https://doi.org/10.1016/j.trb.2018.10.016



# Subloop-based reversal of port rotation directions for container liner shipping network alteration

**Jingxu Chen [a, b, d], Shuai Jia [b], Shuaian Wang [b, c *], Zhiyuan Liu [a, d]**

[a] *Jiangsu Key Laboratory of Urban ITS, Jiangsu Province Collaborative Innovation Center of Modern Urban Traffic Technologies, Southeast University, Si Pai Lou #2, Nanjing, China*

[b] *Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University, Hong Kong*

[c] *The Hong Kong Polytechnic University Shenzhen Research Institute, Nanshan District, Shenzhen, China*

[d] *School of Transportation, Southeast University, Nanjing, China*

* Corresponding author

*E-mail address:* wangshuaian@gmail.com

**ABSTRACT**

Container liner shipping network alteration is a practical manner of shipping network design, which aims to make minor modifications to ameliorate the existing network. In a generic liner shipping network with butterfly ports, each ship route is separated into a set of subloops on the basis of its structure and internal butterfly ports. Reversing the subloop directions has an impact on the network-wide cost including inventory cost, transshipment cost, and slot-purchasing cost. This paper proposes a new destination-based nonlinear model for the subloop-based reversal of port rotation directions with the objective of minimizing the overall network-wide cost. We prove that the addressed problem is NP-hard. Next, the model is transformed to an equivalent mixed-integer linear programming model. Based on the structure of the reformulated model, we develop a Benders decomposition (BD) algorithm and a metaheuristic method to solve practical-size instances. Three acceleration strategies are incorporated into the BD algorithm, which are adding Pareto-optimal cuts, updating big-M coefficients and generating combinatorial Benders cuts. Case studies based on three small examples and an Asia-Europe-Oceania liner shipping network with a total of 46 ports are conducted. Results show that the problem could be efficiently solved by the accelerated BD algorithm and the optimization of subloop directions is conducive to decreasing the network-wide cost especially the inventory cost.

**Keywords:** container shipping; liner shipping network alteration; port rotation direction; mixed-integer linear programming; Benders decomposition

## 1. Introduction

As one of the most important modes of international transportation, container transportation plays a pivotal role in the sustainable development of the world trade and world economy (Brouer et al., 2014a; Mulder and Dekker, 2014; Tran and Haasis, 2015; Angeloudis et al., 2016; Lee and Song, 2017). The total container trade volume was estimated at 1.69 billion tons, equivalent to 175 million twenty-foot equivalent units (TEUs) in 2015 (UNCTAD, 2016). Containers are transported by container shipping companies over their shipping networks. A container liner shipping network operated by a particular shipping company subsumes a set of route services. Each route service is a sequence of port visits (port rotation) sailed by a string of homogenous ships at a regular service frequency. Meantime, the port rotation of each route usually forms a loop in ways that ships visit the first port again after visiting the last one.

In practice, the container shipping network is characterized by the excessively high operating cost, such as Maersk Line spends a two-digit billion USD amount yearly on its operation (Plum et al., 2014a). Therefore, even a small enhancement of the network's utilization efficiency can bring about a significant influence on the overall cost reduction (Song and Dong, 2011, 2012; Jiang et al., 2015). It is known that a cost-effective liner shipping planning process comprises several efficient decisions that span three different levels (Christiansen et al., 2004; Agarwal and Ergun, 2008; Meng et al., 2014): (1) containership fleet size and mix, alliance strategy and network design at the strategic level; (2) frequency setting, speed optimization, fleet deployment and schedule design at the tactical level; and (3) container booking and routing at the operational level. Among these decisions, the liner shipping network design is implemented every three to six months in coping with the variation of demand, which is addressed in this paper.

The liner-shipping network design problem (LSNDP) is to determine which ports each route service with a designated capacity should visit and in what order. Due to its practical importance, LSNDP has gained steadily increasing attention from the research community (Christiansen et al., 2013; Liu et al., 2014; Demir et al., 2016; Karsten et al., 2016; Monemi and Gelareh, 2017; Wang, 2017; Krogsgaard et al., 2018; among many others). We refer the readers to Brouer et al. (2014a) and Meng et al. (2014) for comprehensive reviews of LSNDP. Research on this topic can be classified into two categories. The first category of relevant literature is to *design a liner shipping network from scratch*. The second category is *liner shipping network alteration*, which aims at making minor alterations to ameliorate the existing network.

In the first category, Rana and Vickson (1988) contributed a pioneering work by building a mixed-integer linear programming model for a single ship route design problem. Shintani et al. (2007), Song and Dong (2013) and Plum et al. (2014b) extended the model to incorporate more practical circumstances in the design of port rotation at the route level. Reinhardt and Pisinger (2012) proposed an integer programming model and a branch-and-cut method for designing butterfly ship routes to optimality. A butterfly ship route is a route with at least one port visited twice in a round trip (termed as *butterfly port*). For example, Fig. 1 presents one butterfly ship route where ports Singapore and Colombo are two butterfly ports.

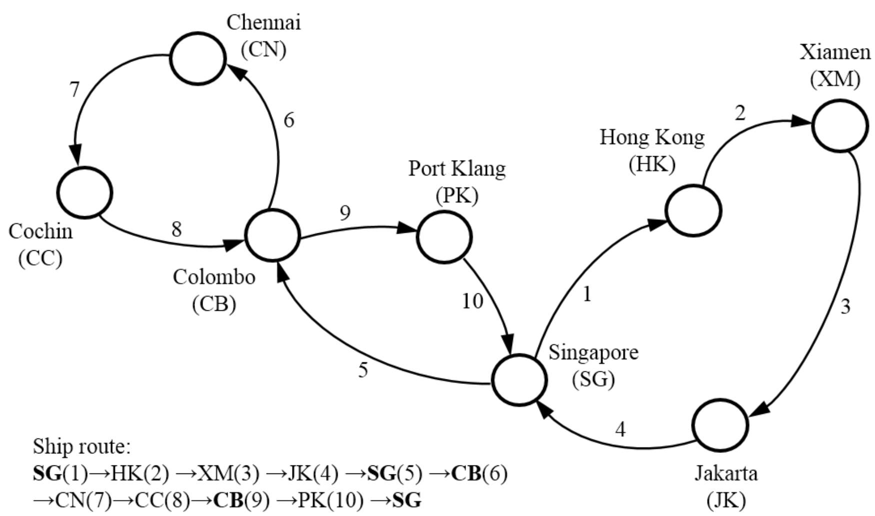



**Fig. 1.** An illustrative ship route with two butterfly ports.

At the network level, Fagerholt (2004), Sambracos et al. (2004) and Karlaftis et al. (2009) worked on the feeder network design problem with one hub port and many feeder ports. Imai et al. (2009), Gelareh et al. (2010, 2011), Sun and Zheng (2016) and Holm et al. (2018) examined the hub and spoke liner shipping network design problem with many hubs and feeder ports. In addition, the transshipment of containers provides the company with more operational flexibilities, which are beneficial to expand the scope of shipping services. In the aspect of considering transshipment operations, there have been a few works in the literature. For example, Agarwal and Ergun (2008) developed a multi-commodity-flow-based space-time network model for LSNDP with cargo routing. Transshipment costs were not considered in the network design stage. Three methods (a two-phase Benders' decomposition-based algorithm, a column generation-based approach and a greedy heuristic) were designed and tested on a network with up to 20 ports and 100 ships. Alvarez

(2009) extended this work by explicitly considering transshipment costs. A combined Tabu search and column generation-based heuristic was proposed and applied to a network with 120 ports and 5 ship types. Brouer et al. (2014b) contributed a seminal benchmark suite for the global container transport network design. This benchmark suite was developed based upon the operating data from the largest global liner shipping company, Maersk Line. The largest instance solved includes 111 ports and 4000 O-D pairs.

In recent literature, some other elements apart from the operating cost and revenue have been taken into account by researchers in LSNDP. Akyüz and Lee (2016) and Karsten et al. (2016, 2017) incorporated the level of service either in the objective function or one constraint of their models, which are represented by the transit time and associated inventory cost. Providing shorter transit time and thereby reducing the inventory cost would gain more market share. Extending the situation where a butterfly port is visited twice in a round trip such as the ones in in Fig. 1, Thun et al. (2017) developed a model for LSNDP which is available to handle the more complex network structure where each port can be visited several times in a route service. The model is solved by a branch-and-price method. Results reveal that butterfly ports are helpful to a decline in the operating cost. Accounting for carbon dioxide ($CO_2$) and sulphur oxides ($SO_x$) emissions, Cariou et al. (2018) explored the network design within emission control areas.

All the above studies mainly focused on designing liner services from scratch. In fact, liner shipping company cannot reshuffle its network overnight due to the continuity of the supply chain relationships with various stakeholders (e.g. shippers and port/terminal operators). The second category of LSNDP provides the other perspective of network design. It aims to design an efficient network via the modifications of the existing network. Currently, the extant literature on the second category is limited, which exhibits two patterns of liner shipping network alteration: (1) the disassembly and reassembly of route services with the same type of ship (Wang et al., 2015); and (2) the alteration of port rotation directions (Wang and Meng, 2013).

For a ship route without butterfly ports, it is a directed loop whose direction is either clockwise or counter-clockwise. Based on this character, Wang and Meng (2013) investigated a special liner shipping network without considering butterfly ports, which means there are only two direction options for each route in the network. They proposed a multi-commodity network flow model to procure the optimal port rotation directions of ship routes. In practice, the butterfly ports are common in a liner shipping network. In a survey of 154 service routes in 2008 (CI, 2009; Song and Dong, 2013), around 55% of ship routes have at least one butterfly port in a single round-trip (these ship routes are called butterfly routes). The existence of butterfly routes makes the port rotation direction optimization problem more complex. For example, the butterfly route in Fig. 1 is a single directed loop, while it can be further decomposed into three *subloops* due to two butterfly ports Singapore and Colombo, and each subloop per se owns its direction option. Therefore, the generic network containing butterfly routes poses more choices of direction modifications, and meantime some practical issues such as transshipment operations and level of service need to be simultaneously considered. However, the port rotation direction optimization of such generic network has not been studied particularly.

In terms of theoretical development, some researchers have derived the computational complexity of the first category of LSNDP. The general liner shipping network from scratch was proved to be NP-hard as it can be reduced to a knapsack problem (Agarwal and Ergun, 2008) and a set covering problem (Brouer et al., 2014). Though the second category of LSNDP is one type of highly constrained network design problem, which makes it easier from the methodological point of view, until recently none of previous studies have explicitly investigated the associated computational complexity. In addition, the proposed models in the second category of LSNDP were primarily solved by off-the-shelf optimization solvers. These solvers are not always efficient and in this case the design of efficient algorithms based on the problem's particular properties is necessitated.

The contribution of this paper is twofold: (i) a destination-based nonlinear model is proposed for the subloop-based reversal of port rotation directions in a generic network containing butterfly ship routes. This model captures container transshipment operations and level of service in the network. We prove the NP-hardness of our problem and similar problems presented in literature; and (ii) the proposed model is transformed into an equivalent mixed-integer linear programming model where an accelerated Benders decomposition algorithm incorporating three acceleration strategies efficiently solves the reformulated model.

The remainder of this study is organized as follows. Section 2 elaborates the problem of subloop-based reversal of port rotation directions. Section 3 builds a mathematical model and discusses its computational complexity. In Section 4, we develop an equivalent linear model, a Benders decomposition method and a metaheuristic method for the problem. Numerical examples are presented in Section 5. Finally, conclusions are provided in the last section.

## 2. Subloop-based reversal of port rotation directions

The parameters used in the model and their notation are summarized in Table 1. Consider a container liner shipping company operating a number of ship routes, which are denoted by the set $R$. The company regularly serves a group of physical ports denoted by the set $P$. Each ship route $r \in R$ has a weekly service frequency maintained by a group of homogeneous ships.

**Table 1** List of Notation

| | |
|---|---|
| ***Indices*** | |
| $i$ | the index of ports of call |
| $r$ | the index of ship routes |
| $s$ | the index of subloops |
| $c$ | the index of sections |
| ***Sets*** | |
| $P$ | a set of physical ports |
| $P_b$ | a set of butterfly ports |
| $I_r$ | a set of ports of call on ship route $r$ |
| $I_{rp}$ | a set of port indices on route $r$ that refer to a butterfly port $p \in P_b$, $I_{rp} \subset I_r$ |
| $R$ | a set of ship routes |
| $\hat{R}$ | a set of butterfly ship routes which contain at least one butterfly port |
| $S_r$ | a set of subloops corresponding to ship route $r$ |
| $\overline{S}_r$ | a set of subloops with only two ports ($\overline{S}_r \subseteq S_r$) |
| $C_s$ | a set of sections corresponding to subloop $s \in S_r$ |
| $W$ | a set of O-D pairs, $W = P \times P$ |
| ***Parameters*** | |
| $N_r$ | the number of ports of call on ship route $r$ |
| $p_{ri}$ | the port corresponding to the $i$th port of call on ship route $r$ |
| $p_{ci}$ | the port corresponding to the $i$th port of call on section $c$ |
| $h(c)$ | the head port of call on section $c$ |
| $t(c)$ | the tail port of call on section $c$ |
| $q^{od}$ | the demand for O-D pair $(o,d) \in W$ |
| $g^{od}$ | the cost for purchasing one slot for O-D pair $(o,d) \in W$ |
| $\alpha$ | the inventory cost rate pertaining to the transit time of containers |
| $\overline{c}_p$ | the transshipment cost at port $p$ |
| $T^{od}$ | the transit time of containers of O-D pair $(o,d) \in W$ |
| $t_p$ | the connection time at port $p$ |
| $t_{ci}$ | the transit time of containers on leg $i$ of section $c$ |
| $E_r$ | the container capacity of ships deployed on ship route $r$ |

Let $N_r$ denote the number of ports of call on ship route *r* and $p_{ri}$ denote the port corresponding to the *i*th port of call. The port rotation (or itinerary) of ship route *r* forms a directed loop, which is expressed as below:

$$p_{r1} \to p_{r2} \to \cdots \to p_{rN_r} \to p_{r1} \ . \tag{1}$$

Define the set of ports of call on ship route *r* as $I_r = \{1, 2, ..., N_r\}$. Defining $p_{r,N_r+1} := p_{r1}$, the voyage between two contiguous ports of call $p_{ri}$ and $p_{r,i+1}$ is called *leg* $i$ of ship route $r$, which can be denoted by the pair of ordered ports $< p_{ri}, p_{r,i+1} >, i \in I_r$. For instance, the ship route shown in Fig. 1 has ten legs - leg 1: $< p_{r1}(\text{SG}), p_{r2}(\text{HK}) >$, leg 2: $< p_{r2}(\text{HK}), p_{r3}(\text{XM}) >$, leg 3: $< p_{r3}(\text{XM}), p_{r4}(\text{JK}) >$, leg 4: $< p_{r4}(\text{JK}), p_{r5}(\text{SG}) >$, leg 5: $< p_{r5}(\text{SG}), p_{r6}(\text{CB}) >$, leg 6: $< p_{r6}(\text{CB}), p_{r7}(\text{CN}) >$, leg 7: $< p_{r7}(\text{CN}), p_{r8}(\text{CC}) >$, leg 8: $< p_{r8}(\text{CC}), p_{r9}(\text{CB}) >$, leg 9: $< p_{r9}(\text{CB}), p_{r10}(\text{PK}) >$, leg 10: $< p_{r10}(\text{PK}), p_{r1}(\text{SG}) >$.

The survey of 154 service routes reveals that ship routes with ports being called more than twice (i.e., ≥3 times) are very rare in practice (CI, 2009; Song and Dong, 2013). In this study, we focus on the case that ports in a service route are called either once or twice. Let $P_b$ denote the set of butterfly ports such as in Fig. 1 $P_b = \{\text{SG}, \text{CB}\}$. Let $I_{rp}$ denote the set of port indices on route *r* that refer to a particular butterfly port $p \in P_b, I_{rp} \subset I_r$. In Fig. 1, we have $I_{r,\text{SG}} = \{1,5\}$ and $I_{r,\text{CB}} = \{6,9\}$.

### 2.1 Subloop and section

To better present the concept of subloop and subloop-based reversal of port rotation directions, we consider a butterfly ship route which may consist of several subloops.

**Definition 1**. In a butterfly ship route, a subloop is a directed closed loop constituted by a set of legs where there are only two legs connecting each butterfly port in one subloop.

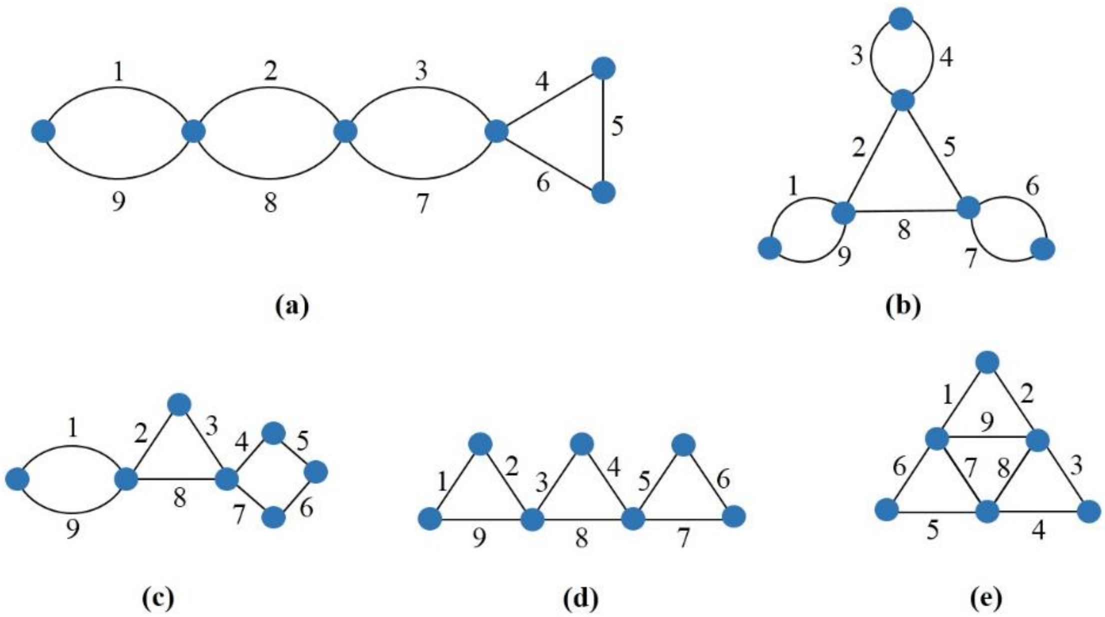
For instance, the ship route in Fig. 1 has three subloops. Subloop 1 subsumes four legs (i.e. leg 1, 2, 3 and 4). Leg 5, leg 9 and leg 10 form subloop 2. Subloop 3 consists of leg 6, leg 7 and leg 8.

The direction of subloop 1 and subloop 2 is clockwise and the direction of subloop 3 is counter-clockwise (legs 1, 2, 3, 4, 5, 9 and 10 do not form a subloop because butterfly port SG connects with four legs). By definition, each butterfly port $p$ in set $P_b$ should appear in two different subloops. Subloop including a single butterfly port is called outer subloop, while subloop with multiple butterfly ports is called inner subloop. In Fig. 1, subloop 1 and subloop 3 are two outer subloops, and subloop 2 is one inner subloop.

**Definition 2**. In a subloop, a section is a sequence of directed legs that start from a butterfly port (tail) to the next butterfly port (head). Other than the tail and head ports of call, none of the remaining ports of call in the section are butterfly ports.

