when optimizing speed, and simple modifications

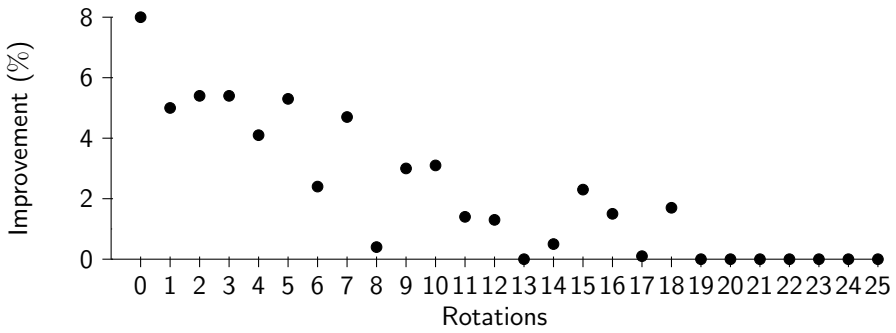

Figure 4: Percentage improvement in profit in the WorldSmall instance when optimizing each rotation. The improvements are sorted according to the runtime for the branch-and-cut implementation without any improvements.

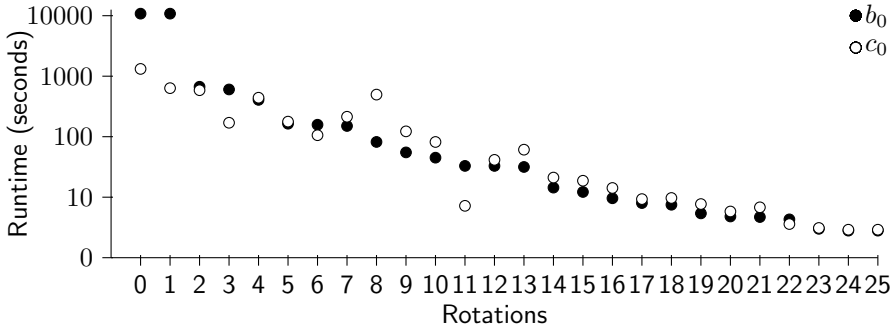

Figure 5: Runtime for each rotation in the WorldSmall instance sorted according to the runtime for the traditional Benders implementation without any improvements b_0. c_0 is the branch-and-cut implementation without any improvements. Notice the logarithmic scale of the y-axis.

that give preference to unchanged flow can easily be incorporated in the multi-commodity flow problem.

Figure 5 shows the runtime for each rotation in the WorldSmall instance for the traditional and branch-and-cut implementation of the Benders algorithm without any algorithmic improvements. The runtimes are sorted according to the traditional implementation. For the rotations that take longer time to optimize, the branch-and-cut implementation is generally faster than the traditional implementation, and it also solves all instances within the time limit, whereas the gap is not closed for two of the rotations using the traditional approach. For the rotations where the optimal solution is found in short time, the traditional implementation converges to the desired solution quality slightly faster.

6.2 Vessel Class Optimization

When the number of vessels is restricted, all configurations of the network may not be possible and e.g. an overall speed decrease of all rotations may use more vessels than are available. Hence, it is desirable to optimize rotations with the same class deployed simultaneously rather than each rotation individually, as it leads to overly optimistic profit improvements corresponding to infeasible deployments. In the following we show results for the vessel class optimization in the WorldSmall instance where a major decision in the speed optimization process is the deployment of vessels to

rotations. The results can be found in Table 2 and some details on solution characteristics for each vessel class (F450, F800, P1200, P2400, PostP, and SuperP where F is feeder and P is Panamax) in the flexible deployment case can be found in Table 3.

Average over all vessel classes in WorldSmall																				
Traditional implementation	Branch-and-cut implementation	LP-relaxation cuts	Warm start cut	Strengthened cuts	Node relaxation cuts	Valid inequalities	Improvement in profit (%)	Final gap (FUB-FLB)/|FUB| (%)	Final UB gap (FUB*-FUB)/|FUB*| (%)	Final LB gap (FUB*-FLB)/|FUB*| (%)	Initial LB gap (FUB*-ILB)/|FUB*| (%)	Runtime for solved classes (seconds)	Solved classes (out of 6)	Improvement in profit (%)	Final gap (FUB-FLB)/|FUB| (%)	Final UB gap (FUB*-FUB)/|FUB*| (%)	Final LB gap (FUB*-FLB)/|FUB*| (%)	Initial LB gap (FUB*-ILB)/|FUB*| (%)	Runtime for solved classes (seconds)	Solved classes (out of 6)
Setting							**Fixed deployment**							**Flexible deployment**						
×							0.2	4.0	0.2	3.8	-	23	5	4.8	34.5	1.3	32.4	-	23	3
×		×					0.1	0.2	0.3	0.2	0.7	336	6	0.3	33.7	5.3	25.1	27.6	27	3
×		×	×				0.1	0.5	0.3	0.2	0.9	1197	6	3.2	21.6	2.7	17.7	20	23	3
×		×	×	×			0.1	0.5	0.3	0.2	0.9	22	5	1.6	23.7	4.1	17.7	20	27	3
×		×	×			×	0.1	2.4	0.3	2.2	2.8	26	5	2.7	52.7	4.5	43.6	45	31	3
	×						0.2	2.6	0.1	2.4	-	7	5	3.3	51.1	1.7	47.0	-	12	3
	×	×					0.2	0.6	0.2	0.4	0.4	20	5	3.3	11.1	1.7	9.0	9.5	47	3
	×	×	×				0.2	0.5	0.1	0.4	0.4	455	6	3.0	11.7	2.0	9.1	9.6	42	3
	×	×	×	×			0.2	0.5	0.1	0.4	0.4	651	6	3.0	11.6	1.9	9.1	9.6	79	3
	×	×	×		×		0.2	0.5	0.1	0.4	0.4	427	6	2.2	12.4	2.7	8.9	9.6	53	3
	×	×	×			×	0.2	0.5	0.3	0.4	0.4	283	6	4.3	9.6	0.8	8.6	8.7	71	3

