https://pubsonline.informs.org/journal/opre

OPERATIONS RESEARCH

Vol. 00, No. 0, Xxxxx 0000, pp. 000–000

Submitted to *Operations Research*

ISSN 0030-364X, EISSN 1526-5463

A Simultaneous Column-and-Row Generation Solution Method for Liner Shipping Network Design

Jun Xia,[a] Zhou Xu,[b,*] Roberto Baldacci[c]

[a]Sino-US Global Logistics Institute, Antai College of Economics & Management, Shanghai Jiao Tong University, Shanghai, China, lgtxiaj@sjtu.edu.cn; [b]Department of Logistics and Maritime Studies, Faculty of Business, The Hong Kong Polytechnic University, Hong Kong, China, lgtzx@polyu.edu.hk; [c]College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar, rbaldacci@hbku.edu.qa

∗Corresponding author

Abstract. The Liner Shipping Network Design (LSND) problem involves creating regular ship rotations to transport containerized cargo between seaports. The objective is to maximize carrier profit by balancing revenue from satisfied demand against operating and transshipment costs. Finding an optimal solution is challenging due to complex rotation structures and joint decisions on fleet deployment, cargo routing, and rotation design. This work introduces a set-partitioning-like formulation for LSND with transshipment costs, featuring an exponential number of variables and constraints. The formulation captures key service components such as ship type, sailing speed, and frequency. Addressing transshipment costs requires numerous rotation-dependent variables and constraints, making even linear programming relaxation difficult to solve. To tackle this, we propose a simultaneous column-and-row generation (SCRG) solution method with novel speed-up techniques. Integrating SCRG into a branch-and-price algorithm, we develop an exact method for LSND and test it on two variants with different rotation configurations. Extensive computational experiments demonstrate the method's effectiveness and efficiency. In addition to advancing solution methods for LSND, this work enhances the SCRG-based method and expands its practical applications.

Funding: This research was supported by the National Natural Science Foundation of China [Grants 72171147, 72031006] and the Research Grants Council of Hong Kong SAR, China [Grant 15221619].

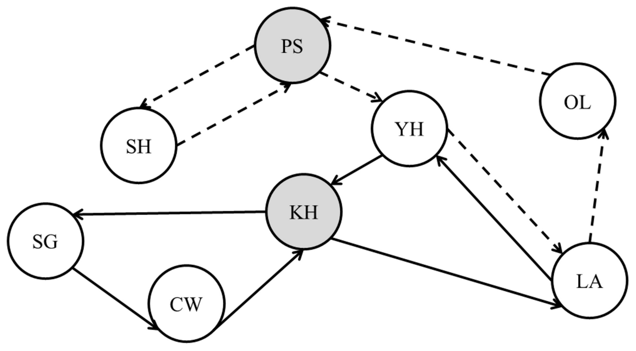
Key words: Liner shipping network design; Transshipment cost; Simultaneous column-and-row generation; Exact method

1. Introduction

In liner shipping, containerships rotate among seaports to transport cargos with a regular service frequency (Cosco-Line 2020, Maersk-Line 2020). These sequential port calls constitute a service network on which carriers operate their lines for cargo transportation. Due to the capricious nature of the shipping industry, carriers need to adjust their service networks periodically to maintain their competitiveness in response to an ever-changing market. Since 2020, the COVID-19 pandemic has significantly influenced global supply chains, with the appearance of many potential new market demands that need to be fulfilled by reconfiguring service networks. In planning the service network, allowing the transfer of containers between ships at an intermediate port improves transportation efficiency and enlarges the market coverage. Nowadays, through transshipment operations, liner shipping services connect more than 80% country pairs (UNCTAD 2017). According to the UNCTAD's (United Nations Conference on Trade and Development) Liner Shipping Connectivity Index (LSCI) (UNCTAD 2021), which is one of the most popular maritime connectivity indices (Wilmsmeier and Hoffmann 2008, Fugazza and Hoffmann 2017), the transshipment operations are increasing during recent years. Nevertheless, this entails considerable transshipment costs that must be considered in the service network planning.

A liner shipping service is defined as a *rotation* that involves a cyclic sequence of port calls maintained by a fleet of capacitated ships for a designated service frequency (e.g., weekly). In a *simple* rotation, each port is visited exactly once during a round trip. If some ports are visited twice or more, the rotation is called *non-simple*, and a port visited more than once is regarded as a *butterfly port*. At a butterfly port, containers can be transferred between two ships operating for the same rotation but with different sailing directions (i.e., westbound or eastbound). We define this transshipment operation as *intra-transshipment*. Containers can also be transferred between ships from different (simple and non-simple) rotations, and we call this *inter-transshipment*.

For example, in Figure 1, both rotations (A) and (B) belong to the non-simple types, with KH and PS as their butterfly ports (represented by gray nodes). A shipment path for the containers from CW to SG starts from CW on rotation (A). When arriving at the butterfly port KH, containers are transferred to another ship heading to SG from rotation (A) via *intra-transshipment* and carried to SG in the next port of call. The shipment path for the containers from PS to SG starts from PS on rotation (B). At YH, which is an intersection port of the two rotations, containers are transferred to a ship from rotation (A) via *inter-transshipment* and next transported to SG by rotation (A).



Rotation (A): SG – CW – KH – LA – YH – KH – SG
Rotation (B): SH – PS – YH – LA – OL – PS – SH

CW	Chiwan
KH	Kaohsiung
LA	Los Angeles
OL	Oakland
PS	Pusan
SG	Singapore
SH	Shanghai
YH	Yokohama

Figure 1 Illustration of intra-transshipment and inter-transshipment. Butterfly ports are represented by gray nodes, intra-transshipment and inter-transshipment occur at KH and YH, respectively.

The liner shipping network design (LSND) problem aims to create a set of rotations to satisfy the container shipment demand. The goal of the LSND is to maximize the carrier's total profit, that is, the total freight revenue from satisfied demands minus the total operating cost, including the costs associated with intra-transshipment and inter-transshipment operations.

The LSND is strongly $\mathcal{NP}$-hard (see Brouer et al. 2014a). When transshipment costs are not considered, this problem can be seen as a variant of the service network design with asset management (SNDAM) problem, where multiple types of assets, which can represent ships, are deployed on cycles to maintain designated service frequencies. For the SNDAM, set-partitioning-like models (see, e.g., Andersen et al. 2009a,b, 2011, Crainic et al. 2014) have been developed based on a multiple commodity flow network, where each arc has a capacity aggregated among the passing cycles to limit the flow of cargos. Traditional column generation (CG) techniques (see, e.g., Lübbecke and Desrosiers 2005) are applicable to solving their linear programming (LP) relaxations, which provide valid dual bounds that can be used to develop exact solution methods. In the SNDAM, transshipping of cargos among cycles is allowed; however, the volumes of cargos transshipped cannot be explicitly captured in the existing models, meaning that the transshipment costs cannot be considered. For the LSND, cargo transshipment operations and their associated costs have been considered in some existing studies (Álvarez 2009, Plum et al. 2014b). Flexible service frequencies are also considered to maximize profits (Giovannini and Psaraftis 2019, Brouer et al. 2014a). In contrast to SNDAM, model formulations of the LSND in these studies are more complicated, resulting in the current works mainly focusing on heuristics, for which optimality gaps of the obtained solutions are rarely reported (see, e.g., Mulder and Dekker 2014, Brouer et al. 2014a,b, Koza et al. 2020). There are only a few existing exact methods developed for the LSND, which are primarily limited to a priori constraints on the number of feasible rotations, the maximum number of operated rotations, or allowing at most one butterfly port (Ameln et al. 2019, Wang and Meng 2012, Reinhardt and Pisinger 2012, Plum et al. 2014b, Balakrishnan and Karsten 2017, Hellsten et al. 2022). To the best of our knowledge, only the exact method of Thun et al. (2017) avoids limitations on the number and structure of the rotations, but it cannot take into account all the possible rotations if more than one butterfly port is allowed.

The results and contributions of this paper can be summarized as follows.

(i) For the first time in the literature, this study provides a new exact method for the LSND with transshipment costs without imposing any a priori restrictions on the number of feasible rotations or the maximum number of operated rotations but considering decisions on general rotation configurations of various service components.

(ii) To model the problem, we define a planning network with specific transshipment nodes and arcs used to explicitly capture transshipped cargos and transshipment costs. Based on that, we develop a mixed-integer linear programming (MILP) formulation, a new set-partitioning-like model for the problem with exponentially many rotation-dependent variables (or columns) and constraints.

(iii) To tackle the challenge of solving the LP-relaxation of the newly developed MILP formulation, a major contribution of this paper is the development of a novel solution method based on *simultaneous column-and-row generation* (SCRG). It is composed of two phases, where the first phase builds a dual solution for a restricted master problem, and the second phase solves a pricing problem that uses the dual solution to check the optimality conditions and to generate new rotation-dependent variables and constraints simultaneously. Concerning existing SCRG solution frameworks (see, e.g., Feillet et al. 2010, Muter et al. 2013), we introduce two innovative techniques, named "LP-based approach" and "post-pricing phase", which leverage the dual information to avoid generating unnecessary columns and thus speed up the convergence of the solution method.

(iv) Our new SCRG-based solution method is embedded within a standard branch-and-price (BP) framework to compute optimal or near-optimal solutions (with their optimality gaps) for the LSND with transshipment costs. We apply it to the following two main variants of the problem: (a) *Standard LSND*, all ships sail at a design speed and all rotations strictly follow weekly frequency, and (b) *General LSND*, each ship of a given type can sail at any speed from a given speed set. Each rotation can have a service frequency from a given frequency set.

Computational experiments are conducted on these two variants to validate their effectiveness and efficiency, and the results show that:

- the "LP-based approach" and the "post-pricing phase" are the two main factors that significantly improve the performance of our solution method and can be adopted as general techniques to improve the convergence of other methods that follow existing SCRG solution frameworks;
- this exact method solves to optimality, for the first time, some difficult LSND instances with transshipment costs (of up to 12 ports), and significantly improves the solutions provided by existing exact and heuristic methods for some benchmark instances (of up to 20 ports).

Our study reveals and demonstrates for the first time in the literature that the LSND can be solved using a SCRG-based solution method, extending the practical applications of the SCRG-based solution method in general. The newly developed optimization techniques in the SCRG-based solution method for the LSND are also potentially useful to solve large-scale linear (integer) programs with column-dependent rows for other complex optimization problems, such as those studied in Feillet et al. (2010) and Muter et al. (2013).

The remainder of this paper is organized as follows. A literature review is presented in §2, followed by the description and formulation of the LSND in §3. §4 is dedicated to our new SCRG-based solution method, whereas §5 describes the exact method. Experimental results are presented in §6, and conclusions and future research directions in §7.

2. Literature review

There is a growing body of studies on the LSND (see Christiansen et al. 2013 and Christiansen et al. 2020 for comprehensive reviews), which have mainly investigated heuristic methods. Álvarez (2009) studied the version of the LSND with only inter-transshipment costs and developed a matheuristic approach, which was applied in a case study with ports of call distributed across the globe. Brouer et al. (2014a) considered a more general LSND problem associated with new decisions on sailing speed and service frequency, as well as additional intra-transshipment costs. They developed an iterated tabu search algorithm to find heuristic solutions and tested it on benchmark instances. Other heuristic algorithms were developed by Mulder and Dekker (2014), Brouer et al. (2014b), and Krogsgaard et al. (2018), all of which applied neighborhood operators to refine the obtained solutions iteratively. Recently, Koza et al. (2020) developed a CG-based matheuristic for a more general version of the LSND, which accounts for speed optimization and additional service-level requirements. In the studies mentioned above, due to the lack of efficient methods for computing strong relaxation bounds, optimality gaps between the solution values of optimal and heuristic solutions were not reported.

For the LSND without transshipment costs (but allowing cargo transshipment), Agarwal and Ergun (2008) proposed a set-partitioning-like model, which consists of an exponential number of rotation-dependent binary variables. They developed two heuristic methods based on CG and Benders decomposition, respectively, which can produce feasible solutions and report their optimality gaps. The experimental results of Agarwal and Ergun (2008) showed that all of their test instances, which included 6 to 20 ports, were solved heuristically, with optimality gaps ranging on average from 2.3% to 12.7%. In our study, we propose and solve a new set-partitioning-like model for the LSND with transshipment costs. This new model can also incorporate additional decisions on various service components, such as the sailing speed and service frequency, which have not been considered in Agarwal and Ergun (2008). It is also complicated, containing an exponential number of rotation-dependent binary variables and rotation-dependent constraints.

Only a few exact methods for the LSND with transshipment costs are known in the literature. Most of them are limited to a single rotation or to a priori restrictions on the number of feasible rotations or the maximum number of operated rotations in a solution, which, however, do not confine our newly proposed solution method. For the LSND problem of only a single rotation to be determined, Plum et al. (2014a) derived a mixed integer programming (MIP) model and developed a branch-price-and-cut algorithm to solve it. Within a time limit of 1 hour, they solved optimality all their test instances of 10 ports and some of their test instances of 15-25 ports. For test instances of 20 ports or more, their obtained solutions had an average optimality gap ranging from 4% to 153%. With a more general setting considered, Wang et al. (2019) developed a branch-and-cut algorithm to optimize the design of a single rotation. Their experimental results on randomly generated test instances showed that test instances of up to 12 ports were solved to optimality in 3 hours. For their largest instances of 20 ports, the obtained solutions had an average optimality gap of around 20%, with the maximum gap reaching 40%.

Given a limited set of candidate routes, Wang and Meng (2012) and Meng and Wang (2011) derived MIP models for the LSND problem with transshipment costs and used a general-purpose MIP solver to solve the models to optimality. However, their models and methods were confined to only a few rotations that had to be explicitly stated in advance.

Reinhardt and Pisinger (2012), Plum et al. (2014b), and Ameln et al. (2019) developed alternative MIP models by using arc-based decision variables, which were, however, confined to a priori restriction on the maximum number of operated rotations in a solution. Under such a restriction, Reinhardt and Pisinger (2012) developed a branch-and-cut algorithm to solve an arc-based model of the LSND problem. Their experiments solved test instances of no more than ten ports and three ships to optimality within around 5.5 hours. For their largest test instances of 15 ports and three ships, the solutions obtained had optimality gaps around 20% on average and 24% at maximum. Plum et al. (2014b) applied a general-purpose MIP solver to directly solve an arc-based model of the LSND problem. They reported experimental results based on two datasets, Baltic and WAF, of the benchmark suite LINER-LIB-2012, which was introduced by Brouer et al. (2014a) to challenge both exact and heuristic methods, as well as to encourage their developments. Experimental results from Plum et al. (2014b) showed that the optimality gaps of their solutions, obtained within 1-3 hours, were often greater than 30% for instances of 12-20 ports. Given a predetermined maximum number of rotations, Ameln et al. (2019) further developed an arc-based model on a layered network and directly applied a general-purpose MIP solver to solve it, which, however, cannot incorporate all possible rotations when the number of butterfly ports in a single rotation can be more than one. They reported experimental results on instances randomly generated from the Baltic and WorldLarge datasets of LINER-LIB-2012. The results showed that for instances of up to 8 ports, most of them could be solved to optimality within 10 hours. For instances of 10 ports, the optimality gaps of the solutions obtained were around 14% on average and 19% at maximum. Among the three existing works mentioned above, both Reinhardt and Pisinger (2012) and Plum et al. (2014b) can incorporate decisions on service frequency, but none considered decisions on sailing speed.

Moreover, the LP-relaxations of arc-based models, such as those mentioned above, generally provide only weak bounds concerning the dual bounds provided by set-partitioning-like models. Indeed, in set-partitioning-like models, the rotation constraints do not need to be explicitly expressed in the formulation; rather, they are implicit from the definition of the set of rotations or variables. Consequently, the formulations provide better lower bounds than those obtained with arc-based formulations because they can be obtained by applying Dantzig-Wolfe decomposition to arc-based formulations (Vanderbeck and Wolsey 2010). This motivates our work on deriving and solving a set-partitioning-like model that is not confined to any a priori restrictions on the number of feasible or the maximum number of operated rotations.

Recently, Thun et al. (2017) developed an exact method for a cost minimization variant of the LSND problem that aims to minimize the total cost with all the cargo demands strictly satisfied, where there are no limitations on the number of rotations and the structure of the rotations. However, similar to Ameln et al. (2019), their solution method also relies on a layered network, which cannot incorporate all possible rotations when the number of butterfly ports in a single rotation can be more than one. It does not consider decisions on the sailing speed or service frequency. The set-partitioning-like model derived in Thun et al. (2017) contains many decision variables, each associated with a combination of a rotation and its cargo flows. Thus, although a CG-based method can be applied to solve the LP relaxation of the model, the corresponding pricing problem needs to determine not only a rotation but also its cargo flows. Thus, such a pricing problem can only be solved by applying an MIP solver on an arc-based model, which can be very time-consuming. Thun et al. (2017) showed that their exact method could solve some randomly generated instances of up to only seven ports within a time limit of 1 hour. Still, for some instances of 7 ports, their method cannot even produce feasible solutions to satisfy all the cargo demands. Our newly derived set-partitioning-like model for the LSND for profit maximization, which we will present later, consists of significantly fewer decision variables, which leads to a more tractable pricing problem and tighter relaxation bound. Our new model considers all possible rotations for any given number of butterfly ports allowed, and it also considers decisions on general rotation configurations of more service components, including the sailing speed and service frequency.

