DTU Library

Single liner shipping service design

Plum, Christian Edinger Munk; Pisinger, David; Salazar-González, Juan-José; Sigurd, Mikkel M.

Published in:
Computers and Operations Research

Link to article, DOI:
10.1016/j.cor.2013.11.018

Publication date:
2014

Document Version
Peer reviewed version

The Single Service Design Problem in Liner Shipping

Christian E. M. Plum[a,c,*], David Pisinger[a], Juan-José Salazar-González[b], Mikkel M. Sigurd[c]

[a] *DTU Management Science, Produktionstorvet, bygning 426, DK-2800 Kgs. Lyngby*
[b] *DEIOC, Facultad de Matemáticas, Universidad de La Laguna, 38721 Tenerife, Spain*
[c] *Maersk line, Esplanaden 50, DK-1098 København K.*

Abstract

The design of container shipping networks is an important logistics problem, involving assets and operational costs in billions of dollars. To guide the optimal deployment of the ships, a single vessel round trip is considered by minimizing operational costs and flowing the best paying demand under commercially driven constraints. This paper introduces the Single Service Design Problem. Arc-flow and path-flow models are presented using state-of-the-art elements from the wide literature on pickup and delivery problems. A Branch-and-Cut-and-Price algorithm is proposed, and implementation details are discussed. The algorithm can solve instances with up to 25 ports to optimality - a very promising result as real-world vessel roundtrips seldom involve more than 20 ports.

Keywords: Traveling salesman problem, Liner shipping, Branch-and-Cut-and-Price, Shortest path, Network design, Green logistics.

---

1. Introduction

Container shipping carriers operate worldwide networks consisting of hundreds of vessels having huge operating costs. Developing methods that can improve the network costs and/or the service level are of huge importance for both the carriers and the customers. Note that most of the market today is based on manufactured products transported on container vessels from distant continents.

Container shipping networks provide transport of containers from port to port at a fixed (usually weekly) schedule with a predetermined trip duration. The networks consist of a number of *services* and a set of similarly sized vessels sailing on a cyclic itinerary of ports. Services meet at certain hub ports where transhipment of containers can take place. The round trip duration is assumed

*Corresponding Author
Email addresses: cemp@man.dtu.dk (Christian E. M. Plum), pisinger@man.dtu.dk (David Pisinger), jjsalaza@ull.es (Juan-José Salazar-González), mikkel.muhldorf.sigurd@maersk.com (Mikkel M. Sigurd)

to be a multiple of a week, and a sufficient number of vessels is assigned to the round trip to ensure a weekly visit to each port. For instance, Figure 1 shows a 6 week round trip with 6 vessels to ensure that each port is visited once a week.

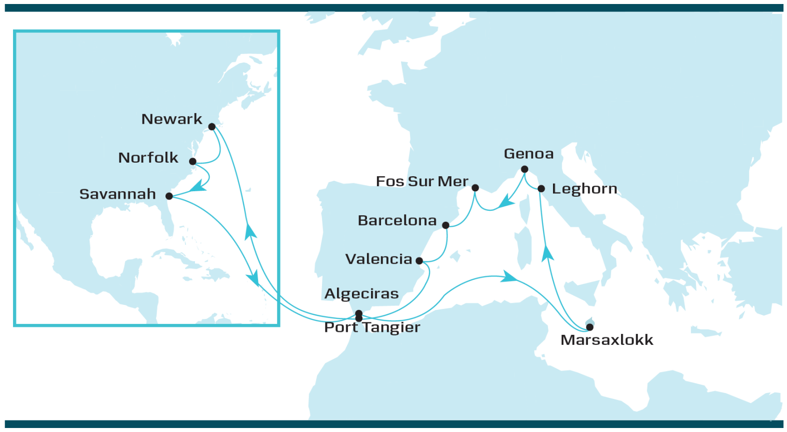

Figure 1: The WestMed Service, transporting containers between U.S. east coast and the western Mediterranean.

A given demand is loaded at its origin port to some service, which may bring the demand directly to its destination or unload it at a hub port for transshipment to another service, ultimately bringing the demand to its destination. See Stopford [25] and Notteboom [19] for a more general introduction to the economics of liner shipping.