Table 2: Computational Results: The algorithm is tested in the traditional implementation of the algorithm and using callbacks. Improvement in profit is the improvement in profit obtained relative to the average speed configuration. Runtime for solved classes is the average time to converge if this is reached within the time limit of 3 hours. It includes the time used at solving the LP-relaxation and up to 1 hour is dedicated to this. FUB is the final upper bound (i.e. the best found solution), FUB* is the best final upper bound found across the different algorithmic settings (i.e. the overall best found solution), FLB is the final lower bound, and ILB is the initial lower bound

Fixed Deployment

As seen in Table 2, the potential improvements in profit are less than 1 % on average for all classes when the fleet is fixed and speed changes of 10 % and 25 % on each sailing leg are allowed. However, even 0.2 % is significant for a global liner shipping network. The detailed results show that in all cases the volume of containers routed is either maintained or increased slightly (less than 1 % increase in volume transported). Generally, the solution times are low but the branch-and-cut implementation is superior. The traditional implementation only solves the problem to within the desired tolerance within the time limit for all six vessel classes when Benders cuts based on the LP-relaxation and the warm start cut is added initially. In the branch-and-cut implementation all six classes are solved when the LP-relaxation is solved initially and a warm start cut is added. The best average solution time, where all six classes were solved, is achieved when valid inequalities are also added to the LP-relaxation as well as at the root node of the integral BMP, and the

algorithmic improvements focusing on the bound does not hurt performance as all six instances are solved. For the five instances that are solved by all configurations, the pure branch-and-cut implementation without any improvements is superior.

Flexible Deployment

Table 2 also shows results for the flexible fleet case and Table 3 shows the solution characteristics for the best found solution for each class. For three of the vessel classes (F450, F800, and SuperP) the solution corresponding to the initial solution cannot be improved in the setting where speed changes of 10 % and 25 % on each sailing leg are allowed, and for these three classes all configurations of the algorithm find the optimal solution within the time limit. For the other three classes (P1200, P2400, and PostP) the best solution is in all cases obtained using the branch-and-cut implementation. Still, on average the traditional implementation with no improvements finds the best solution, but the final lower bound is generally poor. Improvements up to 12.8 % are found and it is clear that most of the improvements are due to changes in deployment such that the overall average sailing speed is lowered. For rotations assigned the vessel classes F450, F800, and SuperP no improvements larger than 1 % can be made, and the detailed computational results for these (not shown) show that the optimization terminates quickly. This characteristic was also found in the single rotation optimization case. For the three remaining vessel classes, P1200, P2400, and PostP significant improvements in the profit can be made. Inspection of the solutions also reveal that the sailing speed on some sailing legs are maintained or increased to meet critical transit times for some demands, and as seen only a few demands cannot be met. For all classes when deployment is flexible the volume of containers routed is either maintained or slightly decreased (less than 0.6 %). Further improvements may be achievable for all classes if different/more sailing speeds are considered. For the best found solution the weekly fuel cost is reduced from $ 68 mio. to $ 64 mio., the weekly time charter rate of 11 additional vessels is $ 1.6 mio. and cargo revenue decreases from $ 132.2 mio to $ 131.7 mio.

If we look at all rotations in the WorldSmall instance using the P1200 vessel class we see a significant improvement in profit and the smallest gap of the classes leading to an improved solution. Here we use 11 extra vessels which is exactly what is available in the instance. If we sum the profit for the results of the individual rotations from Section 6.1, which are all solved to the desired tolerance, a total improvement in profit of 9 % is possible, but it also requires 12 additional vessels, which are not available. For the PostP vessel class seven additional vessels are available, but the optimal solution to the optimization of the individual rotations uses 12 additional vessels in this case. For the P2400 vessel class the best found solution uses the same number of vessels as the optimal solution for each of the individual rotations, so in this case the single rotation optimization could provide a very good warm start solution (but not optimal as transit time critical containers may use two linked rotations from the same class where joint optimization could lead to a lower selected speed on both). As expected there is a larger change in container paths used for the cargo routing when optimizing all rotations in a class rather than a single rotation. On average 11 % of the container paths used in the solutions are not used by the initial solution and the variation is between 0 % and 40 %. The variation is correlated with how much the network is improved, but the largest improvements do not necessarily lead to the largest changes in the container routing.