In this paper, we develop a new exact method based on SCRG to solve the LSND. In addition to the LSND, there are several other complex optimization problems known in the literature that can be naturally formulated as linear (integer) programming models with *column-dependent rows* and can be solved by methods based on some SCRG solution frameworks (see, e.g., Feillet et al. 2010, Muter 2011). Unlike traditional CG, an SCRG-based solution method adds columns (variables) and rows (constraints) simultaneously by solving a *row-generating pricing problem* (Muter et al. 2013). Note that many well-known but fundamentally different methods, such as *column-and-row generation*, *column-and-cut generation*, *branch-price-and-cut*, are named with similar terms in the literature (see, e.g., Katayama et al. 2009, Desrosiers and Lübbecke 2011, Desaulniers et al. 2016). For these methods, rows newly generated are valid inequalities, which are used to strengthen the LP-relaxation of the mathematical formulation, and the generation of the columns and the rows are performed separately. They have recently been applied to the LSND in a matheuristic developed by Koza et al. (2020). Differently from these methods, column-dependent rows generated by SCRG are necessary structural constraints of the mathematical formulation, such as the flow balanced and the ship capacity constraints in the LSND, as we will see later in §3.3. These constraints have to be added simultaneously together with their interdependent variables. Thus, a major challenge in using SCRG comes from generating new columns requiring the dual information on their interdependent rows, which are, however, not yet present in the restricted master problem. In this sense, implementing an SCRG-based method is more complicated than a traditional CG-based method.

Early applications of SCRG were proposed by Zak (2002) and Avella et al. (2006), in which the LP-relaxations of problems studied are solved suboptimally. Problems with exponentially many column-dependent rows, which cannot afford to be enumerated explicitly, have been addressed by Feillet et al. (2010) and Muter et al. (2013). Feillet et al. (2010) described an SCRG solution framework based on the constructions of dual solutions for LP, and they applied it in developing two solution methods for solving the LP-relaxations of a split-delivery vehicle routing problem and a bus rapid transit route design problem, respectively. Muter et al. (2013) proposed another SCRG solution framework based on the solution of a two-stage row-generating pricing problem. By following Muter et al. (2013), similar techniques have been applied to the cutting stock problems (Muter and Sezer 2018, Wang et al. 2020), the integrated airline recovery problems (Maher 2016, Huang et al. 2023), and the time-constrained routing problem (Muter et al. 2018), which are all associated with row-generating pricing problems for generating the column-dependent rows.

As mentioned in Spliet (2024), the effectiveness of the SCRG-based solution methods hinges on their careful design. As we will show later in this paper, the two SCRG-based solution methods in Feillet et al. (2010) are problem-specific and either cannot be used or are inefficient when applied to the LSND. When the technique developed in Muter et al. (2013) is applied to the LSND, the resulting solution method is outperformed by our method. Moreover, existing SCRG solution frameworks overlook the problem of generating unnecessary columns, which will be addressed in our method.

3. Problem description and mathematical formulation

In this section, we present the problem description of the LSND with transshipment costs, imposing no prior restriction on the number of feasible or the maximum number of operated rotations. Decisions on general rotation configurations represent various service components, including but not limited to the ship type, sailing speed, and service frequency. Based on this, a set-partitioning-like MIP model is derived, which captures both intra-transshipment and inter-transshipment costs, and it can be specified to incorporate different variants of the LSND.

3.1. Description of the LSND

We are given a set of ports $H = \{1, ..., |H|\}$, and a set of container shipment demands $K = \{1, ..., |K|\}$, where demand $k \in K$ has q_k containers that are available to be shipped from the origin port $o(k)$ to the destination port $d(k)$ each week, incurring holding and transshipment costs during the transportation. For each demand $k \in K$, shipping a container has a unit revenue p'_k and a unit rejection or penalty cost p''_k. A fleet of ships of different types transports containers. Let $S = \{1, ..., |S|\}$ be the set of ship types, where the ship of type $s \in S$ is associated with a maximum capacity m_s (in TEUs), a design speed u_s (in knots), and a fleet size n_s (i.e., the number of available ships of type s).

A liner shipping service is defined as a *rotation* made up of its configuration and route. The *configuration* of a rotation is a specification of some service components, including but not limited to the ship type, sailing speed, and service frequency, which determine the service's ship deployment, capacity, and cost. Let $C = \{1, 2, \ldots, |C|\}$ indicate the index set of all available configurations. Actual forms of the available configurations depend on the settings of specific applications. For example, in the application studied by Plum et al. (2014b) and Reinhardt and Pisinger (2012), all ships are assumed to sail at their design speeds, and all services are assumed to be weekly. Under this setting, the configuration of a rotation needs to include only the ship types, implying $C = S$.

The *route* of a rotation is a cyclic sequence of port calls. Following some practical rules proposed in the existing studies (see, e.g., Agarwal and Ergun 2008, Brouer et al. 2014a,b), a *feasible* route concerning a given configuration $\sigma \in C$ is required to (i) call at the same port $h \in H$ at most $\psi_h^{(1)}$ times; (ii) pass the same voyage arc $a \in \{(h, h') : h, h' \in H, h \neq h'\}$ at most $\psi_a^{(2)}$ times; (iii) involve at most $\psi^{(3)}$ port calls; (iv) have at most $\psi^{(4)}$ ports that are visited more than once; (v) have a round-trip time no greater than $\psi^{(5)}$ weeks under configuration σ. Hereinafter, by the term route, we refer to a feasible route. We use $\Gamma(\sigma)$ to indicate the index set of all feasible routes concerning configuration $\sigma \in C$. Note that whether a rotation is of a simple or non-simply type depends on its route structure, such as the number of butterfly ports (i.e., $\psi^{(4)}$).

Let R denote the set of all rotations, where each rotation $r \in R$ is represented by a pair (σ_r, γ_r) with $\sigma_r \in C$ and $\gamma_r \in \Gamma(\sigma_r)$ indicating the configuration and the route of rotation r, respectively. Following the existing studies (see, e.g., Agarwal and Ergun 2008, Brouer et al. 2014a,b), we assume that each rotation uses only ships of the same type. Accordingly, for each $r \in R$, let $s(r)$ and $n(r)$ denote the type and the number of ships deployed for rotation r, respectively, and let $m(r)$ denote the weekly capacity provided by the deployed ships of rotation r, and let $w(r)$ denote the weekly operating cost of all the deployed ships of rotation r. The LSND aims to choose a subset of rotations from R to transport the cargo demands from K selectively, without exceeding the ships' capacities and fleet sizes, to maximize the total profit obtained for each week.

3.2. Modeling rotations

We model rotations on a digraph $G = (N, A)$ where the node set N is partitioned as $N = V \cup T$ and A is the arc set. Node set V represents *voyage nodes* whereas set T represents *transshipment nodes*. A voyage node represents a specific port of call. Since a non-simple rotation may involve a butterfly port visited more than once, and two or more rotations may visit the same port, we copy multiple voyage nodes for each port, such that separated voyage nodes represent repeated port calls. Since each feasible route of a rotation call at the same port $h \in H$ at most $\psi_h^{(1)}$ times, it is sufficient to have $\psi_h^{(1)}$ copies of voyage nodes for each port $h \in H$. For the sake of notation, the original set of $|H|$ ports are represented by the first $|H|$ nodes in V. A transshipment node represents the yard area of the port; therefore, set T contains a node for each port. For each node $i \in N$, $h(i) \in H$ denotes the corresponding port.

The set of arcs A is partitioned as $A = A_V \cup A_T$, where $A_V = \{(i,j) : i,j \in V, h(i) \neq h(j)\}$ represents *voyage arcs* whereas A_T represents *transshipment arcs*. Set A_T is further partitioned as $A_T = A_T^{\downarrow} \cup A_T^{\uparrow}$ where $A_T^{\downarrow} = \{(i,j) : i \in V, j \in T, h(i) = h(j)\}$ represents the set of *unloading arcs* and $A_T^{\uparrow} = \{(i,j) : i \in T, j \in V, h(i) = h(j)\}$ the set of *loading arcs*. A transshipment arc, connecting a voyage node to a transshipment node, represents the unloading operation from aboard the ship to the port yard. The contrary connection, from a transshipment node to a voyage node, represents the loading operation. For each voyage node $i \in V$, we indicate with $A_V^-(i) = \{j \in V : (j,i) \in A_V\}$ and $A_V^+(i) = \{j \in V : (i,j) \in A_V\}$ the sets of voyage arcs entering and leaving node i, respectively. For each node $i \in N$, we indicate with $A_T^{\downarrow}(i) = \{j \in N : (j,i) \in A_T\}$ and $A_T^{\uparrow}(i) = \{j \in N : (i,j) \in A_T\}$ the sets of transshipment arcs entering and leaving node i, respectively. Henceforth, if arc a connects the nodes i and j, then (i,j) and a will be used interchangeably to denote the same arc.

Every voyage arc $a \in A_V$ is associated with a length ℓ_a in nautical miles. Every arc $a \in A$ is associated with a cost w_a^k for moving a container of demand $k \in K$ on arc a. In practice, w_a^k for $a \in A_V$ represents the holding cost of a container in the voyage concerning arc a, and w_a^k for $a \in A_T$ represents the transshipment cost for moving a container at a terminal yard for arc a, which can be any loading or unloading cost. It is worth noting that w_a^k can also incorporate costs other than

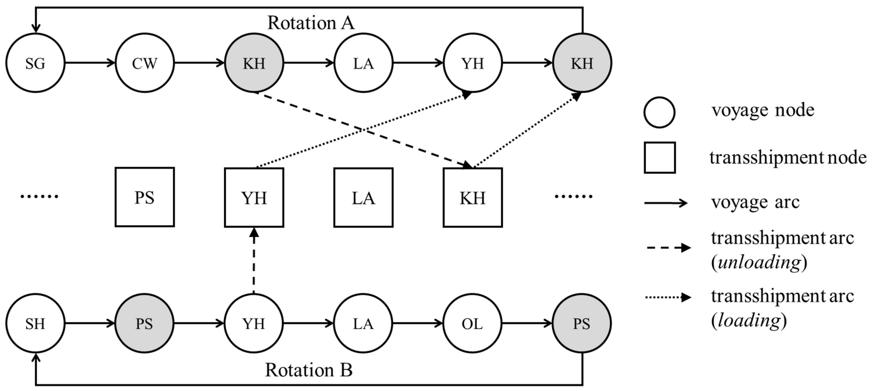


Figure 2 Routes of the two rotations in Figure 1 represented as elementary cycles in the underlying digraph, where port KH and port PS have two copies of voyage nodes.

the loading or unloading costs. For example, suppose the transshipment node of w_a^k for $a \in A_T$ is the origin or the destination port of demand k. In that case, the container movement of demand k can be recognized as an import/export operation, and w_a^k can incorporate a local import/export tariff. Otherwise, the container movement is recognized as a transshipment operation, and w_a^k can be used to incorporate a local transshipment tariff.

Consider each rotation $r = (\sigma_r, \gamma_r) \in R$ with configuration $\sigma_r \in C$ and route $\gamma_r \in \Gamma(\sigma_r)$. Since the digraph $G = (N, A)$ has $\psi_h^{(1)}$ copies of voyage nodes for each port $h \in H$, the route γ_r can be represented by an elementary circuit of G that visits only voyage nodes with no repeats. Let $V(\gamma_r)$ and $A_V(\gamma_r)$ denote the set of voyage nodes and voyage arcs in route γ_r, respectively. Let $A_T(\gamma_r)$ denote the set of transshipment arcs of graph G that leave or enter the voyage nodes in $V(\gamma_r)$.

Figure 2 presents the representations of routes of the two rotations illustrated in Figure 1. The figure shows that each route corresponds to an elementary circuit of distinct voyage nodes in the underlying digraph. Furthermore, unloading and loading transshipment arcs represent intra- and inter-transshipment operations with no need to be distinguished, and the costs associated with these arcs accordingly model the corresponding transshipment costs.

For each rotation $r = (\sigma_r, \gamma_r) \in R$, the type of ships $s(r) \in S$ deployed in r is determined only by the configuration σ_r. Thus, we can represent $s(r)$ by $s(\sigma_r)$. The weekly capacity $m(r)$ provided by the deployed ships in r usually depends only on the type of the deployed ships and the service frequency. Thus, it is also determined only by σ_r. We can also represent $m(r)$ by $m(\sigma_r)$. In contrast, the number of ships $n(r)$ deployed and the round-trip operating cost $w(r)$ cannot be determined only by the configuration σ_r, as they both also depend on the arcs and nodes of the route γ_r of rotation r. Thus, we represent them by $n(\sigma_r, \gamma_r)$ and $w(\sigma_r, \gamma_r)$, respectively.

Accordingly, throughout the remainder of the paper, we replace $s(r)$, $m(r)$, $n(r)$, and $w(r)$, which are introduced to describe problem LSND in §3.1, by $s(\sigma_r)$, $m(\sigma_r)$, $n(\sigma_r, \gamma_r)$, and $w(\sigma_r, \gamma_r)$, for each rotation $r = (\sigma_r, \gamma_r) \in R$. Moreover, for different specific applications, the settings of the LSND are different. As a result, actual forms of the configuration σ_r are different, and so are the actual forms of $s(\sigma_r)$, $m(\sigma_r)$, $n(\sigma_r, \gamma_r)$, and $w(\sigma_r, \gamma_r)$. See the following two examples, two main variants of the LSND, incorporating those problem settings considered in the literature.

EXAMPLE 1 (STANDARD LSND). As mentioned earlier, in the application studied by Plum et al. (2014b) and Ameln et al. (2019), all ships of type $s \in S$ are assumed to sail at their design speeds u_s and all services are assumed to be weekly. We refer to the LSND under this setting as the *Standard LSND*. Accordingly, the configuration σ_r of each rotation $r = (\sigma_r, \gamma_r) \in R$ includes only the ship type $s(\sigma_r) \in S$. Thus, we can simply assume that $C = S$ and $s(\sigma) = \sigma$ for $\sigma \in C$ so that $s(\sigma_r) = \sigma_r$. Since each ship of type $s \in S$ has a maximum capacity m_s and all services are weekly, we obtain that the weekly capacity $m(\sigma_r) = m_{s(\sigma_r)}$. Suppose each port call requires τ days for handling containers at the terminal, where τ is a given input parameter. Since each ship of type $s \in S$ can only sail at its design speed u_s, we know that $\ell_a/(24u_{s(\sigma_r)})$ indicates the travel time associated with each voyage arc $a \in A_V(\gamma_r)$ of the route γ_r of r. To maintain the weekly service frequency, the total round-trip time (in weeks) of rotation r must be equal to the number of ships $n(\sigma_r, \gamma_r)$ deployed in r, which implies that $n(\sigma_r, \gamma_r) = \left\lceil \sum_{a \in A_V(\gamma_r)} \left[\ell_a/(24u_{s(\sigma_r)}) + \tau\right]/7 \right\rceil$. Moreover, following Brouer et al. (2014a), we suppose that a fuel cost $w^{(1)}_{s,a}(u)$ is given for a ship of type s sailing on voyage arc a at speed u, computed as $w^{(1)}_{s,a}(u) = (\ell_a/u) \cdot F_s(u) \cdot \kappa$, where κ denotes the unit ton fuel oil price, and $B_s(u) = b_s(u/u_s)^3$ (with b_s a given coefficient) indicates the bunker consumption rate of a ship of type s sailing at speed u. To berth a ship of type s at port h, it incurs a given fixed port call cost of $w^{(2)}_{s,h}$ and the τ days' port stay cost of $w^{(3)}_{s,h}$. Let $w^{(0)}_s$ denote the weekly fixed cost for every ship of type s. As shown earlier, the total round-trip time (in weeks) equals the number of ships deployed to maintain the weekly frequency. Thus, the weekly operating cost $w(\sigma_r, \gamma_r)$ of all the deployed ships of rotation r equals the round-trip cost per each deployed ship, which implies that $w(\sigma_r, \gamma_r) = w^{(0)}_{s(\sigma_r)} \cdot n(\sigma_r, \gamma_r) + \sum_{a \in A_V(\gamma_r)} w^{(1)}_{s(\sigma_r),a}(u_{s(\sigma_r)}) + \sum_{i \in V(\gamma_r)} \left(w^{(2)}_{s(\sigma_r),h(i)} + w^{(3)}_{s(\sigma_r),h(i)}\right)$.

