Submitted to *Operations Research*

manuscript OPRE-2021-06-343

Vessel Service Planning in Seaports

Lingxiao Wu
HEC Montréal and GERAD, 3000 chemin de la Côte-Sainte-Catherine, Montréal, H3T 2A7, Canada, lingxiao.wu@hec.ca

Yossiri Adulyasak
HEC Montréal and GERAD, 3000 chemin de la Côte-Sainte-Catherine, Montréal, H3T 2A7, Canada, yossiri.adulyasak@hec.ca

Jean-François Cordeau
HEC Montréal and GERAD, 3000 chemin de la Côte-Sainte-Catherine, Montréal, H3T 2A7, Canada, jean-francois.cordeau@hec.ca

Shuaian Wang
Department of Logistics & Maritime Studies, The Hong Kong Polytechnic University, Kowloon, Hong Kong, wangshuaian@gmail.com

Berth allocation and pilotage planning are the two most important decisions made by a seaport for serving incoming vessels. Traditionally, the berth allocation problem and the pilotage planning problem are solved sequentially, leading to suboptimal or even infeasible solutions for vessel services. This paper investigates a vessel service planning problem (VSPP) in seaports that addresses berth allocation and pilotage planning in combination. We introduce a compact mixed-integer linear programming formulation for the problem, which can be solved by general-purpose solvers. To solve large-scale instances, we develop an exact solution approach that combines Benders decomposition and column generation within an efficient branch-and-bound framework. Unlike the traditional three-phase Benders decomposition and column generation method, which does not guarantee optimality, we propose a branching scheme that enables the approach to determine an optimal solution to the VSPP. The approach is enhanced through practical acceleration strategies. Extensive computational results using data instances from one of the world's largest seaports show that these acceleration strategies significantly improve the performance of our solution approach and that it can obtain optimal or near-optimal solutions for instances of realistic scale. We show that our solution approach outperforms the method commonly used for solving similar problems. We perform sensitivity tests to demonstrate the robustness of the approach against variations in problem settings. We also show the benefits brought by integrated optimization by comparing our solution approach with a method that handles berth allocation and pilotage planning sequentially.

Key words: seaport operations, vessel service planning, berth allocation, pilotage planning, Benders decomposition, column generation

1. Introduction

Seaports have long been the key nodes in the global supply chain. In 2018, more than 10 billion tons of cargo were handled by seaports around the world, and the volume is still growing (UNCTAD 2019). Due to the growth of seaborne trade, the number and size of vessels that need to be served in seaports have increased continuously. Seaports and vessels involve huge investments; to recoup these investments, port authorities and shipping companies aim to achieve very high utilization of their capacities, making it critical for vessel services to be efficient (Fransoo and Lee 2013, Roy et al. 2020). In addition, given that vessels are major sources of harmful air pollutants, rapid vessel turnaround in seaports helps to relieve the environmental burden on local communities (Du et al. 2015).

Figure 1 shows the layout of a typical seaport, which generally can be divided into three parts: the anchorage, the navigation channel, and the terminals. The quay of a terminal is divided into a set of berths. Vessels coming from the open sea first reach and wait in the anchorage before traveling through the channel and arriving at the designated berths to be handled. After being served at the berths, the vessels leave the berths, sail to the anchorage by passing through the navigation channel, and finally return to the open sea. In most seaports, pilotage is compulsory to enhance navigational safety. In particular, whenever a vessel is moving in port waters (including sailing in the channel and berthing into or unberthing from the berths), a pilot must be onboard to provide navigational guidance to the vessel master (Wu et al. 2020).

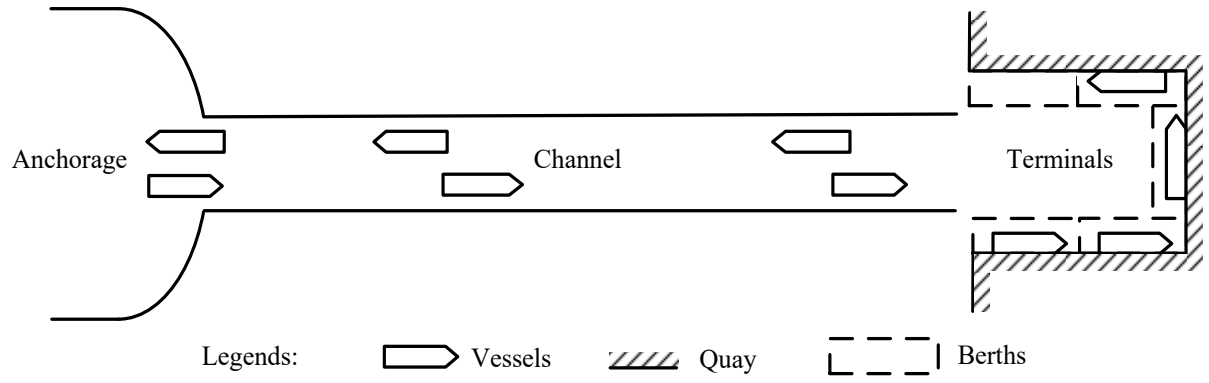


Figure 1 Layout of a Seaport.

Given a set of calling vessels, the service plan made by a seaport includes two main components: berth allocation and pilotage planning. Berth allocation determines the berthing position and time of each vessel, while pilotage planning involves vessel traffic management and pilot scheduling. Berths are the most important resource in a seaport. In busy seaports, vessels often have to wait for

berths to become available due to the limited number of berths compared with the number of calling vessels. Efficient berth allocation promotes efficient cargo dispatch and rapid vessel turnaround. Hence, berth allocation is at the core of seaport operations.

Pilotage planning, another important problem faced by seaport operators, involves the management of vessel traffic in port waters and the assignment of marine pilots for serving the vessels. Vessel movements in port waters (or other restricted waterways) must be carefully controlled to ensure smooth and collision-free traffic, as any collision or blockage in such waterways may lead to serious delays for vessels and incur a considerable loss (e.g., the 2021 Suez Canal obstruction (Russon 2021)). Pilot assistance is indispensable in vessel traffic management in seaports. Marine pilots are scarce talents in many countries. Before becoming a pilot, one should pass rigid entry requirements and go through long-term training periods (e.g., Hong Kong Maritime and Port Board 2018). Given their importance and scarcity, pilots are among the most highly compensated professionals in a seaport. In the U.S., the average yearly salary for marine pilots is more than US$400,000 (NPR News 2012). Moreover, pilotage is a very demanding job that requires intense concentration and high skill levels. Workload management is thus a key component to improve the health of pilots and mitigate fatigue (International Maritime Organization 2001). Therefore, effective and functional pilotage planning is essential to avoid accidents, secure fluency in port operations, reduce port authorities' operational costs, and relieve work-related anxieties.

Traditionally, the berth allocation problem (BAP) and the pilotage planning problem (PPP) are solved sequentially, with the output of the first problem being an input of the second one. However, this sequential solution approach may lead to suboptimal or even infeasible solutions in vessel services.

In this paper, we consider a vessel service planning problem (VSPP) that addresses berth allocation and pilotage planning in an integrated way. To solve large-scale instances, we develop an exact solution approach that embeds Benders decomposition and column generation in a branch-and-bound framework. To the best of our knowledge, our study is the first to solve the BAP and the PPP jointly. From a modeling perspective, the VSPP is similar to the problems in airline or railway operations that involve synchronized scheduling of multiple resources (e.g., the synchronized aircraft routing and crew pairing problem in airlines). However, the uniqueness of the VSPP, as we will discuss in the next section, prohibits the direct applications of existing solution approaches.

Our study makes four main contributions:

1. We propose a research problem that integrates two important problems in seaport operations and develop an exact solution approach that combines Benders decomposition and column generation in an efficient branch-and-bound framework. The approach is shown to outperform a commonly used solution approach for similar problems.

2. The approach uses an efficient label correcting algorithm for generating columns and a tailored branching scheme for identifying integer solutions. We also propose several acceleration strategies that significantly improve the performance of the approach. The structure and the components in the approach are general enough to be applied to other problems that involve synchronized scheduling of multiple resources and can easily be tailored for such purposes.
3. We conduct extensive numerical experiments using data instances from one of the world's largest seaports. The results demonstrate that our solution approach can obtain optimal or near-optimal solutions for real-world instances and that our approach performs robustly under different problem settings.
4. The results also clearly demonstrate the benefits of integrated vessel service planning against the traditional sequential decision method.

The remainder of this paper is structured as follows. We review the relevant studies in Section 2. We formally describe the problem and formulate it as a compact mixed-integer linear programming (MILP) model in Section 3. The solution approach is described in Section 4. Computational experiments are reported in Section 5, followed by conclusions in Section 6. We provide all mathematical proofs in EC.1 of the electronic companion.

2. Literature Review

Seaport operations have received considerable attention in the scientific literature. For surveys of studies in this area, we refer readers to Stahlbock and Voß (2008), Bierwirth and Meisel (2010, 2015), and Carlo et al. (2015). Of many problems related to seaport operations that have been studied in the literature, the ones most closely related to the VSPP are the berth allocation problem, the vessel traffic management problem, and the pilot scheduling problem. In the following subsections, we review studies that focus on these problems.

2.1. Review of Studies on Berth Allocation

The berth allocation problem (BAP) has been the subject of intensive research over the past two decades. Broadly, the BAP can be divided into two streams depending on the spatial constraints on berthing positions: the discrete BAP and the continuous BAP (Imai et al. 2005, Xu and Lee 2018). In the discrete BAP, the quay of the seaport is partitioned into a set of discrete berths, and each berth can handle at most one vessel at a time. In the continuous case, vessels can berth at arbitrary positions within the boundaries of the quay. In this study, we consider discrete berth allocation and thus limit our review to studies in this stream.

The discrete BAP was initially studied by Imai et al. (1997), who presented the first MILP formulation for the problem. In the formulation, all berths are available from the beginning of the

planning horizon, and all vessels are already waiting in the seaport. The objective considered by the authors was to minimize the total turnaround time of vessels. Imai et al. (2001) extended this study by allowing vessels to arrive dynamically during a planning horizon. A Lagrangian relaxation method was proposed to solve the problem. Imai et al. (2003) considered a BAP with service priorities and solved the problem using a genetic algorithm. Cordeau et al. (2005) studied a BAP with time windows on vessel handling completion; they proposed a multi-depot vehicle routing problem formulation and developed a tabu search algorithm for solving the problem. Monaco and Sammarra (2007) strengthened the formulation provided by Imai et al. (2001) and developed a Lagrangian relaxation method for solving the problem. Buhrkal et al. (2011) presented a general set-partitioning model for the discrete BAP that was shown to outperform all other existing models.

2.2. Review of Studies on Vessel Traffic Management

The aim of traffic management in seaports and other restricted waters is to achieve smooth and collision-free traffic flows. The vessel traffic management problem has gained increasing attention from academia in recent years. Using a simulation, Tang et al. (2016) showed that the handling capacity of a seaport may greatly depend on the configuration and capacity of its navigation channel. Building on Tang et al. (2016), most studies in this area have examined ways to manage traffic in seaports based on predetermined berth allocation plans (Zhang et al. 2016, Lalla-Ruiz et al. 2018, Jia et al. 2019, Li and Jia 2019). Zhang et al. (2016) considered a scheduling problem for vessels entering and leaving a seaport and passing through a navigation channel, the objective being to minimize the total waiting time of vessels at berths or in the anchorage. The problem was solved by a metaheuristic based on simulated annealing and genetic search. Lalla-Ruiz et al. (2018) investigated a similar problem where vessel traffic has to be scheduled in two parallel waterways. The authors proved that the problem is NP-hard. A greedy heuristic and a simulated annealing algorithm were proposed to solve the problem. Jia et al. (2019) and Li and Jia (2019) extended the problem studied by Zhang et al. (2016) and Lalla-Ruiz et al. (2018) by considering the use of the anchorage area in the port basin, where vessels can wait near the berths. They proposed Lagrangian relaxation- and column generation-based methods for solving the problem. In addition to vessel traffic management in seaports, Lübbecke et al. (2019) proposed an approach to schedule vessel traffic in the Kiel Canal, a bidirectional waterway in Germany. To mathematically formulate the problem, the authors divided the canal into a set of segments with different nautical rules and showed that the problem shares similarities with *the single-track train scheduling problem*, which can be formulated as a job-shop scheduling problem. A local-search-based heuristic was developed to solve the problem.

The BAP and the traffic management problem in seaports are interconnected and generate more value if solved jointly. The integration of the two problems has been discussed in two studies. Zhen et al. (2017) investigated the BAP with channel capacity limitations such that the total number of vessels simultaneously sailing in a channel cannot exceed a given upper bound. A column generation method was developed to solve the problem. Corry and Bierwirth (2019), noting that channel restrictions based on a maximum throughput capacity may not be sufficient for channel traffic management, proposed and studied a joint BAP and channel traffic control problem. They tackled the problem as a special no-wait flow shop scheduling problem and formulated it as an MILP model. The computational results demonstrated that their model could well solve instances with up to 10 berths and 20 vessels using a general-purpose optimization solver.

2.3. Review of Studies on Pilot Scheduling

Although pilots play a critical role in seaport operations, studies on pilot scheduling are rare. Wermus and Pope (1994) studied a pilot rostering problem to determine work shift start and end times assigned to each pilot, without considering the scheduling of pilotage services. A simple rule-based heuristic was developed to solve the problem. Edwards (2010) investigated a problem for assigning a group of pilots to a set of pilotage tasks, whereby each task corresponds to a vessel entering or leaving a terminal and has a fixed start time. They solved the problem using a heuristic based on column generation.

Two recent works considered the PPP in seaports based on predetermined berth allocation plans. Wu et al. (2020) considered a PPP for jointly scheduling pilots and managing vessel traffic in a seaport. In the problem, work schedules for guiding vessels sailing into and out of berths are generated for pilots working on different shifts. All pilots in the problem were assumed to be homogenous. The authors proved that the PPP is strongly NP-hard and proposed a solution algorithm based on branch-and-price. Jia et al. (2020) investigated an integrated daily vessel traffic and pilot scheduling problem, whereby the seaport has multiple navigation channels and vessels can wait in the anchorage area located near the berths. No working time limits were considered, and pilots were assumed to be able to work all day long. The problem was solved by a Lagrangian relaxation algorithm.

2.4. Review of Other Related Studies

From a modeling perspective, the VSPP jointly solves a resource allocation problem and a personnel scheduling problem. Problems with a similar structure have been studied in other areas. The most well-known examples of such problems include the integrated aircraft routing and crew pairing problem in airlines (Cordeau et al. 2001, Sandhu and Klabjan 2007, Ruther et al. 2017), the

integrated rolling stock and train driver scheduling problem in railways (Dauzère-Pérès et al. 2015), and the integrated vehicle routing and truck driver scheduling problem in road transportation (Goel and Irnich 2017).

The VSPP is different from these problems in multiple ways. First, the problems in airline and railway operations are based on given timetables of flights or trains, and the resources and crew members are scheduled in accordance with these timetables. In the VSPP, however, the times to serve the vessels are also among the decision variables. Second, in the VSPP, pilots can move between the vessels not only through the vessels they serve but also through dedicated transfer vehicles (e.g., pilot boats and helicopters). For airline crews and train or truck drivers, there are no dedicated transfer vehicles that can affect scheduling; rather, personnel movements must conform to the movements of aircraft, trains, and trucks. For this reason, personnel scheduling has more flexibility in the VSPP than in these other problems. A further distinguishing feature of the VSPP relates to regulations on channel traffic, which lead to precedence and non-simultaneity constraints between vessel movements (discussed in the next section); such constraints generally do not exist in the other aforementioned problems.

3. Problem Statement and Compact Formulation

In this section, we first describe the background of the VSPP in Section 3.1 and then introduce the notation used for formulating the problem in Section 3.2. A compact MILP formulation for the VSPP is presented in Section 3.3. We introduce a series of inequalities for strengthening the formulation in Section 3.4.