Adding valid inequalities and the solution of the LP-relaxation initially improves the performance of the algorithm in terms of improving both the lower and upper bound. Adding the strengthened Benders cuts and Benders cuts at node relaxations generally do not improve performance in terms of best solution for vessel class optimization given the time limit, but does improve the bound. The number of basic Benders cuts added by the algorithm (not reported) is generally lower when additional/strengthened cuts are added as one "iteration" takes longer time. (We have implemented the additional cuts discussed in Appendix B and also here the number of iterations is

reduced, but it generally neither improves or deteriorates performance significantly). This means that the derived cuts including the ones discussed in Appendix B do improve the "per iteration" performance, and for the instances that are solved the number of added basic Benders cuts is lower. When we are interested in solutions quickly (or better solutions with less guarantees on quality) it can be advantageous to use a more basic implementation to reduce the time spent on improving the LB. On the other hand the convergence of the LB is very slow when no improvements are made and if the gap is too large only poor solutions may be found. The initial lower bound gap is only reported when the LP-relaxation of the problems is solved initially (otherwise a dash) and here the branch-and-cut implementation usually has better progress, and the detailed results show that more Benders cuts are added within the time limit. When the valid inequalities are added in the traditional implementation, the progress on the LP is very slow whereas they improve performance in the branch-and-cut implementation. Warm starting the algorithm using a known configuration helps to ensure a good initial solution and possible improvements can quickly be assessed.

Vessel class (size in FFE)	Number of (rotations)	Flow (chg. in %)	Additional vessels deployed	Profit (chg. in %)	Final gap (in %)
F450	3	0	0	0	<1
F800	3	0	0	0	<1
P1200	5	-0.3	11	8	6
P2400	5	-0.6	11	13	27
PostP	7	-0.6	6	10	19
SuperP	1	0	0	0	<1

Table 3: Solution characteristics of the best found solution for each vessel class in the WorldSmall instance with flexible vessel deployment. We report the flow as the change in volume of the container routing, the number of additional vessels deployed, the change in overall profit for the network, and the final gap for the best found solution.

6.3 Network Optimization

The majority of cargoes in real liner shipping networks uses up to one transshipment to travel from origin to destination port. Still, a significant amount of cargo uses two or more transshipments. Hence, speed optimization decisions across several rotations, potentially with different capacity, influence the cargo routing. The proposed method allows optimization of an entire network or parts of a network e.g. within a region or some other grouping of rotations based on current container routing. It should be noted that in practice all rotations in a global network are usually not optimized simultaneously.

Table 4 shows the results for optimization of networks where all rotations are optimized simultaneously. We show results for two algorithmic settings. Setting s_1 is the branch-and-cut implementation with a warm start cut and solution of the LP initially. Setting s_2 is as s_1, but we also add the strengthened Benders cuts, Benders cuts at node relaxations and valid inequalities at the LP and the root node. Generally, setting s_1 is faster on small instances and for the larger instances setting s_2 improves the initial and final gap significantly. For instances covering the Baltic and WAF, the algorithm performs very well and finds optimal solutions quickly (showing that no improvement can be made with fixed deployment) in both setting s_1 and s_2. For all instances it is seen that valid inequalities significantly improve the initial gap in setting s_2. In both Pacific and WorldSmall there are issues with convergence of the LP and a significant final gap is reported. In WorldSmall no improvement is found within the time limit, but we know from the vessel class optimization that a significant improvement is achievable. However, in the flexible case, setting s_2 provides a better initial and final gap whereas setting s_1 is better in the more constrained fixed

Network														
Branch-and-cut implementation	LP-relaxation cuts	Warm start cut	Strengthened cuts	Node relaxation cuts	Valid+ inequalities	Improvement in profit (%)	Final gap (FUB-FLB)/|FUB| (%)	Initial LB gap (FUB*-LPLB)/|FUB*| (%)	Runtime (seconds)	Improvement in profit (%)	Final gap (FUB-FLB)/|FUB| (%)	Initial LB gap (FUB*-LPLB)/|FUB*| (%)	Runtime (seconds)
Algorithmic setting						**Fixed deployment**				**Flexible deployment**			
Baltic													
×	×	×				0	0	0	0.01	0	0	0	0.01
×	×	×	×	×	×	0	0	0	0.01	0	0	0	0.01
WAF													
×	×	×				0	0.2	0.2	0.2	1.9	0.9	4.2	0.5
×	×	×	×	×	×	0	0.0	0.0	0.5	1.9	0.7	1.9	4.5
Mediterranean													
×	×	×				2.0	1.1	2.4	t.l.	2.0	1.2	1.8	t.l.
×	×	×	×	×	×	1.9	1.2	1.7	t.l.	2.0	1.2	1.6	t.l.
Pacific													
×	×	×				2.1	2.8	4	t.l.	3.1	48	108	t.l.
×	×	×	×	×	×	1.9	3.2	3.8	t.l.	7.9	70	73	t.l.
WorldSmall													
×	×	×				0.7	3.3	3.6	t.l.	0.0	374	375	t.l.
×	×	×	×	×	×	0.4	5.1	6.1	t.l.	0.0	365	349	t.l.
×	×	×				1.6	3.6	1.7	12h	0.0	265	262	12h
×	×	×	×	×	×	0.4	6.3	4.4	12h	5.2	275	256	12h