For example, each outer subloop in Fig. 1 (i.e. subloop 1 and 3) only has one section since the tail and head ports of call are the same butterfly port. Inner subloop 2 has two sections: the first section is the voyage from butterfly port SG directly to butterfly port CB, and the second section is the voyage from butterfly port CB to port PK, and then to butterfly port SG. It is reasonable to assume that the number of sections between any two butterfly ports does not exceed two. This can be interpreted by the fact that if a butterfly-port pair is served twice in the identical direction in a round trip, it will apparently impair ship utilization and unnecessarily raise voyage time. Then, it is straightforward to observe that any two contiguous subloops can be joined at only one butterfly port.

Fig. 2 illustrates some more complex scenarios of a specific butterfly ship route with nine ports of call. Though the number of ports of call is fixed, a wide range of variation of route topological structures arises in terms of the count of physical ports and the number of subloops and sections (either inner or outer). For example, ship route in Fig. 2a has 6 physical ports and its itinerary is composed of 4 subloops (two outer subloops and two inner subloops). Each inner subloop consists of two sections. It differs from the scenario in Fig. 2b in which the sole inner subloop contains three sections and each section subsumes only one leg originating from one butterfly port to the other butterfly port. In Fig. 2c, the scenario contains a ship route with one inner subloop and two outer subloops. Although the number of subloops in Fig. 2d is identical to that of Fig. 2c, the number of direction options is different (four in Fig. 2c and eight in Fig. 2d). The ship route in Fig. 2e is a bit different from the other scenarios since its subloop splitting result is not unique. As depicted in Fig. 3, it can be divided into either three subloops or two subloops. In the latter case, two subloops nest with each other and this pattern is termed as the inserted subloop.




| Count / Scenario | Physical ports | Subloops Outer | Subloops Inner | Subloops Sum | Legs | Legs in subloop | Sections Outer | Sections Inner | Sections Sum | Direction alteration options |
|---|---|---|---|---|---|---|---|---|---|---|
| a | 6 | 2 | 2 | 4 | 9 | 2, 2, 2, 3 | 2 | 4 | 6 | 2 |
| b | 6 | 3 | 1 | 4 | 9 | 2, 2, 2, 3 | 3 | 3 | 6 | 2 |
| c | 7 | 2 | 1 | 3 | 9 | 2, 3, 4 | 2 | 2 | 4 | 4 |
| d | 7 | 2 | 1 | 3 | 9 | 3, 3, 3 | 2 | 2 | 4 | 8 |
| e | 6 | 0 | 3 | 3 | 9 | 3, 3, 3 | 0 | 6 | 6 | 8 |
| | | 0 | 2 | 2 | | 3, 6 | | | | 4 |

**Fig. 2.** A more complex example of one ship route with nine ports of call.

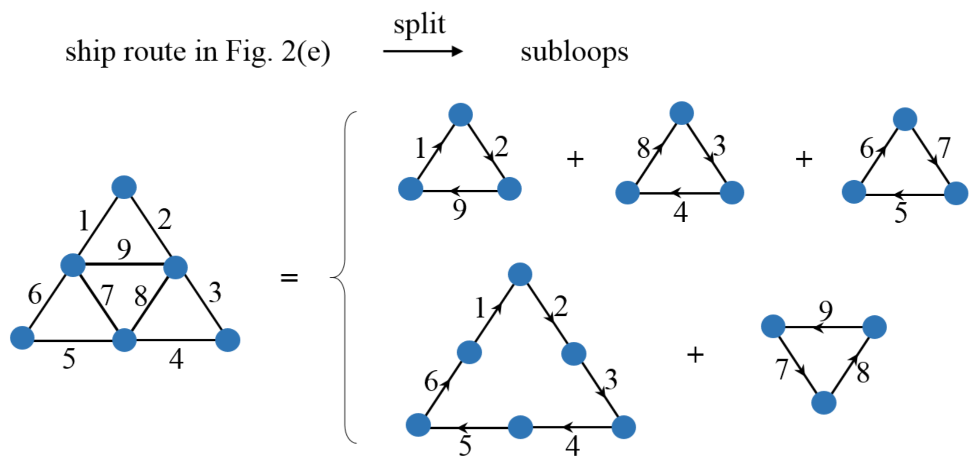



**Fig. 3.** A special case of one ship route with various subloop splitting results.

### 2.2 Reversal of subloop direction

Subloop-based reversal of port rotation directions (SRPRD) is a practical alteration of route structures. For instance, the direction of the right-hand outer subloop (subloop 1) shown in Fig. 1 is clockwise, and ships visit Singapore, Hong Kong, Xiamen, Jakarta sequentially and return to Singapore. If the direction of subloop 1 is reversed to be counter-clockwise, the transit time of containers from Singapore to Hong Kong will be longer while the transit time from Singapore to Jakarta will be shortened. Direction reversal of the other two subloops in Fig. 1 can be performed in an analogous way.

In five scenarios of Fig. 2, each ship route has nine ports of call whereas their possible route alterations are different. For the first two scenarios, though each route contains four subloops, only two direction alteration options are available for each scenario. It is because if there are only two legs in a subloop, there is no difference between clockwise direction and counter-clockwise direction. Then, only direction reversal of the right-hand outer subloop in scenario Fig. 2a and the sole inner subloop in scenario Fig. 2b will take effect on route alteration. It is worth noticing that two subloop splitting results in Fig. 3 correspond to eight and four possible route alterations respectively (including two identical alterations). Such service routes are quite rare in reality, which does not occur in 154 routes in 2008 (CI, 2009; Song and Dong, 2013). In this study, we do not consider the special pattern of butterfly ship routes containing inserted subloops (see Fig. 3). The observation here indicates that for a certain ship route, only those subloops with more than two legs have an impact on the total number of direction options, which presents a power of two functional relationship.

**Remark 1**. Reversal of subloop direction is restructuring each service route while all the ports are visited the same number of times each week before and after alteration.

It is worth noting that the segment-based network alteration in Wang et al. (2015) is different from this study. In Wang et al. (2015), each ship route is disassembled into a set of segments which parallel the definition of sections in this study. Later, segments with the same type of ship are reassembled into several new sub-networks. These sub-networks comprise a more efficient network. Differently, this study focuses on the direction optimization of all the subloops and associated sections. In subloop-based reversal of port rotation directions, each port that was visited by a certain ship route before network alteration is still visited by the same route. Furthermore, all the ships that served the ship route before still serve the identical route. After route alteration, the visiting sequence of ports to be called on the itinerary may be reversed. It will influence the generalized network-wide cost including inventory cost, transshipment cost, and slot-purchasing cost. In the next section, we build a mathematical model to obtain the optimal subloop-based reconstruction of each service route, in an effort to minimize the generalized network-wide cost.

## 3. Mathematical model

Let $\hat{R}$ denote the set of butterfly ship routes which contain at least one butterfly port. For the ease of modeling, we select one butterfly port as the first port of call (i.e. $p_{r1} \in P_b$) for each ship route $r \in \hat{R}$. Then, the remaining ports of call are sequentially numbered since the port rotation is exogenously given. Ship route $r \in \hat{R}$ contains a group of subloops, denoted by set $S_r$. Meantime, we let $\bar{S}_r$ denote the set of subloops with only two ports ($\bar{S}_r \subseteq S_r$). Each subloop $s \in S_r$ consists of a set of sections, denoted by $C_s$. We use lowercase letter $c$ to refer to a particular section. Define the set of ports of call on subloop $s$ as $I_{rs} \subseteq I_r$ and on section $c$ as $I_{rc} \subseteq I_{rs}, c \in C_s$. With a little abuse of the notation, we use $p_{ci}$ to denote the port corresponding to the $i$th port of call on section $c$ ($c \in C_s, s \in S_r$). Here, the sequence number $i$ is in accordance with the original number of port of call in route $r$, i.e., for $c \in C_s, s \in S_r$, we have $p_{ci} = p_{ri}$. Each section $c$ originates from the tail port of call (denoted by $t(c)$) and ends at the head port of call (denoted by $h(c)$). For ship route $r \in \hat{R}$, there always exists one section $c$ ($c \in C_s, s \in S_r$) whose $h(c)$ corresponds to the first port of call $p_{r1}$ (i.e. $h(c)=1$). Here, $h(c)$ is reset to equal $N_r+1$ for the need of modeling. Fig. 4 illustrates the subloops and sections of the ship route in Fig. 1.

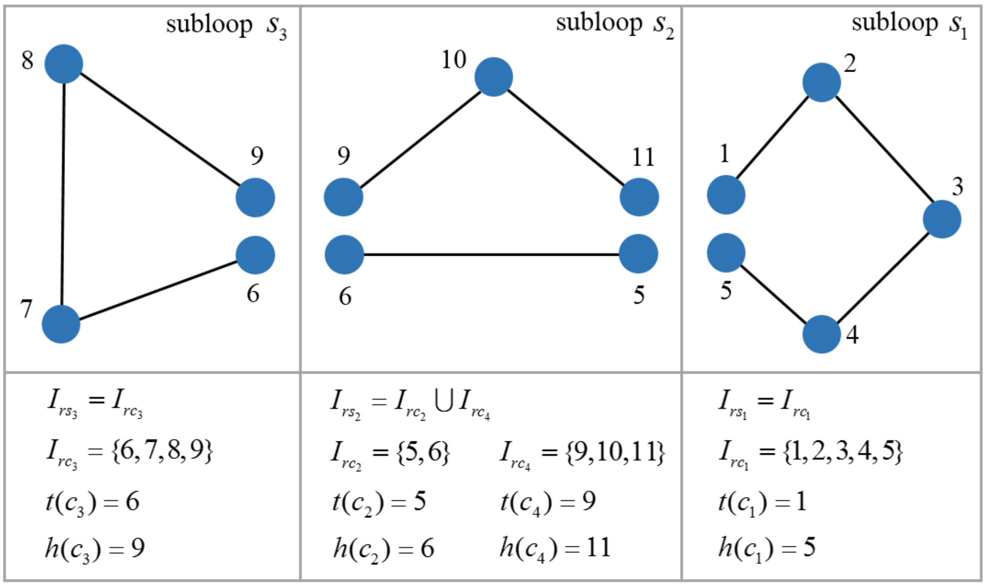



**Fig. 4.** Subloops and ports of call of sections.

For ship route $r \in R \setminus \hat{R}$, an arbitrary port is chosen as the first port of call. As ship route $r \in R \setminus \hat{R}$ per se contains two direction options (clockwise or counter-clockwise direction), it can be treated as one outer subloop $s$ with a sole section $c$. The tail port of call and head port of call of section $c$ are both assumed to be $p_{r,1}$. In this case, $t(c)$ equals 1 while for the ease of modeling, $h(c)$ is set to equal $N_r+1$. Namely, $I_{rc} = I_{rs} = I_r \cup \{N_r+1\}$.

We let $W$ denote the set of O-D pairs, $W = P \times P$. The demand for O-D pair $(o,d) \in W$ is represented by $q^{od}$ (TEUs). If there is no direct service between origins to destinations, containers may be transshipped at port $p \in P$, and the transshipment cost is denoted by $\bar{c}_p$ (USD/TEU). The loading cost at origin ports or discharge cost at destination ports is not explicitly taken into account because they are constant. It takes an additional time when containers are transshipped at port $p$, which is termed as the connection time, denoted by $t_p$ (h). In this study, we make the simplifying assumption that the connection time $t_p$ at each butterfly port $p \in P_b$ is a fixed number. The transit time of containers on leg $i$ of section $c$ ($c \in C_s, s \in S_r$) is represented by $t_{ci}$ (h). The inventory cost rate pertaining to the transit time of containers is denoted by $\alpha$ (USD/TEU/h).

In addition, if the liner shipping company is unable to transport all the containers by its own ships, it may purchase ship slots from other shipping companies. We use $g^{od}$ (USD/TEU) to denote the cost for purchasing one slot for O-D pair $(o,d) \in W$ . For simplicity, we do not consider empty containers (Song and Dong, 2012, 2013; Akyüz and Lee, 2016). At the same time, we let $T^{od}$ denote the transit time of containers of O-D pair $(o,d)$ transported by the purchased slots for formulating the inventory cost. The container capacity of ships deployed on ship route $r \in R$ is denoted by $E_r$ (TEUs).

### 3.1 Subloop reversal variables

To reflect the decisions on port rotation directions, we define $x_s$ ($s \in S_r, r \in R$) as a binary decision variable which equals 1 if the direction of subloop $s$ is reversed and 0 otherwise. Let $X$ denote the set of all the subloop reversal decision variables $x_s$ ($s \in S_r, r \in R$). Meanwhile, to acquire the generalized network-wide cost, we need to determine the optimal container flow in the network. In the next section, we construct a destination-based model for the subloop-based liner shipping network alteration. Some continuous decision variables for container routing on the sections are further defined as below:

$\hat{z}_{c,i}^{d}$ : Number of containers (TEUs/week) originating from any port and destined for port $d \in P$ that are loaded to a ship when it visits the $i$ th port of call on section $c \in C_s, s \in S_r, i \in I_{rc}$ ;

$\tilde{z}_{c,i}^{d}$ : Number of containers (TEUs/week) originating from any port and destined for port $d \in P$ that are discharged from a ship when it visits the $i$ th port of call on section $c \in C_s, s \in S_r, i \in I_{rc}$ ;

$f_{c,i}^{d}$ : Number of containers (TEUs/week) originating from any port and destined for port $d \in P$ that are stowed on board ships which prepare to depart from the $i$ th port of call on section $c \in C_s, s \in S_r, i \in I_{rc}$ .

We first consider the above decision variables of an arbitrary butterfly port $p$ ($\forall p \in P_b$). Port $p$ connects two contiguous subloops $s_1$ and $s_2$ of route $r$ ($s_1, s_2 \in S_r$), and it corresponds to two ports of call $I_{rp} = \{i_1, i_2\}$. Ports of call $i_1$ and $i_2$ are the head port of call and the tail port of call of two sections $c_1$ and $c_1'$ ($c_2'$ and $c_2$) in subloop $s_1$ ($s_2$) respectively. Note that sections $c_1$ and $c_1'$ ($c_2$ and $c_2'$) may represent the identical section, if subloop $s_1$ ($s_2$) is an outer subloop. Fig. 5 depicts four scenarios of butterfly port $p$ under the variation of two decision variables $x_{s_1}$ and $x_{s_2}$.

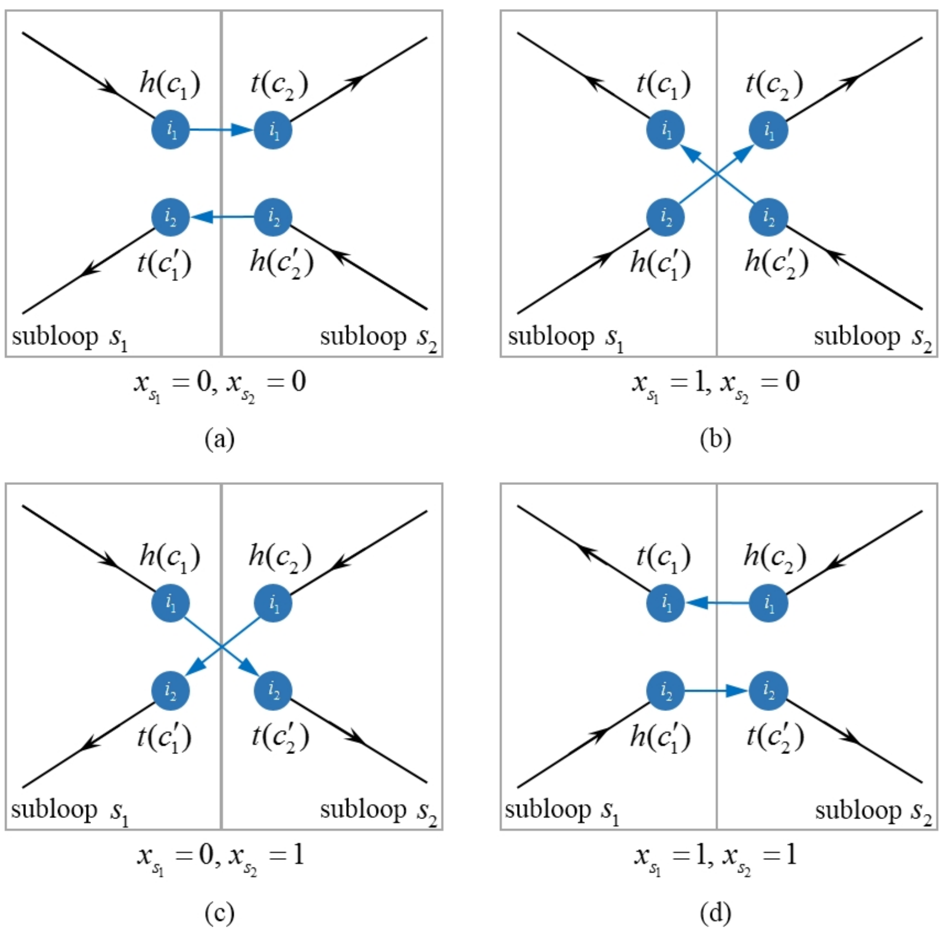



**Fig. 5.** Four scenarios of one butterfly port.

If $x_{s_1} = x_{s_2} = 0$, the head port of call of section $c_1$ and the tail port of call of section $c_2$ are the same port of call. Hence, it is not essential to simultaneously define both loading and discharging for the head of section $c_1$ as well as the tail of section $c_2$. In this case, we require that container loading operations occur at the tail port of call while container discharging operations always occur at the head port of call. This requirement also applies to the other three scenarios in Fig. 5. Mathematically, we have:

$$\begin{array}{l} \hat{z}^d_{c_1,i_1}(1-x_{s_1})=0 \\ \tilde{z}^d_{c_1,i_1}x_{s_1}=0 \end{array} \quad c_1\in C_{s_1}, s_1\in S_r, r\in R \tag{2}$$

$$\begin{array}{l} \tilde{z}^d_{c_2,i_1}(1-x_{s_2})=0 \\ \hat{z}^d_{c_2,i_1}x_{s_2}=0 \end{array} \quad c_2\in C_{s_2}, s_2\in S_r, r\in R \tag{3}$$

$$\begin{array}{l} \hat{z}^d_{c_1',i_2}x_{s_1}=0 \\ \tilde{z}^d_{c_1',i_2}(1-x_{s_1})=0 \end{array} \quad c_1'\in C_{s_1}, s_1\in S_r, r\in R \tag{4}$$

$$\begin{array}{l} \tilde{z}^d_{c_2',i_2}x_{s_2}=0 \\ \hat{z}^d_{c_2',i_2}(1-x_{s_2})=0 \end{array} \quad c_2'\in C_{s_2}, s_2\in S_r, r\in R\ . \tag{5}$$

Furthermore, in each scenario of Fig. 5, we use two dummy legs to link subloops $s_1$ and $s_2$. Take the scenario in Fig. 5a as an instance. When sections $c_1$ and $c_2$ are connected, we let $f^d_{c_1,i_1}$ denote the number of containers (originating from any port and destined for port $d\in P$) "flowing" from the head port of call $h(c_1)$ of section $c_1$ to the tail port of call $t(c_2)$ of section $c_2$ via one dummy leg. Then, the sum of the number of containers that are stowed on the dummy leg from $h(c_1)$ to $t(c_2)$ (denoted by $f^d_{c_1,i_1}$) plus the number of containers that are loaded at the port of call $t(c_2)$ (denoted by $\hat{z}^d_{c_2,i_1}$), should be the same as the number of containers that are stowed at the port of call $t(c_2)$ plus the number of containers that are discharged (denoted by $\tilde{z}^d_{c_2,i_1}$).