3.1. Problem Background

In the VSPP, we are given a set of vessels that dynamically call at a seaport in a planning horizon. These vessels go through a four-stage procedure in their port stay. As shown in Figure 2, after arriving at the anchorage, a vessel waits there until it is ready to sail into the channel (Stage 1). Then, a pilot boards the vessel at the entrance of the channel, navigates it through the channel, and moors it into the designated berth (Stage 2). The pilot leaves the vessel once mooring is completed, and cargo handling at the berth commences (Stage 3). When the vessel is ready to leave the berth, a pilot gets on board and maneuvers the vessel toward the anchorage by passing through the channel (Stage 4). Finally, the pilot deboards the vessel in the anchorage, and the vessel returns to the open sea. Without loss of generality, we assume that each vessel is given time windows for sailing into and out of a berth (i.e., for starting the second and fourth stages).

We consider the following restrictions on vessel traffic in a seaport based on common practices (Corry and Bierwirth 2019, Jia et al. 2019). First, vessels must sail at the same slow and constant

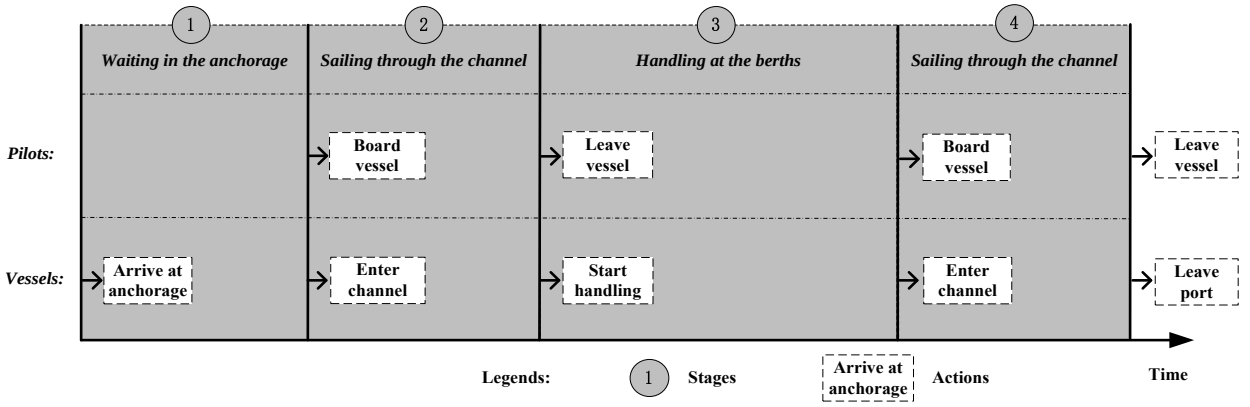


Figure 2 A Vessel's Port Stay.

speed in the channel. Second, vessels sailing in the same direction (i.e., entering or leaving berths) must queue up in the channel, with no overtaking. Third, there must be a minimum headway between a vessel entering the channel and those following it. Fourth, as the water depth is limited and affected by tides, vessels with large drafts may be able to sail in the channel only in high-tide periods. Fifth, to avoid collisions, vessels with extra widths cannot sail simultaneously in opposite directions in the channel. Finally, as vessels are allowed to moor only in the anchorage or at berths, en-route stops are not allowed during the journey between the anchorage and the berths. That is, a vessel has to sail directly to the berth once it leaves the anchorage and has to sail directly to the anchorage once it leaves the berth.

When moving in port waters, a vessel must be navigated by a pilot. The seaport manages a group of pilots and divides the planning horizon into a set of shifts to coordinate pilot assignments. Pilots are scheduled to work during these shifts and can only serve the vessels within their assigned shifts. Depending on the labor policy in a seaport, shifts may or may not overlap. A pilot can serve in multiple vessel movements in one shift, but there is a minimum setup time between two consecutive services conducted by the same pilot. As the repositioning of pilots through the channel can take a considerable amount of time, the setup time between two pilotage services is longer if the first service terminates and the second service starts at different ends of the channel (and is shorter if they occur at the same end of the channel). For example, after a pilot guides a vessel into a berth, the repositioning time is considerably shorter in the case of guiding another vessel out of a berth than in the case of guiding another vessel into a berth. A typical shift lasts eight hours, and each shift includes a rest period during which there are no assignments. Finally, we assume that pilots are identical and that the number of pilots who can work in each shift is not a binding constraint in the problem (but the seaport has to pay for each pilot assigned to a shift).

In the VSPP, the seaport wants to optimize both the service level and operational costs. We quantify the service level by considering delays in cargo dispatch (i.e., delays at the start

of discharging cargo from vessels) and vessel turnaround. Meanwhile, operational costs include vessel handling costs (vessels typically have berth-wise handling costs) and pilot dispatching. The objective of the VSPP is to generate a vessel service plan that achieves the best optimization goal without violating any regulation regarding berth allocation, traffic management, or pilot scheduling.

3.2. Notation

This section describes the notation that we will use to formulate the VSPP as a compact MILP model. For ease of presentation, we use the notation $\mathbb{Z}_{[\underline{z},\overline{z}]}$ to denote the set of integers that are no smaller than $\underline{z}$ and no larger than $\overline{z}$ throughout the paper. In addition, we use $\mathbb{Z}^+$ to denote the set of nonnegative integers and $\mathbb{R}$ to denote the set of real numbers.

We consider the VSPP with a discrete-time finite planning horizon. We denote by $\mathcal{T} = \mathbb{Z}_{[1,\overline{T}]}$ the set of evenly distributed time steps in the planning horizon, which are indexed by t. The planning horizon is divided into a set $\mathcal{S}$ of shifts, indexed by s. Let $\mathcal{T}_s \subseteq \mathcal{T}$ be the set of time steps in shift s, and $\mathcal{T}_s = \mathbb{Z}_{[\underline{T}_s,\overline{T}_s]}$. We have $\bigcup_{s\in\mathcal{S}} \mathcal{T}_s = \mathcal{T}$. Besides, let $\mathcal{S}_t$ be the set of shifts that contain time step t.

Let $\mathcal{K}$, indexed by k, be the set of vessels calling at the seaport in the planning horizon. The port has a set $\mathcal{B}$ of berths indexed by b. Let a_b be the available time of berth b, such that vessels in $\mathcal{K}$ can only moor into berth b at time steps in the set $\mathbb{Z}_{[a_b,\overline{T}]}$. Vessel k can be handled at a berth from a subset $\mathcal{B}_k \subseteq \mathcal{B}$. The handling time of vessel k at berth b is denoted by $h_{k,b}$.

We denote by $\mathcal{I}$, indexed by i, the set of *pilotage tasks* or *tasks* corresponding to pilot services in vessel movements in the seaport. Let $\mathcal{I}^{in} \subseteq \mathcal{I}$ and $\mathcal{I}^{out} \subseteq \mathcal{I}$ be the sets of tasks corresponding to vessels sailing into and out of the berths, respectively. Furthermore, let $\mathcal{I}_k$ be the set of tasks for serving vessel k. Note that $|\mathcal{I}_k| = 2$. The time window to start each task i is given by $\mathcal{E}_i = \mathbb{Z}_{[\underline{E}_i,\overline{E}_i]}$, where $\underline{E}_i$ and $\overline{E}_i$ represent the earliest and latest feasible start times of task i, respectively. The duration of task i is denoted by d_i. For navigational safety, there must be a minimum headway between any two vessels sailing into the channel in the same direction. For modeling this requirement, we define a set $\mathcal{V}$ of task pairs such that $\mathcal{V} := \{(i,j) \in \mathcal{I} \times \mathcal{I} | \text{there exists a minimum headway between } i \text{ and } j,\ i,j \in \mathcal{I}\}$. For each $(i,j) \in \mathcal{V}$, let $f_{i,j}$ be the minimum headway that must be ensured between the start times of tasks i and j. Moreover, a pair of vessels with extra widths are not be allowed to sail in opposite directions in the channel simultaneously. To impose this restriction, we define a set $\mathcal{U}$ of task pairs such that $\mathcal{U} := \{(i,j) \in \mathcal{I} \times \mathcal{I} | i \text{ and } j \text{ cannot be performed simultaneously},\ i,j \in \mathcal{I}\}$.

Let $\mathcal{P}_s$ be set of pilots who can work in shift s. To streamline the formulation, in our model, pilots are "shift-specific" such that $p \in \mathcal{P}_s$ does not correspond to a particular pilot in practice, but refers to a generic pilot who is able to work in shift s. By definition, $\mathcal{P}_s \cap \mathcal{P}_{s'} = \varnothing$ if $s \neq s'$. Because we assume that the number of pilots available in each shift is not binding, when implementing the

model, we let $|\mathcal{P}_s|$ be equal to the maximum number of tasks that can start in shift s. Given $\mathcal{P}_s$, we set $\mathcal{P} = \bigcup_{s\in\mathcal{S}} \mathcal{P}_s$.

Any pilot $p \in \mathcal{P}_s$, $\forall s \in \mathcal{S}$, if scheduled to work in this shift, should be awarded a rest period that starts within a predetermined time window and lasts for at least a given minimum duration. For each s, the time window for pilots working in this shift to start the rest period is denoted by $\mathcal{L}_s$, and $\mathcal{L}_s = \mathbb{Z}_{[\underline{L}_s, \overline{L}_s]}$. Meanwhile, the minimum duration of the rest period in this shift is denoted by g_s. From the modeling perspective, a rest period can also be modeled as a task. Hence, we let $\mathcal{J}_p$ be the set (singleton) of the rest period for pilot p. Corresponding to $i \in \mathcal{J}_p$, we let $\mathcal{E}_i$ and d_i be the time window to start the rest period and its minimum duration, respectively. Here $\mathcal{E}_i = \mathcal{L}_s$ and $d_i = g_s$, $\forall i \in \mathcal{J}_p$, $p \in \mathcal{P}_s$, $s \in \mathcal{S}$. We further define $\mathcal{J} = \bigcup_{p\in\mathcal{P}} \mathcal{J}_p$. For ease of presentation, we use the term *activity* to refer to a task or a rest period in the remainder of this paper.

Pilots may perform multiple activities in a shift. For any two consecutive activities i and j performed by a pilot, we denote by $q_{i,j}$ the minimum setup time between them. In particular, we set $q_{i,j} = 0$ if i or j corresponds to a rest period. We assume that the triangle inequality holds for the setup times between tasks so that $q_{i,j} \le q_{i,i'} + d_{i'} + q_{i',j}, \forall i, i', j \in \mathcal{I},\ i \ne j \ne i'$. Moreover, large constants (big-M) are used for modeling the relationship between the start times of any two activities. To tighten the MILP model, these constants should be set at their smallest feasible values. To this end, $\forall i, j \in \mathcal{I} \cup \mathcal{J}, i \ne j$, let $M_{i,j}$ be the big-M associated with activity pair (i, j), and $M_{i,j}$ is set as

$$M_{i,j} = \begin{cases} \max\{0, \overline{E}_i + d_i + q_{i,j} - \underline{E}_j\}, & \text{if } j \in \mathcal{I}, \\ \overline{E}_i + d_i + q_{i,j}, & \text{if } j \in \mathcal{J}. \end{cases}$$

We consider the following cost components in the VSPP. First, given task i and its start time t, we denote by $c^1_{i,t}$ the penalty cost associated with delays in cargo dispatch (if $i \in \mathcal{I}^{in}$) or vessel turnaround (if $i \in \mathcal{I}^{out}$) for starting this task at this time step. In practice, $c^1_{i,t}$ is set by port authorities based on the service priority of task i and the delay between the earliest start time ($\underline{E}_i$) and time step t. Note that our model can handle any relationship between the penalty cost and the delay. For example, the penalty cost can grow linearly with the delay. The relationship between the penalty cost and the delay can also be a piece-wise linear function that discourages long waiting times for individual vessels. Second, let $c^2_{k,b}$ be the cost of handling vessel k at berth b. Third, the cost of assigning a pilot to work in shift s is denoted by c^3_s.

The following decision variables are used to formulate the VSPP:

$u_{k,b}$: binary variable, which takes value 1 if and only if vessel k is handled at berth b;

$v^{in}_{k,t}$: binary variable, which takes value 1 if and only if vessel k reaches a berth at time step t;

$v^{out}_{k,t}$: binary variable, which takes value 1 if and only if vessel k leaves a berth at time step t;

$w_{k,b,t}$: binary variable, which takes value 1 if and only if vessel k occupies berth b at time step t;

$x_{i,t}$: binary variable, which takes value 1 if and only if task i starts at time step t;

$y_{p,0,j}$: binary variable, which takes value 1 if and only if activity j is the first activity performed by pilot p;

$y_{p,i,j}$: binary variable, which takes value 1 if and only if pilot p performs activity j immediately after activity i;

$y_{p,i,0}$: binary variable, which takes value 1 if and only if activity i is the last activity performed by pilot p;

$z_{s,i}$: binary variable, which takes value 1 if and only if task i is performed by a pilot in shift s;

n_s: integer variable, representing the number of pilots working in shift s.

3.3. The VSPP Formulation

We present a compact MILP formulation for the VSPP, which is an extension of the formulations for the BAP, the vessel traffic management problem, and the PPP studied by Zhen et al. (2017), Jia et al. (2019), Xie et al. (2019), Wu et al. (2020), and Jia et al. (2020). We first present the objective function:

$$[\mathbf{M1}] \quad \min \sum_{i\in\mathcal{I}}\sum_{t\in\mathcal{E}_i} c^1_{i,t}x_{i,t} + \sum_{k\in\mathcal{K}}\sum_{b\in\mathcal{B}_k} c^2_{k,b}u_{k,b} + \sum_{s\in\mathcal{S}} c^3_s n_s. \tag{1}$$

This objective function minimizes the total cost of the vessel service plan, which equals the sum of the penalty cost of starting tasks later than their earliest feasible start times (the first term), the cost of handling vessels (the second term), and the cost of assigning pilots (the third term). The constraints can be divided into four blocks: constraints (2)–(8) regulate berth allocation; constraints (9)–(13) control vessel traffic in the channel; constraints (14)–(23) are related to pilot scheduling; and the domains of the variables are defined in constraints (24)–(30). The first block is the following:

$$\sum_{b\in\mathcal{B}_k} u_{k,b} = 1 \ \forall k \in \mathcal{K} \tag{2}$$

$$\sum_{t\in\mathcal{T}} v^{in}_{k,t} = 1 \ \forall k \in \mathcal{K} \tag{3}$$

$$\sum_{t\in\mathcal{T}} v^{out}_{k,t} = 1 \ \forall k \in \mathcal{K} \tag{4}$$

$$\sum_{t\in\mathcal{T}} t v^{in}_{k,t} \geq \sum_{b\in\mathcal{B}_k} a_b u_{k,b} \ \forall k \in \mathcal{K} \tag{5}$$

$$\sum_{t\in\mathcal{T}} tv^{out}_{k,t} \geq \sum_{t\in\mathcal{T}} tv^{in}_{k,t} + \sum_{b\in\mathcal{B}_k} h_{k,b}u_{k,b} \ \forall k\in\mathcal{K} \tag{6}$$

$$w_{k,b,t} \geq u_{k,b} + \sum_{t'=1}^{t} v^{in}_{k,t'} + \sum_{t'=\min\{t+1,\overline{T}\}}^{\overline{T}} v^{out}_{k,t'} - 2 \ \forall t\in\mathcal{T}, b\in\mathcal{B}, k\in\mathcal{K} \tag{7}$$

$$\sum_{k\in\mathcal{K}} w_{k,b,t} \leq 1 \ \forall t\in\mathcal{T}, b\in\mathcal{B}. \tag{8}$$