EXAMPLE 2 (GENERAL LSND). In addition to the ship type, the sailing speed and the service frequency (i.e., the number of services per week) have also been considered in the LSND literature (see, e.g., Brouer et al. 2014a,b). They can be incorporated in the following setting of the LSND, referred to as the *General LSND*. Suppose that each ship of type $s \in S$ can sail at any speed from a given speed set U_s, and each rotation can have a service frequency from a given frequency set F. The General LSND endogenizes decisions on each rotation's sailing speed and service frequency. Accordingly, each configuration $\sigma \in C$ needs to be associated with not only the ship type $s(\sigma) \in S$, but also the sailing speed and service frequency, denoted by $u(\sigma) \in U$ and $f(\sigma) \in F$, respectively, where $U = \bigcup_{s\in S} U_s$. As a result, for each rotation $r = (\sigma_r, \gamma_r) \in R$, its ship type, sailing speed, and service frequency are represented by $s(\sigma_r)$, $u(\sigma_r)$ and $f(\sigma_r)$, respectively. Following similar arguments in Example 1, we can obtain that the weekly capacity $m(\sigma_r)$ provided by deployed ships of rotation r can be computed by $m(\sigma_r) = f(\sigma_r) m_{s(\sigma_r)}$, the number of ships $n(\sigma_r, \gamma_r)$ deployed in rotation r can be computed by $n(\sigma_r, \gamma_r) = f(\sigma_r) \left\lceil \sum_{a\in A_V(\gamma_r)} [\ell_a/(24u(\sigma_r)) + \tau]/7 \right\rceil$, and the weekly operating cost $w(\sigma_r, \gamma_r)$ of all the deployed ships of rotation r can be computed by $w(\sigma_r, \gamma_r) = w^{(0)}_{s(\sigma_r)} \cdot n(\sigma_r, \gamma_r) + f(\sigma_r) \left[\sum_{a\in A_V(\gamma_r)} w^{(1)}_{s(\sigma_r),a}(u(\sigma_r)) + \sum_{i\in V(\gamma_r)} \left(w^{(2)}_{s(\sigma_r),h(i)} + w^{(3)}_{s(\sigma_r),h(i)} \right) \right]$.

3.3. Mathematical formulation

The formulation of the LSND uses the following three sets of decision variables: (i) non-negative continuous variables $\boldsymbol{g} = [g^k_{r,a} : k \in K, r = (\sigma_r, \gamma_r) \in R, a \in A_V(\gamma_r) \cup A_T(\gamma_r)]$, variable $g^k_{r,a}$ representing the quantity of demand k moved on arc a of rotation r, (ii) non-negative continuous variables $\boldsymbol{x} = [x_k : k \in K]$, variable x_k representing the amount of cargos satisfied for demand k, and (iii) binary variables $\boldsymbol{y} = [y_r : r \in R]$, variable y_r equal to one if rotation r is selected in solution, zero otherwise. The LSND can be formulated into a set-partitioning-like MIP model as follows:

$$\text{(F)}\ \max \sum_{k\in K} p'_k x_k - \sum_{k\in K} p''_k (q_k - x_k) - \sum_{k\in K}\sum_{r\in R}\sum_{a\in A_V(\gamma_r)\cup A_T(\gamma_r)} w^k_a g^k_{r,a} - \sum_{r\in R} w(\sigma_r,\gamma_r) y_r \tag{1a}$$

s.t.

$$\sum_{a\in A^-_V(i)\cap A_V(\gamma_r)} g^k_{r,a} + \sum_{a\in A^\downarrow_T(i)} g^k_{r,a} - \sum_{a\in A^+_V(i)\cap A_V(\gamma_r)} g^k_{r,a} - \sum_{a\in A^\uparrow_T(i)} g^k_{r,a} = 0,\ \forall k\in K,\ r\in R,\ i\in V(\gamma_r), \tag{1b}$$

$$\sum_{a=(j,i)\in A^\downarrow_T(i)}\ \sum_{r\in R: j\in V(\gamma_r)} g^k_{r,a} - \sum_{a=(i,j)\in A^\uparrow_T(i)}\ \sum_{r\in R: j\in V(\gamma_r)} g^k_{r,a} = \begin{cases} -x_k, & o(k) = h(i) \\ 0, & \text{otherwise} \\ x_k, & d(k) = h(i) \end{cases},\ \forall k\in K,\ \forall i\in T, \tag{1c}$$

$$\sum_{r\in R: s(\sigma_r)=s} n(\sigma_r,\gamma_r)y_r \le n_s, \quad \forall s\in S, \tag{1d}$$

$$\sum_{k\in K} g^k_{r,a} - m(\sigma_r)y_r \le 0, \quad \forall r\in R,\, a\in A_V(\gamma_r), \tag{1e}$$

$$g^k_{r,a} - q_k y_r \le 0, \quad \forall k\in K,\, r\in R,\, a\in A_V(\gamma_r), \tag{1f}$$

$$x_k \le q_k, \quad \forall k\in K, \tag{1g}$$

$$x_k\ge 0,\ g^k_{r,a}\ge 0, \quad \forall k\in K,\, r\in R,\, a\in A_V(\gamma_r)\cup A_T(\gamma_r), \tag{1h}$$

$$y_r\in\{0,1\}, \quad \forall r\in R. \tag{1i}$$

The objective function (1a) states to maximize the total profit, where the first two terms compute the total profit by subtracting the penalties of the unsatisfied demands from the revenues of the satisfied demands, the third term computes the cost for moving cargos on the network arcs, including any transshipment cost, and the last term computes the cost of the rotations. Constraints (1b) balance the cargo flows at the voyage nodes for all rotations. Constraints (1c) balance the cargo flows at the transshipment nodes. Note that the two flow balance constraints guarantee that any cargo under loading or unloading operations must go through the transshipment arcs. Constraints (1d) limit the number of utilized ships to be less than or equal to the fleet size associated with each ship type. Constraints (1e) enforce the cargo flows on each rotation to be no greater than the maximum capacity of the deployed ship. Constraints (1f) enforce the cargo flows on each arc to be no greater than their maximum demand volumes. Constraints (1g) impose that the satisfied cargo demands are no greater than their maximum values. The non-negativity and integrity of the decision variables are given in (1h) and (1i).

By grouping the coefficients of variable x_k in (1a) as $p_k = p'_k + p''_k$ for each $k\in K$, the objective function of formulation F can be rewritten as follows:

$$\max \sum_{k\in K} p_k x_k - \sum_{k\in K}\sum_{r\in R}\sum_{a\in A_V(\gamma_r)\cup A_T(\gamma_r)} w^k_a g^k_{r,a} - \sum_{r\in R} w(\sigma_r,\gamma_r)y_r - \sum\nolimits_{k\in K} p''_k q_k, \tag{2}$$

where the last term in the expression is a constant term that, for simplicity of presentation, is disregarded in the following. We denote by LF the LP-relaxation of formulation F.

4. Solving problem LF to optimality: method SCRG-LF

In addition to exponentially many rotation-dependent columns, such as variables $[g^k_{r,a}]$ and $[y_r]$, problem LF also has exponentially many rotation-dependent rows, such as constraints (1b), (1e), and (1f). Consequently, a standard CG procedure cannot compute its optimal solution.

This section describes a novel exact method based on SCRG for problem LF, called SCRG-LF. We define the dual of problem LF, called DLF, as follows. The variables of DLF are given by vectors $[\lambda^k_{r,i}]$, $[\mu^k_i]$, $[\delta_s]$, $[\pi_{r,a}]$, $[\beta^k_{r,a}]$ and $[\varphi_k]$ associated with constraints (1b), (1c), (1d), (1e), (1f) and (1g), respectively. Problem DLF is as follows:

$$\text{(DLF)}\ \min \sum_{s\in S} n_s\delta_s + \sum_{k\in K} q_k\varphi_k, \tag{3a}$$

$$\text{s.t.}\quad \mu^k_{o(k)} - \mu^k_{d(k)} + \varphi_k \geq p_k, \quad \forall k\in K, \tag{3b}$$

$$\lambda^k_{r,j} - \lambda^k_{r,i} + \pi_{r,a} + \beta^k_{r,a} \geq -w^k_a, \quad \forall k\in K,\, r\in R,\, a=(i,j)\in A_V(\gamma_r), \tag{3c}$$

$$\lambda^k_{r,j} - \mu^k_i \geq -w^k_a, \quad \forall k\in K,\, r\in R,\, a=(i,j)\in A^{\uparrow}_T \cap A_T(\gamma_r), \tag{3d}$$

$$-\lambda^k_{r,i} + \mu^k_j \geq -w^k_a, \quad \forall k\in K,\, r\in R,\, a=(i,j)\in A^{\downarrow}_T \cap A_T(\gamma_r), \tag{3e}$$

$$n(\sigma_r,\gamma_r)\delta_{s(\sigma_r)} - \sum_{a\in A_V(\gamma_r)} m(\sigma_r)\pi_{r,a} - \sum_{k\in K}\sum_{a\in A_V(\gamma_r)} q_k\beta^k_{r,a} \geq -w(\sigma_r,\gamma_r), \forall r\in R, \tag{3f}$$

$$\lambda^k_{r,i} \text{ unrestricted}, \quad \forall k\in K,\, r\in R,\, i\in V(\gamma_r), \tag{3g}$$

$$\mu^k_i \text{ unrestricted}, \quad \forall k\in K,\, i\in T, \tag{3h}$$

$$\delta_s\geq 0,\ \pi_{r,a}\geq 0,\ \varphi_k\geq 0,\ \beta^k_{r,a}\geq 0, \quad \forall s\in S,\, k\in K,\, r\in R,\, a\in A_V(\gamma_r). \tag{3i}$$

In formulation DLF, dual variables δ_s and φ_k represent the variation in the objective function of problem LF when the number of available ships of type s and the demand (available containers) of commodity k change by a unit, respectively. Let $\tilde{R}$ be a restricted set of rotations, a subset of R. Given set $\tilde{R}$, we define problem RLF, also called *restricted master problem*, as the problem obtained from problem LF by substituting the set R with $\tilde{R}$. We denote by $DRLF$ the dual of problem RLF, which can also be obtained from problem DLF by substituting the set R with $\tilde{R}$. For the sake of notation, hereinafter we denote by Θ and Π solutions of problem LF and DLF, respectively, and by $z(\Theta)$ and $z(\Pi)$ the corresponding solution values. Similarly, we denote by $\tilde{\Theta}$ and $\tilde{\Pi}$ solutions of problems RLF and $DRLF$, respectively, and by $z(\tilde{\Theta})$ and $z(\tilde{\Pi})$ the corresponding solution values. These solutions are feasible if they satisfy the constraints of problems LF, DLF, RLF, and $DRLF$, respectively.

4.1. SCRG solution frameworks

Based on the solution framework of Feillet et al. (2010), our new method SCRG-LF for solving relaxation LF of the LSND iteratively performs the following three main steps.

(i) *Build a feasible solution* Θ *of* LF. Given $\tilde{\Theta} = (\tilde{\boldsymbol{g}}, \tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}})$, a feasible LF solution $\Theta = (\boldsymbol{g}, \boldsymbol{x}, \boldsymbol{y})$ can be easily defined by setting $\boldsymbol{x} = \tilde{\boldsymbol{x}}$, $\boldsymbol{g}_r = \tilde{\boldsymbol{g}}_r$, $y_r = \tilde{y}_r$, $\forall r \in \tilde{R}$, and $\boldsymbol{g}_r = \mathbf{0}$, $y_r = 0$, $\forall r \in R \setminus \tilde{R}$; clearly we have $z(\Theta) = z(\tilde{\Theta})$;

(ii) *Build a solution* Π *of* DLF. Build solution Π based on solution $\tilde{\Pi}$ such that $z(\Pi) = z(\tilde{\Pi})$;

(iii) *Check if solution* Π *is a feasible* DLF *solution (pricing problem)*. Check if solution Π is a feasible DLF solution. If Π is a feasible DLF solution, by strong duality $z(\tilde{\Theta}) = z(\tilde{\Pi})$, and $z(\Theta) = z(\tilde{\Pi}) = z(\Pi)$, thus showing the optimality of solutions Θ and Π. Otherwise, new columns and rows are generated, and the restricted master problem is updated accordingly.

In the next section (§4.2), we describe the details of method SCRG-LF. In §6.2 we compare our newly proposed method SCRG-LF of solving relaxation LF with the following alternative methods:

• The method derived from the approach of Xia et al. (2015), referred to as XLMX (see §EC.4 of the e-companion). Method XLMX is based on a relaxation of LF obtained by aggregating inequalities (1b), (1e), and (1f), respectively, for all $r \in R$ with $\sigma_r = \sigma$. The derived relaxation of LF can be conveniently solved by using a standard CG technique.

• The method derived from the SCRG solution framework proposed by Muter et al. (2013), referred to as SCRG-LF-MBB (see §EC.6 of the e-companion). In method SCRG-LF-MBB, the DLF solution required at step (ii) of the framework is built by the method adapted from Muter et al. (2013).

• The method derived from the work of Thun et al. (2017) (see §EC.7 of the e-companion). More specifically, following Thun et al. (2017), problem F can be reformulated using an alternative set-partitioning-like model, which is defined by a large number of decision variables, each indicating the adoption of a combination of a rotation and its delivery pattern (i.e., cargo flows of the rotation) to serve the demands. By doing this, a linear relaxation of the model can be solved by a CG technique.

In §EC.5 of the e-companion, we also illustrate why the two solution methods developed in Feillet et al. (2010) cannot be used or are inefficient when applied to solve relaxation LF.

4.2. Description of method SCRG-LF

Following the approach described in §4.1, we develop the main steps of method SCRG-LF as shown in Algorithm 1. We use $\Pi = (\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\delta}, \boldsymbol{\pi}, \boldsymbol{\varphi}, \boldsymbol{\beta})$ and $\tilde{\Pi} = (\tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\delta}}, \tilde{\boldsymbol{\pi}}, \tilde{\boldsymbol{\varphi}}, \tilde{\boldsymbol{\beta}})$ to represent DLF

Algorithm 1 Method SCRG-LF

```
1: Initialization. Define problem $RLF$ by initializing set $\tilde{R}$ to contain all rotations $r = (\sigma_r, \gamma_r)$ such that route $\gamma_r$ consists of exactly two-ports and configuration $\sigma_r \in C$;
2: Solution of the restricted master problem. Solve problem $RLF$ and let $\tilde{\Theta}$ and $\tilde{\Pi} = (\tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\delta}}, \tilde{\boldsymbol{\pi}}, \tilde{\boldsymbol{\theta}}, \tilde{\boldsymbol{\beta}})$ be the corresponding optimal primal and dual solutions of values $z(\tilde{\Theta})$ and $z(\tilde{\Pi})$, respectively;
3: Build a solution $\Pi$ of $DLF$. Based on the optimal $DRLF$ solution $\tilde{\Pi}$, determine $\bar{\Pi} = (\bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\pi}}, \bar{\boldsymbol{\beta}})$ by computing values of the unknown dual variables associated with rotations in $R \setminus \tilde{R}$, and construct a $DLF$ solution $\Pi = \tilde{\Pi} \cup \bar{\Pi}$ such that $z(\Pi) = z(\tilde{\Pi})$ by setting each $\mu_i^k = \tilde{\mu}_i^k$, $\delta_s = \tilde{\delta}_s$, and $\varphi_k = \tilde{\varphi}_k$, setting each $\lambda_{r,i}^k = \tilde{\lambda}_{r,i}^k$, $\pi_{r,a} = \tilde{\pi}_{r,a}$, $\beta_{r,a}^k = \tilde{\beta}_{r,a}^k$ for $r \in \tilde{R}$, and setting each $\lambda_{r,i}^k = \bar{\lambda}_{r,i}^k$, $\pi_{r,a} = \bar{\pi}_{r,a}$, $\beta_{r,a}^k = \bar{\beta}_{r,a}^k$ for $r \in R \setminus \tilde{R}$. Two alternative approaches can be used to construct solution $\Pi$: sequential and LP-based approaches (see §4.3);
4: Pricing problem. Execute a pricing algorithm (see Algorithm 2 later in §4.4) to check if $\Pi$ can be transformed to a feasible $DLF$ solution $\Pi'$ with $z(\Pi') = z(\Pi)$, and compute $\bar{R} \subseteq (R \setminus \tilde{R})$ as a subset of rotations for which dual constraints (3c)–(3f) are violated. Two alternative pricing techniques can be used: with and without the post-pricing phase (see §4.4.1 and §4.4.2).
5: Check optimality condition. If $\bar{R}$ is empty, implying that $\Pi$ can be transformed to a feasible $DLF$ solution $\Pi'$ such that $z(\Pi') = z(\Pi)$, which must be an optimal $DLF$ solution, then stop. Otherwise, update set $\tilde{R} = \tilde{R} \cup \bar{R}$, so that problem $RLF$ is updated by adding all primal variables $[g_{r,a}^k : k \in K, r \in \bar{R}, a \in A_V(\gamma_r) \cup A_T(\gamma_r)]$ and $[y_r : r \in \bar{R}]$, as well as all corresponding rotation-dependent constraints, and then go to Step 2.
```

solutions and $DRLF$ solutions, respectively. We use $\bar{\Pi} = (\bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\pi}}, \bar{\boldsymbol{\beta}})$ to represent a vector that contains values for the dual variables associated with the rotations in the subset $R \setminus \tilde{R}$, where $\bar{\boldsymbol{\lambda}} = [\bar{\lambda}_{r,i}^k : k \in K, r \in R \setminus \tilde{R}, i \in V(\gamma_r)]$, $\bar{\boldsymbol{\pi}} = [\bar{\pi}_{r,a} : r \in R \setminus \tilde{R}, a \in A_V(\gamma_r)]$ and $\bar{\boldsymbol{\beta}} = [\bar{\beta}_{r,a}^k : k \in K, r \in R \setminus \tilde{R}, a \in A_V(\gamma_r)]$.