This relationship can be formulated in an analogous way for the remaining dummy legs of four scenarios. Hence, the container flow conservation equations of butterfly port $p$ are expressed as:

$$\begin{aligned} f^d_{c_1,i_1}(1-x_{s_1})+f^d_{c_1',i_2}x_{s_1}=(f^d_{c_2,i_1}+\tilde{z}^d_{c_2,i_1}-\hat{z}^d_{c_2,i_1})(1-x_{s_2})+(f^d_{c_2',i_2}+\tilde{z}^d_{c_2',i_2}-\hat{z}^d_{c_2',i_2})x_{s_2}, \\ c_1,c_1'\in C_{s_1}, c_2,c_2'\in C_{s_2}, s_1,s_2\in S_r, r\in R \end{aligned} \tag{6}$$

$$\begin{aligned} (f^d_{c_1,i_1}+\tilde{z}^d_{c_1,i_1}-\hat{z}^d_{c_1,i_1})x_{s_1}+(f^d_{c_1',i_2}+\tilde{z}^d_{c_1',i_2}-\hat{z}^d_{c_1',i_2})(1-x_{s_1})=f^d_{c_2,i_1}x_{s_2}+f^d_{c_2',i_2}(1-x_{s_2}), \\ c_1,c_1'\in C_{s_1}, c_2,c_2'\in C_{s_2}, s_1,s_2\in S_r, r\in R. \end{aligned} \tag{7}$$

For ship route $r$ with no butterfly port ($r \in R \setminus \hat{R}$), we also assume that container loading operations occur at the tail port of call and container discharging operations only occur at the head port of call:

$$\begin{aligned} &\tilde{z}^d_{c,1}(1-x_s)=0 \\ &\hat{z}^d_{c,N_r+1}(1-x_s)=0 \\ &\hat{z}^d_{c,1}x_s=0 \\ &\tilde{z}^d_{c,N_r+1}x_s=0 \end{aligned} \quad I_{rc}=I_{rs}=I_r \cup \{N_r+1\}, r\in R\setminus\hat{R}, d\in P. \tag{8}$$

As mentioned previously, if there are only two legs in a subloop, there is no difference between clockwise direction and counter-clockwise direction. Hereon, the decision variables $x_s$ are assumed to equal 0 for all the subloops with only two ports. Mathematically, we have:

$$x_s=0,\quad s\in\bar{S}_r, r\in\hat{R} \tag{9}$$

$$x_s=0,\quad S_r=\{s\}, N_r=2, r\in R\setminus\hat{R}. \tag{10}$$

**3.2 Destination-based model formulation**

We develop a destination-based model for the subloop-based liner shipping network alteration. To reflect transshipment cost and slot-purchasing cost, we define the variables:

$\bar{z}_p$ : Number of transshipment containers (TEUs/week) at port $p\in P$;

$y_{od}$ : Number of containers (TEUs/week) transported by the shipping network for origin-destination (O-D) pair $(o,d)\in W$ .

Based on the above descriptions, the objective is to minimize the generalized network-wide cost, which is formulated as follows:

$$\begin{aligned} \min \ & \sum_{p\in P}(\bar{c}_p+\alpha t_p)\bar{z}_p+\alpha\sum_{r\in R}\sum_{s\in S_r}\left[(1-x_s)\sum_{c\in C_s}\sum_{i=t(c)}^{h(c)-1}t_{ci}\sum_{d\in P}f^d_{ci}\right] \\ & +\alpha\sum_{r\in R}\sum_{s\in S_r}\left[x_s\sum_{c\in C_s}\sum_{i=t(c)+1}^{h(c)}t_{c,i-1}\sum_{d\in P}f^d_{ci}\right]+\sum_{(o,d)\in W}(g^{od}+\alpha T^{od})(q^{od}-y^{od}) \end{aligned} \tag{11}$$

subject to:

$$(f^d_{c,i}+\hat{z}^d_{c,i+1}-f^d_{c,i+1}-\tilde{z}^d_{c,i+1})(1-x_s)=0,\quad i\in I_{rc}\setminus\{h(c)\}, c\in C_s, s\in S_r, r\in R, d\in P \tag{12}$$

$$(f_{c,i}^{d}+\hat{z}_{c,i-1}^{d}-f_{c,i-1}^{d}-\tilde{z}_{c,i-1}^{d})x_s=0,\quad i\in I_{rc}\setminus\{t(c)\},c\in C_s,s\in S_r,r\in R,d\in P \tag{13}$$

$$(f_{c,N_r+1}^{d}+\hat{z}_{c,1}^{d}-f_{c,1}^{d}-\tilde{z}_{c,1}^{d})(1-x_s)=0,\quad I_{rc}=I_{rs}=I_r\cup\{N_r+1\},r\in R\setminus\hat{R},d\in P \tag{14}$$

$$(f_{c,1}^{d}+\hat{z}_{c,N_r+1}^{d}-f_{c,N_r+1}^{d}-\tilde{z}_{c,N_r+1}^{d})x_s=0,\quad I_{rc}=I_{rs}=I_r\cup\{N_r+1\},r\in R\setminus\hat{R},d\in P \tag{15}$$

$$\overline{z}_p=\sum_{r\in R}\sum_{s\in S_r}\sum_{c\in C_s}\sum_{i\in I_{rc},p_{ci}=p}\sum_{d\in P}\hat{z}_{ci}^{d}-\sum_{(p,d)\in W}y^{pd},\quad p\in P \tag{16}$$

$$\sum_{r\in R}\sum_{s\in S_r}\sum_{c\in C_s}\sum_{i\in I_{rc},p_{ci}=p}(\hat{z}_{ci}^{d}-\tilde{z}_{ci}^{d})=\begin{cases}-\sum_{(o,d)\in W}y^{od} & p=d\\ y^{pd} & p\neq d\end{cases},d\in P,p\in P \tag{17}$$

$$\sum_{d\in P}f_{ci}^{d}\leq E_r,\quad i\in I_{rc},c\in C_s,s\in S_r,r\in R \tag{18}$$

$$y^{od}\leq q^{od},\quad (o,d)\in W \tag{19}$$

$$x_s\in\{0,1\},\quad s\in S_r,r\in R \tag{20}$$

$$\hat{z}_{c,i}^{d}\geq 0,\tilde{z}_{c,i}^{d}\geq 0,\quad i\in I_{rc},c\in C_s,s\in S_r,r\in R,d\in P \tag{21}$$

$$f_{c,i}^{d}\geq 0,\quad i\in I_{rc},c\in C_s,s\in S_r,r\in R,d\in P \tag{22}$$

$$y^{od}\geq 0,\quad (o,d)\in W. \tag{23}$$

Eq. (11) is the objective function. The generalized network-wide cost subsumes four terms: the first term is the container transshipment cost and inventory cost pertinent to connection time at transshipment, the second term and third terms correspond to the inventory cost with respect to the sailing time depending on whether the directions of subloops are reversed. The fourth term is the slot-purchasing cost and the associated inventory cost.

Eqs. (12) and (13) are the container flow conservation equations at each port of call except for the head and the tail of a section. Eqs. (14) and (15) are the container flow conservation equations of the head and the tail of the sole section in ship route $r\in R\setminus\hat{R}$. Eq. (16) defines the number of transshipment containers at each port. In detail, containers can be loaded onto a ship at port $p$ since either port $p$ is the origin of these containers or it is a transshipment port. Eq. (17) denotes the number of transported containers to and from port $p$. Eq. (18) enforces the ship capacity constraint. Eq. (19) imposes that the number of transported containers for each O-D pair cannot exceed the associated demand. Eq. (20) presents that $x_s$ are binary variables. Eqs. (21)-(23) are nonnegativity constraints on container flow variables.

Combining the constraints pertinent to butterfly ports and some special subloops in Section 3.1, the destination-based model for the subloop-based reversal of port rotation directions (called model SRPRD1) is expressed as:

*Objective function* (11)

*subject to constraints* (2)-(10) and (12)-(23).

### 3.3 Computational complexity

Computational complexity theory indicates that problems differ in the efforts needed to solve them (e.g. Garey and Johnson, 1979; Papadimitriou, 2003; Ibarra-Rojas and Rios-Solis, 2012; Chen et al., 2015). If an optimization problem belongs to the NP-hard class, a polynomial-time algorithm for the exact solution of this problem does not exist unless P = NP. In this paper, we attempt to prove that the problem of SRPRD belongs to the NP-hard class by proving that its decision version is NP-complete. We propose a polynomial reduction from Partition Problem (PP), whose NP-completeness is assured by Chopra and Rao (1993), to the decision version of SRPRD. With this proof we also guarantee that similar problems like the work of Wang and Meng (2013) are NP-hard. The complexity proof of SRPRD is provided in Appendix A.1.

## 4. Solution methods

Model SRPRD1 is a mixed-integer optimization model with nonlinear objective function and several nonlinear constraints. For solving model SRPRD1, we transform it into an equivalent mixed-integer linear programming model in Section 4.1, by linearizing the objective function (11) and the nonlinear constraints (2)-(8) and (12)-(15). Due to the special problem structure, we develop a Benders decomposition algorithm and a metaheuristic for solving the resulting mixed-integer linear program in Section 4.2 and Section 4.3 respectively.

### 4.1 An equivalent mixed-integer linear programming formulation

In the objective function (11), the second and third cost terms contain nonlinear variables $x_s f_{ci}^d$. First, we adjust the sequence of cost terms in Eq. (11), where all the variables in the first three adjusted terms are linear and variables $x_s f_{ci}^d$ are put in the fourth term:

$$\min \sum_{p\in P}(\bar{c}_p+\alpha t_p)\bar{z}_p+\alpha\sum_{r\in R}\sum_{s\in S_r}\sum_{c\in C_s}\sum_{i=t(c)}^{h(c)-1}t_{ci}\sum_{d\in P}f_{ci}^d+\sum_{(o,d)\in W}(g^{od}+\alpha T^{od})(q^{od}-y^{od})$$
$$+\alpha\sum_{r\in R}\sum_{s\in S_r}x_s\sum_{c\in C_s}\left[\sum_{i=t(c)+1}^{h(c)-1}(t_{c,i-1}-t_{ci})\sum_{d\in P}f_{ci}^d+t_{c,h(c)-1}\sum_{d\in P}f_{c,h(c)}^d-t_{c,t(c)}\sum_{d\in P}f_{c,t(c)}^d\right]. \tag{24}$$

We intend to linearize the fourth term in Eq. (24) by means of the big-M modeling method. Let $\tau_s$ be an auxiliary continuous variable and $M_{1,s}$ ($s\in S_r, r\in R$) be a large positive number.

Since model SRPRD1 is a minimization problem, objective function (24) can be transformed into the following objective function (25) by introducing the additional constraints (26) and (27):

$$\min \sum_{p\in P}(\bar{c}_p+\alpha t_p)\bar{z}_p+\alpha\sum_{r\in R}\sum_{s\in S_r}\sum_{c\in C_s}\sum_{i=t(c)}^{h(c)-1}t_{ci}\sum_{d\in P}f_{ci}^d-\sum_{(o,d)\in W}(g^{od}+\alpha T^{od})y^{od}+\alpha\sum_{r\in R}\sum_{s\in S_r}\tau_s \tag{25}$$

$$\tau_s\ge\sum_{c\in C_s}\left[\sum_{i=t(c)+1}^{h(c)-1}(t_{c,i-1}-t_{ci})\sum_{d\in P}f_{ci}^d+t_{c,h(c)-1}\sum_{d\in P}f_{c,h(c)}^d-t_{c,t(c)}\sum_{d\in P}f_{c,t(c)}^d\right]-M_{1,s}(1-x_s),\quad s\in S_r, r\in R \tag{26}$$

$$\tau_s\ge -M_{1,s}x_s,\quad s\in S_r, r\in R\ . \tag{27}$$

Note that the sum of $(g^{od}+\alpha T^{od})q^{od}$ for all O-D pairs in the objective function (24) is a constant value which is dropped in Eq. (25). For a mixed-integer program with big-M constraints, larger values of big-M parameters can result in a weaker linear relaxation, rendering the model more time-consuming to solve. Hence, it would be better to set an opportune upper limit value for each big-M parameter.

**Proposition 1**. The objective function (24) is equivalent to Eqs. (25)-(27) when the value of $M_{1,s}$ ($s\in S_r, r\in R$) is set to be equal to $\sum_{c\in C_s}\left(\sum_{i=t(c)+1}^{h(c)-1}\left|t_{c,i-1}-t_{ci}\right|+\max(t_{c,h(c)-1},t_{c,t(c)})\right)E_r$ .

The proof of Proposition 1 is provided in Appendix A.2. Constraints (12) and (13) contain three nonlinear parts ($x_s f_{c,i}^d$, $x_s\hat{z}_{c,i}^d$ and $x_s\tilde{z}_{c,i}^d$). Let $M_{2,rd}$ ($r\in R, d\in P$) be a large positive number, constraints (12) and (13) can be linearized as:

$$\begin{aligned}&f_{c,i}^d+\hat{z}_{c,i+1}^d-f_{c,i+1}^d-\tilde{z}_{c,i+1}^d\le M_{2,rd}x_s\\&f_{c,i}^d+\hat{z}_{c,i+1}^d-f_{c,i+1}^d-\tilde{z}_{c,i+1}^d\ge -M_{2,rd}x_s\end{aligned}\quad i\in I_{rc}\setminus\{h(c)\}, c\in C_s, s\in S_r, r\in R, d\in P \tag{28}$$

$$\begin{aligned} f_{c,i}^{d} + \hat{z}_{c,i-1}^{d} - f_{c,i-1}^{d} - \tilde{z}_{c,i-1}^{d} &\leq M_{2,rd}(1-x_s) \\ f_{c,i}^{d} + \hat{z}_{c,i-1}^{d} - f_{c,i-1}^{d} - \tilde{z}_{c,i-1}^{d} &\geq -M_{2,rd}(1-x_s) \end{aligned} \quad i \in I_{rc} \setminus \{t(c)\}, c \in C_s, s \in S_r, r \in R, d \in P \ . \tag{29}$$

**Proposition 2**. When $M_{2,rd}$ ($r \in R, d \in P$) is set as $E_r + \sum_{o \in P} q^{od}$, the two nonlinear constraints (12) and (13) are equivalent to Eqs. (28) and (29).

The proof of Proposition 2 is given in Appendix A.3. Similar to Proposition 2, we set $M_{3,d}$ ($d \in P$) to be $2\sum_{o \in P} q^{od}$. Then, nonlinear constraints (2)-(5) and (8) can be linearized as:

$$\begin{aligned} \hat{z}_{c_1,i_1}^{d} + \tilde{z}_{c_1',i_2}^{d} &\leq M_{3,d} x_{s_1} \\ \tilde{z}_{c_2,i_1}^{d} + \hat{z}_{c_2',i_2}^{d} &\leq M_{3,d} x_{s_2} \\ \hat{z}_{c_1',i_2}^{d} + \tilde{z}_{c_1,i_1}^{d} &\leq M_{3,d}(1-x_{s_1}) \\ \tilde{z}_{c_2',i_2}^{d} + \hat{z}_{c_2,i_1}^{d} &\leq M_{3,d}(1-x_{s_2}) \end{aligned} \quad \begin{aligned} & i_1 = h(c_1) = t(c_2), i_2 = t(c_1') = h(c_2') \\ & c_1, c_1' \in C_{s_1}, c_2, c_2' \in C_{s_2}, s_1, s_2 \in S_r \\ & I_{rp} = \{i_1, i_2\}, r \in \hat{R}, p \in P_b \end{aligned} \tag{30}$$

$$\begin{aligned} \tilde{z}_{c,1}^{d} + \hat{z}_{c,N_r+1}^{d} &\leq M_{3,d} x_s \\ \hat{z}_{c,1}^{d} + \tilde{z}_{c,N_r+1}^{d} &\leq M_{3,d}(1-x_s) \end{aligned} \quad I_{rc} = I_{rs} = I_r \cup \{N_r + 1\}, r \in R \setminus \hat{R}, d \in P \ . \tag{31}$$

After examining the property of the problem, we find that constraints (30) and (31) can be further simplified. These two constraints are based on the assumption that container loading operations occur at the tail port of call while container discharging operations occur at the head port of call. Such assumption can be substituted by the following two cases.

For a butterfly port $p \in P_b$ in route $r \in \hat{R}$, it connects two contiguous subloops $s_1$ and $s_2$ ($s_1, s_2 \in S_r$), and corresponds to two ports of call $I_{rp} = \{i_1, i_2\}$. Ports of call $i_1$ and $i_2$ are also two ports of call in both subloops $s_1$ and $s_2$. We assume that container loading and discharging operations occur at only one subloop (either subloop $s_1$ or $s_2$). It is equivalent to the assumption used in Eqs. (30) because the value of objective function (11) and associated constraints (16)-(17) in model SRPRD1 are not affected.