Constraints (2) require that a suitable berth be allocated for handling each vessel. Constraints (3) and (4) require that each vessel reaches and leaves a berth at one and only one time step, respectively. A vessel cannot moor into a berth before the berth becomes available (constraints (5)). Constraints (6) ensure that a vessel has sufficient time for cargo handling when mooring at a berth. Constraints (7) state that berth b is occupied by vessel k at time step t ($w_{k,b,t}=1$) if (i) the berth is allocated to handle the vessel ($u_{k,b}=1$), (ii) the vessel has reached the berth at the time step ($\sum_{t'=1}^{t} v^{in}_{k,t'}=1$), and (iii) the vessel has not left the berth ($\sum_{t'=\min\{t+1,\overline{T}\}}^{\overline{T}} v^{out}_{k,t'}=1$). At any time, a berth can handle at most one vessel, as required by constraints (8). The second block contains these constraints:

$$\sum_{t\in\mathcal{E}_i} x_{i,t} = 1 \ \forall i\in\mathcal{I} \tag{9}$$

$$\sum_{t'\in\mathbb{Z}_{[t-f_{i,j}+1,t]}\cap\mathcal{E}_i} x_{i,t'} + \sum_{t'\in\{t\}\cap\mathcal{E}_j} x_{j,t'} \leq 1 \ \forall t\in\mathcal{T}, (i,j)\in\mathcal{V} \tag{10}$$

$$\sum_{t'\in\mathbb{Z}_{[t-d_i+1,t]}\cap\mathcal{E}_i} x_{i,t'} + \sum_{t'\in\mathbb{Z}_{[t-d_j+1,t]}\cap\mathcal{E}_j} x_{j,t'} \leq 1 \ \forall t\in\mathcal{T}, (i,j)\in\mathcal{U} \tag{11}$$

$$\sum_{t\in\mathcal{T}} tv^{in}_{k,t} = d_i + \sum_{t\in\mathcal{E}_i} tx_{i,t} \ \forall i\in\mathcal{I}^{in}\cap\mathcal{I}_k, k\in\mathcal{K} \tag{12}$$

$$\sum_{t\in\mathcal{T}} tv^{out}_{k,t} = \sum_{t\in\mathcal{E}_i} tx_{i,t} \ \forall i\in\mathcal{I}^{out}\cap\mathcal{I}_k, k\in\mathcal{K}. \tag{13}$$

Constraints (9) require that each pilotage starts within a given time window. The minimum headway between any vessel sailing into the channel and those following it is imposed by constraints (10). In these constraints, given a time step $t\in\mathcal{T}$ and a pair of tasks $(i,j)\in\mathcal{V}$, if i *starts* within the set of time points denoted by $\mathbb{Z}_{[t-f_{i,j}+1,t]}$, then the minimum headway requirement prevents j from *starting* at time step t. Because we have predetermined time windows for *starting* the tasks, these constraints remain valid when only the time steps within the respective time windows (i.e., $\mathcal{E}_i$ and $\mathcal{E}_j$) are considered. Constraints (11) enforce that tasks in any pair in set $\mathcal{U}$ cannot be performed simultaneously. In constraints (11), given a time step $t\in\mathcal{T}$ and a pair of tasks $(i,j)\in\mathcal{U}$, task i (resp. j) is active if and only if the task *starts* within the set of periods $\mathbb{Z}_{[t-d_i+1,t]}$ (resp. $\mathbb{Z}_{[t-d_j+1,t]}$). Because tasks are only allowed to *start* within their time windows, given $t\in\mathcal{T}$ and $(i,j)\in\mathcal{U}$, task i (resp. j) is active if and only if the task starts within the set of

periods $\mathbb{Z}_{[t-d_i+1,t]} \cap \mathcal{E}_i$ (resp. $\mathbb{Z}_{[t-d_j+1,t]} \cap \mathcal{E}_j$). Hence, for any task pair $(i,j) \in \mathcal{U}$ these constraints prevent tasks i and j from being both active at any time step $t \in \mathcal{T}$. Start times of pilotage tasks associated with a vessel are linked with the times when the vessel enters and leaves a berth in constraints (12) and (13). Note that $\mathcal{I}^{in} \cap \mathcal{I}_k$ and $\mathcal{I}^{out} \cap \mathcal{I}_k$ are singletons, $\forall k \in \mathcal{K}$. The following constraints constitute the third block:

$$\sum_{j \in \mathcal{I} \cup \mathcal{J}_p} y_{p,0,j} \leq 1 \ \forall p \in \mathcal{P} \tag{14}$$

$$y_{p,0,i} + \sum_{j \in \mathcal{I} \cup \mathcal{J}_p \setminus \{i\}} y_{p,j,i} = y_{p,i,0} + \sum_{j \in \mathcal{I} \cup \mathcal{J}_p \setminus \{i\}} y_{p,i,j} \ \forall i \in \mathcal{I} \cup \mathcal{J}_p, p \in \mathcal{P} \tag{15}$$

$$\sum_{j \in \mathcal{I} \cup \mathcal{J}_p} y_{p,0,j} = \sum_{i \in \mathcal{I} \cup \mathcal{J}_p} y_{p,i,0} \ \forall p \in \mathcal{P} \tag{16}$$

$$\sum_{p \in \mathcal{P}_s} y_{p,0,i} + \sum_{p \in \mathcal{P}_s} \sum_{j \in \mathcal{I} \cup \mathcal{J}_p \setminus \{i\}} y_{p,j,i} = z_{s,i} \ \forall i \in \mathcal{I}, s \in \mathcal{S} \tag{17}$$

$$x_{i,t} \leq \sum_{s \in \mathcal{S}_t} z_{s,i} \ \forall t \in \mathcal{E}_i, i \in \mathcal{I} \tag{18}$$

$$\sum_{t \in \mathcal{E}_i} x_{i,t} = \sum_{j \in \mathcal{I} \cup \mathcal{J}_p} y_{p,0,j} \ \forall i \in \mathcal{J}_p, p \in \mathcal{P} \tag{19}$$

$$y_{p,0,i} + \sum_{j \in \mathcal{I} \cup \mathcal{J}_p \setminus \{i\}} y_{p,j,i} = \sum_{j \in \mathcal{I} \cup \mathcal{J}_p} y_{p,0,j} \ \forall i \in \mathcal{J}_p, p \in \mathcal{P} \tag{20}$$

$$\sum_{t \in \mathcal{E}_j} t x_{j,t} \geq \sum_{t \in \mathcal{E}_i} t x_{i,t} + d_i + q_{i,j} + M_{i,j}(y_{p,i,j} - 1) \ \forall i,j \in \mathcal{I} \cup \mathcal{J}_p, i \neq j, p \in \mathcal{P} \tag{21}$$

$$\sum_{t \in \mathcal{E}_j} t x_{j,t} \geq \sum_{t \in \mathcal{E}_i} t x_{i,t} + d_i + q_{i,j} + M_{i,j}(y_{p,i,i'} + y_{p,i',j} - 2) \quad \forall i,j \in \mathcal{I}, i \neq j, i' \in \mathcal{J}_p, p \in \mathcal{P} \tag{22}$$

$$\sum_{p \in \mathcal{P}_s} \sum_{i \in \mathcal{I} \cup \mathcal{J}_p} y_{p,0,i} = n_s \ \forall s \in \mathcal{S}. \tag{23}$$

Each pilot has at most one starting activity in the planning horizon (constraints (14)). Constraints (15) and (16) enforce the activity flow balance for each pilot. Constraints (17) link the z and y variables. Constraints (18) ensure that if a task i starts at time step t, then it should be performed by a pilot who works in a shift that contains the time step. A rest period has to be arranged for a pilot that works in the planning horizon, and it should start within the allowable time window (constraints (19)). Constraints (20) require that the rest period be included in the activity flow of a pilot if the pilot works in the planning horizon. The minimum setup time between two consecutive activities performed by one pilot is imposed by constraints (21). Constraints (22) avoid infeasible setup times in the event that the duration of a rest period is shorter than the time required to reposition a pilot between two tasks that start right before and immediately after the

rest period. We calculate the number of pilots required to work in each shift in constraints (23). The domains for the decision variables are defined in the constraints listed below:

$$u_{k,b} \in \{0,1\} \ \forall b \in \mathcal{B}_k, k \in \mathcal{K} \tag{24}$$

$$v^{in}_{k,t}, v^{out}_{k,t} \in \{0,1\} \ \forall k \in \mathcal{K}, t \in \mathcal{T} \tag{25}$$

$$w_{k,b,t} \in \{0,1\} \ \forall k \in \mathcal{K}, b \in \mathcal{B}, t \in \mathcal{T} \tag{26}$$

$$x_{i,t} \in \{0,1\} \ \forall t \in \mathcal{E}_i, i \in \mathcal{I} \cup \mathcal{J} \tag{27}$$

$$y_{p,0,j}, y_{p,i,j}, y_{p,i,0} \in \{0,1\} \ \forall i,j \in \mathcal{I} \cup \mathcal{J}_p, i \neq j, p \in \mathcal{P} \tag{28}$$

$$z_{s,i} \in \{0,1\} \ \forall i \in \mathcal{I}, s \in \mathcal{S} \tag{29}$$

$$n_s \in \mathbb{Z}^+ \ \forall s \in \mathcal{S}. \tag{30}$$

Our model is formulated for the VSPP faced by a seaport with a bidirectional channel that has a constant nautical condition in each direction. However, the model can easily be adapted to solve the VSPP under different channel configurations. For example, the channels in some seaports are unidirectional (i.e., vessels must sail in the same direction in these channels). To handle the VSPP under such a channel configuration, one can set $\mathcal{U}$ in the model to include all task pairs that consist of one task corresponding to a vessel entering berths and the other corresponding to a vessel leaving berths. In addition, some seaports have channels with multiple segments, some bidirectional and others unidirectional (Corry and Bierwirth 2019). To handle such cases, one can extend $\mathcal{U}$ to be $\mathcal{U}' := \{(i,t,j,t') \in \mathcal{I} \times \mathcal{T} \times \mathcal{I} \times \mathcal{T} | i$ cannot start at time step t if j starts at time step t', $i,j \in \mathcal{I}, t,t' \in \mathcal{T}\}$. By carefully setting $\mathcal{U}'$, one can avoid having two vessels sail in opposite directions in a unidirectional segment at the same time.

3.4. Valid Inequalities

Model M1 can be strengthened by several families of valid inequalities. The first family of inequalities tightens the model by shrinking the time windows for starting the tasks. This is achieved by considering the precedence relationship between tasks and the requirements regarding the rest period in each shift. In particular, a vessel cannot leave its berth before completing cargo handling, and the earliest time step when vessel k can complete cargo handling (denoted by $\underline{F}^{out}_k$) can be calculated by $\underline{F}^{out}_k = \min_{b \in \mathcal{B}_k}\{\max\{\underline{E}_i + d_i, a_b\} + h_{k,b}\}$, where i denotes the pilotage task associated with vessel k sailing into berths. Similarly, to ensure that vessels leave their berths within the given time windows, vessel k must start sailing into the berths no later than $\overline{F}^{in}_k = \overline{E}_j - \min_{b \in \mathcal{B}_k}\{h_{k,b}\} - d_i$, where i and j represent the pilotage tasks associated with vessel k sailing into and out of berths, respectively.

In addition, considering the requirements regarding rest periods, any pilot working in shift s cannot start performing task i at time step t if $t < \underline{L}_s + g_s$ (which disallows resting before the task) and $t > \overline{L}_s - d_i$ (which disallows resting after the task). Based on this observation, for each $i \in \mathcal{I}$ and $s \in \mathcal{S}$, we let $\mathcal{Q}_{s,i} := \{t | t < \underline{L}_s + g_s, t > \overline{L}_s - d_i, t \in \mathcal{E}_i\}$, which is the set of time steps in shift s when task i cannot start. Define $\bar{\mathcal{Q}}_i := \bigcap_{s \in \mathcal{S}} \mathcal{Q}_{s,i}$. It can be readily seen that the following inequalities are valid for the VSPP:

$$x_{i,t} = 0 \ \forall t \in \mathcal{E}_i \cap \mathbb{Z}_{[1, \underline{F}_k^{out} - 1]}, i \in \mathcal{I}^{out} \cap \mathcal{I}_k, k \in \mathcal{K} \tag{31}$$

$$x_{i,t} = 0 \ \forall t \in \mathcal{E}_i \cap \mathbb{Z}_{[\overline{F}_k^{in} + 1, \overline{T}]}, i \in \mathcal{I}^{in} \cap \mathcal{I}_k, k \in \mathcal{K} \tag{32}$$

$$x_{i,t} = 0 \ \forall t \in \mathcal{E}_i \cap \bar{\mathcal{Q}}_i, i \in \mathcal{I}. \tag{33}$$

In the second family of valid inequalities, we tighten M1 by using the relationship between x and z in any *optimal* solution to M1. These inequalities are as follows:

$$z_{s,i} \leq \sum_{t \in \mathcal{T}_s \cap \mathcal{E}_i} x_{i,t} \ \forall i \in \mathcal{I}, s \in \mathcal{S}. \tag{34}$$

The constraints enforce $z_{s,i} = 0$ if i does not start at any time step in shift s, $s \in \mathcal{S}, i \in \mathcal{I}$.

Finally, because all pilots in $\mathcal{P}_s$ are identical $\forall s \in \mathcal{S}$, we can further strengthen the formulation by breaking symmetries in pilot scheduling. To this end, we present $\mathcal{P}_s$ and $\mathcal{I}$ as $\mathcal{P}_s = \{\underline{P}_s, \underline{P}_s + 1, ..., \overline{P}_s\}$ and $\mathcal{I} = \{1, 2, ..., |\mathcal{I}|\}$, respectively. By following Jans (2009) and Adulyasak et al. (2014), we introduce the third family of valid inequalities as follows:

$$\sum_{j \in \mathcal{I} \cup \mathcal{J}_p} y_{p,0,j} \geq \sum_{j \in \mathcal{I} \cup \mathcal{J}_{p+1}} y_{p+1,0,j} \ \forall \underline{P}_s \leq p \leq \overline{P}_s - 1, s \in \mathcal{S} \tag{35}$$

$$\sum_{i'=1}^{i} (y_{p,0,i'} + \sum_{j \in \mathcal{I} \cup \mathcal{J}_p \setminus \{i'\}} y_{p,j,i'}) 2^{(i-i')} \geq \sum_{i'=1}^{i} (y_{p+1,0,i'} + \sum_{j \in \mathcal{I} \cup \mathcal{J}_{p+1} \setminus \{i'\}} y_{p+1,j,i'}) 2^{(i-i')}$$
$$\forall i \in \mathcal{I}, \underline{P}_s \leq p \leq \overline{P}_s - 1, s \in \mathcal{S}. \tag{36}$$

We refer to Model M1 with inequalities (31)–(36) as SM1.

4. A Branch-price-and-Benders-cut Approach

In the previous section, we formulate the VSPP as a compact MILP model. Using a general-purpose optimization solver (e.g., CPLEX), model SM1 can be well solved for instances of small scale. However, this "throwing a model into a solver" method fails to deliver satisfying solutions for large-scale instances. In this section, we propose an exact branch-price-and-Benders-cut (BPBC) approach to solve the VSPP on large instances.

Our approach is based on the set-covering reformulation of SM1 (see Section 4.1). The approach solves the set-covering model by first relaxing the integrality requirements for the columns and

gradually imposing these constraints using a branch-and-bound method (see Section 4.4). At each node of the branch-and-bound tree, one thus obtains a linear programming (LP) relaxation of the set-covering problem. We solve the LP relaxation of the set-covering problem at each node by Benders decomposition (Benders 1962).

The master problem in the decomposition can be viewed as the LP relaxation of the *berth allocation problem with vessel traffic flow control*, and the subproblem can be viewed as the LP relaxation of the *pilot scheduling problem* (see Section 4.2). Both the master problem and the subproblem are solved by Dantzig-Wolfe decomposition (Dantzig and Wolfe 1960) through dynamic column(-and-constraint) generation (see Section 4.3).

When a feasible integer solution is generated at a node, an upper bound for the problem is obtained, which can be used to prune branches of the tree. The approach terminates when there are no active nodes in the tree or when the optimality gap is lower than a given threshold. To improve the performance of the approach, we propose a series of acceleration techniques, which are explained in Section 4.5.