Table 4: Computational Results: The algorithm is tested on networks where all rotations are open to optimization. Improvement in profit is the improvement in profit obtained relative to the average speed configuration. Runtime is the total time to converge to within the desired tolerance including the time used at solving the LP and "t.l." indicates that the time limit of three hours has been reached. LPLB is the lower bound obtained from solving the LP and FUB is the final upper bound. The last row for WorldSmall show results where the time limit has been increased by a factor four and up to 4 hours can be spent on solving the LP.

case. If the time limit is increased by a factor 4 such that we allow 12 hours in total and up to 4 hours at solving the LP, setting s_2 is able to improve the network by 5.2 % whereas setting s_1 still is not. Furthermore, we see that solving the LP initially is very effective in improving the bound (but still it does not converge within the time limit), whereas less improvement is achieved in the integer phase. Generally, for the larger instances setting s_1 shows better performance when the fleet is fixed, and setting s_2 shows better performance when the fleet is flexible.

The network results show that the method in its current form is not suitable for optimization of an entire network as it has problems with convergence for the largest instances. However, for simultaneous optimization of all rotations in a network the algorithmic performance can potentially

be improved by solving the problem in several stages. Initially the single rotation optimization can serve as input to vessel class optimization, which can eventually serve as input to optimization of the full network. To find a feasible solution that can serve as input for the class optimization, the rotations can be ranked according to expected impact and fleet usage can be updated after optimization of each rotation. The solution found in the class optimization will always be a feasible solution for optimizing the entire network.

7 Conclusion

In this paper we introduced a novel model and solution method for liner shipping companies to optimally determine the sailing speed of one or several rotations simultaneously in a global network. In the model and solution method we consider level of service explicitly, and incorporate cargo routing decisions with tight transit time restrictions on each commodity in the entire network. Furthermore, we show in Appendix A that it is possible to extend the model to include optimal time scheduling decisions in the model. The solution method is based on Benders decomposition and column generation and we show that it is able to effectively improve the profit of global size networks. We have used a state of the art algorithm to generate the networks used for testing, and our results show that variable speed on each sailing leg and fleet deployment can lead to large savings in the network design process. Also, we have shown that speed changes can lead to significant changes in the routing of the containers in a global network, and hence it is critical to consider routing implications when optimizing speed in networks where transit time restrictions are tight. In addition to the model and solution method we have proposed several algorithmic enhancements, which all helps improving the bound on the solution. The model and algorithm can be used as a basis for the development of decision support tools in liner shipping companies and it applies in both tactical and operational settings. We believe that even if extended with the time scheduling component described in Appendix A smaller initial gaps may be expected in an actual planning setting since a good configuration is already in place. Often only speed changes will be considered for smaller parts of the networks and potentially close to the existing operation in terms of speed variation. The model and solution method can handle the large and complex planning problems faced by leading liner shipping companies as networks are usually not entirely changed, but merely incrementally improved to new operating conditions. Therefore, future work could be in the direction of solving more restricted problems to improve the upper bounds and obtain solutions faster. Additionally, this can help improve the lower bounds. Finally, searching for improving solutions in the proximity (Fischetti and Monaci, 2014) of the best known solutions rather than based on relaxations may improve the performance.

Acknowledgements

This project was supported by The Danish Maritime Fund under the Competitive Liner Shipping Network Design project and by the Danish Strategic Research Council and EUDP under the GREENSHIP project. We wish to thank managers and planners at Maersk Line for providing insights and data on their current network operations.