A usual intercontinental service has between 5-10 port calls for the more direct trades (e.g. Trans-Atlantic or Trans-Pacific) and 15-20 port calls on the longer trades (e.g. Europe-Asia trades), indicating the problem sizes that can be encountered in reality. Stopford [25] has more details on different service types.

The problem investigated in this paper considers the design of a single capacitated service following a simple cyclic rotation where all ports must be visited, i.e. a Hamiltonian tour. A solution approach for this problem is an important tool for a network planner designing a single service as fierce competition between carriers often require low path durations, while the best paying containers must be prioritized to optimize profits. In practice services are seldom Hamiltonian, partly because important ports are called more than once, partly because waterways as canals must be traversed in both directions. An experienced user knows the ports where several visits may be necessary and, by duplicating them, the problem becomes the Hamiltonian variant addressed in this paper. Canals do not cater to demand and hence should be excluded from the port set, but included in the distances between ports.

The problem is then to transport a set of demands on a generated round trip, where the combined sum of these demands must consider the capacity of all edges. A demand has a maximal path duration which must be respected: a demand can be partly fulfilled, but it must still respect the path duration limit. This problem is called the *Single Service Design Problem*, or in short SSDP. To the best of our knowledge this problem has not been addressed before in the literature.

1.1. Liner shipping

We refer to Christiansen et al. [7] and Christiansen et al. [8] for an overview of early research on Liner Shipping problems. Since these reviews, a number of articles has been published, with various approaches and scopes for Liner Shipping Network Design Problems (LSNDP). The work of Shintani et al. [24] has a detailed description of the cost structure and includes consideration of repositioning empty containers. The network design problem considered by Agarwal and Ergun [1] generates multiple services and handle transshipment. Bender's and column generation based algorithms are implemented. These algorithms scale well to large instances, but transshipment costs are excluded. The model of Alvarez [2] considers transshipment cost and finds solutions for large instances in a heuristical column generation approach. The Branch-and-Cut method of Reinhardt and Pisinger [23] is the first model considering transshipment while allowing for non-simple rotations (with two calls to a single port, a so-called butterfly route). Small instances are solved optimally. The models of Gelareh et al. [11] and Gelareh and Pisinger [10] use a hub location based approach, generating a main service visiting some ports directly, instances of up to 10 ports are solved to optimality. The work of Brouer et al. [6] describes the domain of LSNDP, discusses the relevant scoping, proposes a model of the problem, and presents a number of benchmark instances for the LSNDP based on real world problems. A novel aggregation of demands was presented in Jepsen et al. [15] giving a new model formulation and decomposition method, though it did not perform well in practice. A heuristic algorithm for a short horizon version of the problem is presented by Wang and Meng [26]. A formulation considering empty container repositioning is found in Meng and Wang [16] and a further model dealing with robust schedule design in Wang and Meng [27], but neither of these consider the order of the port calls and take this as an input. A recent overview to the area is given by Meng et al. [17]. This multitude of publications on LSNDP shows that the interest in these problems has increased. Most of these works considers different models and scopes of the problem and optimal methods can only solve small instances (10–15 ports) and, as real world instances are larger, the problem is still open for research (see Brouer et al. [6]).

1.2. Pickup and Delivery problems

The SSDP is related to the well-studied pickup and delivery problems. Parragh et al. [20] and Berbeglia et al. [4] give good introductions to these problems, reviewing existing literature and proposing classification schemes. In the classification of Parragh et al. [20] the SSDP is a Single Dial-A-Ride Problem (SDARP) excluding Time Windows, and with the important difference that no depot is required, i.e. demand can be carried through the depot. In the classification of Berbeglia et al. [4] the SSDP is a **[1-1|PD|1]**: **1-1** as each commodity has one origin and one destination, **PD** as each vertex must be visited exactly once for combined pickup and delivery, and **1** as a single service is generated. An important difference from related problems is the lack of a depot. The multi-commodity one-to-one pickup-and-delivery traveling salesman problem (m-PDTSP) is considered in Hernández-Pérez and Salazar-González [12]: the problem is formulated, and solution methods based on Bender's decomposition are implemented. The SSDP can be seen as an extension of the m-PDTSP with the addition of path duration, and optional demand with associated revenue. An often encountered type of subproblems in pickup and delivery problems are Shortest Path Problems with Resource Constraints (SPPRC) which also appear by decomposing the SSDP. We refer to Irnich and Desaulniers [13], Jepsen et al. [14] or Petersen [21] for a review on SPPRC problems and algorithms.