Hereon, subloop $s_1$ is chosen as the single subloop conducting loading and discharging operations. In this case, the number of containers that are loaded and discharged at both head and tail ports of call of subloop $s_2$ is assumed to equal 0. Then, Eq. (30) is transformed to:

$$\hat{z}^d_{c_2,i_1} + \tilde{z}^d_{c_2,i_1} + \hat{z}^d_{c'_2,i_2} + \tilde{z}^d_{c'_2,i_2} = 0, \quad \begin{array}{l} i_1 = h(c_1) = t(c_2), i_2 = t(c'_1) = h(c'_2) \\ c_1, c'_1 \in C_{s_1}, c_2, c'_2 \in C_{s_2}, s_1, s_2 \in S_r \\ I_{rp} = \{i_1, i_2\}, r \in \hat{R}, p \in P_b. \end{array} \tag{32}$$

Similarly, for route $r \in R \setminus \hat{R}$, we assume that container loading and discharging operations only occur at the head port of call (i.e. $N_r + 1$). It is equivalent to the assumption used in Eqs. (31), and hence Eq. (31) is transformed to:

$$\hat{z}^d_{c,1} + \tilde{z}^d_{c,1} = 0, \quad I_{rc} = I_{rs} = I_r \cup \{N_r + 1\}, r \in R \setminus \hat{R}, d \in P \ . \tag{33}$$

It is worth noticing that Eqs. (32) and (33) are independent of decision variables $x_s$ ($s \in S_r$, $r \in R$). Let $M_{4,r}$ ($r \in R$) be equal to $E_r$. According to Proposition 2 and Eqs. (32)-(33), nonlinear constraints (6)-(7) with respect to butterfly ports $p \in P_b$ in route $r \in \hat{R}$ and nonlinear constraints (14)-(15) pertinent to routes $r \in R \setminus \hat{R}$ can be linearized in an analogous way, which are shown below:

$$\begin{array}{l} f^d_{c_1,i_1} - f^d_{c_2,i_1} \le M_{4,r}(x_{s_1} + x_{s_2}) \\ f^d_{c_2,i_1} - f^d_{c_1,i_1} \le M_{4,r}(x_{s_1} + x_{s_2}) \\ f^d_{c'_1,i_2} - f^d_{c_2,i_1} \le M_{4,r}(1 - x_{s_1} + x_{s_2}) \\ f^d_{c_2,i_1} - f^d_{c'_1,i_2} \le M_{4,r}(1 - x_{s_1} + x_{s_2}) \\ f^d_{c_1,i_1} - f^d_{c'_2,i_2} \le M_{4,r}(1 + x_{s_1} - x_{s_2}) \\ f^d_{c'_2,i_2} - f^d_{c_1,i_1} \le M_{4,r}(1 + x_{s_1} - x_{s_2}) \\ f^d_{c'_1,i_2} - f^d_{c'_2,i_2} \le M_{4,r}(2 - x_{s_1} - x_{s_2}) \\ f^d_{c'_2,i_2} - f^d_{c'_1,i_2} \le M_{4,r}(2 - x_{s_1} - x_{s_2}) \end{array} \quad \begin{array}{l} i_1 = h(c_1) = t(c_2), i_2 = t(c'_1) = h(c'_2) \\ c_1, c'_1 \in C_{s_1}, c_2, c'_2 \in C_{s_2}, s_1, s_2 \in S_r \\ I_{rp} = \{i_1, i_2\}, r \in \hat{R}, p \in P_b \end{array} \tag{34}$$

$$\begin{array}{l} (f^d_{c'_1,i_2} + \tilde{z}^d_{c'_1,i_2} - \hat{z}^d_{c'_1,i_2}) - f^d_{c'_2,i_2} \le M_{2,rd}(x_{s_1} + x_{s_2}) \\ f^d_{c'_2,i_2} - (f^d_{c'_1,i_2} + \tilde{z}^d_{c'_1,i_2} - \hat{z}^d_{c'_1,i_2}) \le M_{2,rd}(x_{s_1} + x_{s_2}) \\ (f^d_{c_1,i_1} + \tilde{z}^d_{c_1,i_1} - \hat{z}^d_{c_1,i_1}) - f^d_{c'_2,i_2} \le M_{2,rd}(1 - x_{s_1} + x_{s_2}) \\ f^d_{c'_2,i_2} - (f^d_{c_1,i_1} + \tilde{z}^d_{c_1,i_1} - \hat{z}^d_{c_1,i_1}) \le M_{2,rd}(1 - x_{s_1} + x_{s_2}) \\ (f^d_{c'_1,i_2} + \tilde{z}^d_{c'_1,i_2} - \hat{z}^d_{c'_1,i_2}) - f^d_{c_2,i_1} \le M_{2,rd}(1 + x_{s_1} - x_{s_2}) \\ f^d_{c_2,i_1} - (f^d_{c'_1,i_2} + \tilde{z}^d_{c'_1,i_2} - \hat{z}^d_{c'_1,i_2}) \le M_{2,rd}(1 + x_{s_1} - x_{s_2}) \\ (f^d_{c_1,i_1} + \tilde{z}^d_{c_1,i_1} - \hat{z}^d_{c_1,i_1}) - f^d_{c_2,i_1} \le M_{2,rd}(2 - x_{s_1} - x_{s_2}) \\ f^d_{c_2,i_1} - (f^d_{c_1,i_1} + \tilde{z}^d_{c_1,i_1} - \hat{z}^d_{c_1,i_1}) \le M_{2,rd}(2 - x_{s_1} - x_{s_2}) \end{array} \quad \begin{array}{l} i_1 = h(c_1) = t(c_2), i_2 = t(c'_1) = h(c'_2) \\ c_1, c'_1 \in C_{s_1}, c_2, c'_2 \in C_{s_2}, s_1, s_2 \in S_r \\ I_{rp} = \{i_1, i_2\}, r \in \hat{R}, p \in P_b \end{array} \tag{35}$$

$$\begin{aligned} f^{d}_{c,N_r+1} - f^{d}_{c,1} &\le M_{4,r} x_s \\ f^{d}_{c,N_r+1} - f^{d}_{c,1} &\ge -M_{4,r} x_s \end{aligned} \quad I_{rc} = I_{rs} = I_r \cup \{N_r + 1\}, r \in R \setminus \hat{R}, d \in P \tag{36}$$

$$\begin{aligned} f^{d}_{c,1} + \hat{z}^{d}_{c,N_r+1} - f^{d}_{c,N_r+1} - \tilde{z}^{d}_{c,N_r+1} &\le M_{2,rd}(1 - x_s) \\ f^{d}_{c,1} + \hat{z}^{d}_{c,N_r+1} - f^{d}_{c,N_r+1} - \tilde{z}^{d}_{c,N_r+1} &\ge -M_{2,rd}(1 - x_s) \end{aligned} \quad I_{rc} = I_{rs} = I_r \cup \{N_r + 1\}, r \in R \setminus \hat{R}, d \in P\ . \tag{37}$$

As a result, model SRPRD1 can be reformulated to an equivalent mixed-integer linear programming (MILP) model (named SRPRD2), which can be solved by off-the-shelf MILP solvers. Model SRPRD2 is expressed as:

*Objective function* (25)

*subject to constraints* (9)-(10), (16)-(23), (26)-(29) and (32)-(37).

## 4.2 Benders decomposition

Model SRPRD2 is a mixed-integer linear programming model with a limited number of binary variables ( $X = \{x_s \mid s \in S_r, r \in R\}$ ). At the same time, all the remaining variables are continuous ( $\{\tau_s, \bar{z}_p, y^{od}, f^{d}_{c,i}, \hat{z}^{d}_{c,i}, \tilde{z}^{d}_{c,i} \mid i \in I_{rc}, c \in C_s, s \in S_r, r \in R,\ p, o, d \in P\}$ ). This property enables Benders decomposition (BD) to be an appropriate solution algorithm to solve model SRPRD2. BD is known to be an efficient method for solving MILP models (Benders, 1962). The basic idea of the method is to iteratively solve a master problem that involves only the binary variables and a dual subproblem which is the dual of the remaining linear program. The extreme rays and points generated by solving the dual subproblem can be used to define the feasibility requirements and the projected costs of the binary variables in the master problem, guiding the search process toward an optimal solution of model SRPRD2.

### 4.2.1 Benders subproblem

For given $X = \bar{X} = \{\bar{x}_s \mid s \in S_r, r \in R\}$ satisfying constraints (9), (10) and (20), model SRPRD2 reduces to the following primal subproblem:

*Primal Subproblem* ( $PSP(\bar{X})$ )

$$\min\ Z(\bar{X}) = \sum_{p \in P} (\bar{c}_p + \alpha t_p) \bar{z}_p + \alpha \sum_{r \in R} \sum_{s \in S_r} \sum_{c \in C_s} \sum_{i = t(c)}^{h(c)-1} t_{ci} \sum_{d \in P} f^{d}_{ci} - \sum_{(o,d) \in W} (g^{od} + \alpha T^{od}) y^{od} + \alpha \sum_{r \in R} \sum_{s \in S_r} \tau_s \tag{38}$$

subject to constraints (16)-(19), (21)-(23), (26)-(29) and (32)-(37), where all the binary variables $x_s\ (s \in S_r, r \in R)$ are replaced with the $\bar{x}_s$ values accordingly.

Let $\vartheta_p$, $\{\varsigma_{od}^{\beta} \mid \beta = 1,2\}$, $\rho_{ci}$, $\{\chi_s^{\beta} \mid \beta = 1,2\}$, $\{\mu_{d,ci}^{\beta} \mid \beta = 1,2,...,4\}$, $\{\gamma_{d,p}^{1,\beta}, \gamma_{d,p}^{2,\beta} \mid \beta = 1,2...,8\}$, $\gamma_{d,p}^{9}$, $\{\phi_{d,s}^{\beta} \mid \beta = 1,2,...,5\}$ denote the dual variables associated with constraints (16)-(19), (26)-(29) and (32)-(37) respectively in the primal subproblem. Then, the dual of the primal subproblem is the following dual subproblem:

*Dual Subproblem* ( $DSP(\bar{X})$ )

$$\begin{aligned}
\max\ F(\bar{X}) = & \sum_{(o,d)\in W} q^{od}\varsigma_{od}^{2} + \sum_{r\in R}\sum_{s\in S_r}\sum_{c\in C_s}\sum_{i\in I_{rc}} E_r\rho_{ci} + \sum_{r\in R}\sum_{s\in S_r} M_{1,s}(1-\bar{x}_s)\chi_s^1 + \sum_{r\in R}\sum_{s\in S_r} M_{1,s}\bar{x}_s\chi_s^2 \\
& + \sum_{r\in R}\sum_{s\in S_r}\bar{x}_s\sum_{c\in C_s}\sum_{i=t(c)}^{h(c)-1}\sum_{d\in P} M_{2,rd}(\mu_{d,ci}^1+\mu_{d,ci}^2) + \sum_{r\in R}\sum_{s\in S_r}(1-\bar{x}_s)\sum_{c\in C_s}\sum_{i=t(c)+1}^{h(c)}\sum_{d\in P} M_{2,rd}(\mu_{d,ci}^3+\mu_{d,ci}^4) \\
& + \sum_{r\in R\backslash\hat{R}}\sum_{s\in S_r} M_{4,r}\bar{x}_s\sum_{d\in P}(\phi_{d,s}^1+\phi_{d,s}^2) + \sum_{r\in R\backslash\hat{R}}\sum_{s\in S_r}(1-\bar{x}_s)\sum_{d\in P} M_{2,rd}(\phi_{d,s}^3+\phi_{d,s}^4) \\
& + \sum_{r\in\hat{R}}\sum_{s_1,s_2\in S_r}\sum_{p\in P_b: I_{rp}=I_{rs_1}\cap I_{rs_2}}\sum_{d\in P}\begin{bmatrix} (\bar{x}_{s_1}+\bar{x}_{s_2})\sum_{k=1}^{2}(M_{4,r}\gamma_{d,p}^{1,k}+M_{2,rd}\gamma_{d,p}^{2,k}) + \\ (1+\bar{x}_{s_1}-\bar{x}_{s_2})\sum_{k=3}^{4}(M_{4,r}\gamma_{d,p}^{1,k}+M_{2,rd}\gamma_{d,p}^{2,k}) + \\ (1-\bar{x}_{s_1}+\bar{x}_{s_2})\sum_{k=5}^{6}(M_{4,r}\gamma_{d,p}^{1,k}+M_{2,rd}\gamma_{d,p}^{2,k}) + \\ (2-\bar{x}_{s_1}-\bar{x}_{s_2})\sum_{k=7}^{8}(M_{4,r}\gamma_{d,p}^{1,k}+M_{2,rd}\gamma_{d,p}^{2,k}) \end{bmatrix}
\end{aligned} \tag{39}$$

subject to:

$$\vartheta_p \le \bar{c}_p + \alpha t_p, \quad p \in P \tag{40}$$

$$\begin{aligned} &\vartheta_o + \varsigma_{dd}^1 - \varsigma_{od}^1 + \varsigma_{od}^2 \le -(g^{od}+\alpha T^{od}), \quad o \ne d \\ &\vartheta_o + \varsigma_{od}^1 + \varsigma_{od}^2 \le -(g^{od}+\alpha T^{od}), \qquad o = d \end{aligned} \quad (o,d)\in W \tag{41}$$

$$\chi_s^1 + \chi_s^2 = -\alpha, \quad s \in S_r, r \in R \tag{42}$$

$$\begin{aligned} &\mu_{d,ci}^1 - \mu_{d,c,i-1}^1 - \mu_{d,ci}^2 + \mu_{d,c,i-1}^2 + \mu_{d,ci}^3 - \mu_{d,c,i+1}^3 - \mu_{d,ci}^4 \\ &\quad + \mu_{d,c,i+1}^4 + \rho_{ci} + (t_{c,i-1}-t_{ci})\chi_s^1 \le \alpha t_{ci} \\ &\mu_{d,c,i-1}^1 - \mu_{d,c,i-1}^2 + \mu_{d,c,i+1}^3 - \mu_{d,c,i+1}^4 - \vartheta_{p_{ci}} + \varsigma_{p_{ci}d}^1 \le 0 \\ &-\mu_{d,c,i-1}^1 + \mu_{d,c,i-1}^2 - \mu_{d,c,i+1}^3 + \mu_{d,c,i+1}^4 - \varsigma_{p_{ci}d}^1 \le 0 \end{aligned} \quad \begin{aligned} &i \in I_{rc}\backslash\{h(c),t(c)\}, c \in C_s, \\ &s \in S_r, r \in R, d \in P \end{aligned} \tag{43}$$

$$\begin{aligned}&\mu^1_{d,ci}-\mu^2_{d,ci}-\mu^3_{d,c,i+1}+\mu^4_{d,c,i+1}-\mu^5_{d,ci}+\mu^6_{d,ci}-\phi^1_{d,s}\\&\quad+\phi^2_{d,s}+\phi^3_{d,s}-\phi^4_{d,s}+\rho_{ci}-t_{ci}\chi^1_s\le\alpha t_{ci}\\&\mu^3_{d,c,i+1}-\mu^4_{d,c,i+1}+\phi^5_{d,s}-\vartheta_{p_{ci}}+\varsigma^1_{p_{ci}d}\le 0\\&-\mu^3_{d,c,i+1}+\mu^4_{d,c,i+1}+\phi^5_{d,s}-\varsigma^1_{p_{ci}d}\le 0\end{aligned}\qquad \begin{aligned}&i{=}t(c),c\in C_s,s\in S_r,\\&r\in R\setminus\hat{R},d\in P\end{aligned}\tag{44}$$

$$\begin{aligned}&-\mu^1_{d,c,i-1}+\mu^2_{d,c,i-1}+\mu^3_{d,ci}-\mu^4_{d,ci}+\mu^5_{d,ci}-\mu^6_{d,ci}+\phi^1_{d,s}\\&\quad-\phi^2_{d,s}-\phi^3_{d,s}+\phi^4_{d,s}+\rho_{ci}+t_{c,i-1}\chi^1_s\le 0\\&\mu^1_{d,c,i-1}-\mu^2_{d,c,i-1}+\phi^3_{d,s}-\phi^4_{d,s}-\vartheta_{p_{ci}}+\varsigma^1_{p_{ci}d}\le 0\\&-\mu^1_{d,c,i-1}+\mu^2_{d,c,i-1}-\phi^3_{d,s}+\phi^4_{d,s}-\varsigma^1_{p_{ci}d}\le 0\end{aligned}\qquad \begin{aligned}&i{=}h(c),c\in C_s,s\in S_r,\\&r\in R\setminus\hat{R},d\in P\end{aligned}\tag{45}$$

$$\begin{aligned}&\mu^1_{d,c_2,i_1}-\mu^2_{d,c_2,i_1}-\mu^3_{d,c_2,i_1+1}+\mu^4_{d,c_2,i_1+1}-\gamma^{1,1}_{d,p}+\gamma^{1,2}_{d,p}-\gamma^{1,5}_{d,p}+\gamma^{1,6}_{d,p}\\&\quad-\gamma^{2,3}_{d,p}+\gamma^{2,4}_{d,p}-\gamma^{2,7}_{d,p}+\gamma^{2,8}_{d,p}+\rho_{c_2,i_1}-t_{c_2,i_1}\chi^1_s\le\alpha t_{c_2,i_1}\\&\mu^1_{d,c_1',i_2}-\mu^2_{d,c_1',i_2}-\mu^3_{d,c_1',i_2+1}+\mu^4_{d,c_1',i_2+1}+\gamma^{1,5}_{d,p}-\gamma^{1,6}_{d,p}+\gamma^{1,7}_{d,p}-\gamma^{1,8}_{d,p}\\&\quad+\gamma^{2,1}_{d,p}-\gamma^{2,2}_{d,p}+\gamma^{2,3}_{d,p}-\gamma^{2,4}_{d,p}+\rho_{c_1',i_2}-t_{c_1',i_2}\chi^1_s\le\alpha t_{c_1',i_2}\\&-\mu^1_{d,c_1,i_1-1}+\mu^2_{d,c_1,i_1-1}+\mu^3_{d,c_1,i_1}-\mu^4_{d,c_1,i_1}+\gamma^{1,1}_{d,p}-\gamma^{1,2}_{d,p}+\gamma^{1,3}_{d,p}-\gamma^{1,4}_{d,p}\\&\quad+\gamma^{2,5}_{d,p}-\gamma^{2,6}_{d,p}+\gamma^{2,7}_{d,p}-\gamma^{2,8}_{d,p}+\rho_{c_1,i_1}+t_{c_1,i_1-1}\chi^1_s\le 0\\&-\mu^1_{d,c_2',i_2-1}+\mu^2_{d,c_2',i_2-1}+\mu^3_{d,c_2',i_2}-\mu^4_{d,c_2',i_2}-\gamma^{1,3}_{d,p}+\gamma^{1,4}_{d,p}-\gamma^{1,7}_{d,p}+\gamma^{1,8}_{d,p}\\&\quad-\gamma^{2,1}_{d,p}+\gamma^{2,2}_{d,p}-\gamma^{2,5}_{d,p}+\gamma^{2,6}_{d,p}+\rho_{c_2',i_2}+t_{c_2',i_2-1}\chi^1_s\le 0\end{aligned}\qquad \begin{aligned}&i_1=h(c_1)=t(c_2),i_2=t(c_1')=h(c_2')\\&c_1,c_1'\in C_{s_1},c_2,c_2'\in C_{s_2},s_1,s_2\in S_r\\&I_{rp}=\{i_1,i_2\},r\in\hat{R},p\in P_b,d\in P\end{aligned}\tag{46}$$

$$\begin{aligned}&\mu^3_{d,c_2,i_1+1}-\mu^4_{d,c_2,i_1+1}+\gamma^9_{d,p}-\vartheta_p+\varsigma^1_{pd}\le 0\\&\mu^3_{d,c_1',i_2+1}-\mu^4_{d,c_1',i_2+1}-\gamma^{2,1}_{d,p}+\gamma^{2,2}_{d,p}-\gamma^{2,3}_{d,p}+\gamma^{2,4}_{d,p}-\vartheta_p+\varsigma^1_{pd}\le 0\\&\mu^1_{d,c_1,i_1-1}-\mu^2_{d,c_1,i_1-1}-\gamma^{2,5}_{d,p}+\gamma^{2,6}_{d,p}-\gamma^{2,7}_{d,p}+\gamma^{2,8}_{d,p}-\vartheta_p+\varsigma^1_{pd}\le 0\\&\mu^1_{d,c_2',i_2-1}-\mu^2_{d,c_2',i_2-1}+\gamma^9_{d,p}-\vartheta_p+\varsigma^1_{pd}\le 0\end{aligned}\qquad \begin{aligned}&i_1=h(c_1)=t(c_2),i_2=t(c_1')=h(c_2')\\&c_1,c_1'\in C_{s_1},c_2,c_2'\in C_{s_2},s_1,s_2\in S_r\\&I_{rp}=\{i_1,i_2\},r\in\hat{R},p\in P_b,d\in P\end{aligned}\tag{47}$$

$$\begin{aligned}&-\mu^3_{d,c_2,i_1+1}+\mu^4_{d,c_2,i_1+1}+\gamma^9_{d,p}-\varsigma^1_{pd}\le 0\\&-\mu^3_{d,c_1',i_2+1}+\mu^4_{d,c_1',i_2+1}+\gamma^{2,1}_{d,p}-\gamma^{2,2}_{d,p}+\gamma^{2,3}_{d,p}-\gamma^{2,4}_{d,p}-\varsigma^1_{pd}\le 0\\&-\mu^1_{d,c_1,i_1-1}+\mu^2_{d,c_1,i_1-1}+\gamma^{2,5}_{d,p}-\gamma^{2,6}_{d,p}+\gamma^{2,7}_{d,p}-\gamma^{2,8}_{d,p}-\varsigma^1_{pd}\le 0\\&-\mu^1_{d,c_2',i_2-1}+\mu^2_{d,c_2',i_2-1}+\gamma^9_{d,p}-\varsigma^1_{pd}\le 0\end{aligned}\qquad \begin{aligned}&i_1=h(c_1)=t(c_2),i_2=t(c_1')=h(c_2')\\&c_1,c_1'\in C_{s_1},c_2,c_2'\in C_{s_2},s_1,s_2\in S_r\\&I_{rp}=\{i_1,i_2\},r\in\hat{R},p\in P_b,d\in P\end{aligned}\tag{48}$$

$$\varsigma^2_{od}\le 0,\quad (o,d)\in W\tag{49}$$

$$\rho_{ci}\le 0,\quad i\in I_{rc},c\in C_s,s\in S_r,r\in R\tag{50}$$

$$\chi^1_s,\chi^2_s\le 0,\quad s\in S_r,r\in R\tag{51}$$

$$\begin{aligned}&\mu^1_{d,ci},\mu^2_{d,ci}\le 0,\quad i\in I_{rc}\setminus\{h(c)\}\\&\mu^3_{d,ci},\mu^4_{d,ci}\le 0,\quad i\in I_{rc}\setminus\{t(c)\}\end{aligned}\qquad c\in C_s,s\in S_r,r\in R,d\in P\tag{52}$$

$$\gamma^{1,\beta}_{d,p}, \gamma^{2,\beta}_{d,p} \le 0, \quad \begin{array}{l} \beta = \{1,2,...,8\}, i_1 = h(c_1) = t(c_2), i_2 = t(c_1') = h(c_2'), c_1, c_1' \in C_{s_1}, \\ c_2, c_2' \in C_{s_2}, s_1, s_2 \in S_r, I_{rp} = \{i_1, i_2\}, r \in \hat{R}, p \in P_b, d \in P \end{array} \tag{53}$$

$$\phi^{\beta}_{d,s} \le 0, \quad \beta = \{1,2,...,4\}, I_{rc} = I_{rs} = I_r \cup \{N_r + 1\}, r \in R \setminus \hat{R}, d \in P \ . \tag{54}$$