Step 1 of Algorithm 1 ensures the feasibility of the initial RLF problem. Similar to the approach in Agarwal and Ergun (2008) for the LSND without transshipment costs, it initializes $\tilde{R}$ to include only two-port rotations that cover all voyage arcs, providing a warm-up start of method SCRG-LF. Based on the optimal $DRLF$ solution $\tilde{\Pi} = (\tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\delta}}, \tilde{\boldsymbol{\pi}}, \tilde{\boldsymbol{\varphi}}, \tilde{\boldsymbol{\beta}})$ obtained in Step 2 for the restricted master problem, Step 3 of Algorithm 1 computes $\bar{\Pi} = (\bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\pi}}, \bar{\boldsymbol{\beta}})$ that contains values for the unknown dual variables associated with rotations in $R \setminus \tilde{R}$. It then defines a DLF solution $\Pi =$

$\tilde{\Pi} \cup \bar{\Pi}$, which combines the values of dual variables in $\tilde{\Pi}$ and $\bar{\Pi}$. Since the objective function (3a) of problem DLF does not depend on the dual variables in $\bar{\Pi}$, we obtain that $z(\Pi) = z(\tilde{\Pi})$.

Step 4 of Algorithm 1 examines whether the solution Π obtained in Step 3 can be transformed to a feasible DLF solution Π' with $z(\Pi') = z(\Pi)$. For this, it is sufficient to execute a pricing algorithm to check if the dual variables in $\bar{\Pi}$ satisfy or can be transformed to satisfy the dual constraints (3c)–(3f) for all $r \in R \setminus \tilde{R}$. If Π can be transformed to a feasible DLF solution Π' with $z(\Pi') = z(\Pi)$, then since $z(\Pi) = z(\tilde{\Pi})$, Π' is a proven optimal DLF solution, and Algorithm 1 stops at Step 5; otherwise, $\tilde{R}$ is updated to include a subset $\bar{R}$ of rotations for which constraints (3c)–(3f) are violated, so that problem RLF is updated, accordingly, and is solved again.

In the following, §4.3 illustrates the two approaches used to compute $\bar{\Pi} = (\bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\pi}}, \bar{\boldsymbol{\beta}})$ for the construction of the DLF solution Π at Step 3 of Algorithm 1. §4.4 describes the pricing algorithm used at Step 4 of Algorithm 1 and proves the correctness and convergence of Algorithm 1.

4.3. Computing $\bar{\Pi} = (\bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\pi}}, \bar{\boldsymbol{\beta}})$

The dual variables composing $\bar{\Pi} = (\bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\pi}}, \bar{\boldsymbol{\beta}})$ cannot be built directly since the number of rotations in $R \setminus \tilde{R}$ is exponential. To tackle this challenge, we first define a vector $\hat{\Pi} = (\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\beta}})$ that consists of values for some auxiliary variables, denoted by $[\hat{\lambda}^k_{\sigma,i} : k \in K, \sigma \in C, i \in V]$, $[\hat{\pi}_{\sigma,a} : \sigma \in C, a \in A_V]$, and $[\hat{\beta}^k_{\sigma,a} : k \in K, \sigma \in C, a \in A_V]$, satisfying the dual constraints (3c)–(3e) for all $r \in R \setminus \tilde{R}$. Specifically, we aim to find $\hat{\Pi} = (\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\beta}})$ that satisfies the linear constraints of the following problem:

$$\hat{\lambda}^k_{\sigma,j} - \hat{\lambda}^k_{\sigma,i} + \hat{\pi}_{\sigma,a} + \hat{\beta}^k_{\sigma,a} \geq -w^k_a, \quad \forall k \in K,\, \sigma \in C,\, a = (i,j) \in A_V, \tag{4a}$$

$$\hat{\lambda}^k_{\sigma,j} \geq -w^k_a + \tilde{\mu}^k_i, \quad \forall k \in K,\, \sigma \in C,\, a = (i,j) \in A^{\uparrow}_T, \tag{4b}$$

$$-\hat{\lambda}^k_{\sigma,i} \geq -w^k_a - \tilde{\mu}^k_j, \quad \forall k \in K,\, \sigma \in C,\, a = (i,j) \in A^{\downarrow}_T, \tag{4c}$$

$$\hat{\lambda}^k_{\sigma,i} \text{ unrestricted}, \quad \forall k \in K,\, \sigma \in C,\, i \in V, \tag{4d}$$

$$\hat{\pi}_{\sigma,a} \geq 0, \hat{\beta}^k_{\sigma,a} \geq 0, \quad \forall k \in K,\, \sigma \in C,\, a \in A_V, \tag{4e}$$

in which the numbers of constraints and variables are in $O(|K||C|(|A| + |V|))$, which are polynomial in $|K|$, $|C|$, $|A|$, and $|V|$. Here, values of $[\tilde{\mu}^k_i]$ are given in $\tilde{\Pi}$ which, as previously mentioned, is an optimal $DRLF$ solution.

Given $\hat{\Pi} = (\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\beta}})$, we can compute $\bar{\Pi} = (\bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\pi}}, \bar{\boldsymbol{\beta}})$ by a mapping that sets $\bar{\lambda}^k_{r,i} = \hat{\lambda}^k_{\sigma,i}$, $\bar{\pi}_{r,a} = \hat{\pi}_{\sigma,a}$ and $\bar{\beta}^k_{r,a} = \hat{\beta}^k_{\sigma,a}$, for each $k \in K$, $\sigma \in C$, $i \in V$, and $r \in R \setminus \tilde{R}$ with $\sigma_r = \sigma$. By doing this, the set of unknown dual variables associated with rotations $r \in R \setminus \tilde{R}$, which has an exponential size, are mapped to a set of auxiliary dual variables associated with configurations $\sigma \in C$, which has a size polynomial in $|K|$, $|C|$, $|A|$, and $|V|$. Since solution $\bar{\Pi}$ satisfies the dual constraints (3c)–(3e) for all $r \in R \setminus \tilde{R}$, to check further that the solution also satisfies the dual constraints (3f), it is necessary to solve the following *pricing problem* at Step 4 of Algorithm 1:

$$\Delta = \min_{r \in R \setminus \tilde{R}} \left\{ w(\sigma_r, \gamma_r) + n(\sigma_r, \gamma_r)\tilde{\delta}_{s(\sigma_r)} - \sum_{a \in A_V(\gamma_r)} m(\sigma_r)\hat{\pi}_{\sigma_r,a} - \sum_{k \in K} \sum_{a \in A_V(\gamma_r)} q_k \hat{\beta}^k_{\sigma_r,a} \right\}, \tag{5}$$

which is defined based on the values of auxiliary dual variables specified in $\hat{\Pi} = (\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\beta}})$. Note that $-\Delta$ computes the maximum reduced cost for the rotations in $R \setminus \tilde{R}$. Accordingly, if $\Delta \geq 0$, which implies that all rotations in $R \setminus \tilde{R}$ are with non-positive reduced costs, then solution $\bar{\Pi}$ must satisfy the dual constraints (3c)–(3f), implying that $\Pi = \tilde{\Pi} \cup \bar{\Pi}$ constructed at Step 3 of Algorithm 1 is a proven optimal DLF solution.

It is worth noting that the above mapping from $\bar{\Pi}$ to $\hat{\Pi}$ is essentially based on a dimension contraction from the rotation set R (which equals $\Gamma \times C$) to the configuration set C, thus reducing the number of dual variables to be polynomial. However, in some situations, such as in the General LSND, it is possible that the configuration set C itself has a large size. To further reduce the number of dual variables, one can partition C into disjoint groups of configurations with the configurations in the same group having the same values of some service components (e.g., having the same value of ship type), so that the dual variables associated with rotations in $R \setminus \tilde{R}$ can be mapped to the auxiliary dual variables associated with the configuration groups. Reducing the number of dual variables still ensures that the solution Π returned by Algorithm 1 is a proven optimal DLF solution. For simplicity of presentation, we consider only the mappings based on configurations $\sigma \in C$. The same method can be adapted to other mappings mentioned here by replacing the configuration indices with the configuration group indices.

In the following, we describe two alternative approaches to compute $\hat{\Pi} = (\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\beta}})$.

A sequential approach We first compute the values of variables $[\hat{\lambda}^k_{\sigma,i}]$, and then define the values of variables $[\hat{\pi}_{\sigma,a}]$ and $[\hat{\beta}^k_{\sigma,a}]$. Variables $[\hat{\lambda}^k_{\sigma,i}]$ are computed as $\hat{\lambda}^k_{\sigma,i} = \tilde{\mu}^k_j$ for $k \in K$, $\sigma \in C$, $(i,j) \in A^{\downarrow}_T$, for which we know that for each $(i,j) \in A^{\downarrow}_T$, node j is the unique transshipment node in T defined for the port $h(i)$ of the voyage node i. Since each w^k_a is non-negative, the values of $[\hat{\pi}_{\sigma,a}]$ and $[\hat{\beta}^k_{\sigma,a}]$ defined above guarantee that constraints (4b) and (4c) are satisfied. We then compute variables $[\hat{\pi}_{\sigma,a}]$ as $\hat{\pi}_{\sigma,a} = 0$ for $\sigma \in C$, $a \in A_V$, and variables $[\hat{\beta}^k_{\sigma,a}]$ as $\hat{\beta}^k_{\sigma,a} = \max\{0, -\tilde{\mu}^k_j + \tilde{\mu}^k_i - w^k_a\}$ for $k \in K$, $\sigma \in C$, $a = (i,j) \in A_V$, in order to satisfy constraints (4a).

An LP-based approach This approach is based on the following proposition.

PROPOSITION 1. *Consider any* $\hat{\Pi}^1 = (\hat{\boldsymbol{\lambda}}^1, \hat{\boldsymbol{\pi}}^1, \hat{\boldsymbol{\beta}}^1)$ *and* $\hat{\Pi}^2 = (\hat{\boldsymbol{\lambda}}^2, \hat{\boldsymbol{\pi}}^2, \hat{\boldsymbol{\beta}}^2)$ *that satisfy constraints* (4a)–(4e). *If* $m(\sigma)\hat{\pi}^1_{\sigma,a} + \sum_{k\in K} q_k \hat{\beta}^{k,1}_{\sigma,a} \le m(\sigma)\hat{\pi}^2_{\sigma,a} + \sum_{k\in K} q_k \hat{\beta}^{k,2}_{\sigma,a}$ *holds for each* $\sigma \in C$ *and* $a \in A_V$*, with at least of such inequalities being strict, then the pricing problem* (5) *under* $\hat{\Pi}^1$ *has an optimal solution value larger than or equal to that under* $\hat{\Pi}^2$.

Proof. See §EC.1.1 of the e-companion to this paper. □

For any $\hat{\Pi}^1$ and $\hat{\Pi}^2$ that satisfy the condition stated in Proposition 1 above, we say $\hat{\Pi}^1$ *dominates* $\hat{\Pi}^2$. The condition implies that a rotation having a non-negative reduced cost under $\hat{\Pi}^1$ cannot have a negative reduced cost under $\hat{\Pi}^1$. This implies that choosing a non-dominated $\hat{\Pi}$ for the constructions of $\bar{\Pi}$ and Π can potentially speed up the convergence of method SCRG-LF.

As shown in §EC.2 of the e-companion, the sequential approach may generate a dominated $\hat{\Pi}$. To obtain a non-dominated $\hat{\Pi}$, we propose an LP-based approach, which solves the LP model below:

$$\min \quad \sum_{\sigma\in C}\sum_{a\in A_V} \eta_{\sigma,a}\left(m(\sigma)\hat{\pi}_{\sigma,a} + \sum_{k\in K} q_k \hat{\beta}^k_{\sigma,a}\right) \text{ s.t. (4a)} - \text{(4e)},$$

where the coefficient $\eta_{\sigma,a}$ for each $\sigma \in C$ and $a \in A_V$ denotes the total number of existing rotations $r \in \tilde{R}$ with $\sigma_r = \sigma$ and $a \in A(\gamma_r)$. It can be seen that the objective of the LP model above is equivalent to maximizing the total reduced cost concerning the existing rotations in $\tilde{R}$. As a result, solving the LP model above reduces the chance that the solution to the pricing problem may unnecessarily generate existing rotations in $\tilde{R}$. This LP-based approach can be adapted and applied to other problems, for which one needs to form an LP model according to the dual formulation of the problems, aiming at yielding a non-dominated dual solution for the subsequent pricing stage.

Let $\hat{\Pi}^3$ indicate the optimal solution to the LP model above. It can be shown that $\hat{\Pi}^3$ must be non-dominated. By contradiction, suppose that there exists a solution $\hat{\Pi}^1$ satisfying constraints (4a)–(4e) and dominating $\hat{\Pi}^3$. We have that $m(\sigma)\hat{\pi}^1_{\sigma,a} + \sum_{k\in K} q_k \hat{\beta}^{k,1}_{\sigma,a} \le m(\sigma)\hat{\pi}^3_{\sigma,a} + \sum_{k\in K} q_k \hat{\beta}^{k,3}_{\sigma,a}$ holds for each $\sigma \in C$ and $a \in A_V$, with at least one of such inequalities being strict. Thus, $\sum_{\sigma\in C}\sum_{a\in A_V} \eta_{\sigma,a}\left(m(\sigma)\hat{\pi}^1_{\sigma,a} + \sum_{k\in K} q_k \hat{\beta}^{k,1}_{\sigma,a}\right) < \sum_{\sigma\in C}\sum_{a\in A_V} \eta_{\sigma,a}\left(m(\sigma)\hat{\pi}^3_{\sigma,a} + \sum_{k\in K} q_k \hat{\beta}^{k,3}_{\sigma,a}\right)$, which contradicts to the fact that $\hat{\Pi}^3$ is an optimal solution to the above LP model.

4.4. Pricing rotations

Given the optimal $DRLF$ solution $\tilde{\Pi} = (\tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{\pi}}, \tilde{\boldsymbol{\beta}})$ and the values of auxiliary variables contained in $\hat{\Pi} = (\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\beta}})$, which are used to construct the DLF solution Π as described in §4.3, the pricing problem (5) for generating rotations of positive reduced costs can be decomposed into a set of independent pricing subproblems, with one for each configuration. To see this, for each configuration $\sigma \in C$ and each route $\gamma \in \Gamma(\sigma)$, we define

$$\Delta(\sigma,\gamma) = w(\sigma,\gamma) + n(\sigma,\gamma)\tilde{\delta}_{s(\sigma)} - \sum_{a \in A_V(\gamma)} m(\sigma)\hat{\pi}_{\sigma,a} - \sum_{k \in K}\sum_{a \in A_V(\gamma)} q_k \hat{\beta}^k_{\sigma,a}. \tag{6}$$

For each $r = (\sigma_r, \gamma_r) \in R$, the value of $-\Delta(\sigma_r, \gamma_r)$ equals the reduced cost with respect to rotation r. Thus, a rotation r of a positive reduced cost corresponds to a rotation r with $\Delta(\sigma_r, \gamma_r) < 0$, and finding a rotation of the maximum reduced cost is equivalent to finding a rotation with the smallest $\Delta(\sigma_r, \gamma_r)$. Accordingly, the pricing problem (5) can be decomposed as follows:

$$\Delta = \min_{\sigma \in C}\{\Delta_\sigma\},$$

where for each configuration $\sigma \in C$,

$$\Delta_\sigma = \min\left\{\Delta(\sigma,\gamma_r) : r = (\sigma_r,\gamma_r) \in R \setminus \tilde{R} \text{ with } \sigma_r = \sigma\right\}. \tag{7}$$

Each pricing subproblem Δ_σ for $\sigma \in C$ corresponds to finding a route γ in graph G that minimizes its weight indicated by $\Delta(\sigma,\gamma)$, where the weights of arcs and nodes of graph G are preset accordingly with the values of dual variables $[\tilde{\delta}_{s(\sigma)}]$, $[\hat{\pi}_{\sigma,a}]$ and $[\hat{\beta}^k_{\sigma,a}]$ contained in $\tilde{\Pi}$ and $\hat{\Pi}$.

It is worth noting that an existing rotation $r \in \tilde{R}$ involved in the restricted master problem can have a positive reduced cost, i.e., $\Delta(\sigma_r,\gamma_r) < 0$, since values of dual variables in $\hat{\Pi}$ do not necessarily satisfy the dual constraints associated with rotation r. What is more, given a rotation r which is not necessarily in $\tilde{R}$ and has a positive reduced cost concerning the objective function of the pricing subproblem (7), one can potentially identify for rotation r some alternative values of dual variables $[\lambda^k_{r,i}]$, $[\pi_{r,a}]$ and $[\beta^k_{r,a}]$ to satisfy dual constraints (3c)–(3e), leading to a non-positive reduced cost, so that the generation of rotation r becomes unnecessary. This is because the definition of the auxiliary variables $[\hat{\lambda}^k_{\sigma,i}]$, $[\hat{\pi}_{\sigma,a}]$ and $[\hat{\beta}^k_{\sigma,a}]$ (see §4.3) is based on the whole set of arcs instead of the specific set of arcs composing rotation r. As a result, a *post-pricing* phase (discussed later in §4.4.2) is needed to avoid generating such unnecessary rotations, which can potentially speed up the convergence of method SCRG-LF.

