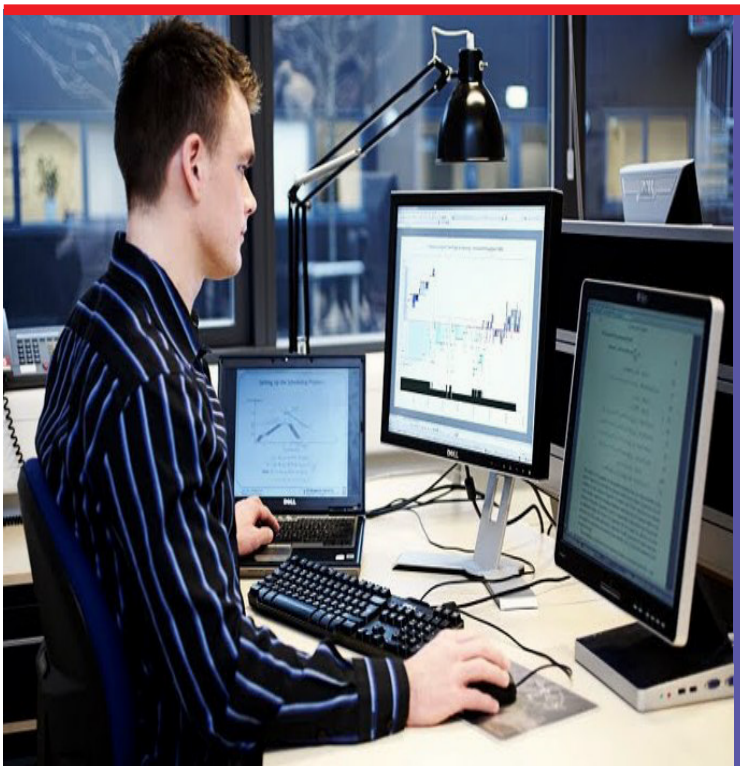
A Branch and Cut algorithm for the container shipping network design problem

Reinhardt, Line Blander; Kallehauge, Brian; Pisinger, David

Publication date:
2010

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

A Branch and Cut algorithm for the container shipping network design problem



Report 20.2010

DTU Management Engineering

Line Blander Reinhardt
Brian Kallehauge
David Pisinger
December 2010

DTU Management Engineering
Department of Management Engineering

A Branch and Cut algorithm for the container shipping network design problem

Line Blander Reinhardt*, Brian Kallehauge*, David Pisinger
Department of Management Engineering, Technical University of Denmark

December 10, 2010

Abstract

The network design problem in liner shipping is of increasing importance in a strongly competitive market where potential cost reductions can influence market share and profits significantly. In this paper the network design and fleet assignment problems are combined into a mixed integer linear programming model minimizing the overall cost. To better reflect the real-life situation we take into account the cost of transhipment, a heterogeneous fleet, route dependant capacities, and butterfly routes. To the best of our knowledge it is the first time an *exact* solution method to the problem considers transhipment cost. The problem is solved with branch-and-cut using clover and transhipment inequalities. Computational results are reported for instances with up to 15 ports.

1 Introduction

Liner shipping routes are characterized by the cyclic routes repeatedly sailed during the scheduled horizon and the transhipment of cargo in hub ports. The process of designing the route network of a liner shipping company is essential for the competitiveness of the company and its ability to sustain and possibly improve the share of the global containerized freight market. The problem of determining the structure of the route network we call the liner shipping network design problem (LS-NDP). Designing efficient routes can reduce the overall cost and the CO_2 emission per container shipped.

To provide a competitive product a liner shipping company must at a minimal cost be able to satisfy the request from customers for shipment of containers. Liner shipping companies usually have a forecast period over which the shipping demands are predicted based on historic data and recent development. The LS-NDP consists of designing vessel routes so that the forecasted requests are satisfied with a minimal cost for the company.

A vessel will repeatedly sail the assigned route throughout the entire planning horizon. This means that the routes are cyclic and the capacity of a link on a route depends on the number of times the link is sailed in the planning horizon.

The LS-NDP gained increasing attention about three decades ago when the container freight started to increase significantly. Recently the interest in the area has increased due to the large focus on CO_2 emission generated by the vessels, and the dramatic change in demands created by the current financial crisis, which has resulted in a need to focus on lowering the costs.

The similarity between LS-NDP, network design and routing problems leads us to the assumption that methods that work well for other scheduling and network design problems will also work well for the LS-NDP. An example of such a method is the branch-and-cut method which has been successfully applied to the vehicle routing problem with time windows (VRPTW) problem, see Bard et al. [4] and Kallehauge et al. [12].

*This research is partly supported by the Danish Maritime Fund

In this paper we present a mathematical formulation of the problem which includes transhipment, transhipment cost and allows a mix of simple and *butterfly* routes. An exact method using branch-and-cut has been developed for solving the presented model. The developed branch-and-cut method has been run on a set of test instances and compared to the CPLEX MIP solver. To our knowledge it is the first time an exact method has been applied to a problem which includes transhipment and the results show that small instances can be solved to optimality. The developed branch-and-cut method clearly outperforms CPLEX. The test results presented in the the computational experiments, Section 5, document that the developed algorithm can be used for planning the routes of a smaller shipping company or a concrete region of the network of a bigger liner shipping company. The LS-NDP problems we solve to optimality are comparable in size to the test instances presented in recent literature on shipping network design using heuristic solution methods (see Agarwal and Ergun [1] and Alvarez [2]).

We will start with a literature review in Section 2. In Section 3, the problem is formulated as a graph theoretical problem and a mathematical model of the LS-NDP is presented. In Section 4 the branch-and-cut algorithm is described, and separation algorithms for the introduced transhipment cuts and connectivity cuts are presented. In Section 5, the tests and results are discussed. Finally, we make some concluding remarks and suggest areas for further research in Section 6.

2 Literature review

In this section we summarize the literature which has been used directly in our work. For a detailed literature review of cargo shipping optimization problems we refer the reader to the survey papers, Ronen [17], Ronen et al. [18], and Christiansen et al. [6]. The reader is also referred to Christiansen et al. [7] for an comprehensive introduction to the areas of optimization in maritime transportation.

In 1991 Rana and Vickson [16] presented a state-of-the-art model for container shipping on the North Atlantic trade routes. They worked with an outbound-inbound principle which, until recently, has been a standard principle in the liner shipping industry. The outbound-inbound principle means that the ports are listed in a predefined order and that a vessel goes through the list in one direction visiting selected ports and upon return goes through the list in the reverse direction until reaching the first port visited on the list. The liner shipping companies still have an inbound-outbound way of viewing some of their routes. However, there is no requirement that the routes must be scheduled this way. For shipping routes along a somewhat straight coastline such as the US West Coast investigated in [16] this is a natural setup. For inter continental routes or routes in enclosed seas such as the Baltic, Mediterranean and Black Sea the overall structure is usually not inbound-outbound. As a result, better routes may be found by relaxing the inbound-outbound restriction.

Rana and Vickson [16], Christiansen and Nygreen [5], Fagerholt [8], Agarwal and Ergun [1] and Alvarez [2] allow for several visits to a port. The allowance of several visits to a port is, in all the mentioned papers (with the exception of [1]) achieved by combining simple routes. In a simple route each port is visited at most once. Agarwal and Ergun [1] solve the problem by using a time-space graph where a port can be visited several times as long as the visit is not on the same weekday.

The shipping companies often wish to schedule the frequency of a departure at a port so that it corresponds to the demand at the port. Fagerholt [8] and Christiansen and Nygreen [5], and Agarwal and Ergun [1] have a weekly frequency requirement on the routes. Fagerholt [8] and Christiansen and Nygreen [5] formulate the weekly frequency by restricting the time of an route to be less than a week. This is applicable to small shipping routes such as regional routes. However, it is clear that when it comes to intercontinental shipping the routes are usually longer than a week. This is handled in Agarwal and Ergun [1] by covering the weekly departures with a sufficient number of vessels of the same type.