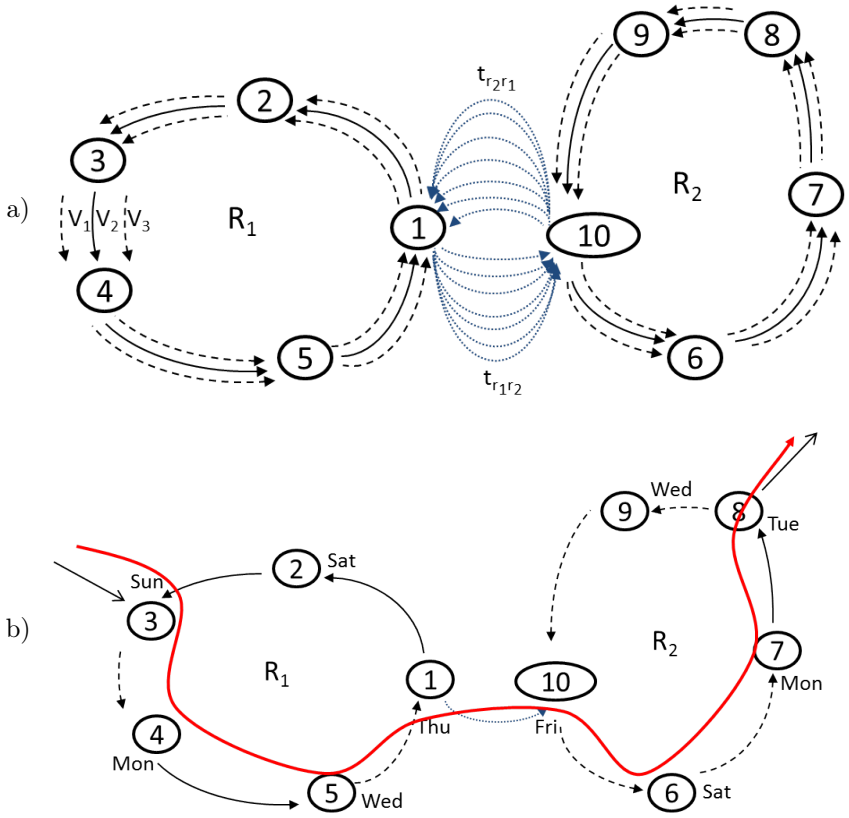

Figure 6: Figure a) a transportation network used for speed and schedule optimization (the problem considered in the BMP). Figure b) the flow through a network at a spcific speed and schedule (the problem considered in the PBSP). The nodes correspond to port calls, which are numbered consecutively around each service. The ten port calls correspond to nine physical ports (the port calls 1 and 10 correspond to the same physical port) and it is possible to transship between rotation R_1 and R_2 by using a transshipment arc from 1 to 10.

A Optimal Time Scheduling

It is possible to add time scheduling constraints to the model to include determination of an optimal time schedule while satisfying the transit time restrictions. The time schedule determines the timing of events, such as port calls, along a service. For each port we introduce a variable to determine the departure time, and at the same time we introduce variables to reflect transshipment time between rotations. These are coupled with the flow variables such that it is possible to consider the influence of schedules in the flow calculations. The time scheduling and transshipment decisions are included in the BMP. Duplicates of the transshipment arcs are considered with different layover time corresponding to all possible schedules, and a binary variable is associated with each of the transshipment arcs. The selected transshipment arc is included in the PBSP. If the departure time for some or all ports are given a priori, this is easy to include by fixing part of the network.

Figure 6 is an extension of the example in Figure 3 and Figure 6 a) shows the transportation network before the schedule and speeds have been selected. The transshipment time from rotation R_1 to R_2 will depend on the schedule of the two rotations, but since both schedule and speed is

variable we need to consider different transshipment arcs (blue dotted lines in the figure). In this case there are seven arcs corresponding to a transshipment time of one to seven days. Figure b) shows an example flow from 3 to 8 where the schedule and speeds have been selected. Rotation R_1 calls port 1 every Thursday and R_2 departs from the same port (represented by a different port call) every Friday, so the transshipment arc used in this case is the one corresponding to one day. This way it is possible to calculate the transit time from 3 to 8 as the length of each sailing leg divided by the selected speed for each of the legs plus the transshipment time corresponding to the selected schedule. If the total transit time from the physical port corresponding to 3 to the physical port corresponding to 8 is longer than what is allowed for a commodity going on this path, there is no feasible path and it may be worth adjusting the speeds and the schedule.

A.1 Modeling

We wish to determine when each port is visited by which rotation and how this influences the achievable transshipment times and thereby the cargo routing. For each rotation the schedule (i.e. arrival and departure times) is determined in all ports for all rotations. The time it takes to transship between two different rotations visiting the same port is determined by the arrival in the port for each of the rotations and some buffer time. In the following we assume for simplicity weekly rotations, but the model can be extended to accommodate bi-weekly frequencies. The set of all port calls is I and in the example of nine physical ports in Figure 6 this corresponds to the port calls 1 to 10 which are consecutively numbered along each service. For each rotation $r \in R$ we assign a starting port call σ_r a priori (port call 1 and 6 in Figure 6) and use this as reference for the schedule of the subsequent ports in each of the rotations. The starting port will always be the call with the lowest index within the rotation. Each service consist of several port calls, and in the case of butterfly rotations the same port may have multiple corresponding port calls. The set of physical ports is Q and the set of port calls in port $q \in Q$ is $I(q)$.The arrival time at port call $i \in I$ given in days is determined by the continuous decision variable $T_i \in \mathbb{R}$ and the integer decision variable $w_{i'i''} \in \mathbb{Z}$ is the offset in weeks between two port calls for the same port, i' and i'', i.e., $w_{i'i''}$ is not necessarily equal to $w_{i''i'}$. Usually two port calls i' and i'' at port p correspond to two different rotations, but for butterfly rotations they can correspond to two calls from the same service. Additionally, $g_{i'i''}$ is the necessary transshipment buffer time between arrival of port call i' and departure of port call i'' in port p. The constant $\bar{t}_i$ specify the length of the stay of port call i and the continuous decision variable $\hat{T}_{i'i''} \in \mathbb{R}$ is the transshipment time from port call i' to port call i'' in a specific port. $A(i)$ is the set of multiarcs between port call i and $i+1$, i.e., arcs at different speeds between two consecutive ports on the same rotation. $(i', i'') \in Q^2(q)$ denotes all the ordered pairs of rotations visiting port q. For each port q with a transshipment opportunity and for each $(i', i'') \in Q^2(q)$, the set of arcs available for transshipment is given by the set $A(i', i'')$. The capacity of the corresponding arc, $u_{i'i''}$, is given by the minimum capacity of the rotation corresponding to port call i' and the rotation corresponding to i'' in a given port. All transshipment arcs have an associated transshipment time, t_a, and we include binary variables, x_a for $a \in A(i', i'')$ that selects whether a transshipment arc is used. The time scheduling part of the model is