Introducing the free variable $\theta$, the master problem for determining the values of the $x_s$ variables is formulated as:

*Master Problem* (*MP*)

$$\min \ \theta \tag{55}$$

subject to constraints (9), (10), (20) and

$$\begin{aligned}
\theta \ge & \sum_{r\in R}\sum_{s\in S_r} x_s \sum_{c\in C_s}\sum_{i=t(c)}^{h(c)-1}\sum_{d\in P} M_{2,rd}(\mu^1_{d,ci} + \mu^2_{d,ci}) + \sum_{r\in R}\sum_{s\in S_r}(1-x_s)\sum_{c\in C_s}\sum_{i=t(c)+1}^{h(c)}\sum_{d\in P} M_{2,rd}(\mu^3_{d,ci} + \mu^4_{d,ci}) \\
& + \sum_{r\in R\setminus\hat{R}}\sum_{s\in S_r} M_{4,r} x_s \sum_{d\in P}(\phi^1_{d,s} + \phi^2_{d,s}) + \sum_{r\in R\setminus\hat{R}}\sum_{s\in S_r}(1-x_s)\sum_{d\in P} M_{2,rd}(\phi^3_{d,s} + \phi^4_{d,s}) \\
& + \sum_{r\in R}\sum_{s\in S_r} M_{1,s}(1-x_s)\chi^1_s + \sum_{r\in R}\sum_{s\in S_r} M_{1,s} x_s \chi^2_s + \sum_{r\in R}\sum_{s\in S_r}\sum_{c\in C_s}\sum_{i\in I_{rc}} E_r \rho_{ci} + \sum_{(o,d)\in W} q^{od}\varsigma^2_{od} \\
& + \sum_{r\in\hat{R}}\sum_{s_1,s_2\in S_r}\sum_{p\in P_b : I_{rp} = I_{rs_1}\cap I_{rs_2}}\sum_{d\in P}\left[\begin{array}{l}
(x_{s_1} + x_{s_2})\sum_{k=1}^{2}(M_{4,r}\gamma^{1,k}_{d,p} + M_{2,rd}\gamma^{2,k}_{d,p}) + \\
(1 + x_{s_1} - x_{s_2})\sum_{k=3}^{4}(M_{4,r}\gamma^{1,k}_{d,p} + M_{2,rd}\gamma^{2,k}_{d,p}) + \\
(1 - x_{s_1} + x_{s_2})\sum_{k=5}^{6}(M_{4,r}\gamma^{1,k}_{d,p} + M_{2,rd}\gamma^{2,k}_{d,p}) + \\
(2 - x_{s_1} - x_{s_2})\sum_{k=7}^{8}(M_{4,r}\gamma^{1,k}_{d,p} + M_{2,rd}\gamma^{2,k}_{d,p})
\end{array}\right], \\
& \left(\varsigma, \chi, \mu, \gamma^1, \gamma^2, \phi\right) \in \Lambda
\end{aligned} \tag{56}$$

where $\Lambda$ is a subset of extreme points of the polyhedron defined by constraints (40)-(54). In MP, constraint (56) is called the optimality cut. In a generic framework of BD, MP contains not only a set of optimality cuts, but also a set of feasibility cuts, which ensures that the solution to the master problem yields a bounded dual subproblem. As the primal subproblem is always feasible and bounded, by strong duality, the dual is also feasible and bounded. Therefore, we only add optimality cuts as deemed necessary.

The BD algorithm is depicted as below. In each iteration, MP is solved, and the resulting objective value serves as a lower bound. The solution $\bar{X}$ of MP is utilized to set up $\text{DSP}(\bar{X})$ which is solved subsequently. Once the optimal solution of $\text{DSP}(\bar{X})$ is obtained, the associated objective value serves as an upper bound. The solution of DSP($\bar{X}$) is used to generate a new optimality cut, which is added to MP. This process is performed repeatedly until pre-specified stop criteria are satisfied. In this study, the algorithm terminates if one of the following conditions is activated: (1) the gap between the lower and upper bound is sufficiently small; (2) the number of iterations exceeds a predefined maximum number; and (3) the running time of the algorithm exceeds a predefined time limit.

### 4.2.2 Improving the performance of the BD algorithm

Our preliminary computational results show that a straightforward implementation of BD in Section 4.2.1 results in slow convergence. Since the BD algorithm was introduced, many researchers have explored various methods to improve its performance (Fontaine and Minner, 2014; Arslan and Karaşan, 2016; Bayram and Yaman, 2017; among many others). In this subsection, we develop three acceleration strategies in an effort to further enhance the overall efficacy of BD.

#### *4.2.2.1 Defining Pareto-optimal cuts*

When the primal subproblem $PSP(\bar{X})$ is degenerate, it is common to get multiple optimal solutions of the dual subproblem $DSP(\bar{X})$, for which cuts of different strengths can be generated. We are interested in generating Pareto-optimal cuts, i.e., cuts that are not dominated by any other cuts. According to Magnanti and Wong (1981), Pareto-optimal cuts can be generated by solving the following auxiliary problem:

$$\max \ F(\tilde{X}) \tag{57}$$

subject to constraints (40)-(54) and

$$F(\bar{X}) = v(DSP(\bar{X})) \tag{58}$$

where $\tilde{X} = \{\tilde{x}_s \mid s \in S_r, r \in R\}$ is a core point, i.e., a point in the relative interior of the convex hull of the feasible subloop reversal vector, and $v(DSP(\bar{X}))$ is the optimal objective value of $DSP(\bar{X})$. We start the first iteration of the BD algorithm with $\tilde{x}_s = 1$ ($s \in S_r, r \in R$). The core point is updated at each iteration $k$ ($k \geq 2$) using the equation $\tilde{x}_s^k = 0.5\tilde{x}_s^{k-1} + 0.5\bar{x}_s^k$.

#### *4.2.2.2 Updating big-M coefficients*

In Section 4.1, model SRPRD1 is linearized by means of the big-M method. Since container flow variables ( $f_{c,i}^{d}, \hat{z}_{c,i}^{d}, \tilde{z}_{c,i}^{d}$ ) are pertinent to the particular demand characters of O-D pairs, there may exist several big-M constraints that will never be binding, weakening the bounds that are generated in each iteration of the BD algorithm.

Based on this observation, we develop an acceleration strategy that dynamically updates the values of the big-M parameters during the search process of the BD algorithm. Let $\mathbf{M}^0$ denote the set of the original big-M values, i.e., $\mathbf{M}^0 = \{M_{1,s}, M_{2,rd}, M_{4,r} \mid s \in S_r, r \in R, d \in P\}$. We begin the BD algorithm by employing the original values in $\mathbf{M}^0$. At each iteration *k*, we calculate the value of the right-hand side of each big-M constraint in $PSP(\bar{X})$. For each $s \in S_r$, let $\widehat{M}_{1,s}^{k}$ be the maximum right-hand-side value of constraints (26) and (27); for each $r \in R, d \in P$, let $\widehat{M}_{2,rd}^{k}$ be the maximum right-hand-side value of constraints (28), (29), and (35); and for each $r \in R$, let $\widehat{M}_{2,rd}^{k}$ be the maximum right-hand-side value of constraints (34) and (36).

We update the big-M values as follows. At each iteration *k*, we let $M'_{1,s} = \max_{k'=1,\ldots,k}(\widehat{M}_{1,s}^{k'})$, $M'_{2,rd} = \max_{k'=1,\ldots,k}(\widehat{M}_{2,rd}^{k'})$, and $M'_{4,r} = \max_{k'=1,\ldots,k}(\widehat{M}_{4,r}^{k'})$. These values constitute a new big-M value set $\mathbf{M}^k = \{M'_{1,s}, M'_{2,rd}, M'_{4,r} \mid s \in S_r, r \in R, d \in P\}$. We update the *k*th iteration big-M coefficients by $k\omega\mathbf{M}^1 + (1-k\omega)\mathbf{M}^0$ ($0 < k\omega < 1$), where $\omega$ is a small positive constant value. Then, the updated big-M coefficients are utilized to address $DSP(\bar{X})$.

*4.2.2.3 Generating combinatorial Benders cuts*

The concept of combinatorial Benders (CB) cuts was initially introduced by Codato and Fischetti (2006). The advantage of CB lies in that it can further alleviate the dependency of the algorithm's performance on the big-M values. When using CB cuts, the dual subproblem $DSP(\bar{X})$ is usually updated by adding the following constraint:

$$F(\bar{X}) \le v(DSP(\bar{X})) - \varepsilon \tag{59}$$

where $v(DSP(\bar{X}))$ is the incumbent objective value resulted by the binary vector $\bar{X}$, and $\varepsilon$ is a sufficiently small positive constant value.

If the dual subproblem $DSP(\bar{X})$ with constraint (59) has no feasible solution, then there exists a minimum infeasible subset (MIS) such that at least one of the $\bar{x}_s$ variables in set $\bar{X}$ should change its value to break the infeasibility. Thus, an alternative binary vector $\bar{X}$ can be obtained by adding the following constraint, which is called the CB cut, to the MP:

$$\sum_{\bar{x}_s=0}(1-\bar{x}_s)x_s+\sum_{\bar{x}_s=1}\bar{x}_s(1-x_s)\geq 1 \tag{60}$$

The CB cut is iteratively added to the MP to search for a feasible binary vector $\bar{X}$ that results in a feasible dual subproblem. Once the dual subproblem is feasible, a new upper bound is thus obtained, though the objective value of the MP no longer serves as a valid lower bound (Codato and Fischetti, 2006). The algorithm terminates when the MP becomes infeasible.

Instead of implementing the standard CB method, we modify this method to improve its performance for the problem of SRPRD. We first run the BD algorithm that incorporates two acceleration strategies described in the previous subsections. We obtain an initial feasible solution of the MP by terminating the algorithm after a fixed number of iterations. With this initial solution, we set up the new dual subproblem that is used to generate CB cuts:

*CB Dual Subproblem* ($CB\text{-}DSP(\bar{X})$)

$$\max\ F(\bar{X}^k) \tag{61}$$

subject to constraints (40)-(54) and

$$F(\bar{X}^k)=\min_{k':1,\ldots,k-1}\left(v(DSP(\bar{X}^{k'}))\right)-\varepsilon^k,\quad k\geq 2 \tag{62}$$

where $\bar{X}^k$ is the feasible solution of the MP generated at the *k*th iteration, and $\varepsilon^k$ is a continuous variable. Note that constraint (62) in $CB\text{-}DSP(\bar{X}^k)$ is in a different form as compared to constraint (59) which is used in standard CB method (Codato and Fischetti, 2006).

In the first iteration of the CB method, we initialize $\varepsilon^1$ with a small positive value. At each iteration $k$ ($k\geq 2$), CB-DSP$(\bar{X}^k)$ is solved with $\varepsilon^k$ being a decision variable. If $\varepsilon^k\leq 0$ in the solution of CB-DSP$(\bar{X}^k)$, then it means that the dual subproblem $DSP(\bar{X})$ with constraint (59) is infeasible. Consequently, a MIS should be identified for generating a CB cut for the master problem. As the problem of identifying MISs is NP-hard (Côté et al., 2014), we endeavor to address it in a greedy fashion, which is reflected in the case of $\varepsilon^k>0$ and the new CB master problem.

Specifically, if $\varepsilon^k > 0$, we first update the best incumbent solution. At the same time, we also generate a cut which is similar to the CB cut in the case of $\varepsilon^k \le 0$ and add it to the new CB master problem. Differed from $\varepsilon^k \le 0$, the CB cut generated by $\varepsilon^k > 0$ will influence the objective value. The specific CB Master Problem (CB-MP) is expressed as follows:

*CB Master Problem* (*CB-MP*)

$$\min \sum_{k'=1}^{k} \bar{\varepsilon}^{k'} \left[ \sum_{\bar{x}_s^{k'}=0} (1-\bar{x}_s^{k'}) x_s + \sum_{\bar{x}_s^{k'}=1} \bar{x}_s^{k'} (1-x_s) \right] \tag{63}$$

subject to constraints (9)-(10), (20) and

$$\bar{\varepsilon}^{k'} = \begin{cases} \left(1-\dfrac{k-k'}{N_s}\right)\varepsilon^{k'} & \varepsilon^{k'} \ge 0, 1-\dfrac{k-k'}{N_s} \ge 0 \\ 0 & \text{otherwise} \end{cases} \quad k'=1,2...,k \tag{64}$$

$$\sum_{\bar{x}_s^{k'}=0} (1-\bar{x}_s^{k'}) x_s + \sum_{\bar{x}_s^{k'}=1} \bar{x}_s^{k'} (1-x_s) \ge 1, \quad k'=1,2...,k \tag{65}$$

where $N_s$ denotes the number of subloops in the shipping network.

In Eq. (63), we utilize the weighted objective function: (1) weigh each CB cut with $\varepsilon^k \le 0$ by 0, and (2) weigh each CB cut with $\varepsilon^k > 0$ by $\bar{\varepsilon}^k$ which is associated with $\varepsilon^k$ and iterations. By iteratively setting to zero some weights of the cuts, we can readily detect alternative MISs, which is important to generate strong CB cuts in the process of the CB separation. In this way, the CB cuts will translate both the feasibility and the optimality requirements. This acceleration strategy terminates when either of the following two stop criteria is met: (1) In CB-MP, the objective value of Eq. (63) equals 0; (2) Iterator exceeds a predefined maximum number.

### 4.3 Metaheuristic method

In addition to the BD algorithm, we also develop a multi-start iterative local search (MILS) algorithm for the problem of SRPRD. The MILS algorithm is a combination of constructive heuristics and local search heuristics (Lourenço et al., 2010; Chen et al., 2018). Constructive heuristics are utilized to generate multi-start initial solutions from scratch, while local search heuristics execute local changes of solutions in the search space until a solution deemed optimal is garnered or the number of iterations is achieved.

First, we define set $S=\bigcup_{r\in R} S_r$. Set $\bar{S}$ denotes the set of subloops with original directions (i.e. $x_s=0, s\in\bar{S}$). In other words, the directions of all the subloops in set $S\setminus\bar{S}$ are reversed. The hybrid MILS solution algorithm proceeds as below (see Algorithm 1). We define a number of main iterations at step 3. In each iteration, we build an initial solution at step 4 (see Algorithm 2). Then, we define the iterations of the iterated local search. In each one, we conduct a hill climbing local search to obtain the best perturbed solution at step 7 (see Algorithm 3). This new solution is accepted only if it yields a further reduction in the generalized cost compared with the incumbent local optimum solution at step 8. The details of the components $Generation(\bar{S})$ and $Climbing(m)$ are described in the following two subsections.

**Algorithm 1.** Hybrid MILS (model SRPRD1)

---

**Input:** model SRPRD1 instance
**Output:** a solution $\bar{m}$ for model SRPRD1
1: Define set $\bar{S}$ as an empty set
2: Search subloops which satisfy Eqs. (9) and (10), and store them in set $\bar{S}$
3: **for** ( $it=1$ to $it=iter_{MULTI}$ ) **do**
4: $m \leftarrow Generation(\bar{S})$
5: $iter=1$
6: **while** ( $iter<iter_{ILS}$ ) **do**
7: $m' \leftarrow Climbing(m)$
8: **if** ( $z(m')<z(m)$ ) **then**
9: $m^* \leftarrow m'$
10: $iter=iter+1$
11: $m \leftarrow Generation($ $\bar{S}$ of solution $m'$ )
12: **else**
13: $iter=iter_{ILS}$
14: **end if**
15: **end while**
16: **if** ( $it==1$ ) **then**
17: $\bar{m}=m^*$
18: **end if**
19: $\bar{m} \leftarrow \mathrm{argmin}\{ z(m^*), z(\bar{m}) \}$
20: **end for**
21: Obtain a near-optimal solution $\bar{m}$

---

### 4.3.1 Initial solution generation

Algorithm 2 shows the generating phase of initial solution *m*. According to Eqs. (9) and (10), the directions of subloops with only two ports are not essential to be reversed. We first seek out these special subloops and store them in initial set $\bar{S}$ (see step 1 and 2 in Algorithm 1). Afterward, we randomly select some subloops from the remaining ones in set $S \setminus \bar{S}$, and store them in set $\hat{S}$. Update set $\bar{S}$ which subsumes all the subloops keeping with original directions. When set $\bar{S}$ is given, model SRPRD1 becomes a linear programming model with pure continuous variables ($\bar{z}_p, y^{od}, f^d_{c,i}, \hat{z}^d_{c,i}, \tilde{z}^d_{c,i}$). It can be efficiently solved by off-the-shelf optimization solvers such as CPLEX.

**Algorithm 2.** *Generation*($\bar{S}$)

---

**Input:** model formulation and set $S \setminus \bar{S}$
**Output:** initial solution *m*
1: Define set $\hat{S}$ as an empty set
2: Randomly select some subloops in set $S \setminus \bar{S}$ (The probability of each subloop being selected is assumed to be $p_r$). and store them in set $\hat{S} \subseteq S \setminus \bar{S}$
3: Update set $\bar{S} \leftarrow \bar{S} \bigcup \hat{S}$
4: Compute the model SRPRD1 considering the updated set $\bar{S}$
5. Obtain the initial solution *m*

---

### 4.3.2 Hill climbing local search

Given an initial solution *m*, the hill climbing local search is utilized to search the local optimum solution by incrementally altering the rotation direction of a single subloop. In Algorithm 3, we define a number of iterations (step 1). In each iteration, we only switch the direction of one subloop. If the selected subloop $s$ is not in set $\bar{S}$, we add it (step 3) while if such subloop $s$ is already in set $\bar{S}$, we delete it (step 6). Afterward, we employ the updated set $\bar{S}$ to generate new solutions using Algorithm 2 and achieve its local optimum at step 9.

**Algorithm 3.** *Climbing*(*m*)

---

**Input:** model formulation, solution *m* and set $\bar{S}$
**Output:** optimum solution $m'$ via local search
1: **for** (all subloops $s \in S$) **do**
2: **if** (subloop $s \in \bar{S}$) **then**
3: $m_k \leftarrow Generation(\bar{S} \setminus \{s\})$

---

**end if**
**if** (subloop $s \in S \setminus \bar{S}$ ) **then**
  $m_k \leftarrow Generation(\bar{S} \cup \{s\})$
**end if**
**end for**
$m' \leftarrow \text{argmin}\{ z(m_s) \,|\, s \in S \}$

## 5 Numerical Examples

In this section, case studies are conducted based on three small examples and a real-case example. The solution methods are coded in Matlab R2017b using CPLEX of version 12.8, and implemented on a desktop with Intel Core Quad CPU Q9550 @ 2.83 GHz and 8.00 G RAM.