The use and influence of transhipment on the liner shipping network design is described by Notteboom and Rodrigue [15]. However only a few decades ago the use of transhipment was much less common. Therefore older articles such as Rana and Vickson [16] do not include transhipment in their route planning. The model solved in [16] was extended in the recent work by Shintani et al [19], where the restrictive visiting order of Rana and Vickson [16] is relaxed as to represent a more realistic set of routes. Moreover, the repositioning of empty containers is included by Shintani et al. [19]. To solve the problem presented in [19] a genetic algorithm is used, however, transhipment is not considered. Christiansen and Nygreen [5] use

column generation to solve the routing problem for ammonia shipping in Norway. In the problem solved in [5] only ammonia is shipped and therefore transhipment is not considered. Fagerholt [8] apply column generation for solving the liner shipping problem along the Norwegian coast. Others, such as Gelareh and Meng [10] exclusively deal with the fleet deployment on a predefined set of routes. In the recent paper by Agarwal and Ergun [1], the authors solve larger problems by using a heuristic based on Benders' Decomposition and compare it to a similar solution method which uses column generation. Recently an article on liner shipping network design optimization has been publish by Alvarez [2] using tabu search and column generation. The model has transhipment costs as part of the overall cost evaluation. Rana and Vickson [16], Christiansen and Nygreen [5] and Fagerholt [8] do not consider transhipment. Even though Notteboom and Rodrigue in [15] emphasize the importance of transhipment in the shipping networks, Agarwal and Ergun in [1] are the first to include transhipment in the liner shipping network design problem. However, they do not include transhipment cost and Alvarez in [2] from 2009 is to our knowledge the first to consider the cost of transhipment when designing the shipping network. In models where each port is represented by one vertex at the points where two cycles are connected a transshipment from an early visit of a vessel to a later visit of the same vessel can occur. As will be discussed in Section 3.2.1 this results in complications in the calculation of transshipment costs. To our knowledge the exact cost of transshipment has not been calculated at the cycle connection points earlier. Note that Argawal and Ergun [1] do not use a single vertex for representing a port and that they do not include transshipment cost. Clearly increasing the number of vertices and thereby the number of edges in the graph will significantly increase the complexity of the problem even though it gives more flexibility in the route structure.

The model by Agarwal and Ergun [1] and Alvarez [2] are so far the most comprehensive representations of the problem faced by liner shipping companies. Alvarez [2] include many relevant parameters in the objective while Agarwal and Ergun [1] only include cost.

Even though shipping companies often have several vessels of the same type it is not always an optimal solution to force the routes to be sailed by the same vessel type and in real-life routes there are some smaller ports which, due to low demand, only require a bimonthly departure and some busy ports might require a biweekly departure.

To the best of our knowledge no results for the LS-NDP, using branch-and-cut, have been presented in the literature. As mentioned in Section 1 good results have been achieved by [4] and [12] when applying branch-and-cut to the VRPTW. Since the VRPTW is somewhat similar to the LS-NDP with its heterogeneous fleet and cyclic routes it is natural to assume that branch-and-cut also will result in good solutions for the LS-NDP.

3 Problem formulation

Let G be a directed graph and let $G(\mathbf{N}, \mathbf{A}, \mathbf{V}, \mathbf{M}, t_{max})$ represent the network with vertex set $\mathbf{N}$, arc set $\mathbf{A}$, a set of vessels $\mathbf{V}$, a set of demands $\mathbf{M}$ and a forecast period with length t_{max}. Each vertex $n \in \mathbf{N}$ represents a port. Each arc $(i, j) \in \mathbf{A}$ is a direct connection between two ports for a given vessel $v \in \mathbf{V}$. Each demand $m \in \mathbf{M}, m = (i, j, d, t)$ is the amount $d \in \mathbb{Z}$ of type t to be shipped from an origin port i to a destination port j. Each vertex j has a cost of transhipping demand m, depending on the type of demand, c_j^m and a service time t_j. Each arc a has a cost c_{ij}^m of carrying demand m on a direct connection from port i to port j. Each arc (i, j) also has an associated time t_{ij}^v reflecting the prefixed time it takes for vessel v to sail a direct connection from port i to port j. Each vessel $v \in \mathbf{V}$ has a capacity C^v. The liner shipping network design problem is to find a connected route for each vessel $v \in \mathbf{V}$ where the customer demands are satisfied and the overall cost is minimized. Since a vessel assigned a route sails continuously during the whole planning horizon, the cost to be minimized is a linear function of the cost of using a selected vessel, the cost of transporting a demand on the arcs and the cost of transhipping at ports. It can be argued that the cost of transporting a demand is negligible however by introducing a small cost corresponding to time, one can be assured that unnecessary extra time or travel is avoided for the demand. In the cost of transporting a demand we only include the time the demand spend on the vessel and not the time the demand uses at a port during transhipment. The objective is to minimize the overall cost so that the required demand can be shipped from their origin to their destination within the time interval of length t_{max}.

Figure 1 shows an example of a network containing two butterfly routes. In Figure 1 transshipment can

occur at the ports A,B and C. At ports A and B transhipment can occur between the two routes moreover at port A transhipment can occur between two visits of route 2 and at port C transhipment can occur between two visits of route 1.

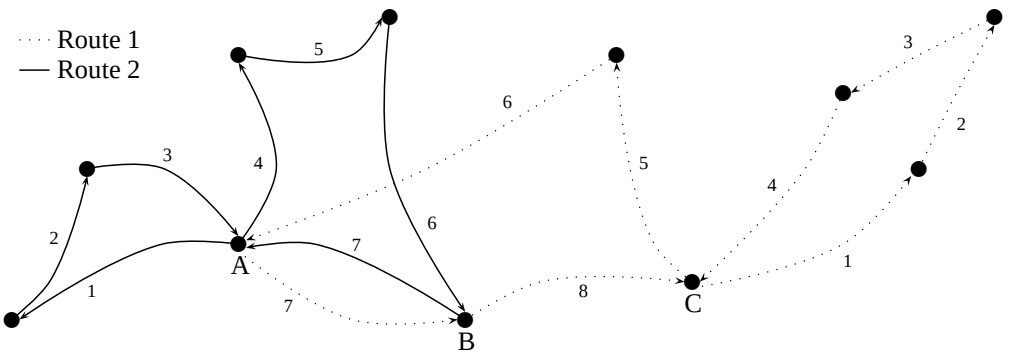


Figure 1: An example of two routes in a liner shipping network. Each route can be constructed by following the arcs in increasing order starting with arc number 1. Transhipment can take place at port A, B and C.

3.1 Mathematical Model

There is no standard mathematical formulation of the liner shipping problem since each liner shipping company has specific constraints based on strategic decisions. As a result of this, several formulations and models have been presented in the literature.

In this section we first present a comprehensive mathematical model for the liner shipping problem which includes transhipment, transhipment cost, simple routes, butterfly routes and a heterogeneous vessel fleet.

3.2 The Network Design Problem

In the model presented by Agarwal and Ergun in [1] a time-space graph structure is used, where the time is the day of week and weekly departure by vessels of the same type is enforced on the same weekday. However in the here presented version of the liner shipping problem a port is allowed to be visited less than once a week and different vessel types are permitted to sail the same route. Allowing for other than weekly departure and different vessel types on a route may result in lower cost.

We have the following variables:

x_{ij}^{mv} the amount of demand m shipped on arc (i, j) by vessel v,

u_{ij}^{v} a binary variable which is 1 if arc (i, j) is the first or the last arc on a route of vessel v with two loops, 0 otherwise,

e_{ij}^{v} a variable enumerating the order of the arcs on the route,

y_{ij}^{v} a binary variable which is 1 if arc (i, j) is in the route of vessel v, 0 otherwise,

f_{j}^{mv} the amount of demand m from vessel v transhipped at port j,

s_{i}^{v} a binary variable which is 1 if i is the port which may connect two loops for vessel v, denoted centerpoint,

f_{jih}^{mv} the amount of demand m from vessel v entering i from j and not leaving on the arc from i to h,

τ_{v} the route travel time of vessel v,

h^{v} a binary variable which is 1 if vessel v is sailing and 0 otherwise,

We use the following parameters

C^v the capacity of vessel v,

t^v_{ij} the time it takes for vessel v to sail arc (i,j)

t_{max} the duration of the forecast period,

t_q the time at quay at any port.

The demands are defined as:

$$b^m_i = \begin{cases} d^m & \text{if } i = o(m) \\ -d^m & \text{if } i = d(m) \\ 0 & \text{otherwise} \end{cases} \quad m \in \mathbf{M}, i \in \mathbf{N}$$

where $o(m)$ is the port of origin of demand m and $d(m)$ is the destination port of demand m.

The four "big-M" coefficients M_1,M_2,M_3 and M_4 are sufficiently large constants. We operate with three different costs: c^v the cost of vessel v sailing, c^m_{ij} the cost of shipping demand m on connection (i,j), and c^m_i the cost of transhipping demand m at port i. This leads to the model:

$$\textbf{Min:} \sum_{m\in\mathbf{M}} \sum_{(i,j)\in\mathbf{A}} c^m_{ij} \sum_{v\in\mathbf{V}} x^{mv}_{ij} + \sum_{m\in\mathbf{M}} \sum_{j\in\mathbf{N}} c^m_j f^{mv}_j + \sum_{v\in\mathbf{V}} c^v h^v \tag{1}$$