Next, we present our algorithm for pricing rotations. It is based on a conventional circuit search procedure (CSA) that detects negative weight elementary circuits on a weighted network, as described in §4.4.1. In §4.4.2, we introduce the post-pricing phase used in the CSA to avoid generating unnecessary rotations. The overall pricing algorithm is then summarized in §4.4.3.

In the following, we denote with $\hat{R}$ and R^* the set of rotations computed by procedure CSA and the forbidden rotations that cannot be included in $\hat{R}$. The final set of rotations returned by the pricing algorithm at Step 4 of Algorithm 1 is denoted by $\bar{R}$.

4.4.1. Procedure CSA without post-pricing phase Similar to the generic dynamic programming algorithm applied to shortest path problems with resource constraints (see, e.g., Irnich and Desaulniers 2005), procedure CSA is a dynamic programming-based algorithm that dynamically expands a state-space graph for each configuration $\sigma \in C$, where each state corresponds to a feasible forward path defined below.

A *forward path* $P = (i_0, i_1, \ldots, i_{k-1}, i_k)$ is an elementary path in G starting from voyage node $i_0 = start(P)$, visiting voyage nodes in $\{i_1, \ldots, i_{k-1}, i_k\}$, and ending at voyage node $i_k = end(P)$. Similar to routes of rotations, we denote by $V(P)$ and $A_V(P)$ the set of voyage nodes and the set of voyage arcs of path P, respectively. A forward path is feasible if it satisfies rules (i)-(v) described in §3.1. Thus, a feasible forward path P may form a partial route of a rotation, and it forms a complete route only when $start(P) = end(P)$.

Replacing σ_r and γ_r in $n(\sigma_r, \gamma_r)$ and $w(\sigma_r, \gamma_r)$ with σ and P, respectively, we can define $n(\sigma, P)$ and $w(\sigma, P)$, which can be regarded as partial values of $n(\sigma_r, \gamma_r)$ and $w(\sigma_r, \gamma_r)$ for rotations r with $\sigma_r = \sigma$ and with γ_r containing P at the beginning. Similar to $\Delta(\sigma, \gamma)$ defined in (6) as the weight of a route γ, we can define $\Delta(\sigma, P)$ below as the weight of the forward path P:

$$\Delta(\sigma, P) = w(\sigma, P) + n(\sigma, P)\tilde{\delta}_{s(\sigma)} - \sum_{a \in A_V(P)} m(\sigma)\hat{\pi}_{\sigma,a} - \sum_{k \in K} \sum_{a \in A_V(P)} q_k \hat{\beta}^k_{\sigma,a}. \tag{8}$$

When $start(P) = end(P)$, the forward path P forms a feasible route $\gamma \in \Gamma(\sigma)$, and thus, $-\Delta(\sigma, P)$ equals the reduced cost $-\Delta(\sigma_r, \gamma_r)$ of the corresponding rotation r with $\sigma_r = \sigma$ and $\gamma_r = \gamma$.

The set $\hat{R}$ of rotations to be returned by procedure CSA is initialized by setting $\hat{R} = \emptyset$. As mentioned earlier in §4.4, there may be some existing rotations in $\tilde{R}$ that have positive reduced costs, which, however, should not be returned by procedure CSA. Accordingly, for procedure CSA, it is given a set R^* of forbidden rotations that cannot be included in $\hat{R}$, where R^* contains all the existing rotations in $\tilde{R}$.

Procedure CSA examines every configuration $\sigma \in C$. For σ currently examined, Procedure CSA iteratively expands forward paths to search rotations with negatively reduced costs. Let $\mathcal{P}$ denote a set of feasible forward paths (concerning σ) that are generated but not expanded. Initially, we set $\mathcal{P} = \{P_1, \ldots, P_{|H|}\}$ where each P_h for $h \in H$ represents a forward path that consists of only one node and that such a node is a voyage node for port h (i.e., $start(P_h) = end(P_h) = i$ for some $i \in V$ with $h(i) = h$).

At each iteration of procedure CSA where σ is examined, the forward path $P \in \mathcal{P}$ having the smallest weight $\Delta(\sigma, P)$ is extracted from $\mathcal{P}$ to expand. The expansions of the forward path P are derived by expanding P with each arc $(end(P), j) \in A_V$ for $j \notin V(P) \setminus \{start(P)\}$. Consider the following two situations after the expansion:

(i) $j = start(P)$. The expansion of forward path P creates a route of a rotation, denoted by γ, so that we can obtain a rotation $r = (\sigma_r, \gamma_r)$ with $\sigma_r = \sigma$ and $\gamma_r = \gamma$. If route σ is not feasible concerning the configuration σ or such a rotation r is in the forbidden set R^*, then rotation r is discarded. Otherwise, route γ is feasible so that $\gamma \in \Gamma(\sigma)$, and if $\Delta(\sigma, \gamma) < 0$, implying that rotation r has a positive reduced cost, then we insert r in the set $\hat{R}$;

(ii) $j \neq start(P)$. The expansion of the forward path P creates a new forward path $P' = P \cup \{j\}$ so that $end(P') = j$; if the path P' is feasible concerning σ, we add P' to the set $\mathcal{P}$.

The iterations of Procedure CSA for σ are terminated when either $\mathcal{P} = \emptyset$ or there are Λ^1 rotations r with $\sigma_r = \sigma$ that have been newly inserted in $\hat{R}$, where $\Lambda^1 > 0$ is a parameter defined a priori.

After all configurations $\sigma \in C$ have been examined, we sort rotations $r \in \hat{R}$ in a non-increasing order of their reduced costs $-\Delta(\sigma_r, \gamma_r)$, and keep only the first $\min\{\Lambda^2, |\hat{R}|\}$ rotations in $\hat{R}$ to return, where $\Lambda^2 > 0$ is a parameter defined a priori. If procedure CSA returns $\hat{R} = \emptyset$, the constructed DLF solution $\Pi = \tilde{\Pi} \cup \bar{\Pi}$ is a feasible DLF solution, implying that problem LF is solved to optimality.

4.4.2. Procedure CSA with post-pricing phase Consider every rotation $r = (\sigma_r, \gamma_r)$ to be inserted in $\hat{R}$ by procedure CSA in situation (i) above after the path expansion, where $\sigma_r \in C$ and $\gamma_r \in \Gamma(\sigma_r)$. We know that $\Delta(\sigma_r, \gamma_r) < 0$ and r is not in the forbidden set R^*.

We introduce a post-pricing phase to check further if it is necessary to insert rotation r in set $\hat{R}$. For this, we solve the following rotation-dependent LP problem, which aims to find alternative values for the rotation-dependent dual variables that result in a negative reduced cost of r:

$$\begin{aligned}
\Delta^*(\sigma_r, \gamma_r) = \max \quad & w(\sigma_r, \gamma_r) + n(\sigma_r, \gamma_r)\tilde{\delta}_{s(\sigma_r)} - \sum_{a \in A_V(\gamma_r)} m(\sigma_r)\pi_{\sigma_r, a} - \sum_{k \in K}\sum_{a \in A_V(\gamma_r)} q_k \beta^k_{\sigma_r, a} \\
\text{s.t.} \quad & \lambda^k_{r,j} - \lambda^k_{r,i} + \pi_{r,a} + \beta^k_{r,a} \geq -w^k_a, \quad \forall k \in K,\ a = (i,j) \in A_V(\gamma_r), \\
& \lambda^k_{r,j} \geq -w^k_a + \tilde{\mu}^k_i, \quad \forall k \in K, a = (i,j) \in A^{\uparrow}_T \cap A_T(\gamma_r), \\
& -\lambda^k_{r,i} \geq -w^k_a - \tilde{\mu}^k_j, \quad \forall k \in K,\ a = (i,j) \in A^{\downarrow}_T \cap A_T(\gamma_r),
\end{aligned} \qquad (9)$$

$$\lambda^k_{r,i} \text{ unrestricted}, \quad \forall k \in K,\, i \in V(\gamma_r),$$

$$\pi_{r,a} \geq 0,\, \beta^k_{r,a} \geq 0, \quad \forall k \in K,\, a \in A_V(\gamma_r).$$

Here, values of variables $[\tilde{\mu}^k_i]$ and $[\tilde{\delta}_{s(\sigma_r)}]$ are given in $\tilde{\Pi}$ which is an optimal solution of $DRLF$. The values of vectors $[\bar{\lambda}^k_{r,i}]$, $[\bar{\pi}_{r,a}]$ and $[\bar{\beta}^k_{r,a}]$ given in solution $\bar{\Pi}$ (with $\bar{\lambda}^k_{r,i} = \hat{\lambda}^k_{\sigma_r,i}$, $\bar{\pi}_{r,a} = \hat{\pi}_{\sigma_r,a}$ and $\bar{\beta}^k_{r,a} = \hat{\beta}^k_{\sigma_r,a}$) also form a feasible solution to problem (9). Hence, $\Delta(\sigma_r,\gamma_r) \leq \Delta^*(\sigma_r,\gamma_r)$ must hold.

Accordingly, if $\Delta^*(\sigma_r,\gamma_r) < 0$, no alternative values exist for the rotation-dependent dual variables that result in a negative reduced cost of r. Thus, we claim that rotation r passes the post-pricing phase and insert it in $\hat{R}$. Otherwise, rotation r is discarded.

The following proposition shows that when the procedure CSA with such a post-pricing phase terminates with $\hat{R} = \emptyset$, problem LF is solved to optimality.

PROPOSITION 2. *When procedure CSA with post-pricing phase terminates with $\hat{R} = \emptyset$, the DLF solution $\Pi = \tilde{\Pi} \cup \bar{\Pi}$ can be transformed to a feasible DLF solution Π' such that $z(\Pi') = z(\Pi)$, and problem LF is solved to optimality.*

Proof. See §EC.1.2 of the e-companion to this paper. □

4.4.3. Pricing algorithm: computing $\bar{R}$ at Step 4 of Algorithm 1 Let us first introduce a heuristic dominance rule below, which can be applied to shorten the running time of procedure CSA with the post-pricing phase.

Heuristic Dominance Rule: For any $\sigma \in C$, a forward path P_1 dominates a forward path P_2 with respect to configuration σ, if $start(P_1) = start(P_2)$, $end(P_1) = end(P_2)$, $V(P_1) \subseteq V(P_2)$, $\sum_{a \in A_V(P_1)} t_a \leq \sum_{a \in A_V(P_2)} t_a$ and $\Delta(\sigma, P_1) \leq \Delta(\sigma, P_2)$.

The heuristic dominance rule above is applied to eliminate forward paths that visit more nodes and consume longer travel times but have larger path weights among the paths with the same starting and ending nodes. However, some forward paths that lead to optimal solutions to the pricing problem may be dominated by some partial paths that lead to forbidden rotations and may, therefore, be eliminated by this heuristic dominance rule. Thus, procedure CSA with both the post-pricing phase and using this heuristic dominance rule cannot ensure that the problem LF is solved optimally when terminating with $\hat{R} = \emptyset$.

Hence, in our pricing algorithm, if procedure CSA with both the post-pricing phase and the use of the heuristic dominance rule produces a non-empty $\hat{R}$, we return $\bar{R} = \hat{R}$, rotations in which all have positive reduced costs. Otherwise, $\hat{R}$ is empty, and we then run the CSA again with the

Algorithm 2 Pricing algorithm used at Step 4 of Algorithm 1

```
1: Initialize the output rotation set $\bar{R} = \emptyset$ and the forbidden rotation set $R^* = \tilde{R}$.
2: Execute procedure CSA with the post-pricing phase and with the use of the heuristic dominance rule to obtain a set $\hat{R}$ of rotations (if any), where each $r \in \hat{R}$ has passed the post-pricing phase, and thus satisfies $\Delta^*(\sigma_r, \gamma_r) < 0$. Update $R^*$ to include all the rotations checked in the post-pricing phase of procedure CSA.
3: Consider the following two cases of $\hat{R}$:
     • Case 1: $\hat{R} \neq \emptyset$. Set $\bar{R} = \hat{R}$, and return $\bar{R}$.
     • Case 2: $\hat{R} = \emptyset$. Execute procedure CSA with the post-pricing phase and without the use of the heuristic dominance rule to obtain an updated set $\hat{R}$ of rotations, where each $r \in \hat{R}$ has passed the post-pricing phase, and thus, satisfies $\Delta^*(\sigma_r, \gamma_r) < 0$. Set $\bar{R} = \hat{R}$, and return $\bar{R}$.
```

post-pricing phase but without the use of the heuristic dominance rule to obtain an updated $\hat{R}$ to return as $\bar{R}$. If the updated $\hat{R}$ is still empty, by Proposition 2, problem LF is solved optimally.

We summarize the above pricing algorithm in Algorithm 2, where at the beginning, the output rotation set $\bar{R}$ is initialized to be empty, and the forbidden rotation set R^* is initialized to contain all the existing rotations in $\tilde{R}$. This pricing algorithm is used at Step 4 of Algorithm 1.

We can now establish Theorem 1 to show the correctness and convergence of Algorithm 1 for method SCRG-LF in solving problem LF.

THEOREM 1. *With Algorithm 2 applied as the pricing algorithm in Step 4, Algorithm 1 solves problem LF to optimality in a finite number of iterations.*

Proof. See §EC.1.3 of the e-companion to this paper. □

5. An exact algorithm for the LSND

This section presents our exact method for solving problem F of the LSND. It is based on a BP approach, where, at each node of the enumeration tree, an upper bound on the optimal solution value is computed by method SCRG-LF in §4.

5.1. Exact algorithm: applying SCRG-LF in branch-and-price

In our BP-based exact method, we apply method SCRG-LF (Algorithm 1) at each node of the enumeration tree to compute an upper bound on the optimal solution value of problem F. The enumeration tree is explored by following a *best-bound* strategy, i.e., the node with the highest upper bound will be explored first. We adopt a binary branching strategy based on variables $[y_r]$. More precisely, we select the variable y_r whose value is closest to 0.5 and then branch by enforcing the selected rotation to be included in the solution or not, i.e., $y_r = 1$ or $y_r = 0$, respectively.

A node of the enumeration tree can be fathomed if the resulting upper bound is less than or equal to the incumbent lower bound of problem F. Nevertheless, at different nodes of the enumeration tree, it can be time-consuming for method SCRG-LF to compute the upper bound by solving relaxation LF to optimality. This is because for proving the optimality of a solution to LF, method SCRG-LF needs to solve the pricing problem to optimality, which may require the pricing algorithm to explore exponentially many partial paths. Method SCRG-LF may also suffer from degenerations (similar to traditional CG algorithms), which slows down the convergence.

To further speed up the computation at each node of the enumeration tree, we allow method SCRG-LF to terminate prematurely with an infeasible solution of DLF, based on which we can compute a valid upper bound (see §5.2) for problem F to fathom the nodes of the enumeration tree. The premature termination of method SCRG-LF only occurs in the following situation: In procedure CSA with post-pricing, we impose that the size of the set $\mathcal{P}$ of partial paths explored cannot exceed an a priori defined limit Λ^3, so that if $|\mathcal{P}|$ becomes greater than Λ^3, and $\hat{R}$ is empty, method SCRG-LF can terminate prematurely.

After the computation of the upper bound at the root node of the enumeration tree, we use the set $\tilde{R}$ of rotations, obtained from method SCRG-LF, to define a restricted MIP formulation for the LSND by substituting the rotation set R in formulation F with $\tilde{R}$. The restricted MIP formulation is then solved to compute an initial lower bound on the optimal solution of the LSND using a general-purpose MIP solver.

5.2. Computing a valid upper bound under premature termination of SCRG-LF

As mentioned in §5.1, in our BP-based exact method, when method SCRG-LF terminates prematurely with a DLF solution Π that has not been proved to be feasible, we need to use this infeasible DLF solution Π to compute a valid upper bound on the optimal solution value of problem F. The computation can be executed at any iteration of our method SCRG-LF (Algorithm 1), aiming to construct a feasible DLF solution derived from Π obtained at Step 3 of Algorithm 1. More precisely, it attempts to compute new values to replace $[\tilde{\delta}_s]$ of the optimal $DRLF$ solution $\tilde{\Pi}$ used in the construction of Π, so that the dual constraints (3f) are satisfied for all $r \in R \setminus \tilde{R}$.