1.3. Overview

The main contributions of this paper are two novel models of the SSDP and a Branch-and-Cut-and-Price solution method for solving the problem. The absence of a depot gives a problem structure not seen in related problems. This requires both the pricing problem and the separation of valid inequalities to be designed in a novel manner. The implemented method solves problem sizes met in real world instances.

In Section 2 an arc-flow model of the SSDP is presented. This model is Dantzig-Wolfe decomposed to a path-flow model to be solved with a column generation algorithm, which effectively handles the multi commodity flow problem with path duration constraints. Details of subtour elimination constraints, pricing problems and branching approaches are given. The proposed algorithm has been implemented and computational results are presented in Section 3, where instances of up to 25 nodes can be solved to optimality. Details of the data instances are provided. Finally Section 4 concludes on the paper and proposes directions for further research. This work is an extension of Plum et al. [22].

2. Mathematical Formulation

In the following we introduce the notation, present an arc-flow model, followed by a path-flow model, and a new solution method for the SSDP.

The service must visit each node $i \in V$ exactly once. Directed arcs $(i, j) \in A$ exist between all nodes, giving the complete directed graph $G = (V, A)$. Let $S \subset V$ be a subset of nodes. Each arc $a = (i, j) \in A$ is associated with a cost c_a representing time charter costs for the vessel, bunker cost for propulsion and port call costs for visiting the port j. Traversing the arc a takes the time t_a. This time depends on the sailed distance and the speed of the vessel. The service has a capacity Q, which must be respected at all traversed arcs. The generated service can transport the commodities $k \in K$. Each commodity k has a source s^k and a destination d^k ($s^k, d^k \in V$), a volume of containers $F^k > 0$, a maximal path duration $t^k > 0$ and unit-revenue for transporting $r^k > 0$. A node i can be the source of one or more commodities, as well as destination for some commodities.

2.1. Arc-Flow Formulation

The problem is to find a maximal profit set of paths in G for a set of commodities k, such that the containers can be moved from their origin to their destination in at most t^k time. All the paths should be a subset of a Hamiltonian tour, where each arc has a corresponding cost and traversal time.

Let x_a be a binary variable indicating whether the service travels on arc $a \in A$. Let f_a^k be the flow of commodity k on each arc a. The problem can then be formulated as:

$$\min \sum_{a \in A} c_a x_a - \sum_{k \in K} r^k \sum_{a \in \delta^-(s^k)} f_a^k \tag{1}$$

subject to

$$x(\delta^-(i)) = 1 \qquad \forall i \in V \tag{2}$$
$$x(\delta^+(i)) = 1 \qquad \forall i \in V \tag{3}$$
$$x(\delta^+(S)) \geq 1 \qquad \forall S \subset V \tag{4}$$
$$\sum_{a \in \delta^+(i)} f_a^k = \sum_{a \in \delta^-(i)} f_a^k \qquad \forall k \in K \text{ and } i \in V \setminus \{s^k, d^k\} \tag{5}$$
$$\sum_{k \in K} f_a^k \leq Q x_a \qquad \forall a \in A \tag{6}$$
$$\sum_{a \in \delta^-(s^k)} f_a^k \leq F^k \qquad \forall k \in K \tag{7}$$
$$\sum_{a \in A} t_a f_a^k \leq t^k \sum_{a \in \delta^-(s^k)} f_a^k \qquad \forall k \in K \tag{8}$$
$$f_a^k \geq 0 \qquad \forall a \in A \text{ and } k \in K \tag{9}$$
$$x_a \in \{0, 1\} \qquad \forall a \in A \tag{10}$$

The objective minimizes the cost of the traversed arcs subtracted the revenue of flowed demand. Constraints (2) and (3) ensure that all nodes have one outgoing and ingoing open edge, where $\delta^+(i)$ and $\delta^-(i)$ denotes the set of ingoing, respectively outgoing, arcs to node i. Constraint (4) are subtour elimination constraints ensuring that the Hamiltonian tour connects all nodes in a single rotation. The conservation of flow is ensured by (5), and the capacity is enforced by (6). Constraints (7) limit the served demand to the upper value F^k. The path duration is ensured by (8) since f_a^k is $\sum_{a' \in \delta^-(s^k)} f_{a'}^k$ for all a in the path moving the demand k. Constraints (9) and (10) set bounds on the decision variables f_a^k and x_a.