$s.t.$

$$\text{(Flow)} \quad \sum_{v\in\mathbf{V}} \sum_{j:(i,j)\in\mathbf{A}} x^{mv}_{ij} - \sum_{v\in\mathbf{V}} \sum_{j:(j,i)\in\mathbf{A}} x^{mv}_{ji} = b^m_i \qquad i \in \mathbf{N}, m \in \mathbf{M} \tag{2}$$

$$\text{(Trans 0)} \quad f^{mv}_i \geq \sum_{j:(j,i)\in\mathbf{A}} x^{mv}_{ji} - \sum_{j:(i,j)\in\mathbf{A}} x^{mv}_{ij} \qquad m \in \mathbf{M}, i \in \mathbf{N}, v \in \mathbf{V} \tag{3}$$

$$\text{(Trans 1)} \quad f^{mv}_i \geq \sum_{j,h\in\mathbf{N},v\in\mathbf{V}} f^{mv}_{jih} - M_1(1 - s^v_i) \qquad m \in \mathbf{M}, i \in \mathbf{N}, v \in \mathbf{V} \tag{4}$$

$$\text{(Trans 2)} \quad f^{mv}_{hij} \geq x^{mv}_{ji} - x^{mv}_{ih} - M_2(2 - y^v_{ji} - y^v_{ih} + u^v_{ji} + u^v_{ih}) \qquad m \in \mathbf{M}, j, i, h \in \mathbf{N}, v \in \mathbf{V} \tag{5}$$

$$\text{(Trans 3)} \quad f^{mv}_{hij} \geq x^{mv}_{ji} - x^{mv}_{ih} - M_3(4 - u^v_{ji} - u^v_{ih} - y^v_{ji} - y^v_{ih}) \qquad m \in \mathbf{M}, j, i, h \in \mathbf{N}, v \in \mathbf{V} \tag{6}$$

$$\text{(Capacity)} \quad \frac{t_p}{\tau_v} C^v y^v_{ij} \geq \sum_{m\in\mathbf{M}} x^{mv}_{ij} \qquad (i,j) \in \mathbf{A}, v \in \mathbf{V} \tag{7}$$

$$\text{(Center)} \quad \sum_{i\in\mathbf{N}} s^v_i = 1 \qquad v \in \mathbf{V} \tag{8}$$

$$\text{(First arc)} \quad \sum_{(ij)\in\mathbf{A}} u^v_{ij} = 2 \qquad v \in \mathbf{V} \tag{9}$$

$$\text{(Out arc)} \quad s^v_i - \sum_{(ij)\in\mathbf{A}} u^v_{ij} \leq 0 \qquad i \in \mathbf{N}, v \in \mathbf{V} \tag{10}$$

$$\text{(In arc)} \quad s^v_i - \sum_{(ji)\in\mathbf{A}} u^v_{ji} \leq 0 \qquad i \in \mathbf{N}, v \in \mathbf{V} \tag{11}$$

$$\text{(Cyclic)} \quad \sum_{j:(i,j)\in\mathbf{A}} y^v_{ij} - \sum_{j:(j,i)\in\mathbf{A}} y^v_{ji} = 0 \qquad i \in \mathbf{N}, v \in \mathbf{V} \tag{12}$$

$$\text{(Connect 0)} \quad \sum_{j:(i,j)\in\mathbf{A}} y^v_{ij} - s^v_i \leq 1 \qquad i \in \mathbf{N}, v \in \mathbf{V} \tag{13}$$

$$\text{(Connect 1)} \quad e^v_{ji} - e^v_{ih} + M_4(y^v_{ih} + y^v_{ji} - 2 - u^v_{ji} - u^v_{ih}) \leq -1 \qquad i, j, h \in \mathbf{N}, v \in \mathbf{V} \tag{14}$$

$$\text{(Ships)} \quad y^v_{ij} - h^v \leq 0 \qquad (i,j) \in \mathbf{A}, v \in \mathbf{V} \tag{15}$$

$$\text{(Time 0)} \quad \tau_v \leq t_{max} \qquad v \in \mathbf{V} \tag{16}$$

$$\text{(Time 1)} \quad \tau_v = \sum_{i,j:(i,j)\in\mathbf{A}} y^v_{ij} \left(t^v_{ij} + t_j\right) \qquad v \in \mathbf{V} \tag{17}$$

$$u_{ij}^v, y_{ij}^v \in \{0,1\} \qquad (i,j) \in \mathbf{A}, v \in \mathbf{V} \tag{18}$$
$$f_{jih}^{mv} \geq 0 \qquad m \in \mathbf{M}, j,i,h \in \mathbf{N}, v \in \mathbf{V} \tag{19}$$
$$e_{ij}^v \in \mathbb{Z}^+ \qquad i,j \in \mathbf{N}, v \in \mathbf{V} \tag{20}$$
$$x_{ij}^{mv} \geq 0 \qquad (i,j) \in \mathbf{A}, m \in \mathbf{M}, v \in \mathbf{V} \tag{21}$$
$$s_i^v \in \{0,1\} \qquad i \in \mathbf{N}, v \in \mathbf{V} \tag{22}$$
$$h^v \in \{0,1\} \qquad v \in \mathbf{V} \tag{23}$$

The objective (1) minimizes the sum of the cost of transporting the demand, the cost of transhipping demand and the cost of using the vessels. Constraints (2) ensure flow conservation that all demand $m \in M$ is satisfied. Constraints (3) ensure that f_i^{mv} is larger than the difference between the incoming demand m and outgoing demand m on a vessel v. Since the objective is to minimize the cost and $C_i^m f_i^{mv}$ is positive then f_i^{mv} will be equal to the amount transhipped. Constraints (4), (5) and (6) together with the constraints (9), (10),(11) and (14) for u_{ij}^v ensure that f_i^{mv} at the vertex i connecting two loops sailed by the same vessel also includes the amount left at the port to be picked up later by the same vessel.

The capacity constraints (7) ensure that the amount shipped on vessel v on arc (i,j) is less than the capacity of the vessel v multiplied by the number of trips which can be completed in the schedule period t_{max}. Note that the value of τ_v is determined in constraint (17) where the right hand side is the time of the route sailed by vessel v. The constraints (8) ensure that for each route exactly one vertex is selected as centerpoint. The constraints (12) ensure that for every port every vessel, which enters the port, also leaves the port. Constraints (13) ensure that a vessel v does not visit its selected start port more than twice. Constraints (14) ensure that all parts of a route sailed by vessel v is connected to the start port s_i^v of vessel v. Constraints (15) ensure that there will be a cost for using vessel v in the objective. Constraints (16) ensure that no route is longer than the schedule period.

In the following sections we will discuss how the requirements special to the LS-NDP can be formulated in a linear model.

3.2.1 Transhipment cost in the Liner Shipping Network Design

Agarwal and Ergun [1] argued that transhipment is the core of liner shipping. We would like to add that transhipment of goods is frequently occurring in liner shipping and the associated cost should not be ignored when designing the network. Transhipment are allowed in the model presented in [1], however the expenses of transhipment were not included in the cost calculation before the work by Alvarez in [2]. To calculate the transhipment cost when satisfying demands in a specific network design one must know the amount of containers, which is transhipped. We define a variable f_i^{vm} which is the amount of containers in demand m transhipped at port i. In the objective function (1) the cost c_j^m of transhipping one unit at port i is included. To find the value of f_i^{vm} we have the constraints (3). When the routes are simple the amount transshipped can be calculated by constraint (3) alone. However when *butterfly* routes exists there can be cargo transhipped at the centerpoint which is not calculated by the constraint (3). This cargo is the containers transhipped between two visits of the same route to the port. Therefore to calculate the exact amount of containers it is important to be able to distinguish between the two visits to the centerpoint. This can be achieved by enumerating the edges on the route and marking the first and last edge on the entire route. The integer variables e_{ij}^v enumerates the edges on the route and the binary variables u_{ij}^v marks the first and last edge on a *butterfly* route. The following constraints ensure that the first and last edge on a *butterfly* route are found:

$$\text{(First arc)} \qquad \sum_{(ij)\in\mathbf{A}} u^v_{ij} = 2 \qquad v \in \mathbf{V} \qquad (24)$$

$$\text{(Out arc)} \qquad s^v_i - \sum_{(ij)\in\mathbf{A}} u^v_{ij} \leq 0 \qquad i \in \mathbf{N}, v \in \mathbf{V} \qquad (25)$$

$$\text{(In arc)} \qquad s^v_i - \sum_{(ji)\in\mathbf{A}} u^v_{ji} \leq 0 \qquad i \in \mathbf{N}, v \in \mathbf{V} \qquad (26)$$

$$\text{(Connect 1)} \qquad e^v_{ji} - e^v_{ih} + M_4(y^v_{ih} + y^v_{ji} - 2 - u^v_{ji} - u^v_{ih}) \leq -1 \qquad i,j,h \in \mathbf{N}, v \in \mathbf{V} \qquad (27)$$

Where s^v_i is the port selected as centerpoint for the route. The constraints (27) ensure that if the route is a *butterfly* route then the last edge (j,i) on the route $u^v_{ji} = 1$ and the first edge (i,h) on the route $u^v_{ih} = 1$.