To achieve this, for any iteration of method SCRG-LF, consider the values of auxiliary variables contained in $\hat{\Pi} = (\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\beta}})$ which are used in the construction of Π at Step 3 of Algorithm 1. As explained in §4.3, solution $\hat{\Pi}$ consists of auxiliary variables $[\hat{\lambda}^k_{\sigma,i} : k \in K, \sigma \in C, i \in V]$, $[\hat{\pi}_{\sigma,a} : \sigma \in C, a \in A_V]$, and $[\hat{\beta}^k_{\sigma,a} : k \in K, \sigma \in C, a \in A_V]$, satisfying the dual constraints (3c)–(3e) for all $r \in R \setminus \tilde{R}$. Consider any rotation set R'' such that each rotation $r \in R''$ does not pass the post-pricing phase of method SCRG-LF, i.e., $\Delta^*(\sigma_r, \gamma_r)$ defined in (9) is non-negative. With R'', for each $\omega \in \{1, 2, ..., n_{s(\sigma)}\}$ and $\sigma \in C$, we define $\Delta(R'', \omega, \sigma)$ as follows, so that $-\Delta(R'', \omega, \sigma)$ equals the maximum reduced cost, as defined in pricing problem (5), of all the rotations $r \in R \setminus R''$ with $n(\sigma_r, \gamma_r) = \omega$ and $\sigma_r = \sigma$:

$$\Delta(R'', \omega, \sigma) = \min_{\substack{r \in R \setminus R'' : \sigma_r = \sigma, \\ n(\sigma_r, \gamma_r) = \omega}} \left\{ w(\sigma_r, \gamma_r) + n(\sigma_r, \gamma_r)\tilde{\delta}_{s(\sigma_r)} - \sum_{a \in A_V(\gamma_r)} m(\sigma_r)\hat{\pi}_{\sigma_r,a} - \sum_{k \in K} \sum_{a \in A_V(\gamma_r)} q_k \hat{\beta}^k_{\sigma_r,a} \right\}, \tag{10}$$

where $\Delta(R'', \omega, \sigma) = +\infty$ if no rotation r in $R \setminus R''$ satisfies $\sigma_r = \sigma$ and $n(\sigma_r, \gamma_r) = \omega$.

By Proposition 3 below, any lower bound $\Delta^-(R'', \omega, \sigma)$ on $\Delta(R'', \omega, \sigma)$ for each $\omega \in \{1, 2, ..., n_{s(\sigma)}\}$ and $\sigma \in C$ can be used to compute new values to replace $[\tilde{\delta}_s]$ of the optimal $DRLF$ solution $\tilde{\Pi}$ for the construction of Π, to obtain an upper bound on the optimal solution value of problem F.

PROPOSITION 3. *Given any set R'' of rotations r satisfying $\Delta^*(\sigma_r, \gamma_r) \geq 0$, let $\Delta^-(R'', \omega, \sigma)$ be any lower bound on $\Delta(R'', \omega, \sigma)$ defined in (10) for each $\omega = \{1, 2, ..., n_{s(\sigma)}\}$ and $\sigma \in C$. Consider the values of dual variables, $[\tilde{\delta}_s : s \in S]$ and $[\tilde{\varphi}_k : k \in K]$, given in the optimal $DRLF$ solution $\tilde{\Pi}$ obtained at Step 2 of Algorithm 1. Define a vector $[\delta^-_s : s \in S]$ where $\delta^-_s = \tilde{\delta}_s - \min_{\omega \in \{1,2,...,n_s\}; \sigma \in C : s(\sigma) = s} \{\min\{0, \Delta^-(R'', \omega, \sigma)\}/\omega\}$ for each $s \in S$. Then, $(\sum_{s \in S} n_s \delta^-_s + \sum_{k \in K} q_k \tilde{\varphi}_k)$ provides an upper bound on the optimal solution value of problem F.*

Proof. See §EC.1.4 of the e-companion to this paper. □

To apply Proposition 3, we need to compute set R'' and lower bounds $\Delta^-(R'', \omega, \sigma)$ on $\Delta(R'', \omega, \sigma)$ for all $\omega \in \{1, 2, ..., n_{s(\sigma)}\}$ and $\sigma \in C$. In our study, we compute R'' and $\Delta^-(R'', \omega, \sigma)$ by using a row generation-based procedure described in §EC.3 of the e-companion.

6. Numerical experiments

This section reports on the computational results of the different methods described in this paper. All algorithms were implemented in C language, and CPLEX 12.8 (IBM 2020) was used as the

LP and MIP solver. The experiments were performed on an Intel Core i7 (3.2 GHz) Desktop PC with 16 GB RAM. Based on the results of preliminary experiments, which were conducted to identify good parameter settings, we used $\Lambda^1 = 50$, $\Lambda^2 = 3$, and $\Lambda^3 = 5000$ for the pricing algorithm described in §4.4 and for the BP-based algorithm presented in §5.1.

Our experiments were conducted on the Standard LSND and the General LSND, the two main variants of the LSND studied in the literature. In §6.1, we describe the test instances used in our experiments. In §6.2, we examine the computational results of the methods used to solve relaxation LF. In §6.3 and §6.4, we compare the results of the methods used to solve problem F of the Standard LSND and the General LSND, respectively. Through these experiments, we also evaluate the benefits of explicitly accounting for transshipment costs and the benefits of endogenizing decisions on sailing speed and service frequency.

6.1. Test instances

The instances used in our experiments are based on two datasets, Baltic (with 12 ports, 22 origin-and-destination pairs for demands, and six ships of two types) and WAF (with 20 ports, 37 origin-and-destination pairs for demands, and 42 ships of two types), these being obtained from the LINER-LIB-2012 benchmark set for the LSND provided by Brouer et al. (2014a). The size of the instances based on the selected datasets is comparable to or larger than the sizes of instances employed in the existing literature on exact methods for the LSND with transshipment costs (see, e.g., Reinhardt and Pisinger 2012, Plum et al. 2014b, Thun et al. 2017), and it is also comparable to the size of instances employed in the study of Agarwal and Ergun (2008) on the CG-based method for the LSND without transshipment cost.

In addition to *benchmark instances* (i.e., six instances taken directly from the LINER-LIB-2012 set), we also considered new *randomly generated instances* based on the LINER-LIB-2012 set to have more coverage on instance scales. Based on the LINER-LIB-2012 set, we generated 20 new random instances that include different numbers of ports (ranging from 7 to 20), different numbers of origin-destination pairs for the demands (ranging from 8 to 37), and different numbers of ships that are available to deploy. The instances are grouped into three test sets for the sake of the solution tests. Set-A consists of the 12 smaller random instances, including the port number ranging from 7 to 12. Set-B consists of the other eight larger random instances, including port numbers ranging from 12 to 20. As previously mentioned, set-C consists of the six benchmark

Table 1 Parameter setting of the test instances.

Name	$\|H\|$	$\|K\|$	$\|S\|$	$\sum_{s\in S} n_s$	$\psi_h^{(1)}$	$\psi_a^{(2)}$	$\psi^{(3)}$	$\psi^{(4)}$	$\psi^{(5)}$
Set-A									
p7-d8-s6	7	8	2	6	1	1	5	0	$+\infty$
p7-d12-s6	7	12	2	6	1	1	5	0	$+\infty$
p8-d11-s6	8	11	2	6	1	1	5	0	$+\infty$
p8-d14-s6	8	14	2	6	1	1	5	0	$+\infty$
p9-d12-s6	9	12	2	6	1	1	5	0	$+\infty$
p9-d16-s6	9	16	2	6	1	1	5	0	$+\infty$
p10-d14-s6	10	14	2	6	1	1	5	0	$+\infty$
p10-d18-s6	10	18	2	6	1	1	5	0	$+\infty$
p11-d16-s6	11	16	2	6	1	1	5	0	$+\infty$
p11-d20-s6	11	20	2	6	1	1	5	0	$+\infty$
p12-d18-s6	12	18	2	6	1	1	5	0	$+\infty$
p12-d22-s6	12	22	2	6	1	1	5	0	$+\infty$
Set-B									
p12-d16-s10	12	16	2	10	1, 2	1	10	1	$+\infty$
p12-d20-s15	12	20	2	15	1	1	$+\infty$	0	$+\infty$
p14-d22-s12	14	22	2	12	1, 2	1	7	2	$+\infty$
p14-d26-s20	14	26	2	20	1	1	11	0	$+\infty$
p16-d26-s18	16	26	2	18	1, 2	1	$+\infty$	2	$+\infty$
p16-d30-s25	16	30	2	25	1, 2	1	10	1	$+\infty$
p18-d28-s32	18	28	2	32	1, 2	1	12	3	$+\infty$
p18-d32-s28	18	32	2	28	1, 2	1	$+\infty$	1	$+\infty$
Set-C									
p12-d22-s5-low	12	22	2	5	2	1	$+\infty$	1	$+\infty$
p12-d22-s6-base	12	22	2	6	2	1	$+\infty$	1	$+\infty$
p12-d22-s7-high	12	22	2	7	2	1	$+\infty$	1	$+\infty$
p20-d37-s33-low	20	37	2	33	2	1	$+\infty$	1	$+\infty$
p20-d37-s42-base	20	37	2	42	2	1	$+\infty$	1	$+\infty$
p20-d37-s51-high	20	37	2	51	2	1	$+\infty$	1	$+\infty$

instances of Brouer et al. (2014a). All the data sets, problem instances, and source codes can be found in the e-companion to this paper.

The test instance in Set-A and Set-B is named by a string “pX-dY-sZ”, implying that the instance involves “X” ports, “Y” origin-destination pairs for the demand, and a total of “Z” available ships. Moreover, the benchmark instance in Set-C is named by a string “pX-dY-sZ-I”, containing an additional scenario indicator $I \in \{low, base, high\}$ to indicate different market conditions. More information on the scenarios can be found in Brouer et al. (2014a).

Table 1 shows the parameter setting of the test instances. It includes the number of ports (“$|H|$”), the number of demands (“$|K|$”), the number of ship types (“$|S|$”), and the total number of available ships (“$\sum_{s\in S} n_s$”). It also includes the following parameters for defining feasible rotations: (i) $\psi_h^{(1)}$, the maximum number of calls for port h; (ii) $\psi_a^{(2)}$, the maximum number of passes on voyage arc

a; (iii) $\psi^{(3)}$, the maximum total number of port calls; (iv) $\psi^{(4)}$, the maximum number of butterfly ports; and (v) $\psi^{(5)}$, the maximum number of weeks for the round trip duration. Their settings accord with the actual practice in liner shipping (see §EC.8.1 of the e-companion for details).

Our additional experiments, presented in §EC.8.5 of the e-companion, show that the computational performance in solving the LSND is affected by the above problem parameters. For example, the difficulty of the LSND grows with the increase in the number of ports, the number of demands, and the total number of available ships. The problems allowing non-simple route structures (such as those allowing butterfly ports) are more challenging than those not allowing them.

The fuel oil price is set to 600 for all the test instances, and the port stay time is set to one day. The unit penalty costs for unsatisfied demands are zero for the randomly generated instances in Set-A and Set-B. For the benchmark instances in Set-C, such penalty costs are set to zero for the Standard LSND and to 1000 for the General LSND, to be consistent with the settings in the literature (Plum et al. 2014b, Brouer et al. 2014a,b).

6.2. Results of solving relaxation LF

To show the efficiency of newly proposed method SCRG-LF (see §4) for solving relaxation LF of the LSND, we have compared method SCRG-LF with method SCRG-LF-MBB based on Muter et al. (2013) (see §EC.6 of the e-companion). To demonstrate the effectiveness of the speed-up techniques used in method SCRG-LF, we have also compared two variants of method SCRG-LF, including one using the sequential approach and the other using the LP-based approach for constructing the DLF solution Π. We have compared them with method SCRG-LF-MBB under two situations, where the post-pricing phase (see §4.4.2) is disabled and enabled, respectively.

The comparison results are shown in Table 2 and Table 3 for situations with and without post-pricing phase in the SCRG methods, respectively. For these comparisons, we considered the set of randomly generated instances of Standard LSND in Set-A, and we imposed a time limit of 300 seconds on each solution method. Our method, SCRG-LF, using the LP-based approach, solves all instances to optimality even without the post-pricing phase. Accordingly, the optimal solution value of relaxation LF is shown in "Obj" under the "LP-based Approach" of Tables 2 and 3. For the other two SCRG methods, the percentage gaps between the solution values of their final solutions obtained and the optimal solution values of relaxation LF are shown in "ΔObj($\%$)" under "SCRG-LF-MBB" and "Sequential Approach" of Tables 2 and 3, and negative gaps indicate that their final solutions are not optimal. For each instance and each SCRG method, Tables 2 and 3 also

Table 2 Solving relaxation LF of Standard LSND by SCRG methods without post-pricing phase: comparison between method SCRG-LF-MBB based on Muter et al. (2013) and our method SCRG-LF.

					SCRG-LF							
	SCRG-LF-MBB				Sequential Approach				LP-based Approach			
Name	ΔObj(%)	Col	Iter	Time(s)	ΔObj(%)	Col	Iter	Time(s)	Obj	Col	Iter	Time(s)
p7-d8-s6	0.00	59	11	0.4	0.00	68	14	0.5	46211.6	39	4	0.2
p7-d12-s6	0.00	166	47	4.2	0.00	791	254	105.7	412423.3	61	12	0.8
p8-d11-s6	0.00	684	215	61.5	0.00	1341	432	300.3	316481.2	77	13	1.6
p8-d14-s6	0.00	101	23	1.9	0.00	374	114	21.4	181231.9	55	7	0.9
p9-d12-s6	-0.53	1139	360	300.8	-0.20	1093	343	300.9	275965.0	143	33	13.4
p9-d16-s6	0.00	1090	342	303.1	-0.85	694	210	302.9	690803.2	113	20	4.2
p10-d14-s6	0.00	904	279	203.9	0.00	1126	351	302.9	285826.3	107	15	4.1
p10-d18-s6	0.00	949	292	301.4	-2.29	607	178	303.2	763932.9	114	18	5.4
p11-d16-s6	0.00	1126	346	301.7	0.00	1000	303	301.2	431630.1	147	22	9.72
p11-d20-s6	-1.19	624	177	300.5	-4.94	561	156	305.4	770558.8	153	25	14.68
p12-d18-s6	0.00	921	270	300.9	-2.92	660	182	300.1	708864.6	141	13	9.48
p12-d22-s6	-1.96	540	142	303.1	-7.84	572	153	300.0	770558.8	161	21	18.1

Notes: values in "Obj" are the optimal solution values of LF obtained by method SCRG-LF with the LP-based approach.

Table 3 Solving relaxation LF of Standard LSND by SCRG methods with post-pricing phase: comparison between method SCRG-LF-MBB based on Muter et al. (2013) and our method SCRG-LF.

						SCRG-LF									
	SCRG-LF-MBB					Sequential Approach					LP-based Approach				
Name	ΔObj(%)	Col	Iter	Rej	Time(s)	ΔObj(%)	Col	Iter	Rej	Time(s)	Obj	Col	Iter	Rej	Time(s)
p7-d8-s6	0.00	39	3	340.7	0.4	0.00	39	4	48.3	0.2	46211.6	39	4	5.3	0.2
p7-d12-s6	0.00	58	10	256.7	1.3	0.00	57	10	779.7	2.9	412423.3	57	11	27.5	0.7
p8-d11-s6	0.00	69	10	1211.7	4.1	0.00	67	9	1330.8	4.1	316481.2	67	9	22.9	1.5
p8-d14-s6	0.00	52	7	240.7	1.2	0.00	54	7	356.1	1.5	181231.9	54	7	12.7	1.0
p9-d12-s6	0.00	97	14	1725.4	9.6	0.00	105	16	2387.0	14.8	275965.0	83	10	60.6	4.6
p9-d16-s6	0.00	88	10	1915.4	8.3	0.00	86	11	2296.9	10.4	690803.2	90	12	66.8	4.4
p10-d14-s6	0.00	88	8	1267.5	12.8	0.00	94	10	1898.8	14.9	285826.3	90	10	50.5	4.4
p10-d18-s6	0.00	103	13	2231.7	16.7	0.00	102	13	2487.8	20.4	763932.9	98	12	57.6	6.1
p11-d16-s6	0.00	109	9	2469.7	47.8	0.00	109	11	4140.2	47.2	431630.1	107	9	60.1	7.5
p11-d20-s6	0.00	131	17	2924.8	62.2	0.00	123	14	3246.6	56.6	770558.8	128	16	69.9	11.6
p12-d18-s6	0.00	135	12	5663.8	67.8	0.00	133	11	4798.5	86.3	708864.6	128	9	53.8	10.0
p12-d22-s6	0.00	147	14	2871.7	65.5	0.00	152	17	3774.9	80.6	770558.8	146	15	74.0	15.3

Notes: values in "Obj" are the optimal solution values of LF obtained by method SCRG-LF with the LP-based approach.

show the number of rotations generated ("Col"), the total computing time in seconds ("Time(s)"), and the number of iterations carried out ("Iter"). Table 3 reports the average number of columns ("Rej") generated by the pricing algorithm but rejected by the post-pricing phase.

From Table 2, it can be seen that when the post-pricing phase is disabled, our new method SCRG-LF with the LP-based approach significantly outperforms its variant with the sequential approach and the method SCRG-LF-MBB. It optimally solves all instances by consuming a much shorter time than the imposed time limit. At the same time, the other two versions of the SCRG method cannot solve some instances with nine or more ports to optimality within the imposed time limit. This demonstrates the efficiency of our new method SCRG-LF and the effectiveness of utilizing the LP-based approach for speed-up of method SCRG-LF (which is due to the benefit of using non-dominated values of dual variables to construct the DLF solution Π).