$$T_i = T_{i-1} + \sum_{a \in A(i-1)} t_a x_a \qquad i \in I \setminus \cup_{r \in R} \{\sigma_r\} \tag{52}$$

$$\hat{T}_{i'i''} = (T_{i''} + \bar{t}_{i''}) - T_{i'} + 7w_{i'i''} \qquad q \in Q,\ (i', i'') \in Q^2(q) \tag{53}$$

$$\sum_{a \in A(i'i'')} t_a x_a \geq \hat{T}_{i'i''} \qquad q \in Q,\ (i', i'') \in Q^2(q) \tag{54}$$

$$\sum_{a \in A(i'i'')} x_a = 1 \qquad q \in Q,\ (i', i'') \in Q^2(q) \tag{55}$$

$$g_{i'i''} \leq \hat{T}_{i'i''} \leq g_{i'i''} + 7 \qquad q \in Q,\ (i', i'') \in Q^2(q) \tag{56}$$

$$T_i \in \mathbb{R}_+ \qquad i \in I \tag{57}$$

$$\hat{T}_{i'i''} \in \mathbb{R}_+ \qquad q \in Q,\ (i', i'') \in Q^2(q) \tag{58}$$

$$w_{i''i'} \in \mathbb{Z} \qquad q \in Q,\ (i', i'') \in Q^2(q) \tag{59}$$

$$x_a \in \{0, 1\} \qquad q \in Q,\ (i', i'') \in Q^2(q),\ a \in A(i', i'') \tag{60}$$

The relation of the departure time for two consecutive ports is given by Constraints (52). Notice that we in Constraints (52) let i run in the elements of I except the starting port call of each rotation. The reason is that we need to avoid a cyclic definition of T_i, which would be infeasible. The transshipment time in port p from port call i' to port call i'' is determined by Constraints (53) and Constraints (54) and (55) makes sure we only select one transshipment arc and that it is feasible. Constraints (56) limits the possible transshipment time. If we have a schedule determined by the hour there are going to be 168 transshipment arcs for each feasible (i', i'')-combination, but we can reduce the number of available transshipment arcs such that we overestimate the transshipment time. There can e.g. be one available arc for each day, i.e., 7 arcs, and for some ports we can have higher accuracy than others by including more arcs.

To illustrate the offset variable consider an instance with hourly accuracy where $\bar{t}_{i''} = 8$ and $g_{i'i''} = 10$ then if $T_{i'} = 24$ and $T_{i''} = 48$ we get that $w_{i'i''} = 0$ and $\hat{T}_{i'i''} = 48 + 8 - 24 = 32$. If there is a too tight schedule, i.e., $T_{i'} = 24$ and $T_{i''} = 24$, we get that the commodity will have to wait because of the buffer time and we get that $w_{i'i''} = 1$ and $\hat{T}_{i'i''} = 24 + 8 - 24 + 168 = 176$. To illustrate the influence of the schedule on longer rotations consider first $T_{i'} = 13 * 24 = 312$ and $T_{i''} = 24$, then we get $w_{i'i''} = 2$ and $\hat{T}_{i'i''} = 24 + 8 - 312 + 2 * 168 = 56$. Conversely if $T_{i'} = 24$ and $T_{i''} = 312$, then we get $w_{i'i''} = -1$ and $\hat{T}_{i'i''} = 312 + 8 - 24 - 168 = 128$.

A.2 Including the Time Scheduling Part in the Benders Decomposition

Similarly to the coupling constraints for the sailing arcs, we can introduce a coupling constraints for the transshipment arcs

$$\sum_{k \in K} \sum_{p \in P(a,k)} y^k_{p_k} \leq x_a u_{i'i''} \qquad a \in A(i', i''),\ q \in Q,\ (i', i'') \in Q^2(q) \tag{61}$$

Then including the time scheduling part in the Benders decomposition lead to a slightly modified sub-problem where we are now considering the transshipment arcs as well such that the PBSP is given by

$$\min \sum_{k \in K} \sum_{p \in P^k} r^k_p y^k_p \tag{62}$$

subject to

$$\sum_{p \in P^k} y_p^k \leq d^k \qquad k \in K \tag{63}$$

$$\sum_{k \in K} \sum_{p \in P(a,k)} y_p^k \leq \bar{x}_a q_a \qquad q \in Q,\ i \in I(q),\ a \in A(i) \tag{64}$$

$$\sum_{k \in K} \sum_{p \in P(a,k)} y_p^k \leq \bar{x}_a u_{i'i''} \qquad q \in Q,\ (i', i'') \in Q^2(q),\ a \in A(i', i'') \tag{65}$$

$$y_p^k \in \mathbb{R}_+ \qquad k \in K, p \in P^k \tag{66}$$

This can still be solved using column generation, but now the column generation sub-problem also contains dual variables corresponding to transshipment arcs. The BMP will be (22)-(28) + (52)-(60).