Our approach is inspired by the *three-phase Benders decomposition and column generation* method, which was proposed by Cordeau et al. (2001) and has been applied to solve many complicated optimization problems (Mercier et al. 2005, Zeighami and Soumis 2019). Note that this method provides valid lower and upper bounds but does not guarantee to converge to optimality (Cordeau et al. 2001), because the integrality constraints for the subproblem in Benders decomposition are relaxed in the entire process of the method except for the last step, where the integer subproblem is solved only once given a feasible integer solution for the master problem. In many studies (e.g., Mercier et al. 2005, Zeighami and Soumis 2019), however, the optimality gap when the algorithm terminates is sufficiently small to guarantee high-quality solutions.

Our approach extends this method in two directions. First, we design a tailored branching scheme that enables the approach to find an optimal solution to the VSPP. Second, we propose several acceleration strategies that have proven to be very effective in improving the performance of the approach. The framework of the BPBC approach and some acceleration strategies are sufficiently general and adaptable to be applied to solve other similar problems.

4.1. The Set-covering Formulation

In the VSPP, the port stay of a vessel can be depicted as a *vessel route* that records the movements of the vessel in the seaport. In particular, given a vessel $k \in \mathcal{K}$ in the VSPP, a vessel route for this vessel specifies the following information for its port stay: (i) the time steps to start Stages 2, 3, and 4 in its port stay and (ii) the berth where it is handled.

Let Ω, indexed by ω, denote the set of feasible vessel routes that satisfy constraints (2)–(6), (9), (12), (13), (24), (25), (27), and (31)–(33). We denote by $\Omega_k \subseteq \Omega$ the routes for vessel k. For each ω, let $k(\omega)$ denote the associated vessel. We define two binary parameters $\alpha_{\omega,i,t}$ and $\beta_{\omega,b,t,t'}$, $\forall i \in \mathcal{I}, b \in \mathcal{B}, t, t' \in \mathcal{T}, \omega \in \Omega$, where $\alpha_{\omega,i,t}$ is equal to 1 if and only if task i is included in the route and the task starts at time step t, and $\beta_{\omega,b,t,t'}$ is equal to 1 if and only if vessel $k(\omega)$ reaches berth b at time step t and leaves the berth at time step t'. Besides, let $\bar{c}_\omega$ denote the cost of the route, which is calculated by $\bar{c}_\omega = \sum_{i\in\mathcal{I}} \sum_{t\in\mathcal{E}_i} \alpha_{\omega,i,t} c^1_{i,t} + \sum_{b\in\mathcal{B}} \sum_{t\in\mathcal{T}} \sum_{t'\in\mathcal{T}} \beta_{\omega,b,t,t'} c^2_{k(\omega),b}$.

The activities performed by a pilot in a shift can also be represented as a *pilot route* that records the tasks and rest period conducted by the pilot. In particular, a pilot route for a pilot in a shift specifies (i) a sequence of pilotage tasks and the rest period to be conducted by the pilot and also (ii) the start time of each activity contained in this sequence.

Let Φ, indexed by ϕ, denote the set of feasible pilot routes that satisfy constraints (14)–(16), (21)–(22), (27) (for all feasible start times of each activity), and (28). Let $\Phi_s \subseteq \Phi$ denote the set of pilot routes for a pilot in shift s. The cost of a pilot route $\phi \in \Phi_s$ is denoted by $\bar{c}_\phi$ and is equal to c^3_s. For each ϕ, let $\gamma_{\phi,i,t}$ be a binary parameter set to 1 if and only if task i is covered in the route and the task starts at time step t, $\forall i \in \mathcal{I}, t \in \mathcal{T}$.

In the set-covering model, we use binary decision variables χ_ω to indicate the flows in vessel routes and binary decision variables μ_ϕ to indicate the flows in pilot routes. The model can be formulated as follows:

$$[\mathbf{M2}] \quad \min \sum_{\omega\in\Omega} \chi_\omega \bar{c}_\omega + \sum_{\phi\in\Phi} \mu_\phi \bar{c}_\phi \tag{37}$$

$$\text{s.t.} \sum_{\omega\in\Omega_k} \chi_\omega = 1 \ \forall k \in \mathcal{K} \tag{38}$$

$$\sum_{\omega\in\Omega} \chi_\omega \sum_{t=1}^{\tau} \sum_{t'=\min\{\tau+1,\overline{T}\}}^{\overline{T}} \beta_{\omega,b,t,t'} \le 1 \ \forall \tau \in \mathcal{T}, b \in \mathcal{B} \tag{39}$$

$$\sum_{\omega\in\Omega} \chi_\omega \sum_{t'=\max\{t-f_{i,j}+1,1\}}^{t} \alpha_{\omega,i,t'} + \sum_{\omega\in\Omega} \chi_\omega \alpha_{\omega,j,t} \le 1 \ \forall t \in \mathcal{T}, (i,j) \in \mathcal{V} \tag{40}$$

$$\sum_{\omega\in\Omega} \chi_\omega \sum_{t'=\max\{t-d_i+1,1\}}^{t} \alpha_{\omega,i,t'} + \sum_{\omega\in\Omega} \chi_\omega \sum_{t'=\max\{t-d_j+1,1\}}^{t} \alpha_{\omega,j,t'} \le 1 \ \forall t \in \mathcal{T}, (i,j) \in \mathcal{U} \tag{41}$$

$$\sum_{\phi\in\Phi} \mu_\phi \gamma_{\phi,i,t} - \sum_{\omega\in\Omega} \chi_\omega \alpha_{\omega,i,t} = 0 \ \forall t \in \mathcal{T}, i \in \mathcal{I} \tag{42}$$

$$\chi_\omega \in \{0,1\} \ \forall \omega \in \Omega \tag{43}$$

$$\mu_\phi \in \{0,1\} \ \forall \phi \in \Phi. \tag{44}$$

The objective function (37) minimizes the sum of vessel movement scheduling cost and pilot scheduling cost. Each vessel is matched with one and only one vessel route (constraints (38)). The

berth capacity limitations are imposed by constraints (39). Constraints (40) enforce the minimum headway requirements for vessels sailing into the channel. The non-simultaneity constraints between tasks are given in (41). Constraints (42) connect vessel routes with pilot routes. Constraints (43) and (44) define the variables to be binary.

The number of feasible vessel routes and pilot routes can be prohibitively large even for small-scale instances, and enumerating all feasible routes is technically impossible. Therefore, our approach solves M2 through column generation (i.e., by gradually generating vessel routes and pilot routes that may be used in an optimal solution). The method starts from a restricted LP relaxation of M2, where only small initial sets of vessel and pilot routes are included (to ensure the feasibility of the model) and the integrality requirements for the routes in constraints (43) and (44) are dropped. Next, vessel routes and pilot routes that may possibly reduce the objective function value are added to the problem, and integrality constraints are imposed gradually by a branch-and-bound method.

At each node of the branch-and-bound tree, one obtains an LP relaxation of M2. However, due to constraints (42) that couple the vessel routes and the pilot routes, the LP relaxation of M2 is difficult to solve by column generation directly. We observe that once the χ_ω variables are fixed, the problem reduces to a pilot scheduling problem in a (generally) sparse space-time network, which is much easier to solve. Based on this motivation, we propose a Benders decomposition method to solve the LP relaxation of M2. We explain the method in the next section.

4.2. Benders Reformulation

In this section, we explain how to decompose the LP relaxation of M2 at a node in the branch-and-bound tree. We start by formulating the *primal Benders subproblem*, then introduce the dual of it, and finally present the *Benders master problem*.

Given a node in the branch-and-bound tree, let $\mathbf{X}$ be the set of vectors for the χ_ω variables that satisfy $0 \le \chi_\omega \le 1$, $\forall \omega \in \Omega$ and satisfy constraints (38)–(41). For any given vector $\bar{\boldsymbol{\chi}} \in \mathbf{X}$, the LP relaxation of M2 reduces to the following primal Benders subproblem (PBSP) involving only μ_ϕ variables:

$$[\mathbf{PBSP}] \quad \min \sum_{\phi \in \Phi} \mu_\phi \bar{c}_\phi \tag{45}$$

$$\text{s.t.} \sum_{\phi \in \Phi} \mu_\phi \gamma_{\phi,i,t} - \sum_{\omega \in \Omega} \bar{\chi}_\omega \alpha_{\omega,i,t} = 0 \ \forall i \in \mathcal{I}, t \in \mathcal{T} \tag{46}$$

$$\mu_\phi \le 1 \ \forall \phi \in \Phi \tag{47}$$

$$\mu_\phi \ge 0 \ \forall \phi \in \Phi. \tag{48}$$

When solving the PBSP through column generation, in the pricing subproblem we only need to consider tasks i and time steps t such that $\sum_{\omega\in\Omega}\bar{\chi}_\omega\alpha_{\omega,i,t}>0$. This enables us to formulate the pricing subproblem in a space-time network with significantly fewer nodes than the original network that contains all time steps for starting a task. Due to this favorable structure, the PBSP can be solved very efficiently.

We let $\boldsymbol{\delta}=(\delta_{i,t}|i\in\mathcal{I},t\in\mathcal{T})$ and $\boldsymbol{\zeta}=(\zeta_\phi|\phi\in\Phi)$ denote the vectors of dual variables associated with constraints (46) and (47), respectively. The dual of the Benders primal subproblem, called the *dual Benders subproblem* (DBSP), can be formulated as follows:

$$[\mathbf{DBSP}]\quad \max \sum_{i\in\mathcal{I}}\sum_{t\in\mathcal{T}}\sum_{\omega\in\Omega}\bar{\chi}_\omega\alpha_{\omega,i,t}\delta_{i,t}+\sum_{\phi\in\Phi}\zeta_\phi \tag{49}$$

$$\text{s.t.}\ \sum_{i\in\mathcal{I}}\sum_{t\in\mathcal{T}}\gamma_{\phi,i,t}\delta_{i,t}+\zeta_\phi\le\bar{c}_\phi\ \forall\phi\in\Phi \tag{50}$$

$$\delta_{i,t}\in\mathbb{R}\ \forall i\in\mathcal{I},t\in\mathcal{T} \tag{51}$$

$$\zeta_\phi\le 0\ \forall\phi\in\Phi. \tag{52}$$

Proposition 1. *The DBSP is always feasible and bounded.*

Proposition 1 indicates that it is sufficient to add only *Benders optimality cuts* in the Benders master problem. Let Δ denote the polyhedron defined by constraints (50)–(52), and let Γ_Δ be the set of extreme points of Δ. Introducing the additional variable η, the LP relaxation of M2 can thus be reformulated as the following Benders master problem (BMP):

$$[\mathbf{BMP}]\quad \min\sum_{\omega\in\Omega}\chi_\omega\bar{c}_\omega+\eta \tag{53}$$

$$\text{s.t.}\quad (38)\text{–}(41)$$

$$\eta\ge\sum_{i\in\mathcal{I}}\sum_{t\in\mathcal{T}}\sum_{\omega\in\Omega}\chi_\omega\alpha_{\omega,i,t}\delta_{i,t}+\sum_{\phi\in\Phi}\zeta_\phi\ \forall(\boldsymbol{\delta},\boldsymbol{\zeta})\in\Gamma_\Delta \tag{54}$$

$$\chi_\omega\le 1\ \forall\omega\in\Omega \tag{55}$$

$$\chi_\omega\ge 0\ \forall\omega\in\Omega \tag{56}$$

$$\eta\ge 0. \tag{57}$$

Observe that the BMP contains a large number of Benders optimality cuts (54). Enumerating these cuts all at once is practically impossible. Therefore, we design an iterative algorithm that generates only a subset of these cuts in a cutting-plane manner. The algorithm will be explained in Section 4.3.

Algorithm 1 Column and Benders Cut Generation (C&BCG).

Initialize the set of Benders cuts as $\tilde{\Gamma}_\Delta = \Gamma^0_\Delta$. ▷ The algorithm can start with $\Gamma^0_\Delta = \varnothing$.
while 1 **do**
  Solve the BMP with the current set $\tilde{\Gamma}_\Delta$ of Benders cuts by column generation (Section 4.3.1); let Z^*_{BMP} and $(\boldsymbol{\chi}^*, \eta^*)$ denote the optimal objective function value and the optimal solution to the BMP, respectively.
  Let $\bar{\boldsymbol{\chi}} = \boldsymbol{\chi}^*$, and solve the PBSP by column generation (Section 4.3.2); let Z^*_{PBSP} and $\boldsymbol{\mu}^*$ denote the optimal objective function value and the optimal solution to the PBSP, respectively; let $(\boldsymbol{\delta}^*, \boldsymbol{\zeta}^*)$ denote the vectors of the values of the associated optimal dual variables.
  if $Z^*_{PBSP} > \eta^*$ **then**
    Update $\tilde{\Gamma}_\Delta = \tilde{\Gamma}_\Delta \cup (\boldsymbol{\delta}^*, \boldsymbol{\zeta}^*)$.
  else
    Break.
  end if
end while
Return Z^*_{BMP}, $\boldsymbol{\chi}^*$, and $\boldsymbol{\mu}^*$.

4.3. Column and Benders Cut Generation

We solve the BMP at each node with a column and Benders cut generation (C&BCG) algorithm. Let Γ^0_Δ be the initial set of Benders cuts i.e., extreme points $(\boldsymbol{\delta}, \boldsymbol{\zeta})$ for the BMP. The framework of the algorithm can be presented in Algorithm 1.

The method to initialize the set of Benders cuts at the beginning of the C&BCG algorithm will be explained in Section 4.5.4. In the C&BCG algorithm, both the BMP and the PBSP are solved by Dantzig-Wolfe decomposition (Dantzig and Wolfe 1960), which iterates between a restricted master problem and a pricing subproblem for column generation. By leveraging the time-space network structure of the problem, we develop efficient polynomial-time algorithms for solving the pricing subproblems for the BMP and the PBSP. Details of the column generation methods for the two problems are explained in Sections 4.3.1 and 4.3.2, respectively.

4.3.1. Column Generation for the BMP To define the pricing problems in the Dantzig-Wolfe decomposition of the BMP, we let $\boldsymbol{\pi} = (\pi_k | k \in \mathcal{K})$, $\boldsymbol{\lambda} = (\lambda_{b,t} | t \in \mathcal{T}, b \in \mathcal{B})$, $\boldsymbol{\theta} = (\theta_{t,(i,j)} | t \in \mathcal{T}, (i,j) \in \mathcal{V})$, $\boldsymbol{\kappa} = (\kappa_{t,(i,j)} | t \in \mathcal{T}, (i,j) \in \mathcal{U})$, and $\boldsymbol{\varphi} = (\varphi_{(\boldsymbol{\delta},\boldsymbol{\zeta})} | (\boldsymbol{\delta}, \boldsymbol{\zeta}) \in \tilde{\Gamma}_\Delta)$ denote the vectors of dual variables associated with constraints (38)–(41) and (54), respectively.