2.2. Path-Flow Formulation

Constraints (5)–(9) can be eliminated through a Dantzig-Wolfe decomposition on the arc-flow model, thus replacing the variables f_a^k by path variables. Let P^k be the set of all feasible paths from s^k to d^k, satisfying the constraints (5)–(9). This set may have an exponential number of elements. Each path $p \in P^k$ is represented as a set of arcs, i.e. $p \subset A$. Let $t_p = \sum_{a \in p} t_a$ be the duration of this path. Let λ_p be a non negative real variable representing the volume of flow of commodity k using path $p \in P^k$. The SSDP can then be formulated as:

$$\min \sum_{a \in A} c_a x_a - \sum_{k \in K} r^k \sum_{p \in P^k} \lambda_p \tag{11}$$

subject to (2)–(4) and

$$\sum_{k \in K} \sum_{p \in P^k : a \in p} \lambda_p \leq Q x_a \qquad \forall a \in A \tag{12}$$

$$\sum_{p \in P^k} \lambda_p \leq F^k \qquad \forall k \in K \tag{13}$$

$$x_a \in \{0, 1\} \qquad \forall a \in A \tag{14}$$

$$\lambda_p \geq 0 \qquad \forall k \in K \tag{15}$$

The objective function minimizes the costs of chosen arcs subtracted the revenue of flowed demand. The capacity is enforced by constraint (12). Convexity constraints (13) ensure that at most the available flow is transported.

The exponential number of subtour elimination constraints (4) can be relaxed initially and inserted when violated, as done in Reinhardt and Pisinger [23]. A lower bound on the optimal value of this model can be attained by solving the LP-relaxation, where the integrality constraints (14) are replaced with constraints $0 \leq x_a \leq 1 \ \forall \ a \ \in \ A$. This LP-relaxation can be solved using a cut-and-price algorithm. Due to the exponential number of variables λ_p, a restricted master problem is obtained by considering a subset $\bar{P} \subseteq P$ of paths. Additional columns of negative reduced costs are generated by solving a pricing subproblem. Let $\pi_a \in \mathbb{R}$ be the dual variables for the capacity constraints (12) and let $\theta^k \leq 0$ be dual variables for the convexity constraints (13). Then the pricing problem becomes:

$$\text{Min: } \sum_{a \in A} \pi_a x_a - \theta^k - r^k \tag{16}$$

subject to constraints (5)–(8).

2.3. Separation of Subtour Elimination Constraints

Given a solution x^* of the LP relaxation of the path-flow formulation, we must search for any violated subtour elimination constraint (4). A violated constraint exists if and only if a minimum-capacity cut in the solution graph $G(x^*)$ has weight less than one. This can be computed in polynomial time.

2.4. Pricing Problem

The pricing problem is an Elementary Shortest Path Problem with Resource Constraints (ESPPRC), which is strongly NP-hard as shown in e.g. Irnich and Desaulniers [13]. The path must have the lowest cost given by arc weights π_a, while respecting path durations. Without the elementarity requirement, the problem can be solved in pseudo-polynomial time. As the pricing problem (16) may contain negative coefficients in the objective, negative cost cycles are likely, but still we apply the non-elementary variant. Negative cycles are broken by introducing a new resource for the number of traversed arcs, by imposing an upper bound of $n-1$ on this resource. The problem is solved by a labeling algorithm, which has the advantage of returning all Pareto optimal paths, in the resources. All these paths are added to the pricing problem, if they have negative reduced costs. When no negative reduced costs can be found for the problem including the resource on number of traversed arcs, then no path with negative reduced costs for the ESPPRC exists.

2.5. Branching

When all violated cuts and negative reduced costs columns have been added to the current node, and fractional binary variables x_a still exists, branching is commenced. Binary branching is used, by selecting the most fractional x_a and adding constraints $x_a \leq 0$, $x_a \geq 1$ to the two branching children, respectively. As this branching is done on variables x_a existing in both the original and reformulated problem space, the branching constraints will be directly imposed in the pricing problem and subtour elimination cuts and no further consideration of this is needed.