To find the demand transhipped from one visit to another visit of the same vessel we introduce the variable f^{mv}_{jih} indicating the transhipment in i when arriving form port j and departing to port h. Then on a *butterfly* route the two visits to a centerpoint i are the one where $u^v_{ji} = u^v_{ih} = 1$ and $y^v_{ji} = y^v_{ih} = 1$ and the one where $u^v_{ki} = u^v_{il} = 0$ and $y^v_{ki} = y^v_{il} = 1$ Clearly if $u^v_{ji} = u^v_{ih} = 1$ and $y^v_{ji} = y^v_{ih} = 1$ then the transhipment at one visit to the port i is $x^{mv}_{ji} - x^{mv}_{ih}$, which can be formulated as:

$$\text{(Trans 3)} \quad f^{mv}_{jih} \geq x^{mv}_{ji} - x^{mv}_{ih} - M_3(4 - u^v_{ji} - u^v_{ih} - y^v_{ji} - y^v_{ih}) \quad m \in \mathbf{M}, j,i,h \in \mathbf{N}, v \in \mathbf{V} \quad (28)$$

If $u^v_{ki} = u^v_{il} = 0$ and $y^v_{ki} = y^v_{il} = 1$ then the transhipment at one visit to the port i is $x^{mv}_{ki} - x^{mv}_{il}$, which can be formulated as:

$$\text{(Trans 2)} \quad f^{mv}_{kil} \geq x^{mv}_{ki} - x^{mv}_{il} - M_2(2 - y^v_{ki} - y^v_{il} + u^v_{ki} + u^v_{il}) \quad m \in \mathbf{M}, k,i,l \in \mathbf{N}, v \in \mathbf{V} \quad (29)$$

This must be included in the value of the f^{mv}_i used in the objective. However the f^{mv}_i only need to be adjusted for the centerpoint of the route. The port i is a centerpoint if $s^v_i = 1$. At the centerpoint the transhipment amount is the sum of the transhipment on the two visits. This can be formulated as:

$$\text{(Trans 1)} \qquad f^{mv}_i \geq \sum_{j,h\in\mathbf{N},v\in\mathbf{V}} f^{mv}_{jih} - M_1(1 - s^v_i) \qquad m \in \mathbf{M}, i \in \mathbf{N}, v \in \mathbf{V} \qquad (30)$$

Constraints (28) calculates the amount unloaded from the vessel at the visit to port i from the end edge to the start edge of the route. Constraints (29) calculates the amount unloaded from the vessel at the other visit to port i. Constraints (30) ensure that this is included in the transhipment on route v at a centerpoint i. The number of constraints in (29) and (28) is $O(|N^3||M||V|)$ which is a significantly large number. The additional binary and integer variables u^v_{ij} and e^v_{ij} may increase the size of the branch and bound tree.

3.2.2 The cyclic structure of liner shipping routes

In the liner shipping network design problem a vessel must leave each port it enters. This is called flow conservation and is modeled by constraints (12).

Clearly it is important to ensure that a route is connected so that it can be sailed by a single vessel and avoid having several disconnected subtours, for example two separate cycles, representing a route. When modeling the constraint that the route must be connected it is often assumed that the route is simple. Although the routes are simple in [1], the time-space graph used by Agarwal and Ergun in [1] allows for multiple visits to a port as long as the visits do not happen on the same day of the week.

In the model presented here we let each route have a port which may be visited twice. This is to model the real life situation with some port used as hubs for the other ports. Notteboom notes in [14] that this form of design is used by Maersk Sealand. In liner shipping such routes are denoted *butterfly* routes.

To model *butterfly* routes the binary variable s^v_i is introduced indicating which port on the route may be used as Hub. To have a polynomial number of constraints ensuring that the routes are connected we have used the approach proposed by Tucker et al. [13] for enumerating the vertices on a simple path. In constraints (14) the arcs on the route are enumerated instead of the ports using the u^v_{ij} variables to mark the

start and end arc of the route. The constraints presented by Tucker et al. [13] are used in vehicle routing problems and variances with simple routes. However in the presented model for the LS-NDP a single port on the route may be visited twice. Allowing the possibility of two visits to the hub port it could be formulated as:

$$\text{(Center)} \qquad \sum_{i \in \mathbf{N}} s_i^v = 1 \qquad v \in \mathbf{V} \tag{31}$$

$$\text{(Butterfly)} \qquad z_i^v - z_j^v + M_4(y_{ij}^v - s_j^v) \leq M_4 - 1 \qquad i,j \in \mathbf{N}, v \in \mathbf{V} \tag{32}$$

Where z_j^v is an positive integer that indicates the order in which the ports are visited. However, since the cost of transshipment is included as described in previous Section 3.2.1 it is needed to know the start and end edges of the route at the hub port. Therefore the constraint is formulated as:

$$\text{(Connect 1)} \qquad e_{ji}^v - e_{ih}^v + M_4(y_{ih}^v + y_{ji}^v - 2 - u_{ji}^v - u_{ih}^v) \leq -1 \qquad i,j,h \in \mathbf{N}, v \in \mathbf{V} \tag{33}$$

This means that constraints concerning s_i^v and u_{ij}^v must be included. Therefore to model connected *butterfly* routes while allowing for calculating the transhipment cost the constraints (9), (10), (11), (12), (13), (31) and (33) are needed. Note that there is an overlap with the constraints needed for calculating the exact transhipment cost.

3.2.3 The number of times a route can be completed in a schedule period

The number of times a link is sailed during the time period affects the capacity on the given link.

In our model a route can at most contain a link once. However, every link on a given route is sailed the number of times the route can be completed by the assigned vessel in the schedule period. Since a link can be sailed on several different routes, the number of times a link is sailed also depends on the number of routes the link appears in.

For example a vessel with the capacity to carry 1000 containers and sailing a route which takes 30 days can in a 30 day forecast period only ship 1000 containers on each leg of the route. However if the same vessel sailed a route which only takes 5 days it could on each leg of the route, ship 6000 containers during the same period. Therefore we include the route length in the capacity constraint in the LS-NDP model. The consideration of route length in liner shipping network design was first introduced by Agarwal and Ergun [1], where vessels of the same type are assigned to a route so that there is always a weekly departure. As mentioned earlier weekly departures are not a strict requirement for all shipping companies. In real-life shipping, ports with smaller demands are visited bi-monthly. Moreover it may happen that a shipping company does not have the right number of ships of a specific type to cover a weekly departure on a route. It is also likely that a better solution has different vessel types assigned to a route.

To include the time of the route in the calculation of the capacity, we multiply the capacity of a vessel with the number of times the route can be completed during the forecast period. This requirement is formulated by the constraints (7) the partial route is included in the capacity calculation as a partial vessel capacity. These constraints are not linear and thus to solve this problem using an integer programming solver it is necessary to linearize the constraints.

We here linearize the equation expressed by constraints (7). This is done by introducing the following variables:

q^v	t_{max}/τ_v, the schedule period divided by the route time.
r_{gh}^v	Some real number greater than the travel time of vessel v on arc (g,h) plus service time at port h multiplied by the times the route can be completed
M	Upper bound for the maximum capacity times maximum route time on any arc.

If $y_{ij}^v = 0$ the flow x_{ij}^{mv} must be equal to 0. Thus we introduce the constraints:

$$\text{(Capacity 2)} \qquad \sum_{m \in \mathbf{M}} x_{ij}^{mv} \leq y_{ij}^v M \qquad (i,j) \in \mathbf{A}, v \in \mathbf{V} \tag{34}$$

These constraints ensure that nothing can be transported on an arc not included in a route. Now we look at the remaining case where the arc (i,j) is traversed. Since $y_{ij}^v = 1$ and $t_{gh}^v + t_h > 0$ then we know that $\tau_v > 0$.

We introduce the variable q^v so that :

$$t_p/\tau_v \geq q^v \quad v \in \mathbf{V} \tag{35}$$

Where $q^v \in \mathbb{R}_0^+$. Note that the constraint (35) is not linear. Since $\tau_v > 0$ for all $v \in \mathbf{V}$, we can express constraint (35) as:

$$t_{max} \geq q^v \tau_v \quad v \in \mathbf{V} \tag{36}$$

However constraint (36) is still not linear. An entry in the sum over $(g,h) \in \mathbf{A}$ on the righthand side is $q^v \left(t_{gh}^v + t_h\right)$ when y_{gh} is one, and zero when y_{gh} is zero. Therefore to linearize this by the "Big M" method from [20] we write the following constraints:

$$\text{(Cap 3)} \qquad r_{gh}^v + M(1 - y_{gh}^v) - q^v(t_{gh} + t_h) \geq 0 \qquad (g,h) \in \mathbf{A}, v \in \mathbf{V} \tag{37}$$

$$\text{(route time)} \qquad t_{max} - \sum_{(g,h)\in\mathbf{A}} r_{gh}^v \geq 0 \qquad v \in \mathbf{V} \tag{38}$$

$$q^v \geq 0 \qquad v \in \mathbf{V} \tag{39}$$

$$r_{gh}^v \geq 0 \qquad (g,h) \in \mathbf{A}, v \in \mathbf{V} \tag{40}$$

Constraints (37) ensure that when y_{gh} is one then $r_{gh}^v \geq q^v(t_{gh} + t_h)$. When y_{gh} is zero then: $r_{gh}^v \geq q^v(t_{gh} + t_h) - M$. Note that M must be chosen so that $q^v(t_{gh} + t_h) - M \leq 0$, and that constraints (38) can replace constraints (16) in the model.