New vessel routes are generated by solving a pricing problem for each vessel $k \in \mathcal{K}$ and each berth that can handle it (i.e., $b \in \mathcal{B}_k$). We denote the pricing problem associated with vessel k and berth b by $\text{MPP}_{k,b}$ (here, the letter "M" represents BMP, and we share use "SPP" to denote pricing problems for the PBSP). The $\text{MPP}_{k,b}$ can be defined as a problem that identifies the arc with the

minimum cost in a space-time network $G_{k,b}=(N_{k,b},A_{k,b})$, where $N_{k,b}$ is the set of nodes and $A_{k,b}$ is the set of arcs. In particular, $N_{k,b}=\{(i,t)|t\in\mathcal{E}_i,i\in\mathcal{I}_k\}$, and $A_{k,b}=\{[(i,t),(i',t')]|t'-t\geq d_i+h_{k,b},(i,t),(i',t')\in N_{k,b},i\in\mathcal{I}^{in},i'\in\mathcal{I}^{out}\}$. The cost of sending a unit flow through arc $[(i,t),(i',t')]$, denoted by $\hat{c}_{[(i,t),(i',t')]}$, is calculated by

$$\begin{aligned}\hat{c}_{[(i,t),(i',t')]}=c^1_{i,t}+c^1_{i',t'}+c^2_{k,b}-\pi_k-\sum_{\tau=t+d_i}^{\tau=t'-1}\lambda_{\tau,b}-\sum_{(i,j)\in\mathcal{V}}\sum_{\tau=t}^{t+f_{i,j}-1}\theta_{\tau,(i,j)}-\sum_{(i',j)\in\mathcal{V}}\sum_{\tau=t}^{t+f_{i',j}-1}\theta_{\tau,(i',j)}\\-\sum_{(j,i)\in\mathcal{V}}\theta_{t,(j,i)}-\sum_{(j,i')\in\mathcal{V}}\theta_{t',(j,i')}-\sum_{(i,j)\in\mathcal{U}}\sum_{\tau=t}^{t+d_i-1}\kappa_{\tau,(i,j)}-\sum_{(i',j)\in\mathcal{U}}\sum_{\tau=t'}^{t'+d_{i'}-1}\kappa_{\tau,(i',j)}\\-\sum_{(j,i)\in\mathcal{U}}\sum_{\tau=t}^{t+d_i-1}\kappa_{\tau,(j,i)}-\sum_{(j,i')\in\mathcal{U}}\sum_{\tau=t'}^{t'+d_{i'}-1}\kappa_{\tau,(j,i')}+\sum_{(\boldsymbol{\delta},\boldsymbol{\zeta})\in\tilde{\Gamma}_\Delta}\varphi_{(\boldsymbol{\delta},\boldsymbol{\zeta})}(\delta_{i,t}+\delta_{i',t'}).\end{aligned}\tag{58}$$

Finally, observe that the minimum-cost arc can be found by enumeration, in $\mathcal{O}(|\mathcal{E}_i||\mathcal{E}_{i'}|)$ time, where i and i' denote the tasks associated with vessel k sailing into and out of berth b, respectively. The newly generated columns with negative reduced costs will then be added to the BMP, which will be solved again. This procedure terminates when no routes with negative reduced costs can be detected.

4.3.2. Column Generation for the PBSP In the PBSP, we are given a vector $\bar{\boldsymbol{\chi}}$ that represents the solution of the χ variables for the associated BMP. Given $\bar{\boldsymbol{\chi}}$, let $H=\{(i,t)|\sum_{\omega\in\Omega}\bar{\chi}_\omega\alpha_{\omega,i,t}>0,t\in\mathcal{E}_i,i\in\mathcal{I}\}$, and $H_s=\{(i,t)|s\in\mathcal{S}_t,(i,t)\in H\}$, $\forall s\in\mathcal{S}$.

New pilot routes for the PBSP are generated by solving $|\mathcal{S}|$ pricing problems, each corresponding to a shift. We denote the pricing problem for shift $s\in\mathcal{S}$ by SPP_s. With a slight abuse of notation, the SPP_s is defined as a shortest-path problem in a space-time network denoted by $G_s=(N_s,A_s)$, where N_s and A_s represent the set of nodes and the set of arcs, respectively. We let $N_s=H_s\cup J_s\cup\{O,D\}$. Here, J_s represents the set of nodes corresponding to starting a rest period. Because the requirements of the rest periods are identical for all pilots working in the same shift, we define J_s as $J_s=\{(i,t),t\in\mathcal{E}_i,i\in\mathcal{J}_p,\exists p\in\mathcal{P}_s\}$. Besides, O and D denote the dummy source and the dummy sink, respectively. Given N_s, we let $A^1_s=\{[(i,t),(i',t')]|(i,t),(i',t')\in N_s,t+d_i+q_{i,i'}\leq t'\}$, $A^2_s=\{[O,(i',t')]|(i',t')\in N_s\}$, and $A^3_s=\{[(i,t),D]|(i,t)\in N_s\}$, and let $A_s=A^1_s\cup A^2_s\cup A^3_s$. The SPP_s is equivalent to finding the shortest elementary path in G_s that travels from O to D and contains exactly one node from J_s. The cost of sending a unit flow through any arc $[m,m']$ in G_s (denoted by $\hat{c}_{[m,m']}$) is calculated by

$$\hat{c}_{[m,m']}=\begin{cases}-\delta_{i,t}, \text{ if } [m,m']\in A^1_s,[m,m']=[(i,t),(i',t')],(i,t)\in H_s,\\ c^3_s, \quad \text{if } [m,m']\in A^2_s,[m,m']=[O,(i',t')],\\ -\delta_{i,t}, \text{ if } [m,m']\in A^3_s,[m,m']=[(i,t),D],(i,t)\in H_s,\\ 0, \quad \text{otherwise.}\end{cases}\tag{59}$$

The network G_s has two special properties that facilitate the development of an efficient solution method. First, since each node is associated with a time step, given a set of nodes in H_s to be visited in a pilot route, the only feasible visiting sequence for these nodes is to travel through them by following their temporal order. Second, the unit flow costs of arcs incident to the nodes in J_s are independent of these nodes. Therefore, while visiting a node in J_s at a given position of a pilot route ensures the feasibility of the route, it does not change the optimal cost for the route. To solve the shortest-path problem, we develop a tailored label correcting algorithm that solves the SPP_s in $\mathcal{O}(|H_s|^2)$ time. The algorithm leverages the special properties of the network. It first identifies partial pilot routes without rest periods using bidirectional label correcting (using the first property) and then constructs feasible pilot routes by incorporating rest periods into these partial routes without changing their costs (using the second property). The algorithm is explained in EC.2 of the electronic companion. The newly generated columns with negative reduced costs will be added to the PBSP, which will be solved again. This procedure continues until no routes with negative reduced costs can be found.

4.4. Branching Scheme

The optimal solution to the BMP at a node in the tree may have fractional flows in vessel or pilot routes. In this case, the node is branched into two child nodes so that (i) the current fractional solution is infeasible for the BMP at both child nodes and (ii) the overall optimal integer solution for the two child nodes remains the same as that for the BMP at the current node. The detailed branching scheme is explained as follows.

We let $\boldsymbol{\chi}^*$ and $\boldsymbol{\mu}^*$ denote the optimal solutions to the BMP and the associated PBSP at a node, respectively. To facilitate branching decisions, we make use of the following additional variable vectors: $\boldsymbol{\Lambda} = (\Lambda_{k,b} | b \in \mathcal{B}_k, k \in \mathcal{K})$, $\boldsymbol{\Xi} = (\Xi_{i,t} | t \in \mathcal{E}_i, i \in \mathcal{I})$, $\boldsymbol{\Pi} = (\Pi_{i,s} | s \in \mathcal{S}, i \in \mathcal{I})$, and $\boldsymbol{\Psi} = (\Psi_{i,j} | i, j \in \mathcal{I} \cup \{0\} \cup \mathcal{J}, i \neq j)$. Here, we use 0 to denote the dummy source or sink of a pilot route. The values of these variables are calculated by:

$$\Lambda_{k,b} = \sum_{\omega \in \Omega} \chi^*_\omega \sum_{t \in \mathcal{T}} \sum_{t' \in \mathcal{T}} \beta_{\omega,b,t,t'} \ \forall b \in \mathcal{B}_k, k \in \mathcal{K} \tag{60}$$

$$\Xi_{i,t} = \sum_{\omega \in \Omega} \chi^*_\omega \alpha_{\omega,i,t} \ \forall t \in \mathcal{E}_i, i \in \mathcal{I} \tag{61}$$

$$\Pi_{i,s} = \sum_{\phi \in \Phi_s} \mu^*_\phi \sum_{t \in \mathcal{T}_s \cap \mathcal{E}_i} \gamma_{\phi,i,t} \ \forall s \in \mathcal{S}, i \in \mathcal{I} \tag{62}$$

$$\Psi_{i,j} = \sum_{\phi \in \Phi} \mu^*_\phi \sigma_{\phi,i,j} \ \forall i, j \in \mathcal{I} \cup \{0\} \cup \mathcal{J}, i \neq j, \tag{63}$$

where $\sigma_{\phi,i,j}$ is a binary parameter which equals 1 if and only if route ϕ travels directly from activity i (or the dummy source) to activity j (or the dummy sink). Note that given $k \in \mathcal{K}$ and $b \in \mathcal{B}_k$,

$\Lambda_{k,b}$ indicates whether vessel k is handled at berth b; given $i \in \mathcal{I}$ and $t \in \mathcal{E}_i$, $\Xi_{i,t}$ indicates whether task i starts at time step t; given $i \in \mathcal{I}$ and $s \in \mathcal{S}$, $\Pi_{i,s}$ indicates whether task i starts in shift s; finally, given $i, j \in \mathcal{I} \cup \{0\} \cup \mathcal{J}$ and $i \neq j$, $\Psi_{i,j}$ indicates whether a pilot starts activity j (or visits the dummy sink) immediately after activity i (or the dummy source). One can easily verify that a solution to the VSPP is integral if and only if all variables in $\mathbf{\Lambda}$, $\mathbf{\Xi}$, $\mathbf{\Pi}$, and $\mathbf{\Psi}$ are integers.

Let $\Lambda_{\bar{k},\bar{b}}$, $\Xi_{\bar{i},\bar{t}}$, $\Pi_{\bar{i},\bar{s}}$, and $\Psi_{\bar{i},\bar{j}}$ denote the variables that are closest to 0.5 among the variables in vectors $\mathbf{\Lambda}$, $\mathbf{\Xi}$, $\mathbf{\Pi}$, and $\mathbf{\Psi}$, respectively. For making the branching decisions, we check whether there are fractional variables in these vectors following the order: (i) $\mathbf{\Lambda}$, (ii) $\mathbf{\Xi}$, (iii) $\mathbf{\Pi}$, and (iv) $\mathbf{\Psi}$. Once a fractional variable is found in $\mathbf{\Lambda}$, (or $\mathbf{\Xi}$, $\mathbf{\Pi}$, and $\mathbf{\Psi}$), we separate the current node by generating one child node with the additional constraint $\Lambda_{\bar{k},\bar{b}} = 0$, (or $\Xi_{\bar{i},\bar{t}} = 0$, $\Pi_{\bar{i},\bar{s}} = 0$, and $\Psi_{\bar{i},\bar{j}} = 0$) imposed on the associated BMP or PBSP and one child node with the additional constraint $\Lambda_{\bar{k},\bar{b}} = 1$, (or $\Xi_{\bar{i},\bar{t}} = 1$, $\Pi_{\bar{i},\bar{s}} = 1$, and $\Psi_{\bar{i},\bar{j}} = 1$) imposed on the associated BMP or PBSP. To preserve the structures of the BMP and the PBSP, all branching constraints are imposed on the corresponding pricing problems for generating columns.

Such a branching scheme, when augmented by the warm start strategy proposed in Section 4.5.4, enables Benders cuts generated at one node to be utilized by more of the nodes explored later in the branch-and-bound tree.

In the BPBC approach, we adopt a best-bound strategy for selecting the nodes to solve. In particular, among all unsolved and unfathomed nodes in the branch-and-bound tree, we solve the one that has the minimum objective function value of the associated BMP.

THEOREM 1. *The BPBC approach can generate an optimal solution for model M2.*

4.5. Computational Enhancements

We now describe the computational enhancements that improve the efficiency of the BPBC approach.

4.5.1. Dynamic Constraint Generation The efficiency of the BPBC approach largely depends on how fast the BMP can be solved by the C&BCG algorithm. We observed that the difficulty for solving the BMP mainly lies in the large number of constraints for berth allocation and vessel traffic management, i.e., constraints (39)–(41). However, we also observed that most of these constraints are inactive in an optimal solution to the BMP, and therefore it is unnecessary to enumerate all of them for solving the BMP.

Based on this observation, we design a dynamic constraint generation method to accelerate the C&BCG algorithm. In particular, only a subset of constraints (39)–(41) are included in the BMP at the beginning of the C&BCG algorithm. Each time the BMP is solved to optimality, a constraint

separation procedure is run to check whether some constraints from (39)–(41) are violated by the incumbent solution. Once identified, such constraints are added into the BMP, which is then solved again. This separation procedure repeats until no violations can be found.

To efficiently generate valid constraints from (39)–(41), the separation procedure leverages some special properties of the problem to avoid some cumbersome enumerations. Details of the revised C&BCG algorithm and the separation procedure are explained in EC.3 of the electronic companion.

4.5.2. Lower-Bound Lifting Inequalities Because parts of the objective function (37) are projected out in the Benders reformulation, the optimality gap of the BMP may be large in the initial stages of the algorithm due to the low quality of the lower bound. A large number of Benders cuts are thus needed to close the gap (Adulyasak et al. 2015). To address this issue, Adulyasak et al. (2015) proposed a method to lift the lower bound of the Benders master problem by using inequalities that contain some information about the parts of the original objective function that were removed. Following this idea, we lift the lower bound of the BMP by using initial cuts, called the lower-bound lifting (LBL) cuts. In particular, we use the LBL cuts to estimate a lower bound of the cost in pilot dispatching and lift the lower bound of the BMP by requiring the variable η to be no smaller than the lower bound of the pilot dispatching cost.

In pilot scheduling, a minimum setup time ($q_{i,j}$) is required when a pilot is repositioned from task i to task j. In the LBL inequalities, we relax the minimum setup time requirement, by setting an identical minimum setup time between task i and any other task (denoted by $\underline{q}_i$) to be $\underline{q}_i = \min_{j\in\mathcal{I}:j\neq i}\{q_{i,j}\}$. Let $\nu_{i,t,s}$ be a binary variable which is equal to 1 if and only if task i starts at time t and is performed by a pilot in shift s. The following LBL cuts are valid for the BMP:

$$\eta \geq \sum_{s\in\mathcal{S}} c_s^3 n_s \tag{64}$$

$$n_s \geq \sum_{i\in\mathcal{I}} \sum_{t'=\max\{\underline{T}_s, t-d_i-\underline{q}_i+1\}}^{t} \nu_{i,t',s} \;\; \forall t\in\mathcal{T}_s, s\in\mathcal{S} \tag{65}$$

$$\sum_{s\in\mathcal{S}_t} \nu_{i,t,s} = \sum_{\omega\in\Omega} \chi_\omega \alpha_{\omega,i,t} \;\; \forall t\in\mathcal{E}_i, i\in\mathcal{I} \tag{66}$$

$$\nu_{i,t,s} \leq 1 \;\; \forall i\in\mathcal{I}, t\in\mathcal{T}, s\in\mathcal{S} \tag{67}$$

$$\nu_{i,t,s} \geq 0 \;\; \forall i\in\mathcal{I}, t\in\mathcal{T}, s\in\mathcal{S} \tag{68}$$

$$n_s \geq 0 \;\; \forall s\in\mathcal{S}. \tag{69}$$

Constraint (64) provides a lower bound for the pilot dispatching cost. Constraints (65) calculate the number of pilots required to work in each shift. Constraints (66) synchronize the scheduling of tasks and the assignment of pilots. The domains of the $\nu_{i,t,s}$ and n_s variables are defined in the last three sets of constraints (they are relaxed to be continuous variables).

We can lift the LBL inequalities by tightening constraints (65) to be

$$n_s \geq \sum_{i\in\mathcal{I}} \sum_{t'=\max\{\underline{T}_s, t-d_i-\underline{q}'_{i,t,s}+1\}}^{t} \nu_{i,t',s} \quad \forall t\in\mathcal{T}_s, s\in\mathcal{S}. \tag{70}$$

For each $i\in\mathcal{I}, t\in\mathcal{T}_s, s\in\mathcal{S}$, $\underline{q}'_{i,t,s} \geq \underline{q}_i$ and is set as follows:

$$\underline{q}'_{i,t,s} = \begin{cases} \overline{T}, & \text{if } \bar{\mathcal{I}}_{i,t,s} = \varnothing, \\ \min\limits_{j\in\bar{\mathcal{I}}_{i,t,s}} \min\limits_{t'\in\mathbb{Z}_{[t+d_i+q_{i,j},\overline{T}_s]}\cap\mathcal{E}_j} t'-d_i, & \text{if } \bar{\mathcal{I}}_{i,t,s} \neq \varnothing, \end{cases}$$

where $\bar{\mathcal{I}}_{i,t,s} = \{j | j\in\mathcal{I}, \mathbb{Z}_{[t+d_i+q_{i,j},\overline{T}_s]}\cap\mathcal{E}_j \neq \varnothing, j\neq i\}$. Observe that $\bar{\mathcal{I}}_{i,t,s}$ is the set of tasks that can be performed by a pilot working in shift s after completing task i which starts at time step t in this shift.