A main contribution of this paper lies in the powerful formulation of the flow and path based models for this new problem. These formulations allows for carrying demand through the depot, while selecting which demand to flow and enforcing the path duration limit. The effectiveness of these formulations allows for efficient algorithmic techniques, as branching in the original space of the x_a variables, separating subtour elimination constraints and solving the pricing problem with an efficient labeling algorithm.

3. Computational Results

The algorithm has been implemented using the `COIN-OR DIP` (Galati and Ralphs [9]) framework to implement the Branch-and-Cut-and-Price method and using `CPLEX 12.1` as LP solver. `Boost`'s graph library (Boost [5]) has an implementation of SPPRC, which is used to solve the pricing problem as described above. `Concorde` (Applegate et al. [3]) has an efficient implementation of a min cut algorithm and `boost` also finds connected components, to check if we have a feasible solution. The implementation has been run on a 4 GB Ram, Intel E8400 3.00 GHz using a single core.

3.1. Instances

The algorithm has been tested on a set of instances inspired by the *class2* and *class3* instances of Hernández-Pérez and Salazar-González [12], which again are based on the description by Mosheiov [18] for instances of TSP with pickup-and-delivery. These have n random points in the square [-500, 500] $\times$ [-500, 500], one of which is located at point (0,0) (formerly the depot). For each problem size, we have 5 randomly generated instances. The travel cost c_a is the Euclidean distance between the points and the travel time is $t_a = c_a/100 \cdot (0.95 + rand(0, 0.1))$. Hence the travel time is proportional to the cost, but with some small deviation, as seen in real-life problems. The vessel has a capacity Q. The commodities have a path duration limit t^k, a volume F^k and an associated revenue r^k. To test the scalability and properties of the algorithm a number of variations of the instances have been created.

Path duration limit. Instances with $t^k \in \{0, 5, 10, 200\}$ have been created. $t^k = 10$ is used as the default setting. All t^k have the same setting in an instance.

Revenue. Instances with $r^k \in \{0, 250, 1000, 10000\}$ have been created, $r^k = 1000$ is used as the default setting. All r^k have the same setting in an instance.

Graph. Instances with 10, 15, 20 and 25 nodes have been run, all with a complete set of edges. These graph sizes resembles real service design problems.

Commodity Density. Instances with fixed (F) number of commodities 5, 10 and 15 have been generated. To test larger commodity sets, instances with (A) sparse commodity density n, (B) populated commodity density $3n$, (C) dense commodity density $(n^2 - n)/2$ and (D) complete commodity density $n^2 - n$ have been tested. Commodity Density (B) is used if nothing else is mentioned.

Capacity. Instances with $Q \in \{0, 10, 30, 200\}$ have been created, $Q = 10$ is used as the default setting. These instances are constructed as to resemble problems that could arise in real service design situations by the relation between capacity, revenue and path duration limit, as the interplay between revenue, cost and operational and commercial restrictions come in play.

3.2. Results

Tables 1-4 shows the results of the developed algorithm on the test instances. Each row in the table corresponds to runs on five randomized instances with the same properties. In the tables column n is the number of nodes, m is the number of commodities, $m = 3n$, commodity density (B), if nothing else is stated. The algorithm has been run with a time limit of 3600 seconds, and *Time* is the average computational time of the five runs. *Timeout* is the number of runs which timed out and hence was not solved to optimality. *Gap* is the percentual gap between the upper and lower bound of all computed instances including both optimally solved and timeouts. The number of added subtour elimination constraints is given by *Cuts*. The number of generated path columns are given by *Columns* and the number of search nodes in the Branch-and-Bound tree is given by *B&B Nodes*. All values are averages over the 5 randomly generated instances.

In general it can be seen that the solution time increases with graph size. Some classes of instances are easily solved as they reduce to a standard TSP problem, these are the cases where $t^k = 0$, $r^k = 0$ and $Q = 0$.

Table 1 reports the effect of different commodity densities. It shows that the small *5*-commodity instances are easily solved, but interestingly the completely dense D-instances are also among the fastest solved for all n. This must be due to the abundance of commodities, making it easier to find the optimal solution, as commodities that fits together without exceeding path duration limit exists.