Now we can let q^v replace t_{max}/τ_v in the constraint formulation (7) and thereby we get inequality:

$$q^v C^v y_{ij}^v \geq \sum_{m\in\mathbf{M}} x_{ij}^m \quad (i,j) \in \mathbf{A}, v \in \mathbf{V} \tag{41}$$

which we again must linearize. Here we note that the left hand side is equal to $q^v C^v$ when $y_{ij}^v = 1$ and zero otherwise. Hence

$$\text{(Capacity 1)} \qquad q^v C^v + M(1 - y_{ij}^v) \geq \sum_{m\in\mathbf{M}} x_{ij}^{mv} \qquad (i,j) \in \mathbf{A}, v \in \mathbf{V} \tag{42}$$

The constraints of type (42) ensure that the flow on all sailed arcs is less than $q^v C^v$. For all arcs not sailed constraints (42) does not add any restrictions given that M is chosen big enough. Recall that constraints (34) ensure that there is not assigned flow to arcs which are not sailed. We include constraints (34), (37), (38), (42) and variable definitions (39) and (40) to replace the non-linear capacity constraints (7) and constraints (16) and (17).

These constraints are included in the integer linear programming (ILP) model used in the test for the branch-and-bound and branch-and-cut method.

3.2.4 The Compact model for Liner Shipping

The linear model for the liner shipping problem, which here is named the *compact model* is the model presented in the beginning of this section where constraints (7), (16) and (17) are replaced by the constraints (34), (37), (38), (42) and variable definitions (39) and (40). As the name indicates the *compact model* has a polynomial number of constraints. Since this model is linear it can be solved directly by an ILP solver. The problem is NP hard as it includes the model [1] as a special case, and the 'big-M' constraints (4),(5),(6),(14),(37) and (42) together with the large number of variables make the problem hard to solve for ILP solvers.

4 The Solution Method

The *compact model* can be solved using branch-and-bound but the 'big-M' constraints may result in large integrality gaps and poor bounds resulting in large search trees. Moreover the many variables make the problem combinatorically hard. As mentioned earlier the branch-and-cut method has successfully been applied to vehicle routing problems (Ascheuer et al. [3]) and other transportation network design problem, (Gendreau et al. [11]). Therefore it is interesting to investigate the possibilities for using the branch-and-cut method on the LS-NDP and compare it with a branch-and-bound method.

4.1 Branch-and-cut

The branch-and-cut method generally give good results on problems with complicating constraints such as non linear constraints or problems with an exponential number of constraints. As in Ascheuer et al. [3] and Gendreau et al. [11] we gradually add the transhipment and connectivity constraints to the formulation when they are violated.

4.1.1 Transhipment cuts

Calculating the amount unloaded from a vessel at a port to be loaded onto the same vessel at a later visit to the port is quite cumbersome. For calculating the transhipment to be picked up at a port by the same vessel we use the constraints (4),(5) and (6). Note that constraints (5) and (6) each represents $|N|^3|M||V|$ constraints. We wish to remove the constraints (4),(5) and (6) and introduce them as cuts when they are violated. Transhipment to be picked up at a port by the same vessel only occurs on a *butterfly* route at the point the two loops meet. The point where the two loops meet are the centerpoint of the route and it is indicated by $s_i^v = 1$.

We have constructed a cut so that if all arcs in a set of arcs T are sailed by a vessel v then if it is a butterfly loop with the centerpoint $s_i^v = 1$, we have two arcs ji and ih in T which are not on the same loop that can be selected as the start and end arc of the route. This is formulated as:

$$u_{ji}^v + u_{ih}^v + 2(|T| - y^v(T)) = 2, \tag{43}$$

where the arcs ji and ih are the first and last arc on the route. For calculating the transhipment between two visits by the same vessel on this route we add the following constraints as cuts:

$$\text{(Tranship 1)} \qquad f_i^{mv} \geq \sum_{j,h \in \mathbf{N}, v \in \mathbf{V}} f_{jih}^{mv} - M(1 - s_i^v) \tag{44}$$

$$\text{(Tranship 2)} \qquad f_{hij}^{mv} \geq x_{ji}^{mv} - x_{ih}^{mv} - M(2 - y_{ji}^v - y_{ih}^v + u_{ji}^v + u_{ih}^v) \tag{45}$$

$$\text{(Tranship 3)} \qquad f_{hij}^{mv} \geq x_{ji}^{mv} - x_{ih}^{mv} - M(4 - u_{ji}^v - u_{ih}^v - y_{ji}^v - y_{ih}^v) \tag{46}$$

Note that for each of the constraints (43) added one of each constraint (44),(45) and (46) is added.

4.1.2 Connectivity cuts

In the network design cases where branch-and-cut has been applied it is assumed that routes are simple. For simple routes in the generalized traveling salesman problem the connectivity constraints have been formulated by Fischetti et al. [9] as:

$$\sum_{i,j \in S} y_{ij} \leq \sum_{h \in \mathbf{N} g \in S \setminus \{k\}} y_{hg}^v - \sum_{e \in \mathbf{N}} y_{el}^v + 1 \qquad v \in \mathbf{V}\ \emptyset \subset S \subset \mathbf{N}, k \in S, l \in \mathbf{N} \setminus S \tag{47}$$

As mentioned before the real routes of the shipping companies are often not simple and we have introduced the concept of *butterfly* routes in Section 3.2.2. This extension introduces new and weaker connectivity constraints. Since the first and last edge is selected by the transhipment cuts (see Section 4.1.1) the edge order can be ignored here.

The connectivity constraints from [9] have been modified to allow *butterfly* routes. We will present a connectivity cut which will allow for $\psi + 1$ subtours which all have exactly one point in common. Because the subtours must go through exactly one common point, we call this type of cut a *clover-cut*. The clover-cut removes any route which is not connected but it keeps routes which are pseudo-simple.

Lemma 1 *For any cyclic disconnected* clover *path there exists a set S, and two vertices k and l for which the following clover-cut inequality is violated. Moreover no S,k and l violating a connected* clover *path exists. This can be expressed as:*

$$\sum_{i,j \in S} y_{ij}^{v} \leq \sum_{h \in \mathbf{N} g \in S \setminus \{k\}} y_{hg}^{v} + \psi s_k^{v} - \sum_{e \in \mathbf{N}} y_{el}^{v} + \psi s_l^{v} + 1 \qquad v \in \mathbf{V}, \emptyset \subset S \subset \mathbf{N}, k \in S, l \in \mathbf{N} \setminus S \quad (48)$$

Proof
Let v_1 be the route. First we prove that we cannot find a S, k and l for which the clover-cut inequality does not hold for a connected pseudo-simple path.

Case 1 on S: Assume that there is no part of the route v_1 outside of S, where v_1 is a connected pseudo-simple route. Then $\sum_{e \in \mathbf{N}} y_{el}^{v_1} = 0$. Moreover by constraints (13) we have that all vertices with $s_i^{v_1} = 0$ has at most one ingoing arc and the one vertex with $s_i^{v_1} = 1$ has at most ψ ingoing arcs. Therefore for all $k \in S$ and $l \in \mathbf{N} \setminus S$ it must holds that

$$\sum_{i,j \in S} y_{ij}^{v_1} \leq \sum_{h \in \mathbf{N} g \in S \setminus \{k\}} y_{hg}^{v_1} + \psi s_k^{v_1} - \sum_{e \in \mathbf{N}} y_{el}^{v_1} + \psi s_l^{v_1} + 1 \quad (49)$$

Case 2 on S: Assume that there is a part of the route for v_1 outside of S and that v_1 is a pseudo-simple route. In this case clearly there must be at least one arc $y_{ij}^{v_1}$ where $i \in \mathbf{N} \setminus S, j \in S$.

$s_k^{v_1} = 0 \wedge s_l^{v_1} = 0$**:** Then $0 \leq \sum_{e \in \mathbf{N}} y_{el}^{v_1} \leq 1$. In this case the clover-cut holds if the following inequality holds $\sum_{i,j \in S} y_{ij}^{v_1} \leq \sum_{h \in \mathbf{N} g \in S \setminus \{k\}} y_{hg}^{v_1}$. Since $s_k^{v_1} = 0$ and therefore there is at most one arc entering vertex k and since the whole route is not in S this inequality is trivially true for connected pseudo-simple routes.

$s_k^{v_1} = 1 \wedge s_l^{v_1} = 0$**:** The inequality becomes $\sum_{i,j \in S} y_{ij}^{v_1} \leq \sum_{h \in \mathbf{N} g \in S \setminus \{k\}} y_{hg}^{v_1} - \sum_{e \in \mathbf{N}} y_{el}^{v_1} + 1 + \psi$. In this case $0 \leq \sum_{e \in \mathbf{N}} y_{el}^{v_1} \leq 1$. Therefore it is enough to show that $\sum_{i,j \in S} y_{ij}^{v_1} \leq \sum_{h \in \mathbf{N} g \in S \setminus \{k\}} y_{hg}^{v_1} + \psi$ which clearly holds for a connected pseudo-simple route since $\psi \geq \sum_{h \in \mathbf{N}} y_{hk}^{v_1}$.