A constant demand multiplier (*DM*) is employed to reflect the variations of demand. For instance, *DM* = 1 denotes the original demand, and *DM* = 2 implies that the demand of each O-D pair is doubled. The parameters used are set as follows: the transshipment cost is $\bar{c}_p$=150 USD/TEU, the unit inventory cost rate is $\alpha$ = 0.2 USD/TEU/h, and the connection time is $t_p$ = 3.5 days (84 h) for all ports. The slot-purchasing cost $g^{od}$ and the transit time $T^{od}$ are assumed to be:

$$g^{od} := 1000 + 0.2 \times \text{Distance between the two ports (n mile)}, \quad \forall (o,d) \in W \tag{66}$$

$$T^{od} := 7 \times 24 + \text{Distance between the two ports (n mile)/15 knots}, \quad \forall (o,d) \in W \ . \tag{67}$$

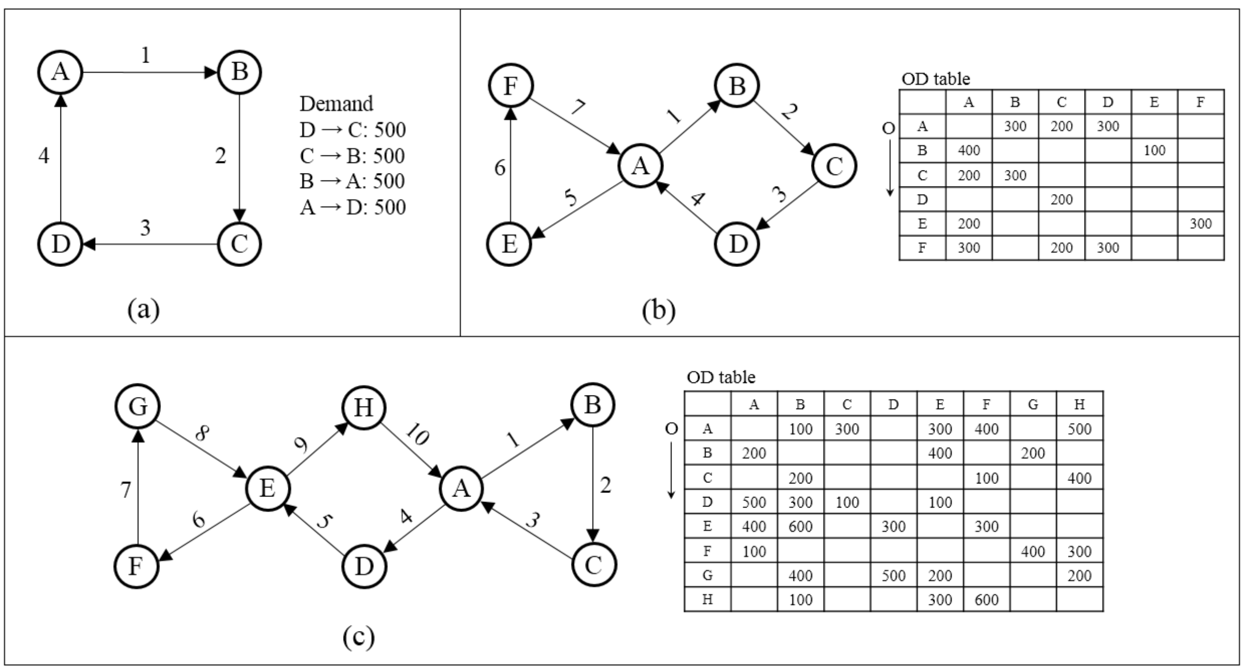
### 5.1 Small examples

Three small examples, set out in Fig. 6, are utilized to analyze the potential benefits of subloop-based reversal of port rotation directions at the outset. Fig. 6 presents the original port rotation and demand for O-D pairs. Since the number of binary variables in set $X$ is not more than three, we just enumerate all the scenarios for the set $\bar{X} = \{\bar{x}_s \,|\, s \in S_r, r \in R\}$ and calculate the associated $PSP(\bar{X})$ via CPLEX. The computation time for each scenario is less than 0.1 second.

The first example in Fig. 6a is a simple case with four special O-D pairs. The ship route has a single subloop and the size of ships is taken as 1500 TEUs. When *DM* = 1, if the port rotation direction is clockwise, the maximum volume of containers that can be transported is 2000 TEUs. The liner shipping company needs to purchase ship slots from other companies once *DM* > 1. By contrast, if the port rotation direction is reversed, all the 6000 containers can be shipped even if *DM* = 3. In Fig. 7a, though the maximum inventory cost of containers in the shipping process (the shaded area in the figure) is identical, the total cost heavily depends on the port rotation direction.

In Fig. 6b, the ship route contains two outer subloops and the size of ships is set as 3000 TEUs. Fig. 7b depicts the comparison results of four scenarios for the set $\bar{X}$ under three demand multipliers. Hereon, for the ease of representation, we use binary form to indicate the direction of subloops from right to left. For instance, “10” denotes the scenario that only the direction of the right outer subloop is reversed.

When *DM* = 1, the company is able to transport all the containers in four scenarios and only the inventory cost exists. Scenario “10” apparently outperforms scenario “01” with around 54% reduction in the inventory cost. When *DM* = 1.5, transshipment cost appears in scenario “01”. It is because ships in some legs begin to be fully loaded. The company prefers to conduct transship operations in an effort to transport more containers, rather than buying slots from other companies. When *DM* reaches 2, scenarios “10” and “11” are still capable to accommodate the double quantity of containers, whereas the company has to employ slot-purchasing in scenarios “00” and “01”. It means that in this case the right outer subloop is the key subloop whose direction has a pronounced effect on the total cost.




OD table (b)

| O ↓ | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| A | | 300 | 200 | 300 | | |
| B | 400 | | | | 100 | |
| C | 200 | 300 | | | | |
| D | | | 200 | | | |
| E | 200 | | | | | 300 |
| F | 300 | | 200 | 300 | | |

OD table (c)

| O ↓ | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| A | | 100 | 300 | | 300 | 400 | | 500 |
| B | 200 | | | | 400 | | 200 | |
| C | | 200 | | | | 100 | | 400 |
| D | 500 | 300 | 100 | | 100 | | | |
| E | 400 | 600 | | 300 | | 300 | | |
| F | 100 | | | | | | 400 | 300 |
| G | | 400 | | 500 | 200 | | | 200 |
| H | | 100 | | | 300 | 600 | | |

**Fig. 6.** Three small examples.

The third example in Figure 6c is a ship route consisting of three subloops and its associated capacity is taken as 5000 TEUs. The box plot exhibits the cost distribution of eight scenarios under five demand multipliers (see Fig. 7c). Results show that the direction of the sole inner subloop appreciably affects the total cost, and such influence becomes continuously larger as the demand multiplier increases. Here, we can divide eight scenarios into two groups according to whether the inner subloop is reversed. The best scenario is always in the group with the reversed direction of inner subloop while the worst scenario only arises in the other group. The main difference in the total cost of two groups is the number of purchasing slots. When the direction of the inner subloop is unchanged, the company needs to buy more slots from other companies, as shown in the table of Fig. 7c.

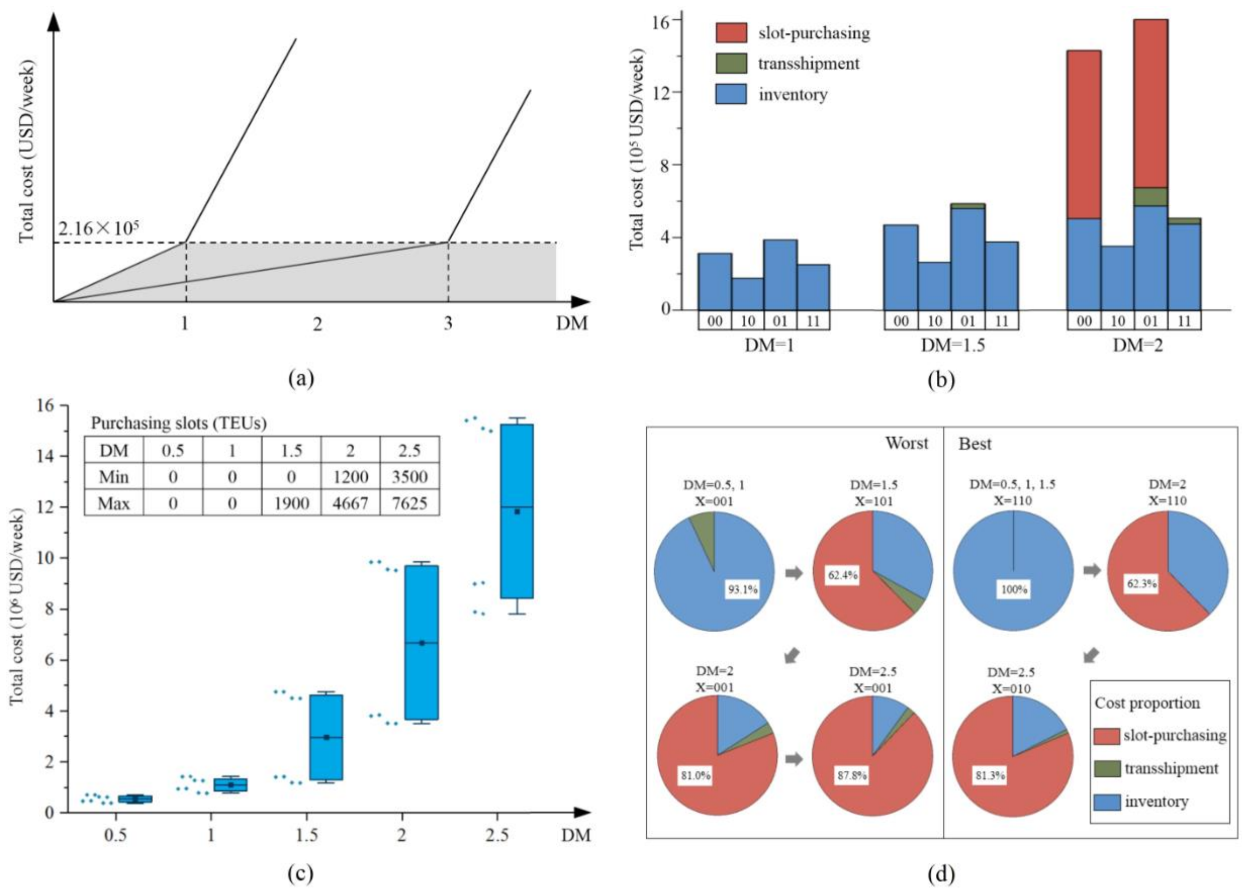



**Fig. 7.** Results of three small examples.

We further analyze the best and worst scenarios under various demand multipliers as depicted in Fig. 7d. First, we find that neither the best nor the worst scenario is fixed. For example, the best scenario changes from “110” to “010” when *DM* increases from 2 to 2.5. Furthermore, when *DM* is small (e.g. *DM* = 0.5), some transshipment containers occur in the worst scenario. It is because the round-trip time is relatively long which may lead to a high inventory cost of some O-D pairs if the setting of subloop directions is inappropriate for them. For these O-D demand, a trade-off arises between transshipment cost and inventory cost. As the demand multiplier increases, more and more legs become fully loaded due to the constraint of ship capacity. Once transshipment operations cannot contribute to using the remaining resources of ships, the company needs to buy more slots from other companies. When *DM* attains 1.5, the slot-purchasing cost finally occupies the majority of total cost (over 80% for both scenarios).

In addition, Proposition 3 links the relation between any two demand multipliers, and the associated proof is provided in Appendix A.4.

**Proposition 3.** Let $\mathrm{Opt}(DM)$ denote the optimal objective value of model SRPRD1 when the demand multiplier equals $DM$ ( $DM > 0$ ). Then, the following inequality always holds:

$$\mathrm{Opt}(DM_1) \geq \frac{DM_1}{DM_2}\mathrm{Opt}(DM_2), \quad DM_1 \geq DM_2 > 0 \ . \tag{68}$$

### 5.2 A case study of a liner shipping network

The proposed models and algorithms are applied to an Asia-Europe-Oceania shipping network of a global liner shipping company. This network has totally 46 ports, which are identical to the ones in Fig. 9 of Wang and Meng (2013). The company operates ten ship routes including 98 legs, as described in Table 2. Table 2 also presents the size of ships (TEUs) deployed on each ship route. There are a total of 652 O-D pairs with container shipment demand. These ship routes are further decomposed into 24 subloops and 29 sections.

**Table 2** The existing 10 routes and deployed ships

| ID | Ship type | Ports of call |
|---|---|---|
| 1 | 5000 | Fremantle → Sydney → Melbourne → Adelaide → Fremantle → Jakarta → Singapore → Port Klang |
| 2 | 5000 | Kaohsiung→ Ningbo → Shanghai → Yantian → Kaohsiung → Sydney → Melbourne → Brisbane |
| 3 | 5000 | Brisbane → Xiamen → Shanghai → Qingdao → Busan → Kobe → Yokohama → Brisbane → Sydney → Melbourne |
| 4 | 3000 | Kwangyang → Kobe → Nagoya → Tokyo → Kwangyang → Dalian → Xingang → Ningbo → Hong Kong → Manila → Laem Chabang → Ho Chi Minh → Manila |
| 5 | 10000 | Shanghai → Busan → Shanghai → Manila → Singapore → Port Klang → Colombo → Jakarta→ Singapore → Hong Kong → Chiwan → Xiamen |
| 6 | 3000 | Jakarta → Singapore → Laem Chabang → Ho Chi Minh → Jakarta → Colombo → Chennai → Chittagong |
| 7 | 1500 | Colombo → Cochin → Nhava Sheva → Karachi → Jebel Ali → Salalah |
| 8 | 5000 | Salalah → Sokhna → Aqabah → Jeddah → Salalah → Port Klang → Singapore |
| 9 | 5000 | Le Havre → Port Klang → Hong Kong → Le Havre → Thamesport → Rotterdam → Hamburg → Rotterdam → Antwerp → Zeebrugge |
| 10 | 10000 | Antwerp → Rotterdam → Bremerhaven → Hamburg → Antwerp → Southampton → Le Havre → Singapore → Hong Kong → Ningbo → Shanghai → Tokyo → Busan → Shanghai → Manila → Singapore |

The BD algorithm contains two stages. The first stage is the general BD algorithm incorporating Pareto-optimal cuts and updating big-M coefficients. The maximum number of iterations in the first stage is taken as 20, and the parameter $\omega$ used in updating big-M coefficients is taken as 0.05. Then, stage 2 generates CB cuts to continue accelerating the performance of the BD algorithm. In this study, the second stage terminates either when the objective value of CB-MP equals 0 or when the maximum number of CB cuts attains 200.

Fig. 8 depicts the iterative process of the BD algorithm with three acceleration strategies when $DM$ = 1. Stage 1 illustrates how the upper bound (UB) and lower bound (LB) of model SRPRD2 are updated across 20 iterations. It is worth noticing that $\sum_{(o,d)\in W}(g^{od}+\alpha T^{od})q^{od}$ is a constant value which is dropped from the objective function, and hence the values of UB and LB are negative. In stage 2, the convergence trend of LB and CB-MP is shown in right side of Fig. 8. Finally, after generating 60 CB cuts, the BD algorithm terminates since the objective value of CB-MP in Eq. (63) becomes 0.

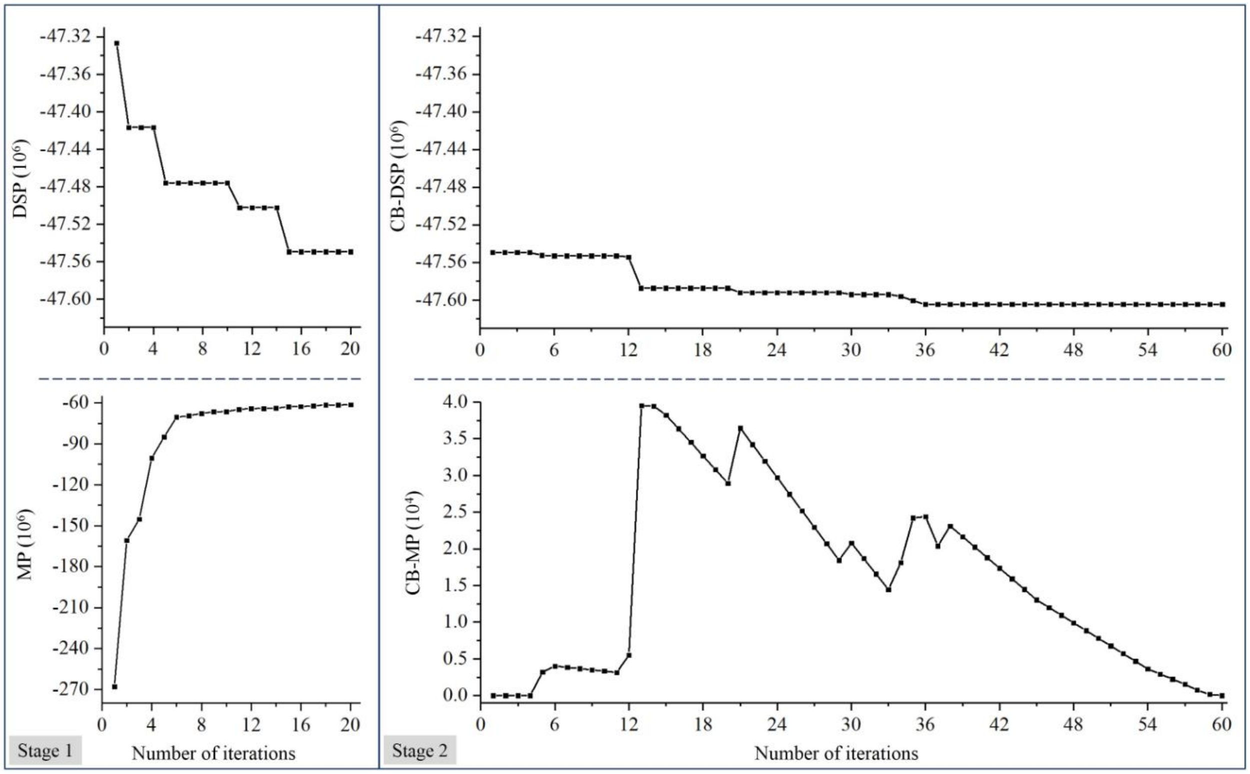



**Fig. 8.** Convergence process of the BD algorithm.

To evaluate the effectiveness of the proposed BD algorithm, we compare it with CPLEX and MILS algorithm under three demand multipliers. Here, we use the default settings of CPLEX to directly solve model SRPRD2. In MILS, the parameters $iter_{MULTI}$ and $iter_{ILS}$ are taken as 1 and 30, respectively. For each demand multiplier, we randomly generate 3 solutions of set $X$ . These random solutions and the original solution $X = \{x_s = 0 \mid s \in S_r, r \in R\}$ are seen as the initial inputs of both BD and MILS.

The computational results of three algorithms are shown in Table 3. For all cases, the BD algorithm can obtain the optimal solution. The average number of CB cuts is 80.5 and the longest computational time is not more than 15 min. The MILS metaheuristic method obtains a local optimum in one twelfth of the cases, and the count of iterations varies from 9 to 16. The computational time of CPLEX is 49.7 times of the computational time of BD and 22.8 times of the computational time of MILS on average. Results reveal the soundness of the proposed BD algorithm for solving the problem of SRPRD.