In Table 2, problem complexity can be seen to increase with increasing t^k. The number of *B&B Nodes* decreases for the larger instances with high t^k values, as each node becomes harder to solve with non binding path duration constraints.

In Table 3 problem complexity also increases with revenue, but not for the largest graphs, where the gap stabilizes or decreases for very large revenues, as the problem shifts from balancing cost against revenues, to maximizing the revenues.

The dependence on the vessel's capacity can be seen in Table 4, where the complexity of the instance appears to be proportional with the capacity. The largest instances that can be solved to optimality have 25 nodes and commodity density A or D. These problem types represent real world problems well and it proves the methods applicability in a real world setting as a decision support tool to generate services in a complex operational and commercial setting.

Generation of subtour elimination constraints (*Cuts*) increases with the size of the graph, for the instances solved to optimality. For instances reaching the time limit the time for each iteration increases with graph size, and thus *decreases* the number of subtour elimination constraints generated. The number of path columns generated (*Columns*) increases with increasing *Path duration limit* as more paths become feasible. There is also some dependence with the commodity density and capacity. The number of *B&B Nodes* follows the same pattern as the number of *Cuts*, i.e. increasing with larger graph size. As the time of each iteration increases the number of branches decreases due to timeout.

4. Conclusion and Further work

We have presented the SSDP, a pickup and delivery problem which differs from related pickup and delivery problems by not considering a depot, having optional demands, and having to respect path durations for the demand. The inclusion of path durations and optional demand is a new, and probably more realistic, way of seeing liner shipping network design.

A novel arc-flow model as well as a path-flow model have been proposed, and a Branch-and-Cut-and-Price algorithm has been devised for the path-flow model. This algorithm effectively deals with the path duration limits in subproblems for each demand, while it chooses the vessel round trip, demand paths

n	Commodity Density	m	Time	Timeout	Gap	Cuts	Columns	B&B Nodes
10	F	5	1	0	0 %	42	124	20
10	F	10	2	0	0 %	79	467	48
10	F	15	3	0	0 %	99	872	58
10	A	10	160	0	0 %	3936	1141	5510
10	B	30	18	0	0 %	291	3988	203
10	C	45	19	0	0 %	300	3511	191
10	D	90	6	0	0 %	130	4197	48
15	F	5	3	0	0 %	111	506	74
15	F	10	885	0	0 %	4465	6336	5519
15	F	15	292	0	0 %	1881	7518	1961
15	A	15	3	0	0 %	50	1225	12
15	B	45	1034	0	0 %	2278	33037	2035
15	C	105	3602	5	7 %	3327	62769	1448
15	D	210	117	0	0 %	484	29509	201
20	F	5	28	0	0 %	246	1454	175
20	F	10	1237	1	4 %	5000	7557	5585
20	F	15	3017	4	5 %	5213	22944	3735
20	A	20	3610	5	inf	601	41362	216
20	B	60	3605	5	12 %	1093	72048	415
20	C	190	3612	5	8 %	893	54188	220
20	D	380	620	0	0 %	702	54569	269
25	F	5	562	0	0 %	2275	7489	2324
25	F	10	3185	4	10 %	2394	35609	1904
25	F	15	3607	5	14 %	1109	46511	497
25	A	25	1713	1	1 %	848	150721	510
25	B	75	3619	5	39 %	457	52158	120
25	C	300	3613	5	12 %	644	79878	112
25	D	600	849	0	0 %	441	45677	189

Table 1: Computational results with varying commodity density. Average values of 5 instances.

n	t^k	Time	Timeout	Gap	Cuts	Columns	B&B Nodes
10	0	0	0	0 %	7	30	3
10	5	1	0	0 %	16	101	6
10	10	18	0	0 %	291	3988	203
10	200	3072	3	1 %	6331	162744	4969
15	0	1	0	0 %	11	45	8
15	5	7	0	0 %	124	298	109
15	10	1034	0	0 %	2278	33037	2035
15	200	3604	5	13 %	1445	163834	546
20	0	2	0	0 %	5	60	1
20	5	55	0	0 %	372	856	349
20	10	3605	5	12 %	1093	72048	415
20	200	3614	5	29 %	548	63474	114
25	0	2	0	0 %	1	75	1
25	5	496	0	0 %	2215	3541	2100
25	10	3619	5	39 %	457	52158	120
25	200	3614	5	25 %	529	104782	105