B Generating Additional Benders Cuts

It is possible to obtain additional Benders cuts in each iteration of the algorithm based on a solution to the multi-commodity flow problem. When the sub-problem has been solved we modify the problem to have an additional constraint restricting the objective to be at least as good as the optimal solution but maximizing some distance to the solution, e.g.:

$$\max \sum_{k \in K} \sum_{p \in P^k} |\bar{y}_p^k - y_p^k| \tag{67}$$

subject to

$$\sum_{p \in P^k} y_p^k \leq d^k \qquad k \in K \tag{68}$$

$$\sum_{k \in K} \sum_{p \in P(a,k)} y_{p_k}^k \leq q_a x_a \qquad (a) \in A \tag{69}$$

$$\sum_{k \in k} \sum_{p \in P^k} r_p^k y_p^k \leq \sum_{k \in K} \sum_{p \in P^k} r_p^k \bar{y}_p^k \tag{70}$$

$$y_p^k \in \mathbb{R}_+ \qquad k \in K,\ p \in P^k \tag{71}$$

where $\bar{y}_p^k$ is an optimal solution and we use the set of already generated paths to find an alternative solution. However, the objective (67) is non-linear so we modify it such that if $\bar{y}_p^k = 0$ then we take $(y_p^k - \bar{y}_p^k)$ and if $\bar{y}_p^k = d^k$ then we take $(\bar{y}_p^k - y_p^k)$. If $0 < \bar{y}_p^k < d^k$ we use $(\bar{y}_p^k - y_p^k)$ as the objective with probability $\frac{\bar{y}_p^k}{d^k}$ and $y_p^k - \bar{y}_p^k$ otherwise. Leaving out the constant term, the objective becomes

$$\max \sum_{k \in K} \left(\sum_{p \in P^k : \bar{y}_p^k = 0} y_p^k - \left(\sum_{p \in P^k : \bar{y}_p^k = d^k} y_p^k \right) + \sum_{p \in P^k : 0 < \bar{y}_p^k < d^k} X_p y_p^k \right) \tag{72}$$

Where X_p is a random variable of ± 1 with probability $\frac{\bar{y}_p^k}{d^k}$. Using only a reduced set of columns when solving the primal problem will lead to the optimal primal solution, but the corresponding dual solution may not be optimal/feasible as only a subset of constraints are considered. To obtain appropriate dual values, the objective is changed to the original objective, such that constraint (67) is removed and the problem is resolved using the solution as a warm start. The new dual solution is checked by solving the pricing problem in multi-commodity flow problem once, and if no reduced cost columns are returned an additional cut is added based on the new dual variables. If a reduced cost column is found, we do not add a cut.

Similarly we could solve the multi-commodity flow problem with the set of columns already obtained with an interior point method to obtain an alternative Benders cut based on a solution centered towards the interior. Again we need to check the multi-commodity flow pricing problem and only add the cut if no reduced cost paths are found.

References

Atamtürk, A. and Rajan, D. (2002). On splittable and unsplittable flow capacitated network design arc-set polyhedra. *Mathematical Programming*, 92(2):315–333.

Bai, L. and Rubin, P. A. (2009). Combinatorial benders cuts for the minimum tollbooth problem. *Operations research*, 57(6):1510–1522.

Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W., and Vance, P. H. (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations research*, 46(3):316–329.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik*, 4(1):238–252.

Boccia, M., Sforza, A., Sterle, C., and Vasilyev, I. (2008). A cut and branch approach for the capacitated p-median problem based on fenchel cutting planes. *Journal of Mathematical Modelling and Algorithms*, 7(1):43–58.

Boyd, E. A. (1994). Fenchel cutting planes for integer programs. *Operations Research*, 42(1):53–64.

Brouer, B., Alvarez, J., Plum, C., Pisinger, D., and Sigurd, M. (2014a). A base integer programming model and benchmark suite for liner shipping network design. *Transportation Science*, 48(2):281–312.

Brouer, B., Desaulniers, G., Karsten, C., and Pisinger, D. (2015). A matheuristic for the liner shipping network design problem with transit time restrictions. In Corman, F., Voß, S., and Negenborn, R., editors, *Computational Logistics*, volume 9335 of *Lecture Notes in Computer Science*, pages 195–208. Springer International Publishing.

Brouer, B. D., Desaulniers, G., and Pisinger, D. (2014b). A matheuristic for the liner shipping network design problem. *Transportation Research Part E: Logistics and Transportation Review*, 72:42–59.

Cheaitou, A. and Cariou, P. (2012). Liner shipping service optimisation with reefer containers capacity: an application to northern europe–south america trade. *Maritime Policy & Management*, 39(6):589–602.