**Table 3** Computational results of three algorithms

| DM | CPLEX | | Case | BD | | | MILS | | |
|---|---|---|---|---|---|---|---|---|---|
| | Obj ($10^6$) | Time (h:m:s) | | Obj ($10^6$) | Count of CB cuts | Time (h:m:s) | Obj ($10^6$) | Number of iterations | Time (h:m:s) |
| 0.8 | -38.2471 | 8:40:15 | 1 | -38.2471 | 80 | 0:10:10 | -38.2471 | 13 | 0:25:58 |
| | | | 2 | -38.2471 | 93 | 0:11:15 | -38.2471 | 11 | 0:20:56 |
| | | | 3 | -38.2471 | 51 | 0:7:49 | -38.2471 | 12 | 0:23:20 |
| | | | 4* | -38.2471 | 89 | 0:10:55 | -38.2471 | 10 | 0:18:44 |
| 1 | -47.6061 | 10:25:34 | 1 | -47.6061 | 60 | 0:8:31 | -47.6061 | 13 | 0:25:49 |
| | | | 2 | -47.6061 | 97 | 0:11:36 | -47.6061 | 11 | 0:20:59 |
| | | | 3 | -47.6061 | 73 | 0:9:35 | -47.5977 | 9 | 0:16:25 |
| | | | 4 | -47.6061 | 80 | 0:10:12 | -47.6061 | 10 | 0:18:47 |
| 1.2 | -55.6257 | 6:17:23 | 1 | -55.6257 | 108 | 0:12:35 | -55.6257 | 12 | 0:23:21 |
| | | | 2 | -55.6257 | 76 | 0:9:49 | -55.6257 | 16 | 0:33:40 |
| | | | 3 | -55.6257 | 68 | 0:9:10 | -55.6257 | 11 | 0:21:10 |
| | | | 4 | -55.6257 | 91 | 0:11:5 | -55.6257 | 9 | 0:16:28 |

* Case 4 denotes the original solution, i.e. $X = \{x_s = 0 \mid s \in S_r, r \in R\}$

We add the constant term $\sum_{(o,d)\in W}(g^{od} + \alpha T^{od})q^{od}$ to the objective value again and compare the results of the original network and the modified network with the optimal setting of subloop directions. Table 4 describes the change of cost components before and after alteration. In the last column "reversed subloops", "5(2)" denotes that the direction of the second subloop in route 5 is reversed. For the optimized network, the directions of nine subloops are reversed when *DM* equals 0.8 and 1, while there are eight subloops whose directions are reversed when *DM* becomes 1.2.

Besides, results show that the subloop-based reversal of port rotation directions contributes to the reduction of the total network-wide cost 5.18% on average among three demand multipliers. The most influenced cost component is inventory cost which descends from 11.79% to 12.03% when *DM* increases from 0.8 to 1.2. It is observed that, the reduction of total cost continuously increases with a growth of the demand multiplier, but with a diminishing marginal return (the decline percentage drops from 5.61% to 4.47%). The reason is that as the demand multiplier increases, more legs are fully loaded and the company tends to purchase more slots whose cost is assumed to be notably larger than the cost by its own ships. High slot-purchasing cost undermines the benefit of the reversal of subloop directions.

**Table 4** Comparisons of the original network and the optimized network

| DM | Total demand | Solution | Total cost | Transshipment cost | Inventory cost | Slot cost | Reversed subloops |
|---|---|---|---|---|---|---|---|
| 0.8 | 17,653 | a* | 4,251 | 2,360 | 1,891 | 0 | **9**: 1(1), 2(1), 3(1), 5(2) |
| | | b | 4,503 | 2,360 | 2,144 | 0 | 6(1), 8(1), 8(2), 10(2) |
| | | | 5.61% | 0.00% | 11.79% | - | 10(3) |
| 1 | 22,054 | a | 5,516 | 2,923 | 2,355 | 238 | **9**: 1(1), 2(1), 3(1), 5(2) |
| | | b | 5,836 | 2,921 | 2,675 | 239 | 6(1), 8(1), 8(2), 10(2) |
| | | | 5.47% | -0.04% | 11.94% | 0.51% | 10(3) |
| 1.2 | 26,465 | a | 8,121 | 3,324 | 2,693 | 2,103 | **8**: 1(1), 2(1), 3(1), 5(2) |
| | | b | 8,501 | 3,344 | 3,062 | 2,095 | 6(1), 8(1), 10(2), 10(3) |
| | | | 4.47% | 0.59% | 12.03% | -0.41% | |

* a denotes the optimal solution; b denotes the original solution

The case of $DM$ = 1 is worth further discussion. Table 5 shows the optimized network and the underlined ports denote that the directions of their corresponding subloops are reversed. Results reveal that (i) all the subloops of some routes whose directions are completely reversed, such as route 8; (ii) only a part of subloops of some routes are selected to alter their directions (e.g. route 1 and 2); and (iii) some routes are unchanged, such as route 4 and 7. All the segments in the network are intact and meantime the number of ships in each route does not change after alteration.

**Table 5** The optimized network in the case of $DM$ = 1

| Route ID | Ports of call |
|---|---|
| 1 | Fremantle → Adelaide → Melbourne → Sydney → Fremantle → Jakarta → Singapore → Port Klang |
| 2 | Kaohsiung → Yantian → Shanghai → Ningbo → Kaohsiung → Sydney → Melbourne → Brisbane |
| 3 | Brisbane → Yokohama → Kobe → Busan → Qingdao → Shanghai → Xiamen → Brisbane → Sydney → Melbourne |
| 4 | Kwangyang → Kobe → Nagoya → Tokyo → Kwangyang → Dalian → Xingang → Ningbo → Hong Kong → Manila → Laem Chabang → Ho Chi Minh → Manila |
| 5 | Shanghai → Busan → Shanghai → Xiamen → Chiwan → Hong Kong → Singapore → Port Klang → Colombo → Jakarta → Singapore → Manila |
| 6 | Jakarta → Ho Chi Minh → Laem Chabang → Singapore → Jakarta → Colombo → Chennai → Chittagong |
| 7 | Colombo → Cochin → Nhava Sheva → Karachi → Jebel Ali → Salalah |
| 8 | Salalah → Jeddah → Aqabah → Sokhna → Salalah → Singapore → Port Klang |
| 9 | Le Havre → Port Klang → Hong Kong → Le Havre → Thamesport → Rotterdam → Hamburg → Rotterdam → Antwerp → Zeebrugge |
| 10 | Antwerp → Rotterdam → Bremerhaven → Hamburg → Antwerp → Singapore → Manila → Shanghai → Tokyo → Busan → Shanghai → Ningbo → Hong Kong → Singapore → Le Havre → Southampton |

Furthermore, we analyze the variation of ship occupancy which is termed as the difference of the container flow on each leg before and after alteration divided by the associated route capacity. Fig. 9 plots the variations of ship occupancy on 98 legs. It indicates that the container flow patterns before and after alteration are apparently distinct. At the same time, the ship occupancy on these legs before alteration is larger since there are more points above the horizontal axis. It is because the optimal setting of subloop directions is beneficial to decrease the sailing time of many O-D pairs and the overall inventory cost is decreased by 11.94% when *DM* equals 1.

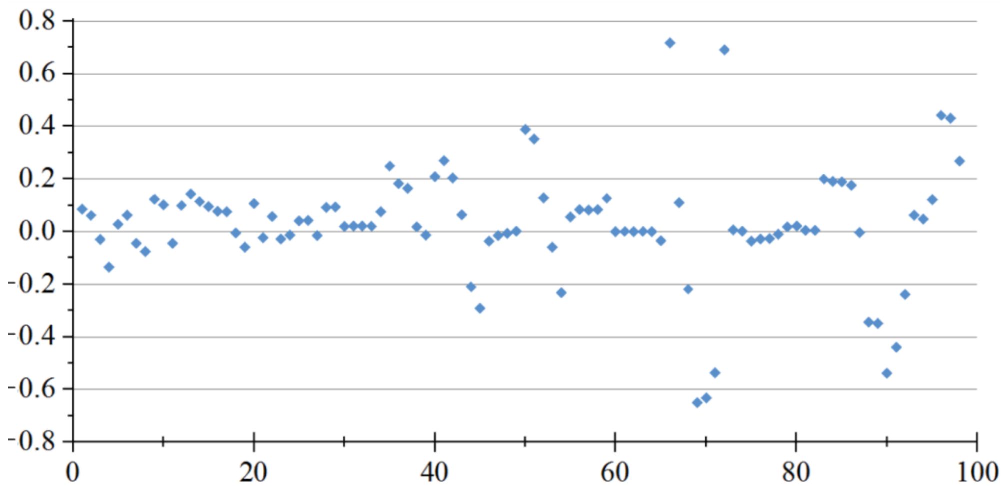



**Fig. 9.** Variation of ship occupancy on legs before and after alteration: (before – after)/capacity.

## 6 Conclusions

This paper focused on the optimization of port rotation directions in a generic liner shipping network with butterfly ports. Each ship route is separated into a set of subloops on the basis of its structure and internal butterfly ports. Through optimizing the subloop directions, the modified subloops constitute a new network which is coherent with the original network as all the subloops and associated sections are intact. In particular, each port that was visited before is still visited now, and all the ships that served the ship route before still serve the same route. Nevertheless, the subloop-based reversal of port rotation directions (SRPRD) not only influences the level of service, but also the transshipment operations and shipping capacity. We proposed a new destination-based nonlinear programming formulation for SRPRD with the objective of minimizing the generalized network-wide cost including inventory cost, transshipment cost, and slot-purchasing cost.

The problem of SRPRD was proved to be NP-hard, and this complexity proof also ensures the NP-hardness of problem presented in Wang and Meng (2013). Later, we transformed the proposed model to an equivalent mixed-integer linear programming model. The structure of the reformulated model makes it well suited for decomposition. We developed a Benders decomposition (BD) algorithm incorporating three acceleration strategies which subsume adding Pareto-optimal cuts, dynamically updating big-M coefficients and generating combinatorial Benders cuts respectively. In numerical instances, three small examples were utilized to analyze the potential benefits of subloop-based reversal of port rotation directions. Then, a case study based on an Asia-Europe-Oceania liner shipping network with a total of 46 ports was conducted. Computational results demonstrate the efficiency of the BD algorithm as it is comparable to a metaheuristic method and meanwhile considerably faster than solving the reformulated model with an off-the-shelf solver. Results show that the optimization of subloop directions contributes to decreasing the total cost of the original network by 5.18% on average, and the most influenced cost component is inventory cost with over 11.5% reduction.

Future research directions are as follows: First, in this study the demand between each OD pair is assumed fixed. In future, it is worthwhile to investigate container liner shipping network alteration with stochastic demand. Second, some practical circumstances of the liner shipping business could be considered. For instance, container terminal operations such as berth time windows, which affect the arrival and departure time of ships at each port of call may be incorporated in modeling. We recommend that future studies could focus on these issues.

## Acknowledgements

The authors would like to thank the Editor-in-Chief and the anonymous reviewers for their valuable comments and suggestions. This study is supported by the National Natural Science Foundation of China (No. 71771050, No. 71701178, No. 71831008), the Natural Science Foundation of Jiangsu Province in China (BK20180402), and the China Postdoctoral Science Foundation (2017M621596).

## Appendix - Proofs

This supplement contains all the proofs of results contained in the main paper.

**A.1.** Computational complexity proof.

**Proof.** We propose a polynomial reduction from Partition Problem (PP), whose NP-completeness is assured by Chopra and Rao (1993), to the decision version of SRPRD. First, we show the decision versions of both problems.

---

*Partition Problem* (*PP*):
INPUT: A multiset $A$ of $N$ positive integers $\{a_1, a_2, ..., a_N\}$.
QUESTION: Is there a partition of set $E$ into two subsets $A_1$ and $A_2$ such that the sum of numbers in set $A_1$ equals half of the sum of numbers in set $A$ (that is, the sum of numbers in set $A_2$ also equals half of the sum of numbers in set $A$)?

---

A simple example of this problem is that given $A = \{1,1,2,3,5\}$, the valid solution is the two sets $A_1 = \{1,2,3\}$ and $A_2 = \{1,5\}$.

---

*Subloop-based reversal of port rotation directions* (*decision version called dSRPRD*):
INPUT: Parameters $\alpha$, $t_{ci}$, $E_r$, $t_p$, $\bar{c}_p$, $g^{od}$, $T^{od}$, $q^{od}$ $\forall i \in I_{rc}, c \in C_s, s \in S_r, r \in R, p \in P,$ $(o,d) \in W$, and a scalar $K$.
QUESTION: Are there any values for decision variable set $\{x_s, \bar{z}_p, f_{ci}^d, \hat{z}_{ci}^d, \tilde{z}_{ci}^d, y^{od}\}$ $\forall i \in I_{rc},$ $c \in C_s, s \in S_r, r \in R, p \in P, (o,d) \in W$ that satisfy constraints (2)-(10) and (12)-(23) and such that

$$\sum_{p \in P} (\bar{c}_p + \alpha t_p) \bar{z}_p + \alpha \sum_{r \in R} \sum_{s \in S_r} \left[ (1 - x_s) \sum_{c \in C_s} \sum_{i=t(c)}^{h(c)-1} t_{ci} \sum_{d \in P} f_{ci}^d \right]$$
$$+ \alpha \sum_{r \in R} \sum_{s \in S_r} \left[ x_s \sum_{c \in C_s} \sum_{i=t(c)+1}^{h(c)} t_{c,i-1} \sum_{d \in P} f_{ci}^d \right] + \sum_{(o,d) \in W} (g^{od} + \alpha T^{od})(q^{od} - y^{od}) \le K ?$$

---

Next, we discuss the hardness of the problem for dSRPRD. The description of two steps for the complexity proof is as follows:

1. dSRPRD $\in$ NP

We assume that there exists an algorithm which generates a solution for dSRPRD. Determining if the solution is feasible for dSRPRD needs several steps. First, examine whether constraints(2)-(10) and (12)-(23) are satisfied for each decision variable of the solution ( $x_s, \bar{z}_p, f_{ci}^d, \hat{z}_{ci}^d, \tilde{z}_{ci}^d, y^{od}$ ).

Then, verify if the sum of four cost terms in the objective function (11) based on the obtained solution is not more than $K$ (one constraint). Considering all the constraints, it takes a polynomial number of steps to verify the feasibility of a dSRPRD solution, i.e. $\text{dSRPRD} \in \text{NP}$.

2. $\text{PP} \prec_p \text{dSRPRD}$

Consider an arbitrary instance of PP (a multiset *A* of *N* positive integers $\{a_1, a_2, ..., a_N\}$). We build a particular instance of dSRPRD, called dSRPRD*, as below. Set the number of ship routes to be $|R| = N$ (ship capacity of route *r* corresponds to the integer $a_r$ in set *A*). Assume that the shipping network only contains three physical ports (port 1, 2 and 3). There are totally 6 O-D pairs, and demand of each pair is identical which is set to equal $0.5\sum_{r\in R} a_r$. The inventory cost parameter $\alpha$ is set to be 0. The transshipment cost $\bar{c}_p + \alpha t_p$ is set to be $+\infty$ which means that no transship operations are allowed. The slot-purchasing cost $g^{od} + \alpha T^{od}$ is set to be 1. At the same time, we let the scalar $K$ be equal to zero.

By dSRPRD* definition, only the fourth cost term of the objective function (11) remains because the other three terms equal 0. In this case, the task of dSRPRD* is to check whether there exists a valid solution that meet constraints (2)-(10) and (12)-(23) and the objective value equals zero.

In practice, instance dSRPRD* has the following properties which are beneficial to prove the solution equivalence.

**Property 1**. The maximum number of containers transported by ship route *r* is $3a_r$.

**Property 2**. The feasible solution of dSRPRD* exists only if the number of containers transported by route *r* with clockwise direction is $3a_r$ and the number of containers transported by route $r'$ with counter clockwise direction is $3a_{r'}$.

Fig. 10 depicts the influence of port rotation direction on shipping capacity. Given three O-D pairs (pair 1-3, 2-1, 3-2), if ship route *r* serves three physical ports and port rotation direction is clockwise as shown in Fig. 10a, then the maximum number of containers served is $1.5a_r$. By contrast, if ship route *r* serves three physical ports and port rotation direction is counter-clockwise (see Fig. 10b), the optimal choice to ship as many containers as possible is to ship $3a_r$ containers. Therefore, Property 1 holds.

The total demand of 6 O-D pairs is $6\times 0.5\sum_{r\in R} a_r = \sum_{r\in R} 3a_r$. According to Property 1, we naturally obtain Property 2 (otherwise the company cannot transport all the containers by its own ships; it may purchase ship slots and the objective value exceeds zero). Note that Property 2 is the necessary condition of instance dSRPRD*.

Next, we prove the solution equivalence, i.e., the answer of an instance of problem PP is "YES" if and only if, the answer of the related instance of problem dSRPRD* is "YES".

($\Rightarrow$) Let $A_1 = \{a_1,...,a_{N_1}\}$ be a solution of problem PP with a "Yes" answer. Then, we procure that $\sum_{A_1} a_r = 0.5\sum_{A} a_r$. Since ship capacity of each route $r$ corresponds to a particular integer $a_r$ in set $A$, we assign all the ship routes whose corresponding integer $a_r$ is in set $A_1$ to the first group, and assign the remaining routes to the second group.

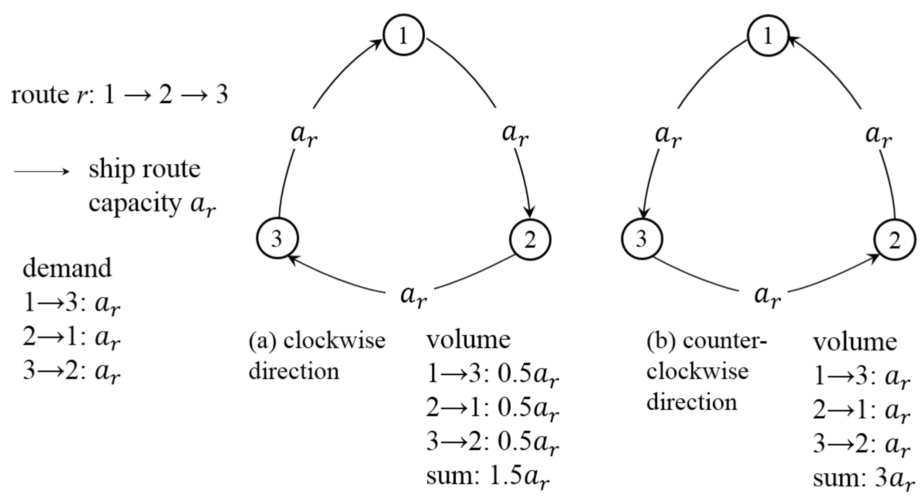



**Fig. 10.** Influence of port rotation direction on shipping capacity.

The itinerary of ship route $r$ in the first group is set to be $p_{r3} \to p_{r2} \to p_{r1}$ which exclusively serves three O-D pairs (pair 1-3, 2-1, 3-2). Then, the maximum number of containers transported between port 1 and 3 is $\sum_{A_1} a_r = 0.5\sum_{A} a_r$ (see Fig. 10b). It means that all the containers of pair 1-3 can be transported by ship routes in the first group. Similarly, we can obtain the same conclusions with respect to pair 2-1, 3-2, and the other three O-D pairs in the second group.

Therefore, there is no need to purchase ship slots from other companies, and the objective value of Eq. (11) equals $K$ ($K=0$).