Proposition 2. *The LBL inequalities* (64) *and* (66)–(70) *are valid for the BMP.*

Finally, we note that after adding these inequalities, the BMP has been changed, not only with more decision variables (columns) but also with more constraints. As a sequence, the pricing problems and their solution algorithms should also be modified to incorporate the changes. Consider the pricing problem for vessel k and berth b (denoted by $\text{MPP}'_{k,b}$). The $\text{MPP}'_{k,b}$ can be defined as a problem that identifies the arc with the minimum cost in a three-dimensional task-time-shift network. We present the details of the pricing problems for the BMP with the LBL cuts in EC.4 of the electronic companion.

To solve the $\text{MPP}'_{k,b}$, we extend the enumeration method for the original pricing problem $\text{MPP}_{k,b}$ by not only enumerating all feasible pairs of start times for the vessel to sail into and out of the berth but also enumerating all feasible pilot assignment patterns (in terms of working shifts) for the associated pilotage tasks. Observe that this enumeration method for solving the $\text{MPP}'_{k,b}$ still has polynomial time complexity and solves the problem in $\mathcal{O}(|\mathcal{E}_i||\mathcal{E}_{i'}||\mathcal{S}|^2)$ time, where i and i' denote the tasks corresponding to vessel k sailing into and out of berth b, respectively.

4.5.3. Variable Fixing In each iteration of the column generation procedure for solving the BMP, one has to solve $\sum_{k\in\mathcal{K}}|\mathcal{B}_k|$ pricing problems. In this section, we show that the column generation procedure can be accelerated by fixing the value of χ_ω at zero if χ_ω cannot take value 1 in an optimal integer solution for M2.

When solving the BMP at a node in the branch-and-bound tree, at each stage of column generation, we obtain (i) the optimal objective function value denoted by Υ, and (ii) the minimum reduced cost obtained by solving pricing problem $\text{MPP}_{k,b}$ (or $\text{MPP}'_{k,b}$) which is denoted by $r_{k,b}$, where $k\in\mathcal{K}, b\in\mathcal{B}_k$. Then we have:

LEMMA 1. *$\Upsilon + \sum_{k\in\mathcal{K}} \sum_{b\in\mathcal{B}_k: r_{k,b}<0} r_{k,b}$ is a lower bound for the BMP at the current node.*

For notational simplicity, let $LB = \Upsilon + \sum_{k\in\mathcal{K}} \sum_{b\in\mathcal{B}_k: r_{k,b}<0} r_{k,b}$, and let $\Omega_{k,b}$ denote the set of columns corresponding to vessel k being handled at berth b, where $b \in \mathcal{B}_k$, $k \in \mathcal{K}$. We further have:

PROPOSITION 3. *Given a pair of b and k such that $b \in \mathcal{B}_k, k \in \mathcal{K}$ and $r_{k,b} \geq 0$, $LB + r_{k,b}$ gives a lower bound for the BMP with the constraint $\sum_{\omega\in\Omega_{k,b}} \chi_\omega = 1$ at the current node and any child nodes generated from this node.*

Suppose that UB is the best (integer) upper bound for model M2. Then, consider a pair of b and k that satisfies $LB + r_{k,b} \geq UB$. In this case, in any optimal solution for the BMP with the constraint $\sum_{\omega\in\Omega_{k,b}} \chi_\omega = 1$ at the current node, the resulting objective function value is no smaller than the upper bound. Due to constraints (38), in an integer solution for the BMP at any node, the value of $\sum_{\omega\in\Omega_{k,b}} \chi_\omega$ either takes 1 or 0. Hence, in any integer solution for the BMP with the objective function value smaller than UB at the current node, one must have $\sum_{\omega\in\Omega_{k,b}} \chi_\omega = 0$. To accelerate the column generation procedure, we "fix" $\chi_\omega = 0$, $\forall \omega \in \Omega_{k,b}$ by not solving the pricing problem $\text{MPP}_{k,b}$ (or $\text{MPP}'_{k,b}$) at the current node and any child nodes generated from this node.

4.5.4. Warm Start for the BMP We warm start the C&BCG algorithm for solving the BMP at a node by using a set of initial Benders cuts in the first iteration of the algorithm and by using a set of initial columns in each subsequent iteration.

Let Θ_ι denote the set of branching constraints regarding the variables in $\mathbf{\Pi}$ and $\mathbf{\Psi}$, as defined in Section 4.4, imposed on the PBSP at a node denoted by ι. The method for initializing Benders cuts for the BMP is based on Proposition 4 and Corollary 1.

PROPOSITION 4. *For two nodes ι and ι', if we have $\Theta_{\iota'} \subseteq \Theta_\iota$, then the Benders cuts that are valid for the BMP at node ι' are also valid for the BMP at node ι.*

COROLLARY 1. *The Benders cuts that are valid for the BMP at the root node are also valid for the BMP at any other node.*

When solving the BMP at node ι, we let the initial set of Benders cuts (denoted by Γ^0_Δ in the C&BCG algorithm) include all Benders cuts that are generated for solving the BMP at any node ι' such that $\Theta_{\iota'} \subseteq \Theta_\iota$ (including the root node).

Moreover, in each iteration of the C&BCG algorithm (except for the first iteration), we also initialize a set of columns for the BMP. In particular, the set of initial columns, denoted by Ω_0, is set as $\Omega_0 = \{\omega | \chi^*_\omega > 0, \omega \in \tilde{\Omega}\}$, where $\tilde{\Omega}$ and χ^*_ω denote the set of columns (vessel routes) generated for solving the BMP in the previous iteration in the algorithm and their optimal solutions, respectively.

4.5.5. Primal Heuristics A high-quality upper bound for M2 helps prune branches in the branch-and-bound tree in the early stages. It can also accelerate the column generation procedure for solving the BMP through variable fixing. To quickly identify high-quality upper bounds, the approach leverages two methods to generate feasible integer solutions for M2.

The first method detects integer solutions within the C&BCG algorithm for solving the BMP. In particular, in any iteration of the algorithm, let $(\boldsymbol{\chi}^*, \eta^*)$ and Z_1^* be the optimal solution and the optimal objective function value of the BMP, respectively, and let $\boldsymbol{\mu}^*$ and Z_2^* be the optimal solution and the optimal objective function value of the associated PBSP, respectively. If all values in $\boldsymbol{\chi}^*$ and $\boldsymbol{\mu}^*$ are integral, then we obtain a feasible integer solution $(\boldsymbol{\chi}^*, \boldsymbol{\mu}^*)$ to M2 and a corresponding upper bound equal to $Z_1^* - \eta^* + Z_2^*$.

The second method is used each time the C&BCG algorithm completes solving the BMP and delivers a fractional solution for it. In this case, we run a heuristic that tries to construct a feasible integer solution for M2 based on the current fractional solution. Details of the heuristic are explained in EC.5 of the electronic companion.

5. Computational Experiments

We have performed extensive computational experiments to confirm the applicability and effectiveness of our model and algorithm. In this section, we first introduce the experimental settings in Section 5.1 and the procedures for generating instances in Section 5.2. We then present the computational results, which consist of four parts. In the first part, we examine the impacts of the acceleration techniques on the performance of the BPBC approach. In the second part, we compare the performance of the approach with that of a standard MILP solver applied to models M1 and SM1 and that of the three-phase Benders decomposition and column generation method proposed by Cordeau et al. (2001). In the third part, we evaluate the robustness of the BPBC approach against variations in problem settings. In the last part, we analyze the value of the integration by comparing the results delivered by our approach with those obtained by a method that solves the BAP and the PPP sequentially. Interested readers can find our code implementation, data sets used, and associated user instructions as part of the electronic companion to this paper.

5.1. Experimental Settings

We performed the experiments on an Intel Core i7 2.20 GHz PC with 32 GB RAM. All algorithms were coded in C++ calling CPLEX 12.6.

In each iteration of the column generation procedure for the BMP or the PBSP, the approach solves a set of independent pricing problems. Although all of the pricing problems for the BMP and the PBSP have polynomial time complexity, their large number still hinders the solution

process. To alleviate this computational burden, our approach solves multiple pricing problems simultaneously by using parallel computing. In the experiments, the pricing problems for the BMP and the PBSP were solved in a six-thread environment (i.e., at most six pricing problems are solved simultaneously). Meanwhile, we also let CPLEX run on six threads for solving the MILP models M1 and SM1. Finally, the time limit for any approach to solve any instance was set to 3,600 seconds.

5.2. Instance Generation

We created a set of instances for the experiments based on real operational data from the Hong Kong Container Port, which is the eighth largest container port in the world (The Marine Department of Hong Kong 2020a).

Let L, B, and V be the length of the planning horizon (days), the number of berths, and the number of vessel arrivals in an instance, respectively. In all instances, we let $L \in \{1, 2\}$ and $B \in \{10, 15\}$. Then, given each combination of L and B, we set $V = R \cdot L \cdot B$, where R is the vessel arrival rate (vessel arrivals per berth per day) and $R \in \{1.0, 1.2, 1.4\}$. Hence, there are 12 combinations of L, B, and V. For each combination, we created five random instances, yielding 60 instances in total. In all instances, the unit time is set to 10 minutes; i.e., a time step lasts 10 minutes, and a day contains 144 time steps. A day is divided into three shifts, covering 0:00-8:00, 8:00-16:00, and 16:00-24:00. The planning horizon in an instance starts from 0:00 on a given day. Details of the operational data of the Hong Kong Container Port and the settings of other parameters in the instances are explained in EC.6 of the electronic companion.

5.3. Impacts of the Computational Enhancements

We proposed five acceleration strategies in Section 4.5 to improve the performance of the BPBC approach. The impacts of these strategies on the performance of the BPBC approach are analyzed in the following two subsections. In the first subsection, we investigate the joint impact of all strategies and the impact of dynamic constraint generation, which is shown to be the most powerful enhancement. Then, in the second subsection, we focus on the improvements brought by other strategies.

5.3.1. Joint Impact of Enhancements and Impact of Dynamic Constraint Generation To examine the joint impact of all acceleration strategies and the impact of dynamic constraint generation, we solved 30 instances with a planning horizon of one day using four different approaches, which were (i) the BPBC without any acceleration strategy, (ii) the BPBC with dynamic constraint generation but without any other acceleration strategy, (iii) the BPBC without

dynamic constraint generation but with all the other acceleration strategies, and (iv) the BPBC approach (with all acceleration strategies).

Table 1 reports the performance of the four approaches. The first column shows the settings of (L, B, V) for a group of five instances. Columns *No Enhancements*, *DCG*, *No DCG*, and *BPBC* are associated with the first, second, third, and fourth approaches, respectively. For each instance group, we report the number of feasible solutions, the number of optimal solutions, the average optimality gap (in percentage terms), as well as the average computational time (in CPU seconds), obtained by each of the approaches, in columns *Fsl*, *Opt*, *Gap* and *CPU*, respectively.

Table 1 Computational Results of the BPBC Approach with Different Acceleration Strategies (Part I).

	No Enhancements				DCG				No DCG				BPBC			
(L, B, V)	Fsl	Opt	Gap	CPU	Fsl	Opt	Gap	CPU	Fsl	Opt	Gap	CPU	Fsl	Opt	Gap	CPU
$(1, 10, 10)$	5	5	0.0	814.2	5	5	0.0	75.9	5	5	0.0	62.6	5	5	0.0	5.0
$(1, 10, 12)$	5	4	0.0	1399.5	5	5	0.0	113.1	5	5	0.0	53.1	5	5	0.0	2.9
$(1, 10, 14)$	2	0	60.0	3600.0	5	5	0.0	363.7	5	5	0.0	369.2	5	5	0.0	14.9
$(1, 15, 15)$	2	0	63.0	3600.0	4	3	20.0	1625.7	5	5	0.0	739.2	5	5	0.0	49.7
$(1, 15, 18)$	1	1	80.0	3301.5	4	3	20.0	1760.1	5	5	0.0	519.1	5	5	0.0	17.4
$(1, 15, 21)$	0	0	100.0	3600.0	2	2	60.0	3058.6	5	2	0.5	3037.7	5	5	0.0	168.9
Total	15	10	50.5	2719.5	25	23	16.7	1166.2	30	27	0.1	796.8	30	30	0.0	43.1

Note. We take the optimality gap as 100% if a method failed to obtain a feasible solution for an instance.

Let us first compare the performance of the first and fourth approaches (i.e., the BPBC approaches with no and all acceleration strategies). The BPBC approach with all acceleration strategies found optimal solutions for all the 30 instances, while the approach not using these strategies obtained feasible and optimal solutions for only 15 and 10 instances, respectively. Both approaches solved all instances in the first group to optimality, but the BPBC approach with all acceleration strategies was 161 times faster than the approach without them for solving these instances. Even bounded by an upper limit, the computational times used by the BPBC approach with all acceleration strategies were on average merely 1/63 of those used by the approach without these strategies. These results indicate that the proposed acceleration strategies can significantly improve the performance of the BPBC approach such that more instances can be solved and solution times are reduced by one to two orders of magnitude after using the strategies.

By using dynamic constraint generation, we allow the BPBC to generate constraints on the fly when solving the BMP. This strategy helps generate smaller LPs that can be solved more efficiently. The differences in the performance of the third and fourth approaches clearly demonstrate the impact of dynamic constraint generation. In particular, the strategy enables more instances to be solved to optimality and reduces the solution times by more than 90%.

On top of dynamic constraint generation, other acceleration strategies are also critical for improving the performance of the BPBC approach. As demonstrated in columns *DCG* and *BPBC* in Table 1, compared with the BPBC with all acceleration strategies, the approach with only dynamic constraint generation took 26 times more time on average to solve the instances and it could not provide optimal solutions for seven instances and failed to find feasible solutions for five instances.

5.3.2. Impacts of Other Enhancements In this subsection, we investigate the impacts of other acceleration strategies. To this end, we solved 60 instances from 12 groups using the BPBC approach under different combinations of these acceleration strategies.

Table 2 reports the performance of the BPBC approach on the 60 instances when each of these acceleration strategies was not applied and when all of them were used. The first column shows the settings of (L, B, V) for a group of five instances. Column *BPBC* indicates the BPBC with all strategies, and other columns show the results when different strategies were not used. The abbreviations LBL, VF, WS, and PH correspond to the lower-bound lifting inequalities, variable fixing, warm start for the BMP, and primal heuristics, respectively. For each group of instances, we report the number of instances for which feasible solutions were detected, the number of instances solved to optimality, the average optimality gap (in percentage terms), and the average computational time (in CPU seconds) for each solution method in the corresponding columns *Fsl*, *Opt*, *Gap*, and *CPU*, respectively.

Table 2 Computational Results of the BPBC Approach with Different Acceleration Strategies (Part II).