Table 2: Computational results with varying path duration limits. Average values of 5 instances.

and quantity of each demand to respect the vessel capacity in the master problem. The solution method has been implemented and extensive testing shows that it is able to solve problem instances with $n = 25$ nodes and commodity density (A) or (D) to optimality in less than 3600 seconds. The model and

n	r^k	Time	Timeout	Gap	Cuts	Columns	B&B Nodes
10	0	0	0	0 %	7	30	3
10	250	20	0	0 %	300	4414	210
10	1000	18	0	0 %	291	3988	203
10	10000	22	0	0 %	324	4401	234
15	0	1	0	0 %	11	45	8
15	250	1046	0	0 %	2687	36270	2597
15	1000	1034	0	0 %	2278	33037	2035
15	10000	976	0	0 %	2120	31345	1927
20	0	2	0	0 %	5	60	1
20	250	3606	5	18 %	1051	83624	391
20	1000	3605	5	12 %	1093	72048	415
20	10000	3603	5	11 %	1035	65633	395
25	0	2	0	0 %	1	75	1
25	250	3616	5	58 %	465	61431	123
25	1000	3619	5	39 %	457	52158	120
25	10000	3613	5	21 %	486	53531	127

Table 3: Computational results with varying revenue. Average values of 5 instances.

n	Q	Time	Timeout	Gap	Cuts	Columns	B&B Nodes
10	0	1	0	0 %	6	110	4
10	10	18	0	0 %	291	3988	203
10	30	90	0	0 %	665	8757	520
10	200	109	0	0 %	789	8934	628
15	0	1	0	0 %	8	231	4
15	10	1034	0	0 %	2278	33037	2035
15	30	3110	3	11 %	3600	101576	2781
15	200	3146	4	15 %	2957	90291	1926
20	0	2	0	0 %	4	412	2
20	10	3605	5	12 %	1093	72048	415
20	30	3608	5	inf	404	99737	108
20	200	3619	5	76 %	398	107869	102
25	0	5	0	0 %	15	632	10
25	10	3619	5	39 %	457	52158	120
25	30	3607	5	124 %	368	173356	76
25	200	3609	5	153 %	346	168005	67

Table 4: Computational results with varying capacity. Average values of 5 instances.

developed solution method is generally applicable to a wide range of problems, as well as for liner shipping specific problems. If one wished to capture more of the rich problems faced in liner shipping network design, the model and solution method could be extended to: include time windows, as a carrier will often have a limited number of berth hours available at some port. Another extension would be to allow multiple port calls to some ports, as port calls both in- and outbound on a service can improve path duration for both imports and exports, this would require a model allowing non-simple cycles. The developed solution method can solve problem instances with up to said 25 nodes, which makes it applicable to the design of real world inter continental services, typically calling 10 to 20 ports.

Acknowledgments

The authors would like to thank Berit Dangaard Brouer for valuable comments, Matthew Galati for his help using **COIN-OR DIP** and The Danish Council for Strategic Research for support to the ENERPLAN project.

References

[1] Richa Agarwal and Ozlem Ergun. Ship scheduling and network design for cargo routing in liner shipping. *Transportation Science*, 42(2):175–196, 2008.

[2] Jose Fernando Alvarez. Joint routing and deployment of a fleet of container vessels. *Maritime Economics and Logistics*, 11:186–208, 2009.

[3] David L. Applegate, Robert E. Bixby, Vasek Chvátal, and William J. Cook. Concorde tsp solver, 2001. Available at www.tsp.gatech.edu/concorde.html.

[4] Gerardo Berbeglia, Jean-Francois Cordeau, Irina Gribkovskaia, and Gilbert Laporte. Static pickup and delivery problems: a classification scheme and survey. *TOP, Trabajos de Investigación Operativa*, 15:1–31, 2007.

[5] Boost. The boost graph library. Available at www.boost.org.