$s_k^{v_1} = 0 \wedge s_l^{v_1} = 1$**:** Since $\sum_{e \in \mathbf{N}} y_{el}^{v_1} \leq \psi$. Then clearly if $\sum_{i,j \in S} y_{ij}^{v_1} - 1 \leq \sum_{h \in \mathbf{N} g \in S \setminus \{k\}} y_{hg}^{v_1}$ the clover cut will hold. This is trivially true since $\sum_{h \in \mathbf{N} g \in S \setminus \{k\}} y_{hg}^{v_1} \geq \sum_{i,j \in S} y_{ij} + \sum_{m \in \mathbf{N} \setminus S, n \in S} y_{mn}^{v_1} - 1$. Where $\sum_{m \in \mathbf{N} \setminus S, n \in S} y_{mn}^{v_1} \geq 0$ and thus $\sum_{h \in \mathbf{N} g \in S \setminus \{k\}} y_{hg}^{v_1} \geq \sum_{i,j \in S} y_{ij} - 1$.

Therefore this cut holds for all connected pseudo-simple paths.

Now we will prove that there exists S, k and l so that the clover-cut does not hold when v_1 is disconnected.

Let v_{1_1} and v_{1_2} be two disconnected components of v_1. Let S contain exactly the vertices of v_{1_1}. Clearly by constraints (12) v_{1_1} and v_{1_2} are cyclic. Since v_{1_2} is in $\mathbf{N} \setminus S$ and since $\mathbf{N} \setminus S \geq 2$ and $S \geq 2$, we can choose l on v_{1_2} so that $s_l^{v_1} = 0$ and k on v_{1_1} so that $s_k^{v_1} = 0$. By the choice of S there is no arcs entering S and therefore we have that $\sum_{i,j \in S} y_{ij}^{v_1} > \sum_{h \in \mathbf{N} g \in S \setminus \{k\}} y_{hg}^{v_1}$. Moreover as l is on v_{1_2} and $s_l^{v_1} = 0$ then $\sum_{e \in \mathbf{N}} y_{el}^{v_1} = 1$. Thus clearly $\sum_{i,j \in S} y_{ij}^{v_1} > \sum_{h \in \mathbf{N} g \in S \setminus \{k\}} y_{hg}^{v_1} + \psi s_k^{v_1} - \sum_{e \in \mathbf{N}} y_{el}^{v_1} + \psi s_l^{v_1} + 1$.

□

For our case with *butterfly* routes $\psi = 1$ and the cut becomes:

$$\sum_{i,j \in S} y_{ij}^{v} \leq \sum_{h \in \mathbf{N} g \in S \setminus \{k\}} y_{hg}^{v} + s_k^{v} - \sum_{e \in \mathbf{N}} y_{el}^{v} + s_l^{v} + 1 \qquad v \in \mathbf{V}, \emptyset \subset S \subset \mathbf{N}, k \in S, l \in \mathbf{N} \setminus S \quad (50)$$

4.1.3 Initial Problem for the branch-and-cut Algorithm

When solving the problem using branch-and-cut the following relaxed problem is used as the initial problem to which the violated capacity and connectivity constraints are added.

$$LSNDPR = \textbf{Min: } \sum_{m\in\mathbf{M}}\sum_{(i,j)\in\mathbf{A}} c_{ij}^m \sum_{v\in\mathbf{V}} x_{ij}^{mv} + \sum_{m\in\mathbf{M}}\sum_{j\in\mathbf{N}} c_j^m \sum_{v\in\mathbf{V}} f_j^{mv} + \sum_{(i,j)\in\mathbf{A}} c_{ij} \sum_{v\in\mathbf{V}} y_{ij}^v \qquad (51)$$

$s.t.$

(Transhipment) $$f_i^{mv} \geq \sum_{j:(j,i)\in\mathbf{A}} x_{ji}^{mv} - \sum_{j:(i,j)\in\mathbf{A}} x_{ij}^{mv} \qquad m\in\mathbf{M}, i\in\mathbf{N}, v\in\mathbf{V} \qquad (52)$$

(Flow) $$\sum_{v\in\mathbf{V}}\sum_{j:(i,j)\in\mathbf{A}} x_{ij}^{mv} - \sum_{v\in\mathbf{V}}\sum_{j:(j,i)\in\mathbf{A}} x_{ji}^{mv} = b_i^m \qquad i\in\mathbf{N}, m\in\mathbf{M} \qquad (53)$$

(Cap 0) $$\sum_{m\in\mathbf{M}} x_{ij}^{mv} \leq y_{ij}^v M \qquad (i,j)\in\mathbf{A}, v\in\mathbf{V} \qquad (54)$$

(Cap 2) $$r_{gh}^v + M(1-y_{gh}^v) - q^v(t_{gh}+t_h) \geq 0 \qquad (g,h)\in\mathbf{A}, v\in\mathbf{V} \qquad (55)$$

(route time) $$t_{max} - \sum_{(g,h)\in\mathbf{A}} r_{gh}^v \geq 0 \qquad v\in\mathbf{V} \qquad (56)$$

(Cap 1) $$q^v C^v + M(1-y_{ij}^v) \geq \sum_{m\in\mathbf{M}} x_{ij}^{mv} \qquad (i,j)\in\mathbf{A}, v\in\mathbf{V} \qquad (57)$$

(Cyclic) $$\sum_{j:(i,j)\in\mathbf{A}} y_{ij}^v - \sum_{j:(j,i)\in\mathbf{A}} y_{ji}^v = 0 \qquad i\in\mathbf{N}, v\in\mathbf{V} \qquad (58)$$

(connected) $$\sum_{j:(i,j)\in\mathbf{A}} y_{ij}^v - s_i^v \leq 1 \qquad i\in\mathbf{N}, v\in\mathbf{V} \qquad (59)$$

(Start vertex) $$\sum_{i\in\mathbf{N}} s_i^v = 1 \qquad v\in\mathbf{V} \qquad (60)$$

(Ships) $$y_{ij}^v - h^v \leq 0 \qquad (i,j)\in\mathbf{A}, v\in\mathbf{V} \qquad (61)$$

$$y_{ij}^v \in \{0,1\} \qquad (i,j)\in\mathbf{A}, v\in\mathbf{V} \qquad (62)$$

$$x_{ij}^{mv} \geq 0 \qquad (i,j)\in\mathbf{A}, m\in\mathbf{M}, v\in\mathbf{V} \qquad (63)$$

$$f_i^{mv} \geq 0 \qquad m\in\mathbf{M}, i\in\mathbf{N}, v\in\mathbf{V} \qquad (64)$$

$$s_i^v \in \{0,1\} \qquad i\in\mathbf{N}, v\in\mathbf{V} \qquad (65)$$

$$q^v \geq 0 \qquad v\in\mathbf{V} \qquad (66)$$

$$r_{gh}^v \geq 0 \qquad (g,h)\in\mathbf{A}, v\in\mathbf{V} \qquad (67)$$

Note that the problem has been relaxed by removing the constraints:

(Butterfly-Cut:)

$$\sum_{i,j\in S} y_{ij}^v \leq \sum_{h\in\mathbf{N}}\sum_{g\in S\setminus\{k\}} y_{hg}^v + s_k^v - \sum_{e\in\mathbf{N}} y_{el}^v + s_l^v + 1 \qquad v\in\mathbf{V}, \emptyset\subset S\subset\mathbf{N}, k\in S, l\in\mathbf{N}\setminus S$$

(Transhipment Cuts:)

$$u_{ji}^v + u_{ih}^v + 2(|T| - y^v(T)) = 2 \qquad ij, ih\in T\in B(E), S(ij,T)\neq S(ih,T)$$

$$f_i^{mv} \geq \sum_{j,h\in\mathbf{N}, v\in\mathbf{V}} f_{jih}^{mv} - M(1-s_i^v) \qquad m\in\mathbf{M}, i\in\mathbf{N}, v\in\mathbf{V}$$

$$f_{hij}^{mv} \geq x_{ji}^{mv} - x_{ih}^{mv} - M(2-y_{ji}^v - y_{ih}^v + u_{ji}^v + u_{ih}^v) \qquad m\in\mathbf{M}, j,i,h\in\mathbf{N}, v\in\mathbf{V}$$

$$f_{hij}^{mv} \geq x_{ji}^{mv} - x_{ih}^{mv} - M(4-u_{ji}^v - u_{ih}^v - y_{ji}^v - y_{ih}^v) \qquad m\in\mathbf{M}, j,i,h\in\mathbf{N}, v\in\mathbf{V}$$

Where $B(E)$ is the set of butterfly routes on E and $S(ij,T)$ is the simple tour on which arc (i,j) is located. The cuts are added during the branching process on the initially relaxed variables y_{ij}^v and s_i^v. Clearly the violated cuts should be added as early as possible. However the cuts are defined for integer variables and for non-integer values the cuts may not be violated.