	No LBL				No VF				No WS				No PH				BPBC			
(L, B, V)	Fsl	Opt	Gap	CPU	Fsl	Opt	Gap	CPU	Fsl	Opt	Gap	CPU	Fsl	Opt	Gap	CPU	Fsl	Opt	Gap	CPU
$(1, 10, 10)$	5	5	0.0	11.1	5	5	0.0	5.5	5	5	0.0	15.6	5	5	0.0	9.7	5	5	0.0	5.0
$(1, 10, 12)$	5	5	0.0	17.4	5	5	0.0	3.6	5	5	0.0	5.9	5	5	0.0	3.9	5	5	0.0	2.9
$(1, 10, 14)$	5	5	0.0	44.0	5	5	0.0	16.7	5	5	0.0	44.4	5	5	0.0	20.0	5	5	0.0	14.9
$(1, 15, 15)$	5	5	0.0	115.3	5	5	0.0	82.9	5	5	0.0	208.4	5	5	0.0	86.1	5	5	0.0	49.7
$(1, 15, 18)$	5	5	0.0	118.4	5	5	0.0	26.7	5	5	0.0	65.4	5	5	0.0	36.9	5	5	0.0	17.4
$(1, 15, 21)$	5	5	0.0	321.9	5	5	0.0	195.3	5	4	0.0	1524.5	5	5	0.0	231.2	5	5	0.0	168.9
$(2, 10, 20)$	5	5	0.0	184.8	5	5	0.0	222.9	5	5	0.0	494.9	5	5	0.0	458.5	5	5	0.0	115.1
$(2, 10, 24)$	5	5	0.0	274.6	5	5	0.0	146.8	5	5	0.0	689.9	5	5	0.0	207.7	5	5	0.0	108.4
$(2, 10, 28)$	5	3	0.1	1913.9	5	3	0.1	1578.8	5	3	0.2	2167.5	3	3	40.0	1844.7	5	3	0.1	1603.7
$(2, 15, 30)$	5	4	0.0	1373.2	5	5	0.0	1062.0	5	3	0.2	1950.7	4	4	20.0	1539.4	5	5	0.0	879.6
$(2, 15, 36)$	5	4	0.0	1889.9	5	4	0.0	1375.5	5	2	0.5	2455.3	4	4	20.0	1386.3	5	4	0.0	1150.1
$(2, 15, 42)$	5	1	1.2	3199.8	5	2	0.7	2407.6	4	0	21.2	3600.0	2	2	60.0	2428.0	5	2	0.8	2349.1
Total	60	52	0.1	788.7	60	54	0.1	593.7	59	47	1.8	1101.9	53	53	11.7	687.7	60	54	0.1	538.7

Note 1. We take the optimality gap as 100% if a method failed to obtain a feasible solution for an instance.

Note 2. Abbreviations: LBL: lower-bound lifting inequalities; VF: variable fixing; WS: warm start for the BMP; PH: primal heuristics.

As shown in Table 2, the BPBC approach obtained optimal or near-optimal solutions for all instances within the time limit. This indicates that the approach can well solve instances on real-world scale.

The results confirm that the acceleration strategies can significantly improve the performance of the BPBC approach. First, by lifting the lower bounds of the BMP, the LBL cuts accelerate the convergence of the Benders decomposition for solving the BMP and reduce the solution times. Such cuts are tighter for instances with smaller sizes. In particular, for instance groups with $L = 1$, the usage of LBL reduced the average computational times by 48% to 85%. Second, VF saves computational times by reducing the number of pricing problems to be solved in column generation. The results in Table 2 indicate that VF is most powerful for instances with medium sizes (i.e., those in the fourth to eighth instance groups). This is because compared with instances with larger sizes, high-quality upper bounds for these instances are more likely to be found in early stages. Meanwhile, compared with instances with smaller sizes, these instances require more iterations of column generation before being solved to optimality. By using WS, the BPBC approach solves the BMP with a set of columns for initializing column generation in each iteration of Benders decomposition and can start the Benders decomposition with an initial set of Benders cuts. This strategy reduces the number of iterations for column generation and also reduces the number of iterations in the Benders decomposition for solving the BMP. Thanks to this strategy, the average solution times were decreased by more than 60% in most instance groups. Finally, the use of primal heuristics enabled the BPBC approach to generate feasible solutions for all instances, which results in a significant decrease in the average optimality gap. High-quality upper bounds also help prune nodes in the branch-and-bound tree in early stages, which led to decreases in the solution times.

The computational results in Section 5.3 indicate that the acceleration strategies are critical to the performance of the BPBC approach. Moreover, these strategies can also provide insights for designing solution approaches based on Benders decomposition and/or column generation for other problems.

5.4. Comparisons with Existing Methods

We next compare the performance of the BPBC approach with two existing methods, including a commonly used optimization solver (CPLEX) solving the MILP models and the three-phase Benders decomposition and column generation method proposed by Cordeau et al. (2001) for solving synchronized scheduling problems.

5.4.1. Comparisons with CPLEX In this section, we compare the performance of the BPBC approach with methods that use optimization solvers to solve MILP models M1 (the original formulation) and SM1 (M1 with valid inequalities [31]–[36]). We adopted CPLEX 12.6 (using six threads) as the optimization solver.

For the comparisons, we used 15 instances from three groups with $L = 1$, $B = 10$, and $V \in \{10, 12, 14\}$. Table 3 summarizes the results produced by CPLEX on M1 and SM1 and the BPBC approach. For each method, we show the number of instances for which feasible solutions were found in column *Fsl* and the number of instances solved to optimality in column *Opt*. Columns *Gap* and *CPU* report the average optimality gap in percentage terms and the average CPU time in seconds delivered by each method for solving the instances in each group, respectively.

Table 3 Computational Results of CPLEX and the BPBC Approach.

	M1				SM1				BPBC			
(L, B, V)	Fsl	Opt	Gap	CPU	Fsl	Opt	Gap	CPU	Fsl	Opt	Gap	CPU
$(1, 10, 10)$	0	0	100.0	3600.0	5	4	0.5	1890.3	5	5	0.0	5.0
$(1, 10, 12)$	0	0	100.0	3600.0	4	0	26.0	3600.0	5	5	0.0	2.9
$(1, 10, 14)$	0	0	100.0	3600.0	3	0	56.0	3600.0	5	5	0.0	14.9
Total	0	0	100.0	3600.0	12	4	42.2	3030.1	15	15	0.0	7.6

Note. We take the optimality gap as 100% if a method failed to obtain a feasible solution for an instance.

For the 15 instances, CPLEX on M1 failed to find a feasible solution for any instance within the time limit. In comparison, CPLEX on SM1 found feasible solutions for 12 instances, and four were solved to optimality within the time limit. These confirm that the inequalities (31)–(36) can significantly improve the performance of an MILP solver for solving the instances. CPLEX on SM1 can well solve instances with 10 berths and 10 vessel arrivals a day, which are comparable to the sizes of the practical VSPP faced by small and medium seaports. For example, the Port of Freeport in the U.S. has seven berths and serves three vessel calls per day on average (Texas Department of Transportation 2020), the SGICT Terminal in China (one of the largest container terminals in Shanghai, China) operates seven berths and handles 12 to 14 vessels (including barges) a day (Ding et al. 2016), and the Port of Felixstowe in the U.K. (one of the largest container ports in the country) operates nine berths (The Port of Felixstowe 2021).

Moreover, as shown in Table 3, the BPBC approach solved all 15 instances to optimality, and the average computational time was 7.6 seconds. Hence, the BPBC approach significantly outperforms CPLEX for solving the VSPP instances. Therefore, while one can rely on a general-purpose solver to generate a feasible solution to the VSPP on small or medium instances, for instances with large sizes, a tailored solution approach (e.g., the BPBC approach) must be used.

5.4.2. Comparisons with the Approach of Cordeau et al. (2001) In this section, we compare the performance of the BPBC approach with that of the three-phase Benders decomposition and column generation method proposed by Cordeau et al. (2001) (which is shortened as TPBC). To adjust the TPBC for the VSPP, we incorporated the C&BCG algorithm in Algorithm 1 into the same algorithmic framework used by Cordeau et al. (2001).

For the comparisons, we used 30 instances from six groups with $L = 1$. Table 4 summarizes the results produced by the TPBC method and the BPBC approach for solving these instances. For each instance group, we report the numbers of instances for which feasible solutions were found and the numbers of instances solved to optimality in columns *Fsl* and *Opt*, respectively. Columns *Gap* and *CPU* report the average optimality gaps and the average computational times, respectively.

Table 4 Computational Results of the TPBC Method and the BPBC Approach.

	TPBC				BPBC			
(L, B, V)	Fsl	Opt	Gap	CPU	Fsl	Opt	Gap	CPU
$(1, 10, 10)$	5	3	0.3	178.3	5	5	0.0	5.0
$(1, 10, 12)$	5	2	0.2	334.8	5	5	0.0	2.9
$(1, 10, 14)$	5	1	0.8	1303.8	5	5	0.0	14.9
$(1, 15, 15)$	5	0	0.7	2091.2	5	5	0.0	49.7
$(1, 15, 18)$	1	1	80.0	3272.2	5	5	0.0	17.4
$(1, 15, 21)$	0	0	100.0	3600.0	5	5	0.0	168.9
Total	21	7	30.3	1796.7	30	30	0.0	43.1

Note. We take the optimality gap as 100% if a method failed to obtain a feasible solution for an instance.

As shown in Table 4, the BPBC approach delivered optimal solutions for all 30 instances, while the TPBC method found feasible and optimal solutions for only 21 and seven instances, respectively. Note that the TPBC method cannot guarantee the optimality of a solution, even when the method completes the solution procedure within the time limit (3,600 seconds). In terms of solution speed, the BPBC approach was 40 times faster than the TPBC method on average. Therefore, the BPBC approach outperforms the TPBC method in that it can provide proven optimal solutions for the VSPP instances within much less computational times. Because the key differences between the two methods lie in the branching scheme and the application of the acceleration strategies, these results also attest their value for improving the performance of the BPBC approach.

5.5. Sensitivity Analyses

In this section, we evaluate the performance of the BPBC approach and the changes in solution structures when the problem settings are changed. In particular, we consider the changes in shift arrangements and cost parameters, which are the most changeable settings in the VSPP. Port operators may choose to use more flexible shift arrangements to achieve lower labor costs. They

may also adjust the cost parameters to adapt to the changes in the levels of relative importance among different objective indicators (i.e., delays in cargo dispatch, delays in vessel turnaround, berth handling costs, and pilot dispatching costs).

Our experiments were based on the 15 instances with $L = 1$ and $B = 10$, and we changed the settings of shift arrangements and cost parameters in these instances to create new ones. In our experiments, we considered four shift arrangements in which shifts still last eight hours but the intervals between the start times of two consecutive shifts were set as one, two, four, and eight hours, respectively. Note that a day contains 24, 12, six, and three shifts when the intervals between the start times of two consecutive shifts are one, two, four, and eight hours, respectively, and that shifts can overlap in the first three settings. In all instances, the planning horizons start at 0:00 on a given day, and the first shifts start at the same time.

The tests also used different settings of cost parameters. In particular, recall that in the model, we use $c^1_{i,t}$ with $i \in \mathcal{I}^{in}$ and $t \in \mathcal{T}$ to denote the penalty costs associated with the delays in cargo dispatch, $c^1_{i,t}$ with $i \in \mathcal{I}^{out}$ and $t \in \mathcal{T}$ to denote the penalty costs associated with the delays in vessel turnaround, $c^2_{b,k}$ with $b \in \mathcal{B}$ and $k \in \mathcal{K}$ to denote the berth handling costs, and c^3_s with $s \in \mathcal{S}$ to denote the pilot dispatching costs. In the experiments, the cargo dispatch delay costs, the vessel turnaround delay costs, and the berth handling costs were changed to $\vartheta^{in} c^1_{i,t}$ ($i \in \mathcal{I}^{in}$, $t \in \mathcal{T}$), $\vartheta^{out} c^1_{i,t}$ ($i \in \mathcal{I}^{out}$, $t \in \mathcal{T}$), and $\vartheta^{berth} c^2_{b,k}$ ($b \in \mathcal{B}$, $k \in \mathcal{K}$), respectively. Here ϑ^{in}, ϑ^{out}, and ϑ^{berth} are coefficients selected from $\{0.5, 1.0, 1.5, 2.0\}$. As for the pilot dispatching costs, we considered five values of c^3_s ($s \in \mathcal{S}$), including 48, 60, 72, 84, and 96. For each of the 15 instances, we created 68 variants with different shift and cost settings, leading to a total of 1020 tests. The detailed results of these tests are reported in EC.7.1 of the electronic companion.

The BPBC approach delivered feasible solutions in all tests, and 995 instances were solved to optimality within the time limit. These results demonstrate that the BPBC approach performs robustly against variations in problem settings, indicating that it can be well applied for solving the VSPP under different scenarios. Based on the obtained results, we further analyzed the performance of the approach and investigated the changes in the solution structure (in terms of the number of working pilots and the overall objective function value) under different problem settings. The corresponding results are presented and discussed in EC.7.1 of the electronic companion.

5.6. Benefits of Integration

The last part of our experiments focuses on the benefits of solving the BAP and the PPP in the VSPP using an integrated optimization method. To this end, we first developed a method that solves the VSPP by solving the BAP with vessel traffic flow control and the resulting PPP sequentially. We have also adjusted the acceleration strategies to fit them into the sequential

optimization method. As in the sensitivity analyses, we used this sequential optimization method to solve the 15 instances from three groups with $L = 1$, $B = 10$, and $V \in \{10, 12, 14\}$ under 68 different settings of shift arrangements and cost parameters. This led to 1020 tests.

The sequential method efficiently solved all the 1020 cases. In particular, optimal solutions were obtained in all tests. The average solution time was less than 0.1 second and the maximum was 0.3 seconds. Therefore, as "side products" of the BPBC approach, the corresponding components in this approach can be well applied to solve the BAP with vessel traffic flow control and the PPP. Also, as one can see from the computational results reported in EC.7.1 of the electronic companion, solving the BAP and the PPP jointly takes much more time than solving them sequentially. This confirms again that the difficulty for solving the VSPP largely lies in the trade-off between vessel service levels and pilot dispatching costs.

In section EC.7.2 of the electronic companion, we can see that, compared with the sequential optimization method, the solutions generated by the BPBC approach require two to five fewer pilots on average, resulting in a 14% to 39% average reduction in the pilot dispatching costs. Depending on the settings of shifts and costs, the BPBC approach reduces total vessel service costs by an average of 0.8% to 5.5% compared with the sequential optimization method. These results demonstrate that by solving the BAP and the PPP jointly, the BPBC approach generates significant benefits for port authorities compared with a sequential decision method.

In the sequential optimization method, decisions in berth allocation and vessel traffic management are made based on the berth and channel capacities alone, without considering the scheduling of pilots and the overall objective in the VSPP. In contrast, the BPBC approach handles berth allocation, vessel traffic management, and pilot scheduling in an integrated manner, by considering the trade-offs among different objectives. Therefore, on the one hand, the sequential optimization method generates relatively unfavorable pilotage task input from the berth allocation problem for the subsequent pilot scheduling problem, leading to higher pilot dispatching costs and higher overall vessel service costs. On the other hand, by using the integrated optimization approach, seaport operators can work out vessel service plans that achieve the overall best performance.

Moreover, in the VSPP, pilot scheduling has a greater impact on the overall objective function value if the cost for dispatching a pilot becomes relatively higher or more pilots are needed to perform a given set of tasks. This explains why the gaps between the two solution methods get wider when pilot dispatching costs get higher (or other costs get lower) and υ gets larger.

6. Conclusions

In this paper, we have introduced a VSPP in seaports that addresses the BAP and the PPP in combination. We have formulated the problem as a compact MILP model and have proposed

several valid inequalities to strengthen the LP relaxation of the model. This model can be solved efficiently by a general-purpose solver for small-scale instances. To solve large-scale instances, we have proposed a BPBC approach. The approach is further improved by several computational enhancements: dynamic constraint generation, LBL inequalities, variable fixing, warm start for the BMP, and primal heuristics. The computational results show that these enhancements significantly improve the performance of the BPBC approach.

The performance of the BPBC approach has been compared with that of CPLEX on MILP models and that of a commonly used method for solving similar problems. The results attest that the BPBC approach outperforms both methods. We have also tested the performance of the BPBC approach on instances under different scenarios. The results show that the approach performs robustly against variations in problem settings. Finally, we have compared the performance of the BPBC approach and a sequential optimization method, and the results demonstrate that integrating the BAP and the PPP in the VSPP brings significant benefits to port authorities.

Our BPBC approach is proposed for solving the VSPP but it can also provide insights for designing approaches for solving other problems. To begin with, our approach combines Benders decomposition and column generation within a branch-and-bound framework. Such a framework can be applied in algorithms for solving other problems that involve synchronized scheduling.