[6] Berit D. Brouer, J. Fernando Alvarez, Christian E. M. Plum, David Pisinger, and Mikkel M. Sigurd. A base integer programming model and benchmark suite for liner shipping network design. *Transportation Science*, 2013. Accepted by Transportation Science.

[7] Marielle Christiansen, Kjetil Fagerholt, and David Ronen. Ship routing and scheduling: status and perspectives. *Transportation Science*, 38(1):1–18, 2004.

[8] Marielle Christiansen, David Ronen, Bo Nygreen, and Kjetil Fagerholt. Maritime transportation. In *Handbooks in Operations Research and Management Sciences*, pages 189–284. North Holland, 14 edition, 2007.

[9] Matthew Galati and Ted .K. Ralphs. Coin-or dip (decomposition for integer programming), 2009. Available at projects.coin-or.org/Dip.

[10] Shahin Gelareh and David Pisinger. Fleet deployment, network design and hub location of liner shipping companies. *Transportation Research Part E: Logistics and Transportation Review*, 47:947 – 964, 2011.

[11] Shahin Gelareh, Stefan Nickel, and David Pisinger. Liner shipping hub network design in a competitive environment. *Transportation Research Part E: Logistics and Transportation Review*, 46(6):991–1004, November 2010.

[12] Hipólito Hernández-Pérez and Juan-José Salazar-González. The multi-commodity one-to-one pickup-and-delivery traveling salesman problem. *European Journal of Operational Research*, 196(3):987 – 995, 2009.

[13] Stefan Irnich and Guy Desaulniers. Shortest path problems with resource constraints. In *Column Generation*, pages 33–65. Springer US, 2005.

[14] Mads K. Jepsen, Bjørn Petersen, Simon Spoorendonk, and David Pisinger. Introducing subset row inequalities in a branch-and-cut-and-price algorithm for the vehicle routing problem with time windows. *Operations Research*, 56:497–511, 2008.

[15] Mads K. Jepsen, Berit Løfstedt, Christian E. M. Plum, David Pisinger, and Mikkel M. Sigurd. A path based model for a green liner shipping network design problem. In *Proceedings of the IAENG, ICOR 2011*, 2011.

[16] Qiang Meng and Shuaian Wang. Liner shipping service network design with empty container repositioning. *Transportation Research Part E: Logistics and Transportation Review*, 47(5):695–708, 2011.

[17] Qiang Meng, Shuaian Wang, Henrik Andersson, and Kristian Thun. Containership routing and scheduling in liner shipping: Overview and future research directions. *Transportation Science*, 2013. To appear in.

[18] Gur Mosheiov. The travelling salesman problem with pick-up and delivery. *European Journal of Operational Research*, 79(2):299 – 310, 1994.

[19] Theo E. Notteboom. Container shipping and ports: An overview. *Review of Network Economics*, 3, 2004.

[20] Sophie Parragh, Karl Doerner, and Richard Hartl. A survey on pickup and delivery problems. *Journal fur Betriebswirtschaft*, 58:81–117, 2008.

[21] Bjørn Petersen. *Shortest Paths and Vehicle Routing*. PhD thesis, 2011.

[22] Christian E. M. Plum, David Pisinger, Juan-José Salazar-González, and Mikkel M. Sigurd. The multi-commodity one-to-one pickup-and-delivery traveling salesman problem with path duration limits. In *Proceedings of the International MultiConference of Engineers and Scientists 2012 Vol II*, 2012.

[23] Line Blander Reinhardt and David Pisinger. A branch and cut algorithm for the container shipping network design problem. *Flexible Services And Manufacturing Journal*, 2011.

[24] Koichi Shintani, Akio Imai, Etsuko Nishimura, and Stratos Papadimitriou. The container shipping network design problem with empty container repositioning. *Transportation Research Part E: Logistics and Transportation Review*, 43(1):39–59, 2007.

[25] Martin Stopford. *Maritime Economics*. Routledge, third edition, 2009.

[26] Shuaian Wang and Qiang Meng. Schedule design and container routing in liner shipping. *Transportation Research Record: Journal of the Transportation Research Board*, 2222: 25–33, 2011.

[27] Shuaian Wang and Qiang Meng. Robust schedule design for liner shipping services. *Transportation Research Part E: Logistics and Transportation Review*, 48(6):1093 – 1106, 2012.