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

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

Cordeau, J.-F., Soumis, F., and Desrosiers, J. (2000). A benders decomposition approach for the locomotive and car assignment problem. *Transportation science*, 34(2):133–149.

Cordeau, J.-F., Soumis, F., and Desrosiers, J. (2001). Simultaneous assignment of locomotives and cars to passenger trains. *Operations research*, 49(4):531–548.

Costa, A. M. (2005). A survey on benders decomposition applied to fixed-charge network design problems. *Computers & Operations Research*, 32(6):1429–1450.

Fischetti, M. and Monaci, M. (2014). Proximity search for 0-1 mixed-integer convex programming. *Journal of Heuristics*, 20(6):709–731.

Fortz, B. and Poss, M. (2009). An improved benders decomposition applied to a multi-layer network design problem. *Operations research letters*, 37(5):359–364.

Gelareh, S. and Meng, Q. (2010). A novel modeling approach for the fleet deployment problem within a short-term planning horizon. *Transportation Research Part E: Logistics and Transportation Review*, 46(1):76–89.

Gendron, B. (2011). Decomposition methods for network design. *Procedia-Social and Behavioral Sciences*, 20:31–37.

Kaparis, K. and Letchford, A. N. (2010). Separation algorithms for 0-1 knapsack polytopes. *Mathematical programming*, 124(1-2):69–91.

Karsten, C. V., Pisinger, D., Ropke, S., and Brouer, B. D. (2015). The time constrained multi-commodity network flow problem and its application to liner shipping network design. *Transportation Research Part E: Logistics and Transportation Review*, 76:122–138.

Kontovas, C. and Psaraftis, H. N. (2011). Reduction of emissions along the maritime intermodal container chain: operational models and policies. *Maritime Policy & Management*, 38(4):451–469.

Magnanti, T. L. and Wong, R. T. (1981). Accelerating benders decomposition: Algorithmic enhancement and model selection criteria. *Operations research*, 29(3):464–484.

Man (2013). Basic principles of ship propulsion. Technical report, Man Diesel & Turbo.

Meng, Q. and Wang, S. (2011). Optimal operating strategy for a long-haul liner service route. *European Journal of Operational Research*, 215(1):105–114.

Meng, Q., Wang, S., Andersson, H., and Thun, K. (2014). Containership routing and scheduling in liner shipping: overview and future research directions. *Transportation Science*, 48(2):265–280.

Notteboom, T. E. and Vernimmen, B. (2009). The effect of high fuel costs on liner service configuration in container shipping. *Journal of Transport Geography*, 17(5):325–337.

Psaraftis, H. N. and Kontovas, C. A. (2013). Speed models for energy-efficient maritime transportation: A taxonomy and survey. *Transportation Research Part C: Emerging Technologies*, 26:331–351.

Psaraftis, H. N. and Kontovas, C. A. (2015). Slow steaming in maritime transportation: Fundamentals, trade-offs, and decision models. In *Handbook of Ocean Container Transport Logistics*, pages 315–358. Springer.

Ronen, D. (2011). The effect of oil price on containership speed and fleet size. *Journal of the Operational Research Society*, 62(1):211–216.

Wang, S. and Meng, Q. (2012a). Liner ship fleet deployment with container transshipment operations. *Transportation Research Part E: Logistics and Transportation Review*, 48(2):470–484.

Wang, S. and Meng, Q. (2012b). Liner ship route schedule design with sea contingency time and port time uncertainty. *Transportation Research Part B: Methodological*, 46(5):615–633.

Wang, S. and Meng, Q. (2012c). Sailing speed optimization for container ships in a liner shipping network. *Transportation Research Part E: Logistics and Transportation Review*, 48(3):701–714.

Xia, J., Li, K. X., Ma, H., and Xu, Z. (2015). Joint planning of fleet deployment, speed optimization, and cargo allocation for liner shipping. *Transportation Science*, ( ): .

Zacharioudakis, P. G., Iordanis, S., Lyridis, D. V., and Psaraftis, H. N. (2011). Liner shipping cycle cost modelling, fleet deployment optimization and what-if analysis. *Maritime Economics & Logistics*, 13(3):278–297.

We introduce a decision support tool for liner shipping companies to optimally determine the sailing speed and needed fleet for a global network. As a novelty we incorporate cargo routing decisions with tight transit time restrictions on each container such that we get a realistic picture of the utilization of the network. Furthermore, we show that it is possible to extend the model to include optimal time scheduling decisions such that the time associated with transshipments is also reflected accurately. To solve the speed optimization problem we propose an exact algorithm based on Benders decomposition and column generation that exploits the separability of the problem. Computational results show that the method is applicable to liner shipping networks of realistic size and that it is important to incorporate cargo routing decisions when optimizing speed.

DTU Management Engineering
Institut for Systemer, Produktion og Ledelse
Danmarks Tekniske Universitet

Produktionstorvet
Bygning 424
2800 Kongens Lyngby
Tlf. 45 25 48 00
Fax 45 93 34 35

www.man.dtu.dk