4.1.4 Connectivity cut separation algorithm

As mentioned earlier a cut is added when the corresponding constraint is violated. The cut added is the first found violated cut. To check if the connectivity is violated we can use depth first search to check if all ports assigned to a vessel can be reached from any other port assigned to the same vessel. The depth first algorithm needs to be run for each vessel and therefore has complexity $O(|V|(|N|+|A|))$. The variables y_{ij}^v are used to define a connection between two ports. Since the integrality of variables y_{ij}^v and s_i^v is relaxed the cut of the form:

$$\sum_{i,j\in S} y_{ij}^v \leq \sum_{h\in\mathbf{N}g\in S\setminus\{k\}} y_{hg}^v + \psi s_k^v - \sum_{e\in\mathbf{N}} y_{el}^v + \psi s_l^v + 1 \qquad v\in\mathbf{V}, \emptyset\subset S\subset\mathbf{N}, k\in S, l\in\mathbf{N}\setminus S \quad (68)$$

may not be violated by the solution. Therefore additional conditions must be determined before adding the cut to the problem. The following conditions must hold when the connectivity cut (68) is violated for the integer relaxed problem.

- There are two vertices i and j visited by vessel v for which $y_{ij}^v = 0$.
- The flow on two disconnected components of a route when added must be greater than 1.
- There exists two disconnected vertices i and j visited by vessel v such that $s_i^v = 0$ and $s_j^v = 0$.

When all of the three condition listed above are fulfilled a cut for every vessel and every $l\in T$ and $k\in S$ is added to the relaxed integer program.

Given a graph G, the separation algorithm is run for each vessel v as follows:

Connectivity-Separation-Algorithm(G,v)

1: $i \leftarrow$ a port with $y_{ij}^v > 0$;
2: $numberconected \leftarrow DFS(i,v)$;
3: **if** $numberconnected$= ports on route for v **then**
4: **for all** ports i,j visited by v where $s_i^v = 0$ and $s_j^v = 0$ and $\sum_{h\in N} y_{ih}^v + \sum_{h\in N} y_{jh}^v > 1$ **do**
5: **for all** vessels v **do**
6: Add cut: $\sum_{g,h\in S} y_{gh}^v \leq \sum_{g\in\mathbf{N}h\in S\setminus\{i\}} y_{hg}^v + \psi s_i^v - \sum_{e\in\mathbf{N}} y_{ej}^v + \psi s_j^v + 1$
7: **end for**
8: **end for**
9: **end if**

Where $f(i) = \sum_{h\in N} y_{hi}^v$, for $i\in N$ is the flow on vessel v through the vertex i. The depth first search DFS in line 2 selects a vertex with an edge with an edge weight greater than zero and uses the edges with an edge weight greater than zero to do the depth first search from this vertex. The DFS returns the number of ports the DFS has visited. The set of ports on route v is the ports with an in-edge with edge weight greater than zero.

Test results using this algorithm are reported in Section 5.

4.1.5 Transhipment cut separation algorithm

For finding transhipment cuts only *butterfly* routes are checked. For a *butterfly* route an arc leaving the center point is selected as start arc and by using this the last arc on the route is found. Then it is investigated if the difference in flow on the last and the first arc plus the difference of flow of the two other arcs at the

centerpoint is greater than the the transhipment variable f_i^{mv}. If so the transhipment cuts are added. The transhipment cuts are of the form:

$$u_{ji}^v + u_{ih}^v + 2(|T| - y^v(T)) = 2$$
$$f_i^{mv} \geq \sum_{j,h \in \mathbf{N}, v \in \mathbf{V}} f_{jih}^{mv} - M(1 - s_i^v)$$
$$f_{hij}^{mv} \geq x_{ji}^{mv} - x_{ih}^{mv} - M(2 - y_{ji}^v - y_{ih}^v + u_{ji}^v + u_{ih}^v)$$
$$f_{hij}^{mv} \geq x_{ji}^{mv} - x_{ih}^{mv} - M(4 - u_{ji}^v - u_{ih}^v - y_{ji}^v - y_{ih}^v)$$

The set T must contain all arcs in the ***butterfly*** route. The cuts are only introduced if the solution is integer.

Given a graph G and a vessel v the capacity cut separation algorithm can be written as follows:

Transhipment-Separation-Algorithm(G, v)

1: $firstloop \leftarrow true$;
2: **for all** vertices i **do**
3: **if** $s_i^v \geq 1$ **then**
4: $centerpoint \leftarrow i$;
5: **end if**
6: **end for**
7: **if** less than 2 arcs leaving $centerpoint$ used by vessel v **then**
8: **return** "No cut found";
9: **end if**
10: $startarc \leftarrow$ an arc (i, j) used by vessel v leaving $centerpoint = i$;
11: $Arc \leftarrow startarc$;
12: $T \leftarrow T \cup Arc$;
13: **while** Arc $(ij)^v$ exists $\wedge(j \neq centerpoint \vee firstloop)$ **do**
14: **if** $j = centerpoint$ **then**
15: $firstloop \leftarrow false$;
16: $flend \leftarrow Arc$;
17: $Arc \leftarrow$ arc leaving j used by vessel v which is not $startarc$;
18: $slstart \leftarrow Arc$;
19: **else**
20: $Arc \leftarrow$ arc used by vessel v leaving j;
21: **end if**
22: $T \leftarrow T \cup Arc$;
23: **end while**
24: **if** (Arc $(ij)^v$ exists) $\wedge(j = centerpoint) \wedge (\neg firstloop)$ **then**
25: $endarc \leftarrow$ Arc ;
26: **for all** $m \in \mathbf{M}$ **do**
27: **if** $f_{centerpoint}^{mv} < \max\{x_{flend}^{vm} - x_{slstart}^{vm}, 0\} + \max\{x_{endarc}^{vm} - x_{startarc}^{vm}, 0\}$ **then**
28: **return** $(startarc, endarc, slstart, flend, T)$;
29: **end if**
30: **end for**
31: **end if**
32: **return** "No cut found";

In the algorithm the start and the centerpoint of a ***butterfly***-route is found at lines 2 to 6. It is checked at line 7 if the route of the vessel v is a ***butterfly***-route. If it is not a ***butterfly***-route then we will not need transhipment cuts. Then an arc exiting the start vertex is selected as the start arc for the route and the while loop walks through the route until all arcs have been visited and the last arc is then selected as the last arc

on the route. In lines 24 to 32 it is investigated if there is a transhipment at the start vertex going between the two visits of vessel v. If this is the case then the cuts are added.

Note that the while loop is a depth first search and therefore the algorithm has complexity $O(|N|)$ for line 1 to 6 and $O(|\mathbf{A}|)$ for line 7 to 23 and $O(|\mathbf{M}|)$ for line 24 to 32. This gives a running time of $O(|\mathbf{N}| + |\mathbf{A}| + |\mathbf{M}|)$ for the algorithm. Note that if the route of vessel v is not a butterfly route the algorithm will terminate after a asymptotic running time of $O(|\mathbf{N}|)$.

5 Computational Experiments

The branch-and-cut algorithm was implemented in C++ using CPLEX version 10.2 and Concert version 2.4 where the Connectivity and Transhipment cuts were added when violated. This is compared with CPLEX version 10.2 MIP-solver on the compact model. Tests have been run on a dual Intel CPU with 2.67 GHz.

5.1 Test cases

We have randomly generated cases which are constructed to reflect real-life network design problems. The graphs have 5 to 15 ports and includes up to 6 vessels. The forecast includes up to 12 demands.

In all of the test cases simple and *butterfly* routes are permitted, and the time of the evaluated period is 150 days. In the tests we use three different vessel types. Inspired by Agarwal and Ergun in [1] the demands in the tests are selected randomly from the complete set of origin destination pairs and the size of the demand is randomly selected between 1% to 80% of the capacity of the biggest vessel. The time of sailing between ports varies between 5 and 45 days. The cost of sailing between two ports is dependent on the vessel type and the time it takes to sail the distance. The cost of transshipment at a port is randomly selected between two predefined values. The ports are in all tests fully connected and the arcs are directed. The tests are terminated after 20 000 seconds which corresponds to 333.33 minutes. In order to ensure that the algorithms are tested without bias a new network is randomly generated for each test run.

5.2 Results

				Branch and Cut				Branch and Bound	
				Cuts					
instance	ports	vessels	demands	con	trans	Gap %	Time (min)	Gap %	Time (min)
a	5	3	9	6	140	0	0.05	0	0.22
b	5	3	9	13	196	0	0.13	0	0.33
c	5	3	9	16	252	0	0.08	0	0.23
d	5	3	9	21	364	0	0.15	0	0.40
e	5	3	9	11	252	0	0.16	0	0.87
a	7	3	9	29	560	0	15.28	0	46.91
b	7	3	9	29	476	0	0.93	0	3.36
c	7	3	9	30	504	0	0.21	0	5.99
d	7	3	9	17	280	0	0.53	0	3.73
e	7	3	9	67	1204	0	23.16	0	169.95
a	7	6	9	4	84	0	7.93	0	183.93
b	7	6	9	29	364	0	9.03	0.02	333.33
c	7	6	9	38	700	0	16.96	0.01	333.33
d	7	6	9	39	532	0	244.11	0.02	333.33
e	7	6	9	8	168	0	12.09	0.01	333.33
a	7	3	12	40	962	0	9.62	0	118.71
b	7	3	12	18	370	0	0.38	0	4.44
c	7	3	12	5	111	0	0.11	0	102.88
d	7	3	12	66	1628	0	24.70	0	166.88
e	7	3	12	26	740	0	0.41	0	5.48

Table 1: Test results with 5 to 7 ports and 9 to 12 demands given in the column of the same name. Comparing the described Branch and cut algorithm and the CPLEX branch and bound.