($\Leftarrow$) Let $\{x_s, \bar{z}_p, f_{ci}^d, \hat{z}_{ci}^d, \tilde{z}_{ci}^d, y^{od}\}$ be a solution of instance dSRPRD* with a "Yes" answer. As the network has no butterfly port ($\hat{R}=\varnothing$), each ship route is treated as one outer subloop $s$ with three ports of call. Ship routes can be divided into two groups depending on the port rotation direction. We let $x_s=0$ denote the first group $R_1$ of routes with clockwise direction and $x_s=1$ denote the second group $R \setminus R_1$ of routes with counter-clockwise direction (see Fig. 10). By Property 3, the number of containers transported in the first group (serving pair 1-2, 2-3, 3-1) equals $\sum_{r\in R_1} 3a_r$. Meantime, the objective value equals zero means that demand of three O-D pairs (1-2, 2-3, 3-1) is satisfied by ships routes in the first group, namely $\sum_{r\in R_1} 3a_r = 3\times 0.5\sum_{r\in R} a_r$. When all the integer values $a_r$ corresponding to ship capacity of routes in the first group constitute a subset $A_1$, we obtain that $A_1$ is a valid solution of problem PP.

As per the above solution equivalence, considering an optimization problem, where its decision version (dSRPRD) is NP-complete, then, the optimization problem (SRPRD) belongs to the NP-hard class. □

**A.2.** When $M_{1,s}$ ($s\in S_r, r\in R$) is set to equal $\sum_{c\in C_s}\left(\sum_{i=t(c)+1}^{h(c)-1}\left|t_{c,i-1}-t_{ci}\right| + \max(t_{c,h(c)-1}, t_{c,t(c)})\right)E_r$, the objective function (24) is equivalent to Eqs. (25)-(27).

**Proof.** The difference between Eq. (24) and Eq. (25) is the fourth term. For the clarity of representation, we let $\kappa_s$ be equal to:

$$\kappa_s = \sum_{c\in C_s}\left[\sum_{i=t(c)+1}^{h(c)-1}(t_{c,i-1}-t_{ci})\sum_{d\in P} f_{ci}^d + t_{c,h(c)-1}\sum_{d\in P} f_{c,h(c)}^d - t_{c,t(c)}\sum_{d\in P} f_{c,t(c)}^d\right]. \tag{69}$$

Then, the fourth term of Eq. (24) is the sum of $x_s\kappa_s$ for all the subloops in the network.

As Eq. (25) is a minimization objective function, it is easy to verify that either of constraint (26) and (27) is binding. Therefore, depending on whether subloop $s$ ($s\in S_r, r\in R$) is reversed, auxiliary variable $\tau_s$ can be expressed as:

$$\tau_s = \begin{cases} \max\left(\kappa_s, -M_{1,s}\right), & x_s = 1 \\ \max\left(\kappa_s - M_{1,s}, 0\right), & x_s = 0. \end{cases} \tag{70}$$

When $M_{1,s}$ is not less than $|\kappa_s|$, $\tau_s = x_s\kappa_s$ always holds in Eq. (70). In this case, the objective function (24) is equivalent to Eqs. (25)-(27). Next, we explore the relationship between $|\kappa_s|$ and $\sum_{c\in C_s}\left(\sum_{i=t(c)+1}^{h(c)-1}\left|t_{c,i-1}-t_{ci}\right|+t_{c,h(c)-1}+t_{c,t(c)}\right)E_r$, as shown below:

$$\begin{aligned}
|\kappa_s| &= \left|\sum_{c\in C_s}\left[\sum_{i=t(c)+1}^{h(c)-1}(t_{c,i-1}-t_{ci})\sum_{d\in P} f_{ci}^{d} + t_{c,h(c)-1}\sum_{d\in P} f_{c,h(c)}^{d} - t_{c,t(c)}\sum_{d\in P} f_{c,t(c)}^{d}\right]\right| \\
&\le \sum_{c\in C_s}\left|\sum_{i=t(c)+1}^{h(c)-1}(t_{c,i-1}-t_{ci})\sum_{d\in P} f_{ci}^{d} + t_{c,h(c)-1}\sum_{d\in P} f_{c,h(c)}^{d} - t_{c,t(c)}\sum_{d\in P} f_{c,t(c)}^{d}\right| \\
&\le \sum_{c\in C_s}\left(\left|\sum_{i=t(c)+1}^{h(c)-1}(t_{c,i-1}-t_{ci})\sum_{d\in P} f_{ci}^{d}\right| + \left|t_{c,h(c)-1}\sum_{d\in P} f_{c,h(c)}^{d} - t_{c,t(c)}\sum_{d\in P} f_{c,t(c)}^{d}\right|\right) \\
&\le \sum_{c\in C_s}\left(\sum_{i=t(c)+1}^{h(c)-1}\left|t_{c,i-1}-t_{ci}\right|\sum_{d\in P} f_{ci}^{d} + \max(t_{c,h(c)-1}\sum_{d\in P} f_{c,h(c)}^{d}, t_{c,t(c)}\sum_{d\in P} f_{c,t(c)}^{d})\right) \quad \text{(due to Eq. 22)} \\
&\le \sum_{c\in C_s}\left(\sum_{i=t(c)+1}^{h(c)-1}\left|t_{c,i-1}-t_{ci}\right| + \max(t_{c,h(c)-1}, t_{c,t(c)})\right)E_r \quad \text{(due to Eq. 18)} \\
&= M_{1,s}.
\end{aligned} \tag{71}$$

When $M_{1,s}$ ($s\in S_r, r\in R$) is set to be equal to $\sum_{c\in C_s}\left(\sum_{i=t(c)+1}^{h(c)-1}\left|t_{c,i-1}-t_{ci}\right| + \max(t_{c,h(c)-1}, t_{c,t(c)})\right)E_r$, $M_{1,s} \ge |\kappa_s|$ is satisfied. This completes the proof. □

**A.3**. When $M_{2,rd}$ ($r\in R, d\in P$) is set as $E_r + \sum_{o\in P} q^{od}$, the two nonlinear constraints (12) and (13) are equivalent to Eqs. (28) and (29).

**Proof.** Based upon whether subloop $s$ ($s\in S_r, r\in R$) is reversed, subloops in the network are separated into two groups. The first group is composed of subloops whose direction is unchanged (i.e. $x_s = 0$). In this group, Eq. (12) is identical to Eq. (28) while $M_{2,rd}$ should be not less than $\left|f_{c,i}^{d} + \hat{z}_{c,i-1}^{d} - f_{c,i-1}^{d} - \tilde{z}_{c,i-1}^{d}\right|$ in order that Eq. (29) is always satisfied. The second group contains all the subloops whose direction is reversed (i.e. $x_s = 1$), in which Eq. (13) is the same as Eq. (29). Meanwhile, $M_{2,rd}$ needs to be not less than $\left| f_{c,i}^d + \hat{z}_{c,i+1}^d - f_{c,i+1}^d - \tilde{z}_{c,i+1}^d \right|$ so that Eq. (28) always holds. Now, we proceed to discuss the relation between $E_r + \sum_{o \in P} q^{od}$ and the maximum of two absolute values:

$$\begin{aligned}
&\max\left(\left| f_{c,i}^d + \hat{z}_{c,i-1}^d - f_{c,i-1}^d - \tilde{z}_{c,i-1}^d \right|, \left| f_{c,i}^d + \hat{z}_{c,i+1}^d - f_{c,i+1}^d - \tilde{z}_{c,i+1}^d \right|\right) \\
&= \max\left(\left|( f_{c,i}^d - f_{c,i-1}^d) + (\hat{z}_{c,i-1}^d - \tilde{z}_{c,i-1}^d)\right|, \left|( f_{c,i}^d - f_{c,i+1}^d) + (\hat{z}_{c,i+1}^d - \tilde{z}_{c,i+1}^d)\right|\right) \\
&\le \max\left(\left| f_{c,i}^d - f_{c,i-1}^d \right| + \left|\hat{z}_{c,i-1}^d - \tilde{z}_{c,i-1}^d\right|, \left| f_{c,i}^d - f_{c,i+1}^d \right| + \left|\hat{z}_{c,i+1}^d - \tilde{z}_{c,i+1}^d\right|\right) \\
&\le \max\left(\max( f_{c,i}^d, f_{c,i-1}^d) + \max(\hat{z}_{c,i-1}^d, \tilde{z}_{c,i-1}^d), \max( f_{c,i}^d, f_{c,i+1}^d) + \max(\hat{z}_{c,i+1}^d, \tilde{z}_{c,i+1}^d)\right) \\
&\le \max( f_{c,i-1}^d, f_{c,i}^d, f_{c,i+1}^d) + \max(\hat{z}_{c,i-1}^d, \tilde{z}_{c,i-1}^d, \hat{z}_{c,i+1}^d, \tilde{z}_{c,i+1}^d) \\
&\le E_r + \sum_{o \in P} q^{od} \quad \text{(due to Eq. 18)} \\
&= M_{2,rd}.
\end{aligned} \tag{72}$$

Note that in some cases, the two absolute values in Eq. (72) may not simultaneously exist. For instance, at the tail port of call $t(c)$, there is no $(i-1)$th port of call. Nevertheless, $E_r + \sum_{o \in P} q^{od}$ is also not less than the single absolute value $\left| f_{c,i}^d + \hat{z}_{c,i+1}^d - f_{c,i+1}^d - \tilde{z}_{c,i+1}^d \right|$. Hence, when $M_{2,rd}$ ($r \in R, d \in P$) is set as $E_r + \sum_{o \in P} q^{od}$, constraints (12) and (13) are equivalent to Eqs. (28) and (29). □

**A.4.** Let $\text{Opt}(DM)$ denote the optimal objective value of model SRPRD1 when the demand multiplier equals $DM$ ($DM > 0$). Then, inequality (68) always holds.

**Proof.** First, we let $X_{DM_1}$ and $\Omega_{DM_1}$ denote the resulting set of subloop reversal variables $\{x_s \mid DM_1\}$ and the set of container flow variables $\{y^{od}, \hat{z}_{c,i}^d, \tilde{z}_{c,i}^d, f_{c,i}^d, \bar{z}_p \mid DM_1\}$ respectively, which correspond to $\text{Opt}(DM_1)$. Analogously, the sets $X_{DM_2}$ and $\Omega_{DM_2}$ correspond to $\text{Opt}(DM_2)$.

Given $DM_1 \ge DM_2 > 0$, we multiply all the continuous variables in set $\Omega_{DM_1}$ by $\dfrac{DM_2}{DM_1}$ and obtain a new set, denoted by $\tilde{\Omega}_{DM_1}$. It is easy to verify that $\tilde{\Omega}_{DM_1}$ is a feasible container flow set when the demand multiplier equals $DM_2$ and the set of subloop reversal variables is $\{x_s \mid DM_1\}$. Here, we use $\Psi$ to denote the objective value of model SRPRD1 when the two associated sets are given. Then, we can get the following relationship:

$$\begin{aligned}
\mathrm{Opt}(DM_1) &= \Psi(X_{DM_1}, \Omega_{DM_1}) \\
&= \frac{DM_1}{DM_2}\Psi(X_{DM_1}, \tilde{\Omega}_{DM_1}) \quad \text{linear model} \\
&\geq \frac{DM_1}{DM_2}\Psi(X_{DM_2}, \Omega_{DM_2}) \quad \text{optimal soltuion} \\
&= \frac{DM_1}{DM_2}\mathrm{Opt}(DM_2)
\end{aligned} \tag{73}$$

This completes the proof. □

**REFERENCES**


Agarwal, R., Ergun, O., 2008. Ship scheduling and network design for cargo routing in liner shipping. Transportation Science, 42, 175-196.

Akyüz, M. H., Lee, C. Y., 2016. Service type assignment and container routing with transit time constraints and empty container repositioning for liner shipping service networks. Transportation Research Part B: Methodological, 88, 46-71.

Alvarez, J. F., 2009. Joint routing and deployment of a fleet of container vessels. Maritime Economics & Logistics, 11(2), 186-208.

Angeloudis, P., Greco, L., Bell, M. G. H., 2016. Strategic maritime container service design in oligopolistic markets. Transportation Research Part B: Methodological, 90, 22-37.

Arslan, O., Karaşan, O. E., 2016. A Benders decomposition approach for the charging station location problem with plug-in hybrid electric vehicles. Transportation Research Part B: Methodological, 93, 670-695.

Bayram, V., Yaman, H., 2017. Shelter location and evacuation route assignment under uncertainty: A Benders decomposition approach. Transportation Science, 52(2), 416-436.

Benders, J., 1962. Partitioning procedures for solving mixed-variables programming problems. Numerische Mathematik, 4(1), 238-252.

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

Brouer, B. D., 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.

Cariou, P., Cheaitou, A., Larbi, R., Hamdan, S., 2018. Liner shipping network design with emission control areas: A genetic algorithm-based approach. Transportation Research Part D: Transport and Environment, 63, 604-621.

Chen, J., Wang, S., Liu, Z., Wang, W., 2017. Design of suburban bus route for airport access. Transportmetrica A: Transport Science, 13(6), 568-589.

Chen, J., Wang, S., Liu, Z., Guo, Y., 2018. Network-based optimization modeling of manhole setting for pipeline transportation. Transportation Research Part E: Logistics and Transportation Review, 113, 38-55.

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

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.

Chopra, S., Rao, M. R., 1993. The partition problem. Mathematical Programming, 59, 87-115.

CI. 2009. Containerisation International Yearbook 2009, Lloyd’s MIU, London.

Codato, G., Fischetti, M., 2006. Combinatorial Benders’ cuts for mixed-integer linear programming. Operations Research, 54(4), 756-766.

Côté, J. F., Dell'Amico, M., Iori, M., 2014. Combinatorial Benders' cuts for the strip packing problem. Operations Research, 62(3), 643-661.

Demir, E., Burgholzer, W., Hrušovský, M., Arıkan, E., Jammernegg, W., Van Woensel, T., 2016. A green intermodal service network design problem with travel time uncertainty. Transportation Research Part B: Methodological, 93, 789-807.

Fagerholt, K., 2004. Designing optimal routes in a liner shipping problem. Maritime Policy and Management, 31(4), 259-268.

Fontaine, P., Minner, S., 2014. Benders decomposition for discrete–continuous linear bilevel problems with application to traffic network design. Transportation Research Part B: Methodological, 70, 163-172.

Garey, M.R., Johnson, D.S., 1979. Computers and Intractability: A Guide to the Theory of NP-Completeness (first ed.), W.H. Freeman & Co. New York, NY.

Gelareh, S., Nickel, S., Pisinger, D., 2010. Liner shipping hub network design in a competitive environment. Transportation Research Part E: Logistics and Transportation Review, 46 (6), 991-1004.

Gelareh S, Pisinger D, 2011. Fleet deployment, network design and hub location of liner shipping companies. Transportation Research Part E: Logistics and Transportation Review, 47(6): 947-964.

Holm, M. B., Medbøen, C. A. B., Fagerholt, K., Schütz, P., 2018. Shortsea liner network design with transhipments at sea: a case study from Western Norway. Flexible Services and Manufacturing Journal, 1-22.

Ibarra-Rojas, O. J., Rios-Solis, Y. A., 2012. Synchronization of bus timetabling. Transportation Research Part B: Methodological, 46(5), 599-614.

Imai, A., Shintani, K., Papadimitriou, S., 2009. Multi-port vs. Hub-and-Spoke port calls by containerships. Transportation Research Part E: Logistics and Transportation Review, 45 (5), 740-757.

Jiang, J., Lee, L. H., Chew, E. P., Gan, C. C., 2015. Port connectivity study: An analysis framework from a global container liner shipping network perspective. Transportation Research Part E: Logistics and Transportation Review, 73, 47-64.

Karlaftis, M. G., Kepaptsoglou, K., Sambracos, E., 2009. Containership routing with time deadlines and simultaneous deliveries and pick-ups. Transportation Research Part E: Logistics and Transportation Review, 45 (1), 210-221.

Karsten, C. V., Brouer, B. D., Desaulniers, G., Pisinger, D., 2016. Time constrained liner shipping network design. Transportation Research Part E: Logistics and Transportation Review, 105, 152-162.

Karsten, C. V., Brouer, B. D., Pisinger, D., 2017. Competitive liner shipping network design. Computers & Operations Research, 87, 125-136.

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

Lee, C. Y., Song, D. P., 2017. Ocean container transport in global supply chains: Overview and research opportunities. Transportation Research Part B: Methodological, 95, 442-474.

Liu, Z., Meng, Q., Wang, S., Sun, Z., 2014. Global intermodal liner shipping network design. Transportation Research Part E: Logistics and Transportation Review, 61, 28-39.

Lourenço, H., Martin, O., Stützle, T., 2010. Iterated Local Search: Framework and Applications. Handbook of Metaheuristics (second ed.), Springer, Boston, MA.

Magnanti, T. L., Wong, R. T., 1981. Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. Operations Research, 29(3), 464-484.

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

Monemi, R. N., Gelareh, S., 2017. Network design, fleet deployment and empty repositioning in liner shipping. Transportation Research Part E: Logistics and Transportation Review, 108, 60-79.

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

Papadimitriou, C. H., 2003. Computational Complexity. John Wiley and Sons Ltd. Chichester, UK.

Plum, C. E., Pisinger, D., Sigurd, M. M., 2014a. A service flow model for the liner shipping network design problem. European Journal of Operational Research, 235(2), 378-386.

Plum, C. E., Pisinger, D., Salazar-González, J. J., Sigurd, M. M., 2014b. Single liner shipping service design. Computers & Operations Research, 45, 1-6.

Rana, K., Vickson, R.G., 1988. A model and solution algorithm for optimal routing of a time-chartered containership. Transportation Science, 22 (2), 83-95.

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

Sambracos, E., Paravantis, J. A., Tarantilis, C. D., Kiranoudis, C. T., 2004. Dispatching of small containers via coastal freight liners: the case of the Aegean Sea. European Journal of Operational Research, 152 (2), 365-381.

Shintani, K., Imai, A., Nishimura, E., Papadimitriou, S., 2007. The container shipping network design problem with empty container repositioning. Transportation Research Part E: Logistics and Transportation Review, 43 (1), 39-59.

Song, D. P., Dong, J. X., 2011. Effectiveness of an empty container repositioning policy with flexible destination ports. Transport Policy, 18 (1), 92-101.

Song, D. P., Dong, J. X., 2012. Cargo routing and empty container repositioning in multiple shipping service routes. Transportation Research Part B: Methodological, 46 (10), 1556-1575.

Song, D. P., Dong, J. X., 2013. Long-haul liner service route design with ship deployment and empty container repositioning. Transportation Research Part B: Methodological, 55, 188-211.

Sun, Z., Zheng, J., 2016. Finding potential hub locations for liner shipping. Transportation Research Part B: Methodological, 93, 750-761.

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.

Tran, N. K., Haasis, H. D., 2015. Literature survey of network optimization in container liner shipping. Flexible Services and Manufacturing Journal, 27(2-3), 139-179.

UNCTAD, 2016. Review of Maritime Transportation 2016. Paper presented at the United Nations Conference on Trade and Development. New York and Geneva. <http://unctad.org/en/PublicationsLibrary/rmt2016_en.pdf>. (accessed 10 August 2018).

Wang, S., Meng, Q., 2013. Reversing port rotation directions in a container liner shipping network. Transportation Research Part B: Methodological, 50, 61-73.

Wang, S., Liu, Z., Meng, Q., 2015. Segment-based alteration for container liner shipping network design. Transportation Research Part B: Methodological, 72, 128-145.

Wang, S. 2017. Formulating cargo inventory costs for liner shipping network design. Maritime Policy & Management, 44(1), 62-80.