Column “Col” under the LP-based approach in Table 2 shows that when the post-pricing phase is disabled, the total number of columns (or rotations) generated by method SCRG-LF-MBB does not grow as significantly as the other two versions of the SCRG method, concerning the growth of the instance size. This indicates that unlike the other two versions of the SCRG method, method SCRG-LF with the LP-based approach is hardly affected by the degeneracy, which is known to be *tailing-off effect* in applications of the standard simplex-based column generation method (Lübbecke and Desrosiers 2005).

For the situation where the post-pricing phase is enabled, the results in Table 3 indicate that the method SCRG-LF with the LP-based approach still significantly outperforms its variant with the sequential approach and the method SCRG-LF-MBB. Moreover, by comparing the results in Table 2 and Table 3, it can be seen that the post-pricing phase improves all the three versions of the SCRG method. This demonstrates the particular usefulness of the post-pricing phase in speeding up the SCRG method to solve relaxation LF, not only for our newly proposed method SCRG-LF but also for those adapted from the existing SCRG-based methods, such as SCRG-LF-MBB. In particular, method SCRG-LF with the LP-based approach also solves all the instances optimally and is, on average, faster than SCRG-LF without post-pricing. In contrast, methods SCRG-LF-MBB and SCRG-LF with the sequential approach can solve all instances optimally within the imposed time limit. For method SCRG-LF with the LP-based approach, although the computational advantage from post-pricing seems to be minor in solving a single LF, such an advantage can be accumulated, and it results in significant computational benefits when more LF problems are computed in the branch-and-price search. This motivates us to implement the SCRG with the LP-based approach and the post-pricing phase in our BP-based exact method.

From columns “Col” and “Iter” of Table 3, we can see that by enabling the post-pricing phase, both the numbers of rotations generated and the numbers of iterations needed by method SCRG-LF-MBB and by method SCRG-LF with the sequential approach are significantly reduced. From columns “Rej” and “Time(s)” of Table 3, it can be seen that for method SCRG-LF-MBB and method SCRG-LF with the sequential approach, the post-pricing phase rejects quite several columns that are generated by the pricing algorithm. These rejections occur at the cost of more computation time in solving relaxation LF. In comparison, method SCRG-LF with the LP-based approach performs more efficiently, as the number of rejections of such unnecessary columns is much smaller than those of the other two versions of the SCRG method.

Furthermore, we have also compared the new method SCRG-LF with other methods that can be used to solve relaxations of the LSND, including the CPLEX LP solver applied to relaxation LF, method XLMX based on the approaches of Xia et al. (2015) and a relaxation of LF (see §EC.4 of the e-companion), the method based on model TAC adapted from the work of Thun et al. (2017) and an LP-relaxation of an alternative set-partitioning-like model (see §EC.7 of the e-companion). For instances of the Standard LSND in Set-A, all feasible rotations can be generated a priori, thus allowing the application of the CPLEX LP solver to solve relaxation LF. According to the comparison results (see §EC.8.2 of the e-companion for details), our new method SCRG-LF significantly outperforms method XLMX, method TAC, and the CPLEX LP solver. The comparison results show that method SCRG-LF is the only method capable of optimally solving all instances within the imposed time limit, and its running time is much shorter. The upper bounds provided by relaxation LF are also significantly tighter than those provided by method XLMX and TAC.

Since relaxation LF provides the best LP-relaxation for the LSND, we have implemented method SCRG-LF with the LP-based approach and the post-pricing phase in our exact method, which, as shown above in Table 2 and 3, performs the best in solving relaxation LF,

6.3. Results of solving problem F of Standard LSND

This section reports on the results of our BP-based exact method described in §5, hereafter called method BP, to evaluate its performance in solving problem F of Standard LSND.

First, we compare method BP with the MIP solver of CPLEX on the set of randomly generated instances from Set-A. In our experiment for this, a time limit of two hours was imposed on both the CPLEX solver and method BP. The time for method BP to compute the lower bound at root node was limited to ten minutes. The CPLEX solver was applied to the MIP formulation of problem F based on the complete set of rotations, which had to be explicitly enumerated. We only ran the CPLEX solver on Set-A instances since the MIP formulations are too large for larger instances.

The obtained results are given in Table 4. For each instance and each method, Table 4 shows the solution values (profits) of the obtained solutions ("Obj"), the percentage gap of the best upper bound and the best lower bound found by the method ("Gap(%)"), the total number of rotations generated ("Col"), and the total computing time in seconds ("Time(s)"). For method BP, the table also shows the solution obtained at the root node of the enumeration tree (under "Root Node Solution"), as well as the solution finally obtained (under "BP").

Table 4 Solving problem F of Standard LSND: comparison between CPLEX MIP solver and our exact method BP on randomly generated instances in Set-A.

	CPLEX MIP Solver			Root Node Solution				BP				
Name	Obj	Gap(%)	Time(s)	Obj	Gap(%)	Col	Time(s)	Obj	Gap(%)	Col	Node	Time(s)
p7-d8-s6	45894	0.00	11.5	45894	0.69	39	0.2	45894	0.00	41	2	0.6
p7-d12-s6	412325	0.00	605.2	412325	0.02	57	0.1	412325	0.00	57	1	0.2
p8-d11-s6	315504	0.00	505.1	315503	0.31	67	1.1	315503	0.00	87	11	18.2
p8-d14-s6	177762	0.00	2600.0	177762	1.91	54	1.7	177762	0.00	58	8	11.7
p9-d12-s6	215629	88.81	7200.7	248983	9.78	83	8.3	248983	0.00	194	67	212.9
p9-d16-s6	676285	16.09	7200.6	679974	1.57	90	8.8	679974	0.00	294	277	778.8
p10-d14-s6	0	100	7200.5	278054	2.72	90	6.9	278054	0.00	111	9	36.8
p10-d18-s6	440940	104.14	7200.8	708652	7.24	98	15.1	717781	0.00	210	85	410.3
p11-d16-s6	0	100	7201.4	431630	0.00	107	11.0	431630	0.00	107	1	11.0
p11-d20-s6	0	100	7201.8	725766	5.81	128	13.1	725782	0.00	446	497	4792.0
p12-d18-s6	n/a	n/a	106.2†	681659	3.84	128	16.8	681659	0.00	227	56	485.9
p12-d22-s6	n/a	n/a	87.8†	721497	6.37	146	73.0	725782	0.00	486	507	6602.9

Notes: "†" indicates the "out-of-memory" status. "n/a" indicates that no solution to LF is obtained. When the solution value ("Obj") equals zero, only a trivial feasible solution with no rotation operated is obtained, resulting in a 100% gap between the best upper bound and the best lower bound found.

Table 5 Solving problem F of Standard LSND by exact method BP on randomly generated instances in Set-B.

	Root Node Solution				BP				
Name	Obj	Gap(%)	Col	Time(s)	Obj	Gap(%)	Col	Node	Time(s)
p12-d16-s10	2106991	10.65	145	105.5	2106991	3.88	411	239	7206.3
p12-d20-s15	2697833	6.39	110	112.0	2697833	3.04	347	178	7203.7
p14-d22-s12	2340210	12.59	170	136.3	2340210	7.24	370	101	7204.4
p14-d26-s20	3503897	5.97	166	249.4	3554237	1.49	313	46	7204.4
p16-d26-s18	3236887	18.45	252	1201.7	3236887	14.16	373	11	7314.6
p16-d30-s25	3830746	14.15	244	1000.9	3834203	9.50	413	41	7385.0
p18-d28-s32	4811084	23.43	300	1475.5	4811084	16.38	432	7	7784.7
p18-d32-s28	5322757	13.46	307	2606.3	5322757	11.91	504	9	7926.9

As shown in Table 4, method BP solves all instances to optimality within the time limit, significantly outperforming the CPLEX MIP solver, which fails to solve more than half of the instances to optimality. For almost half of the instances, the CPLEX solver produces no solution or only a trivial feasible solution with no rotation being operated. This also highlights that the LSND represents a very challenging optimization problem.

To further attest to the effectiveness of method BP in solving difficult LSND instances, we also tested BP on the set of randomly generated instances in Set-B with increased problem scales and relaxed rotation settings and on the set of six Baltic and WAF benchmark instances from the LINERLIB-2012 in Set-C, for which computational results have been reported in the literature. Table 5 gives the results of method BP on the Set-B instances using the same notation introduced for Table 4. The results show that none of the instances has been optimally solved by method BP within the imposed time limit, thus representing difficult LSND instances for method BP. Indeed, for these instances, the time to compute the root node solution ranges from 2 to 45 minutes. This is mainly due to the increased number of ports, ships, and origin-destination demand pairs in the mentioned instances and the relaxed settings for defining feasible rotations, which slow down the solution of problem RLF and the pricing problem. However, despite their difficulty, many instances of such scales are solved within two hours with high-quality sub-optimal solutions by the BP method. From columns "Gap(%)" in Table 4 and Table 5, we can see that the average percentage gaps of the lower bounds computed at the root node of method BP are 3.36% and 13.14% for instances of Set-A and Set-B, respectively. The enumeration phase of method BP improves the lower bounds for two instances of Set-A and two instances of Set-B, with average percentage improvements equal to 0.94% and 0.76%, respectively. For Set-A instances, method BP closes the optimality gaps of all the instances, whereas for Set-B instances, the enumeration phase significantly reduces the average optimality gap from 13.14% to 8.45%.

Concerning the six Baltic and WAF benchmark instances in Set-C, Plum et al. (2014b) computed both upper and lower bounds based on an MIP model, but with an upper limit imposed on the number of rotations, and with all ports being allowed more than one visit. Therefore, only the lower bounds computed by Plum et al. can be compared with the results of our method BP. The MIP model of Plum et al. was run on an Intel i5 (2.53 GHz) equipped with 3 GB of RAM and was solved using the CPLEX 12.2 MIP solver, with a time limit of one hour and three hours for the Baltic and WAF groups of instances, respectively. According to the SuperPi (1M) benchmark (http://www.superpi.net/), our machine is about 10% faster than that used by Plum et al. (2014b).

The six instances considered in Plum et al. (2014b) have also been tested by a heuristic method proposed by Brouer et al. (2014a), but under different settings of the LSND problem. In particular, Brouer et al. (2014a) took into account also biweekly services (i.e., once per two weeks) and speed optimization in their LSND problem. For the sake of comparison, we, therefore, implemented and adapted their heuristic method to compute valid lower bounds for the six instances considered under our setting of the Standard LSND problem. The heuristic method of Brouer et al. (2014a) was inspired by the approach presented in Álvarez (2009). It applies a tabu search on a set of rotations iteratively generated by an MIP neighborhood. It includes heuristic measures on cargo composition and is designed to generate simple and on-simple rotations of high quality.

The results on the six benchmark instances of Set-C are reported in Table 6. We use PPS and BAPPS to denote the methods used in Plum et al. (2014b) and our implementation of the heuristic method proposed by Brouer et al. (2014a), respectively. We imposed time limits of one hour and three hours for the Baltic and WAF groups, respectively, on both methods, BAPPS and BP. For each instance and each method, Table 6 shows the best lower bound obtained ("Obj") and the computing

Table 6 Solving problem F of Standard LSND: comparison of method PPS of Plum et al. (2014b), method BAPPS based on Brouer et al. (2014a), and our exact method BP on the benchmark instances in Set-C.

	PPS		BAPPS		BP				
Name	Obj	Time(s)	Obj	Time(s)	Obj	Gap(%)	Col	Node	Time(s)
p12-d22-s5-low	427485	3600	442306	2614	662767	5.58	391	60	3600
p12-d22-s6-base	408771	3600	505306	3002	748618	2.76	378	43	3600
p12-d22-s7-high	636152	3600	571954	3379	790618	2.28	322	45	3600
p20-d37-s33-low	1940817	10800	4195167	10800	5175561	14.91	372	5	10800
p20-d37-s42-base	3372618	10800	4525371	10800	5762190	17.21	413	7	10800
p20-d37-s51-high	3899767	10800	4630588	10800	6089836	14.84	418	7	10800

time in seconds (“Time(s)”). For method BP, Table 6 also gives the percentage gap (“Gap(%)”) of its best upper bound and the best lower bound found, the number of rotations generated (“Col”), and the total number of nodes of the enumeration tree (“Node”).

Table 6 shows that none of the six instances can be solved optimally by either PPS or BP within the imposed time limit, highlighting the difficulty of solving the LSND to optimality. However, the differences between the solution values computed by BP, PPS, and BAPPS show that the quality of the solutions obtained can be greatly improved by method BP and that method BP can provide, on average, high-quality solutions within the imposed time limit.

Moreover, we have compared the network design solutions produced by the BP, PPS, and BAPPS methods on the six benchmark instances in Set-C. The results, illustrated in Table EC.3 of the e-companion, show that the BP solution is advantageous for utilizing the current fleet of ships to serve more demand pairs. The resulting BP solutions involve more rotations and port calls for all these six benchmark instances, and the average capacity utilization ratios over the voyage arcs are the highest for five of these six benchmark instances.

We have also investigated the benefits of considering the transshipment costs in the Standard LSND. As illustrated in §EC.8.3 of the e-companion, the solutions that explicitly account for transshipment costs can lead to significantly improved solutions with higher total profits. For the six Baltic and WAF benchmark instances, a total profit improvement from a minimum of about 3% to a maximum of about 23% has been achieved by considering the transshipment costs.

6.4. Results of solving problem F of General LSND

To evaluate the performance of our method BP in solving the General LSND problem, we tested method BP on the six benchmark instances in Set-C. We compare its results with the benchmark results available in the literature, including Plum et al. (2014b), Brouer et al. (2014a), and Brouer

Table 7 Solving problem F of General LSND: comparison of method PPS of Plum et al. (2014b) and our exact method BP on the benchmark instances considering the optimization of service frequency.

	PPS		BP				
Name	Obj	Time(s)	Obj	Gap(%)	Col	Node	Time(s)
p12-d22-s5-low	-265117	3600	210433	46.46	353	11	3600
p12-d22-s6-base	-134687	3600	461384	34.16	325	10	3600
p12-d22-s7-high	-183348	3600	646154	16.24	261	3	3600
p20-d37-s33-low	411317	10800	5114795	19.65	510	8	10800
p20-d37-s42-base	1059352	10800	5824197	14.91	456	6	10800
p20-d37-s51-high	1281583	10800	6264127	13.94	449	5	10800

et al. (2014b), which were produced under different configuration settings concerning the optimization of the sailing speed and service frequency for the LSND. Accordingly, we compare our new exact method BP with the exact method PPS of Plum et al. (2014b) on instances considering optimization of the service frequency, with the heuristic method BDP of Brouer et al. (2014b) on instances considering optimization of the sailing speed, and with the heuristic method BAPPS of Brouer et al. (2014a) on instances considering optimization of both the sailing speed and service frequency. Moreover, using the experiment settings considered in the literature of these benchmark methods, we set the penalty cost for rejecting a unit of cargo equal to 1000. As the optimization of the sailing speed and service frequency is considered, the size of the rotation configuration set is enlarged, which inevitably leads to computational challenges for method BP. To guarantee sufficient explorations on the lower and upper bounds, we increased the time limit for method BP to compute the lower bound at the root node to one hour and increased the total time limit of method BP to three hours.

Table 7 compares the performance of our method BP and method PPS on the six benchmark instances of the General LSND where ships' sailing speeds are predetermined to adopt their design speeds, but service frequencies, which can be either weekly (once per week) or biweekly (once per two weeks), need to be optimized. The table's notation of columns is the same as Table 6. Table 7 shows that although neither method solves these General LSND instances to optimality within the imposed time limit, our method BP produces significantly better solutions than method PPS. For the first three instances, solutions of method PPS all have negative profits, while solutions of method BP all have positive profits. For the other three instances, solutions of method BP improve solutions of method PPS by about four to ten times in the solution values.

Table 8 and Table 9 compare our new exact method BP with heuristic methods BDP and BAPPS on the six benchmark instances of General LSND where the sailing speeds, which belong to some given speed sets, need to be optimized. According to Brouer et al. (2014b), the benchmark results

Table 8 Solving problem F of General LSND: comparison of method BDP of Brouer et al. (2014b) and our exact method BP on the benchmark instances considering the optimization of sailing speed.

	BDP	BP				
Name	Obj	Obj	Ub	ΔObj(%)	Obj$^+$	ΔObj$^+$(%)
p12-d22-b-low	-137369	-145898	10226.0	-6.21	-134412	2.15
p12-d22-b-base	246605	251169	331169.0	1.85	266510	8.07
p12-d22-b-high	430593	415516	500476.4	-3.50	430675	0.02
p20-d37-w-low	4578177	4481997	5349587.5	-2.10	4599098	0.46
p20-d37-w-base	5590381	5922462	6346280.5	5.94	5934951	6.16
p20-d37-w-high	6264469	6496608	6836831.5	3.71	6514383	3.99

produced by method BDP were under the setting where the speed sets were continuous intervals $[10,14]$ and $[10,17]$ for the two types of ships. According to Brouer et al. (2014a), the benchmark results produced by method BAPPS were under the setting where the speed sets were discrete sets $U_1 = \{10, 11, ..., 14\}$ and $U_2 = \{10, 11, ..., 17\}$ for the two types of the ships. For our method BP, we followed Brouer et al. (2014a) to adopt $U_1 = \{10, 11, ..., 14\}$ and $U_2 = \{10, 11, ..., 17\}$ as the discrete speed sets for the two types of the ships. Moreover, for a solution produced by our method BP, the discrete sailing speed obtained for each rotation can be adjusted to a continuous sailing speed that equals the ratio of the length and the duration of the rotation. This adjustment can reduce time slack on a rotation, possibly slow down the sailing speed, and improve the solution value due to fuel cost savings by slow steaming. We, therefore, also examine such improved solutions of method BP with speed adjustment applied in our experiments.