In Table 1 the test results for test cases with 5 and 7 ports are shown.

The first four columns in Table 1 indicate respectively the instance name, number of ports, vessels and demands in the test instance. The next four columns contain information related to solving the problem

using branch and cut. The first two columns reports the number of connectivity cuts added and the number of transhipment cuts added when solving the instance with the developed branch-and-cut algorithm. The next two columns, column 7 and 8, gives the lower and upper bound. The next column, column 9 reports the gap and column 10 gives time in minutes needed to solve the test instance.

The next columns reports Lower and upper bound, gap and the time in minutes needed to solve the test instance with the CPLEX MIP-solver.

The solution time is reported in minutes so that it can be easily compared to the times reported in [1]. In the instances with 5 ports an optimal solution is reached within a minute for all cases in both the branch-and-cut scheme and the CPLEX MIP-solver. However the branch-and-cut scheme does find the optimal solution faster than the CPLEX MIP-solver.

From the table it is clear that the branch-and-cut algorithm outperforms the CPLEX MIP-solver on these instances.

				Branch and Cut						Branch and Bound			
				Cuts					Time				Time
instance	port	v	dem	con	trans	LB	UB	Gap %	(min)	LB	UB	Gap %	(min)
a	10	3	7	32	462	60365	60365	0.00%	92.83	54818	61165	10.38%	333.33
b	10	3	7	44	484	50071	50071	0.00%	157.85	43487	50071	13.15%	333.33
a	10	6	9	212	3668	75296	100699	25.23%	333.33	74498	109882	32.20%	333.33
b	10	6	9	119	2352	68253	74155	7.96%	333.33	66213	85671	22.71%	333.33
a	15	3	7	8	22	50508	61305	17.61%	333.33	50505	*	*	333.33
b	15	3	7	101	1430	41457	49042	15.47%	333.33	40969	*	*	333.33
a	15	3	9	42	392	58874	78348	24.85%	333.33	58804	*	*	333.33
b	15	3	9	158	2604	63099	79249	20.38%	333.33	64130	*	*	333.33

Table 2: Test results with 10 to 15 ports. Comparing the described Branch and cut algorithm and the CPLEX branch and bound.

Tables 2 show the results for test cases with 10 to 15 ports. The columns are the same as in Table 1, except that two columns representing lowerbound (LB) and upperbound (UB) for the branch and cut and for the branch and bound algorithms are included. The lowerbound and upperbound are reported since some of the test instances in Table 2 are not solved to optimality. For each configuration of ports, ships and vessels two different instances are solved. The tests are stopped after 20 000 seconds and the best feasible solution found is reported. A gap between the best found feasible solution and the best known lowerbound is reported. In all of these test instances the best known lowerbound is the maximum of the lowerbounds found by CPLEX in the branch-and-cut algorithm and the standard CPLEX MIP-solver. For all the test cases with 15 ports it was not possible for the CPLEX MIP-solver to find a feasible solution within the given time limit. For the cases reported in Table 2 the branch-and-cut does better or at least as well as the CPLEX MIP-solver both with respect to time and solution quality. However, for the very last instance listed in Table 2 with 15 ports the lowerbound of the CPLEX MIP-solver is slightly better than that of the branch and cut algorithm even though the CPLEX-MIP solver was not able to find a solution. From the tests reported in Table 1 and 2 it is clear that increasing the number of ports increases the solution time of the problem therefore a time-space graph may not result in good solutions even though the connectivity cuts would be tightened.

6 Conclusion

In this paper we have formulated a new model of the liner shipping network design problem. We have included *butterfly* routes in the route structure, included cost of transhipment in the objective and we have included the time of the route in the calculations of the capacity to present a realistic model. On the presented model we have generalized clover-cuts and transhipment cuts. We have described and implemented a separation algorithm for the branch-and-cut method. The proposed algorithm is so far the only exact solution method allowing transhipment and calculating the transhipment cost. To the best of our knowledge, branch-and-cut methods have not been applied to the liner shipping network design problem before. The test results in general show improvements to the CPLEX branch and bound and cases with 15 ports and a reasonable demand set can be solved to acceptable precision using this method. The problem size and

solution time is comparable to the tests reported in Agarwal and Ergun [1]. The test results show that the branch-and-cut method is promising for the liner shipping network design problem.

The contributions of this paper is a general formulation of the problem including transhipment and transhipment costs. Moreover we have applied the branch-and-cut method to the formulation and thereby developed an exact solution method which can solve instances of the same size as some of the recently developed heuristic methods. The developed branch-and-cut method can be used for planning the routes of a smaller shipping company such as a feeder company or for planning a region of the network of a bigger liner shipping company. Since the algorithm finds optimal solutions it can also be used to benchmark heuristic algorithms.

Acknowledgments

The authors wish to thank Christian Edinger Munk Plum, Berit Løfstedt, Shahin Gelareh and Jose Fernando Alvarez for valuable comments.

References

[1] R. Agarwal and O. Ergun. Ship scheduling and network design for cargo routing in liner shipping. *Transportation Science*, 42:175–196, 2008.

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

[3] N. Ascheuer, M. Jünger, and G. Reinelt. A branch & cut algorithm for the asymmetric traveling salesman problem with precedence constraints. *Computational Optimization and Applications*, 17: 61–84, 2000.

[4] J. F. Bard, G. Kontoravdis, and G. Yu. A branch-and-cut procedure for the vehicle routing problem with time windows. *Transportation Science*, 36:250–269, 2002.

[5] M. Christiansen and B. Nygreen. A method for solving ship routing problems with inventory constraints. *Annals of Operations Research*, 81:357–378, 1998.

[6] M. Christiansen, K. Fagerholt, and D. Ronan. Ship routing and scheduling: Status and perspectives. *Transportation Science*, 38:1–18, 2004.

[7] M. Christiansen, K. Fagerholt, B. Nygreen, and D. Ronen. Maritime transportation. In C. Barnhart and G. Laporte, editors, *Handbook in OR & MS*, volume 14, chapter 4. Elsevier Science B.V., 2007.

[8] K. Fagerholt. Optimal fleet design in a ship routing problem. *International Transactions in Operational research*, 6:453–464, 1999.

[9] M. Fischetti, J. Salazar, and P. Toth. A branch-and-cut algorithm for the symmetric generalized traveling salesman problem. *Operations Research*, 45:378–394, 1997.

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

[11] M. Gendreau, G. Laporte, and F. Semet. A branch-and-cut algorithm for the undirected selective traveling salesman problem. *Networks*, 32:263–273, 1998.

[12] B. Kallehauge, N. Boland, and O. B. G. Madsen. Path inequalities for the vehicle routing problem with time windows. *Networks*, 49:273–293, 2007.

[13] C. E. Miller, A. W. Tucker, and R. A. Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the Association for Computing Machinery*, 7:326–329, 1960.

[14] T. Notteboom. The time factor in liner shipping services. *Maritime Economics & Logistics*, 8:19–39, 2006.

[15] T. Notteboom and J. Rodrigue. Containerisation, box logistics and global supply chains: The integration of ports and liner shipping networks. *Maritime Economics & Logistics*, 10:152–174, 2008.

[16] K. Rana and R. Vickson. Routing container ships using lagrangean relaxaton and decomposition. *Transportation Science*, 25:201–214, 1991.

[17] D. Ronen. Cargo ships routing and scheduling: Survey of models and problems. *European Journal of Operational Research*, 12:119–126, 1983.

[18] D. Ronen. Ship scheduling: The last decade. *European Journal of Operational Research*, 71:325–333, 1993.

[19] K. Shintani, A. Imai, E. Nishimura, and S. Papadimitriou. The container shipping network design problem with empty container repositioning. *Transportation Research Part E*, 43:39–59, 2007.

[20] H. P. Williams. *Model Building in Mathematical Programming*. Wiley, New York, 1999.

The network design problem in liner shipping is of increasing importance in a strongly competitive market where potential cost reductions can influence market share and profits significantly. In this paper the network design and fleet assignment problems are combined into a mixed integer linear programming model minimizing the overall cost. To better reflect the real-life situation we take into account the cost of transhipment, a heterogeneous fleet, route dependant capacities, and butterfly routes. To the best of our knowledge it is the first time an exact solution method to the problem considers transhipment cost. The problem is solved with branch-and-cut using clover and transhipment inequalities. Computational results are reported for instances with up to 15 ports.

DTU Management Engineering
Department of Management Engineering
Technical University of Denmark

Produktionstorvet
Building 424
DK-2800 Kongens Lyngby
Denmark
Tel. +45 45 25 48 00
Fax +45 45 93 34 35

www.man.dtu.dk