Table 8 shows the results of our new exact method BP and the heuristic method BDP, where sailing speeds need to be optimized, but service frequencies are fixed to be weekly. For method BDP, column "Obj" reports the solution (profit) values of its best-known Baltic and WAF solutions produced, which are available at https://github.com/blof/LINERLIB/. For our method BP, in addition to the solution values ("Obj") obtained, we report their improvement percentages ("ΔObj($\%$)") against the solution values of the solutions produced by method BDP. We also report the solution values ("Obj$^+$") for solutions of the method BP with speed adjustment applied and their improvement percentages ("ΔObj$^+$") against the solution values of the solutions produced by method BDP. We also report the upper bound ("Ub") produced by the method BP.

From Table 8, we can see that although these instances are not solved to optimality, our method BP with speed adjustment outperforms method BDP in producing better solutions, with up to 8.07% improvements. With the speed adjustment applied, solutions obtained by BP are significantly improved, particularly for Baltic instances (i.e., the first three instances). For the WAF instances, the strength of our exact method BP over the heuristic method BDP is more remarkable,

Table 9 Solving problem F of General LSND: comparison of method BAPPS of Brouer et al. (2014a) and our exact method BP on the benchmark instances considering the optimization of sailing speed and service frequency.

	BAPPS	BP				
Name	Obj	Obj	Ub	ΔObj(%)	Obj$^+$	ΔObj$^+$(%)
p12-d22-b-low	235044	218605	362067.9	-6.99	234172	-0.37
p12-d22-b-base	325305	458775	648536.0	41.03	472226	45.16
p12-d22-b-high	609700	702692	788281.6	15.25	710156	16.48
p20-d37-w-low	4486261	5117998	5957101.0	14.08	5196478	15.83
p20-d37-w-base	5565389	6310642	6869302.0	13.39	6314192	13.45
p20-d37-w-high	6220044	6940677	7350122.5	11.59	6940677	11.59

further validating the effectiveness of our exact method in searching profitable solutions against existing heuristic methods for instances of such scales.

Table 9 shows the results of our new exact method BP and the heuristic method BAPPS where both sailing speeds and service frequencies must be optimized. The table's notation of columns is the same as Table 8. The solution values reported for the best-known solutions obtained by method BAPPS of Brouer et al. (2014a) are obtained from Plum et al. (2014b). The results in Table 9 show that although no instances are solved to optimality, our new exact method BP produced significantly better solutions than the heuristic method BAPPS for the five benchmark instances out of the six. When speed adjustments are not applied, 11.59% to 41.01% improvements are achieved for these five instances. When the speed adjustments are applied, the improvements are further increased slightly for these five instances, and for the remaining instance (p12-d22-b-low), the solution quality of method BP has significantly improved, with the percentage gap of its solution value against that of the solution of method BAPP improved from -6.99% to -0.37%. These again demonstrate the effectiveness of the speed adjustment for method BP, as well as the strength of our exact method BP against the existing heuristic methods in solving General LSND for instances of such scales. Moreover, by comparing the solution values of our method BP shown in Table 7, Table 8 and Table 9, we can see that the flexibility in choosing sailing speeds and service frequencies can indeed bring a substantial increase in the profit of the final LSND solutions, demonstrating the benefits of considering the optimization of the sailing speed and service frequency.

Furthermore, we compare the network design solutions produced by BDP and BP methods on the six benchmark instances in Set-C of the General LSND. Solutions produced using the BAPPS method are unavailable in the literature. As illustrated in §EC.8.4 of the e-companion, the BP solutions are advantageous to select proper sailing speeds for the rotations operated under different market conditions. This finding accords with the facts in the liner shipping industry that carriers adopt a slow-steaming strategy under an overcapacity market (such as the liner shipping market after the financial crisis of 2008) while speeding up the liner services when the capacity supply is insufficient (such as the liner shipping market during the Covid-19 pandemic since 2020).

7. Conclusions and future research

This paper studied the liner shipping network design (LSND) problem with transshipment costs, which is today's core of liner shipping operations. To model the problem, we considered decisions on general rotation configurations, where a configuration specifies various service components, such as the ship type, sailing speed, service frequency, etc. With the aim of profit maximization, the model identifies profitable demands to serve and can produce a network design solution considering existing and potential demand markets.

We proposed a new set-partitioning-like formulation with an exponential number of variables and constraints to capture the transshipment costs. The proposed formulation is associated with rotation-dependent variables and constraints, so the traditional column generation (CG) method cannot be applied. To solve the linear programming relaxation of the new formulation, we developed a simultaneous column-and-row generation (SCRG) method, together with several novel speed-up techniques. The SCRG method was embedded in an exact branch-and-price algorithm to find optimal or near-optimal LSND solutions without any a priori restriction on the number of feasible or the maximum number of operated rotations. The newly proposed solution method was examined on test instances from the literature and newly generated for two main LSND variants based on different rotation configurations: the Standard LSND and the General LSND. Our new exact method outperformed several alternative solution methods known in the existing literature. It significantly improved the best-known lower and upper bounds on the optimal solutions to several benchmark problem instances. This study has enriched the solution methods for the LSND, enhanced the SCRG-based solution method in general, and extended its practical applications.

Our work has opened up several promising future research directions. First, in this paper, we have proposed a new SCRG-based solution method for the LSND. To further improve it, we can develop alternative approaches to computing dual solutions and enhance the algorithm for solving the pricing problem. In particular, it would be of great interest to investigate how to avoid the generation of unnecessary rotations more effectively. Although this work focused on developing an exact method that can be directly used to plan regional rotations, the SCRG-based solution method can also be used to develop heuristic methods for solving larger-sized instances of the LSND. For example, it can be applied to plan inter-region rotations, where a region represents a cluster of regional ports (i.e., FarEast and NorthEurope) with their demands being aggregated (see, e.g., Xia et al. 2015). By solving the pricing problem approximately or using port aggregation and disaggregation techniques (see, e.g., Mulder and Dekker 2014), one could transform our branch-and-price algorithm to a heuristic. In the future, we will investigate how such a heuristic performs in solving larger-sized LSND instances, how to identify a suitable size for the configuration set, and improve its performance. we will also investigate how our SCRG-based solution method can be applied to tackling other challenging LSND variants, such as those with transit time restrictions (Karsten et al. 2018), transit time costs (Trivella et al. 2021), and load-dependent fuel consumption (Xia et al. 2015), for which sailing speeds need to be optimized for each voyage arc.

Second, the proposed SCRG-based solution method can be adapted to incorporate additional rotation constraints arising in real-world situations by modifying the rotation generation phase of the SCRG-based solution method. Moreover, the SCRG-based solution method is developed to solve the LSND with general rotation configurations, which can incorporate not only those common service components, such as the ship type, sailing speed, and service frequency, but also those emerging ones. For example, green technologies, such as scrubbers and shore powers, are being adopted on ships nowadays to meet the emission control requirements at the ports and voyage legs of the rotation. However, despite some studies in the liner shipping fleet deployment literature (see, e.g., Zhen et al. 2020), decisions on green technology adoptions for rotations have not been studied for the LSND. As such decisions can be incorporated into the configurations of rotations, our new SCRG-based solution method can be adapted to solve this new variant of the LSND.

Finally, we have enhanced our SCRG-based solution method for the LSND by developing several new speed-up techniques to prevent the generation of unnecessary columns, which have been overlooked in previous studies. These techniques are potentially useful in solving other large-scale linear (integer) programming models with column-dependent rows, such as those for the bus rapid transit route design problem (Feillet et al. 2010), the multi-stage cutting stock problem (Muter et al. 2013), and the quadratic set covering problem (Muter et al. 2013). Future studies on applications to these problems and the adaption of our new speed-up techniques hold great promise.

Acknowledgments

The authors thank the reviewers, associate editor, and editor for helpful suggestions and thorough review.

References

Agarwal R, Ergun Ö (2008) Ship scheduling and network design for cargo routing in liner shipping. *Transportation Science* 42(2):175–196.

Álvarez JF (2009) Joint routing and deployment of a fleet of container vessels. *Maritime Economics & Logistics* 11(2):186–208.

Ameln M, Sand Fuglum J, Thun K, Andersson H, Stalhane M (2019) A new formulation for the liner shipping network design problem. *International Transactions in Operational Research* 1–22.

Andersen J, Christiansen M, Crainic TG, Grohaug R (2011) Branch and price for service network design with asset management constraints. *Transportation Science* 41(1):33–49.

Andersen J, Crainic TG, Christiansen M (2009a) Service network design with asset management: Formulations and comparative analyses. *Transportation Research Part C: Emerging Technologies* 17(2):197–207.

Andersen J, Crainic TG, Christiansen M (2009b) Service network design with management and coordination of multiple fleets. *European Journal of Operational Research* 193(2):377–389.

Avella P, D'Auria B, Salerno S (2006) A LP-based heuristic for a time-constrained routing problem. *European Journal of Operational Research* 173(1):120–124.

Bajgiran OS, Cire AA, Rousseau LM (2017) A first look at picking dual variables for maximizing reduced cost fixing. Salvagnin D, Lombardi M, eds., *Integration of AI and OR Techniques in Constraint Programming*, 221–228 (Springer).

Balakrishnan A, Karsten CV (2017) Container shipping service selection and cargo routing with transshipment limits. *European Journal of Operational Research* 263(2):652–663.

Brouer BD, Alvarez JF, Plum CE, Pisinger D, Sigurd MM (2014a) A base integer programming model and benchmark suite for liner-shipping network design. *Transportation Science* 48(2):281–312.

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

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

Christiansen M, Hellsten E, Pisinger D, Sacramento D, Vilhelmsen C (2020) Liner shipping network design. *European Journal of Operational Research* 286(1):1–20.

Cosco-Line (2020) www.lines.coscoshipping.com, accessed February 23, 2020.

Crainic TG, Hewitt M, Toulouse M, Vu DM (2014) Service network design with resource constraints. *Transportation Science* 50(4):1380–1393.

Desaulniers G, Rakke JG, Coelhk LC (2016) A branch-price-and-cut algorithm for the inventory-routing problem. *Transportation Science* 50(3):1060–1076.

Desrosiers J, Lübbecke M (2011) *Branch-price-and-cut algorithms* (Wiley Encyclopedia of Operations Research and Management Science).

Feillet D, Gendreau M, Medaglia AL, Walteros JL (2010) A note on branch-and-cut-and-price. *Operations Research Letters* 38(5):346–353.

Fugazza M, Hoffmann J (2017) Liner shipping connectivity as determinant of trade. *Journal of Shipping and Trade* 2(1).

Giovannini M, Psaraftis HN (2019) The profit maximizing liner shipping problem with flexible frequencies: logistical and environmental considerations. *Flexible Services and Manufacturing Journal* 31:567–597.

Hellsten EO, Sacramento D, Pisinger D (2022) A branch-and-price algorithm for solving the single-hub feeder network design problem. *European Journal of Operational Research* 300(3):902–916.

Huang L, Xiao F, Zhou J, Duan Z, Zhang H, Liang Z (2023) A machine learning based column-and-row generation approach for integrated air cargo recovery problem. *Transportation Research Part B* 178:102846.

IBM (2020) *IBM ILOG CPLEX Optimization Studio CPLEX User's Manual*.

Irnich S, Desaulniers G (2005) Shortest path problems with resource constraints. Desaulniers G, Desrosiers J, Solomon MM, eds., *Column Generation*, 33–65 (Springer US).

Karsten CV, Ropke S, Pisinger D (2018) Simultaneous optimization of container ship sailing speed and container routing with transit time restrictions. *Transportation Science* 52(4):769–787.

Katayama N, Chen M, Kubo M (2009) A capacity scaling heuristic for the multicommodity capacitated network design problem. *Journal of Computational and Applied Mathematics* 232(1):90–101.

Koza DF, Desaulniers G, Ropke S (2020) Integrated liner shipping network design and scheduling. *Transportation Science* 54(2):512–533.

Krogsgaard A, Pisinger D, Thorsen J (2018) A flow-first route-next heuristic for liner shipping network design. *Networks* 72(3):358–381.

Lübbecke M, Desrosiers J (2005) Selected topics in column generation. *Operations Research* 53(6):1007–1023.

Maersk-Line (2020) www.maerskline.com, accessed February 23, 2020.

Maher SJ (2016) Solving the integrated airline recovery problem using column-and-row generation. *Transportation Science* 50(1):216–239.

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

Mulder J, Dekker R (2014) Methods for strategic liner shipping network design. *European Journal of Operational Research* 235(2):367–377.

Muter İ (2011) *Simultaneous column-and-row generation*. Ph.D. thesis, Sabanci University, Turkey.

Muter İ, Birbil Şİ, Bülbül K (2013) Simultaneous column-and-row generation for large-scale linear programs with column-dependent-rows. *Mathmatical Programming* 142(1-2):47–82.

Muter İ, Birbil Şİ, Bülbül K (2018) Benders decomposition and column-and-row generation for solving large-scale linear programs with column-dependent-rows. *European Journal of Operational Research* 264(1):29–45.

Muter İ, Sezer Z (2018) Algorithms for the one-dimensional two-stage cutting stock problem. *European Journal of Operational Research* 271(1):20–32.

Plum CE, Pisinger D, Salazar-Gonzalez JJ, Sigurd MM (2014a) Single liner shipping service design. *Computers & Operations Research* 45:1–6.

Plum CE, Pisinger D, Sigurd MM (2014b) A service flow model for the liner shipping network design problem. *European Journal of Operational Research* 235(2):378–386.

Reinhardt LB, Pisinger D (2012) A branch and cut algorithm for the container shipping network design problem. *Flexible Services and Manufacturing Journal* 24(3):349–374.

Spliet R (2024) A simple perspective on simultaneous column and row generation. *Operations Research Forum* 5:69.

Thun K, Andersson H, Christiansen M (2017) Analyzing complex service structures in liner shipping network design. *Flexible Services and Manufacturing Journal* 29(3-4):535–552.

Trivella A, Corman F, Koza DF, Pisinger D (2021) The multi-commodity network flow problem with soft transit time constraints: Application to liner shipping. *Transportation Research Part E: Logistics and Transportation Review* 150:102342.

UNCTAD (2017) Review of maritime transport 2017 Annual report.

UNCTAD (2021) Review of maritime transport 2021 Annual report.

Vanderbeck F, Wolsey LA (2010) Reformulation and decomposition of integer programs. Jünger M, Liebling TM, Naddef D, Nemhauser GL, Pulleyblank WR, Reinelt G, Rinaldi G, Wolsey LA, eds., *50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art*, 431–502 (Berlin, Heidelberg: Springer Berlin Heidelberg).

Wang D, Xiao F, Zhou L, Liang Z (2020) Two-dimensional skiving and cutting stock problem with setup cost based on column-and-row generation. *European Journal of Operational Research* 286:547–563.

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

Wang Y, Meng Q, Kuang H (2019) Intercontinental liner shipping service design. *Transportation Science* 53(2):344–364.

Wilmsmeier G, Hoffmann J (2008) Liner shipping connectivity and port infrastructure as determinants of freight rates in the caribbean. *Maritime Economics & Logistics* 10(1-2):130–151.

Xia J, Li KX, Ma H, Xu Z (2015) Joint planning of fleet deployment, speed optimization, and cargo allocation for liner shipping. *Transportation Science* 49(4):922–938.

Yang Y (2024) Deluxing: Deep lagrangian underestimate fixing for column-generation-based exact methods. *Operations Research* Articles in Advance, URL https://doi.org/10.1287/opre.2023.0398.

Zak EJ (2002) Row and column generation technique for a multistage cutting stock problem. *Computers & Operations Research* 29(9):1143–1156.

Zhen L, Wu Y, Wu Y, Wang S, Laporte G (2020) Green technology adoption for fleet deployment in a shipping network. *Transportation Research Part B: Methodological* 139:388–410.

Author biographies

Jun Xia is an associate professor in the Sino-US Global Logistics Institute at the Antai College of Economics & Management, Shanghai Jiao Tong University. His research interests include combinatorial optimization and mathematical programming, with a particular emphasis on their practical applications in transportation planning and logistics.

Zhou Xu is a professor in the Department of Logistics and Maritime Studies at the Faculty of Business, Hong Kong Polytechnic University. His research interests include combinatorial optimization, mathematical programming, and algorithm design and analysis, particularly with reference to transportation, logistics, and supply chain management.

Roberto Baldacci is an associate professor of Operations Research at the College of Science and Engineering, Hamad Bin Khalifa University. His major research interests are in the area of transportation planning, logistics and distribution, and the solution of vehicle routing and scheduling problems over street networks. His research activities are in the theory and applications of mathematical programming. He has worked in the design of new heuristic and exact methods for solving combinatorial problems, such as routing and location problem.