In addition, the proposed acceleration strategies also set examples for enhancing algorithms that use Benders decomposition and/or column generation. First, the impact of dynamic constraint generation illustrates the value of using cutting-plane techniques in solving an LP when there are a large number of constraints but most of them remain inactive in an optimal solution. Second, the successful application of the LBL inequalities indicates that the convergence of a Benders decomposition algorithm can be accelerated by incorporating partial information from the subproblem into the master problem. Third, our variable fixing strategy demonstrates how to accelerate column generation in a system with multiple pricing problems and verifies that such a strategy works well even when the corresponding master problem in the Dantzig-Wolfe decomposition is solved in a column-constraint(-and-Benders-cut) generation manner. Fourth, the warm start strategy presents a framework of utilizing Benders cuts generated at other nodes of a branch-and-bound tree when the Benders subproblem involves integrality constraints. Considering the possibility of "warm starting" a node can also lead to a more efficient branching scheme. In addition, the strategy also shows how to speed up the column generation incorporated in Benders decomposition by using initial sets of columns. Finally, the primal heuristics used in our approach provide insights regarding how to leverage the partial solutions determined during the solution process to quickly generate high-quality feasible solutions for problems that are solved through Benders decomposition and/or column generation.

The current study can be extended in several ways. First, while we assume deterministic parameters, the arrival and handling times of vessels are uncertain in practice. Thus, a promising extension direction of this research is to design solution methods for the VSPP under uncertainty.

Second, in our problem, we consider the most common seaport layout, i.e., one consisting of an (uncapacitated) anchorage located at the outermost part of the seaport, a set of berths, and one navigation channel that lies between them. In some large seaports, however, there may be another anchorage that lies between the channel and the berths. Such "inner" anchorages provide temporary mooring areas for vessels before entering the terminals, and they generally have limited capacities. Some large seaports also may have multiple navigation channels. Future studies can consider extending the models and solution approach developed in this paper to solve the VSPP in seaports with different layouts.

Third, like most airline crew pairing problems that aim at generating anonymous pairings, the VSPP also creates anonymous activity schedules for pilots in a shift. In practice, pilots in a seaport can be heterogeneous in terms of capabilities of serving vessels of different sizes, labor costs, and shift availabilities. In addition, when assigning individual pilots to work shifts, various laboring regulations must be respected (e.g., the minimum gap between two consecutive shifts assigned to one pilot and personalized requirements regarding fatigue mitigation within a working shift). Hence, a natural extension of the current study is to further consider the scheduling of individual pilots (i.e., rostering of pilots) in the VSPP such that the heterogeneity in pilots is respected.

Finally, along with pilots, tugboats also play an important role in berthing and unberthing vessels in a seaport. In many seaports, tugboats are managed by companies that are independent of port authorities, e.g., the Port of Hong Kong and the Port of Montreal (The Marine Department of Hong Kong 2020b, The Port of Montreal 2020). Nevertheless, it would be interesting to develop methods to further integrate towage services in the VSPP.

Acknowledgments

All authors gratefully acknowledge the valuable comments received from the editors and three anonymous referees on an earlier version of this article.

References

Adulyasak Y, Cordeau JF, Jans R (2014) Formulations and branch-and-cut algorithms for multivehicle production and inventory routing problems. *INFORMS Journal on Computing* 26(1):103–120.

Adulyasak Y, Cordeau JF, Jans R (2015) Benders decomposition for production routing under demand uncertainty. *Operations Research* 63(4):851–867.

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

Bierwirth C, Meisel F (2010) A survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research* 202(3):615–627.

Bierwirth C, Meisel F (2015) A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research* 244(3):675–689.

Buhrkal K, Zuglian S, Ropke S, Larsen J, Lusby R (2011) Models for the discrete berth allocation problem: a computational comparison. *Transportation Research Part E: Logistics and Transportation Review* 47(4):461–473.

Carlo HJ, Vis IF, Roodbergen KJ (2015) Seaside operations in container terminals: literature overview, trends, and research directions. *Flexible Services and Manufacturing Journal* 27(2-3):224–262.

Cordeau JF, Laporte G, Legato P, Moccia L (2005) Models and tabu search heuristics for the berth-allocation problem. *Transportation Science* 39(4):526–538.

Cordeau JF, Stojković G, Soumis F, Desrosiers J (2001) Benders decomposition for simultaneous aircraft routing and crew scheduling. *Transportation Science* 35(4):375–388.

Corry P, Bierwirth C (2019) The berth allocation problem with channel restrictions. *Transportation Science* 53(3):708–727.

Dantzig GB, Wolfe P (1960) Decomposition principle for linear programs. *Operations Research* 8(1):101–111.

Dauzère-Pérès S, De Almeida D, Guyon O, Benhizia F (2015) A Lagrangian heuristic framework for a real-life integrated planning problem of railway transportation resources. *Transportation Research Part B: Methodological* 74:138–150.

Ding Y, Jia S, Gu T, Li CL (2016) SGICT builds an optimization-based system for daily berth planning. *Interfaces* 46(4):281–296.

Du Y, Chen Q, Lam JSL, Xu Y, Cao JX (2015) Modeling the impacts of tides and the virtual arrival policy in berth allocation. *Transportation Science* 49(4):939–956.

Edwards H (2010) Pilot assignment to ships in the Sea of Bothnia. *Energy, Natural Resources and Environmental Economics*, 411–425 (Springer).

Fransoo JC, Lee CY (2013) The critical role of ocean container transport in global supply chain performance. *Production and Operations Management* 22(2):253–268.

Goel A, Irnich S (2017) An exact method for vehicle routing and truck driver scheduling problems. *Transportation Science* 51(2):737–754.

Hong Kong Maritime and Port Board (2018) Pilots in Hong Kong. `https://www.hkmpb.gov.hk/en/pilots.html`, Accessed May 1, 2021.

Imai A, Nagaiwa K, Tat CW (1997) Efficient planning of berth allocation for container terminals in Asia. *Journal of Advanced Transportation* 31(1):75–94.

Imai A, Nishimura E, Papadimitriou S (2001) The dynamic berth allocation problem for a container port. *Transportation Research Part B: Methodological* 35(4):401–417.

Imai A, Nishimura E, Papadimitriou S (2003) Berth allocation with service priority. *Transportation Research Part B: Methodological* 37(5):437–457.

Imai A, Sun X, Nishimura E, Papadimitriou S (2005) Berth allocation in a container port: using a continuous location space approach. *Transportation Research Part B: Methodological* 39(3):199–221.

International Maritime Organization (2001) Guidelines on fatigue mitigation and management. Technical report, URL `https://wwwcdn.imo.org/localresources/en/OurWork/HumanElement/Documents/1014.pdf`, Accessed May 1, 2021.

Jans R (2009) Solving lot-sizing problems on parallel identical machines using symmetry-breaking constraints. *INFORMS Journal on Computing* 21(1):123–136.

Jia S, Li CL, Xu Z (2019) Managing navigation channel traffic and anchorage area utilization of a container port. *Transportation Science* 53(3):728–745.

Jia S, Wu L, Meng Q (2020) Joint scheduling of vessel traffic and pilots in seaport waters. *Transportation Science* 54(6):1495–1515.

Lalla-Ruiz E, Shi X, Voß S (2018) The waterway ship scheduling problem. *Transportation Research Part D: Transport and Environment* 60:191–209.

Li S, Jia S (2019) The seaport traffic scheduling problem: formulations and a column-row generation algorithm. *Transportation Research Part B: Methodological* 128:158–184.

Lübbecke E, Lübbecke ME, Möhring RH (2019) Ship traffic optimization for the Kiel Canal. *Operations Research* 67(3):791–812.

Mercier A, Cordeau JF, Soumis F (2005) A computational study of Benders decomposition for the integrated aircraft routing and crew scheduling problem. *Computers & Operations Research* 32(6):1451–1476.

Monaco MF, Sammarra M (2007) The berth allocation problem: a strong formulation solved by a Lagrangean approach. *Transportation Science* 41(2):265–280.

NPR News (2012) Harbor pilots reap high rewards for dangerous job. `https://www.npr.org/2012/03/21/149091141/harbor-pilots-reap-high-rewards-for-dangerous-job`, Accessed May 1, 2021.

Roy D, de Koster R, Bekker R (2020) Modeling and design of container terminal operations. *Operations Research* 68(3):686–715.

Russon MA (2021) The cost of the Suez Canal blockage. `https://www.bbc.com/news/business-56559073`, Accessed May 1, 2021.

Ruther S, Boland N, Engineer FG, Evans I (2017) Integrated aircraft routing, crew pairing, and tail assignment: branch-and-price with many pricing problems. *Transportation Science* 51(1):177–195.

Sandhu R, Klabjan D (2007) Integrated airline fleeting and crew-pairing decisions. *Operations Research* 55(3):439–456.

Stahlbock R, Voß S (2008) Operations research at container terminals: a literature update. *OR Spectrum* 30(1):1–52.

Tang G, Wang W, Song X, Guo Z, Yu X, Qiao F (2016) Effect of entrance channel dimensions on berth occupancy of container terminals. *Ocean Engineering* 117:174–187.

Texas Department of Transportation (2020) Texas port profiles. Technical report, URL `https://ftp.dot.state.tx.us/pub/txdot-info/mrt/port-profiles.pdf`, Accessed May 1, 2021.

The Marine Department of Hong Kong (2020a) Port of Hong Kong in figures. Technical report, URL `https://www.mardep.gov.hk/en/fact/pdf/portstat_pamphlet20.pdf`, Accessed May 1, 2021.

The Marine Department of Hong Kong (2020b) Towage & salvage in Hong Kong. `https://www.mardep.gov.hk/en/pub_services/sdfiles/towasalv.html`, Accessed May 1, 2021.

The Port of Felixstowe (2021) Home page of the Port of Felixstowe. `https://www.portoffelixstowe.co.uk/`, Accessed May 1, 2021.

The Port of Montreal (2020) Services and fees in the Port of Montreal. `https://www.port-montreal.com/en/goods/operations/services-and-fees`, Accessed May 1, 2021.

UNCTAD (2019) Review of maritime transportation 2019. *Paper presented at the United Nations Conference on Trade and Development* (New York and Geneva), URL `http://unctad.org/en/PublicationsLibrary/rmt2019_en.pdf`, Accessed May 1, 2021.

Wermus M, Pope JA (1994) Scheduling harbor pilots. *Interfaces* 24(2):44–52.

Wu L, Jia S, Wang S (2020) Pilotage planning in seaports. *European Journal of Operational Research* 287(1):90–105.

Xie F, Wu T, Zhang C (2019) A branch-and-price algorithm for the integrated berth allocation and quay crane assignment problem. *Transportation Science* 53(5):1427–1454.

Xu Z, Lee CY (2018) New lower bound and exact method for the continuous berth allocation problem. *Operations Research* 66(3):778–798.

Zeighami V, Soumis F (2019) Combining Benders' decomposition and column generation for integrated crew pairing and personalized crew assignment problems. *Transportation Science* 53(5):1479–1499.

Zhang X, Lin J, Guo Z, Liu T (2016) Vessel transportation scheduling optimization based on channel-berth coordination. *Ocean Engineering* 112:145–152.

Zhen L, Liang Z, Zhuge D, Lee LH, Chew EP (2017) Daily berth planning in a tidal port with channel flow control. *Transportation Research Part B: Methodological* 106:193–217.

Electronic Companion

EC.1. Mathematical Proofs

This section presents proofs to lemmas, propositions, theorems, and corollaries in the main text.

EC.1.1. Proof of Proposition 1

Proof of Proposition 1. Given any $\bar{\boldsymbol{\chi}} \in \mathbf{X}$, let $H = \{(i,t) | \sum_{\omega\in\Omega} \bar{\chi}_\omega \alpha_{\omega,i,t} > 0, t \in \mathcal{E}_i, i \in \mathcal{I}\}$. Given $\bar{\boldsymbol{\chi}}$, the corresponding PBSP is feasible if and only if there exists a solution $\boldsymbol{\mu}'$ such that $\sum_{\phi\in\Phi} \mu'_\phi \gamma_{\phi,i,t} = \sum_{\omega\in\Omega} \bar{\chi}_\omega \alpha_{\omega,i,t}$, $\forall (i,t) \in H$. We can show that such a $\boldsymbol{\mu}'$ exists for any $\bar{\boldsymbol{\chi}}$.

Recall that $\bigcup_{s\in\mathcal{S}} \mathcal{T}_s = \mathcal{T}$ and that all vessel routes in model M2 satisfy constraints (33). Hence, corresponding to each $(i,t) \in H$, one can always construct a feasible pilot route that visits i at time t. In particular, the pilot route can consist of only task i (which starts at time t) and a rest period (which starts at a feasible time) and is completed by a pilot in a suitable shift. Then, based on these pilot routes, one can easily construct a feasible solution for the PBSP.

As a result, given any $\bar{\boldsymbol{\chi}}$, the PBSP is always feasible. Furthermore, considering that the cost parameters $\bar{c}_\phi$ are finite and due to constraints (46)–(48), any feasible solution of the PBSP must be bounded. Therefore, as the dual of the PBSP, the DBSP is also feasible and bounded. □

EC.1.2. Proof of Theorem 1

Proof of Theorem 1. At each node of the branch-and-bound tree, the BMP and the PBSP are defined on a subset $\Omega' \subseteq \Omega$ of vessel routes and a subset $\Phi' \subseteq \Phi$ of pilot routes, due to the branching constraints. The approach solves the BMP to optimality with column-and-Benders-cut generation. Let $(\boldsymbol{\chi}^*, \eta^*)$ and $\boldsymbol{\mu}^*$ denote the optimal solution to the BMP and the associated PBSP, respectively. According to Benders (1962), we have that (i) $\boldsymbol{\chi}^*$ and $\boldsymbol{\mu}^*$ constitute an optimal solution to the LP relaxation of M2 defined on the same subsets of vessel routes and pilot routes (i.e., the LP relaxation of M2 imposed with the same branching constraints) and that (ii) the BMP and the LP relaxation of M2 have the same optimal objective function value.

Now consider our branching scheme. One can verify the solution $(\boldsymbol{\chi}^*)$ generated by the C&BCG algorithm to the BMP is fractional if and only if there is at least one fractional value in $\boldsymbol{\Lambda}$ or $\boldsymbol{\Xi}$. Further, given any integer solution $(\bar{\boldsymbol{\chi}})$ for the BMP, the solution generated by column generation to the corresponding PBSP is fractional if and only if there is at least one fractional value in $\boldsymbol{\Pi}$ or $\boldsymbol{\Psi}$.

Therefore, by imposing branching constraints on the BMP and the PBSP at the associated nodes of the tree, and by solving the BMP at each node to optimality, the approach gradually

reaches a solution to the BMP that has the minimum objective function value among all integer solutions (i.e., solutions where all values in the solution $\boldsymbol{\chi}^*$ are integral and all values in the solution $\boldsymbol{\mu}^*$ for the corresponding PBSP are integral as well). Correspondingly, the approach reaches an integer solution that has the minimum cost among all integer solutions for the LP relaxation of M2. Therefore, the BPBC approach delivers an optimal solution for M2. □

EC.1.3. Proof of Proposition 2

Proof of Proposition 2. Consider the BMP without these LBL inequalities. Given any feasible solution of the χ variables, denoted by $\bar{\boldsymbol{\chi}}$, to the BMP, let $\eta^*(\bar{\boldsymbol{\chi}})$ denote the associated optimal solution of the η variable in the BMP. To prove that the LBL inequalities are valid for the BMP, it suffices to show that the following linear program generates a lower bound to $\eta^*(\bar{\boldsymbol{\chi}})$:

$$[\mathbf{P}_{LBL}(\bar{\boldsymbol{\chi}})] \quad \min \sum_{s\in\mathcal{S}} c_s^3 n_s \tag{EC.1}$$

$$\text{s.t. } (67)-(70)$$

$$\sum_{s\in\mathcal{S}_t} \nu_{i,t,s} = \sum_{\omega\in\Omega} \bar{\chi}_\omega \alpha_{\omega,i,t} \ \forall t\in\mathcal{E}_i, i\in\mathcal{I}. \tag{EC.2}$$

We have shown in Proposition 1 that the DBSP is always feasible and bounded. Given $\bar{\boldsymbol{\chi}}$, let $\boldsymbol{\delta}^*$ and $\boldsymbol{\zeta}^*$ denote an optimal solution to the DBSP. Because $(\boldsymbol{\delta}^*, \boldsymbol{\zeta}^*)\in\Gamma_\Delta$, we have $\eta^*(\bar{\boldsymbol{\chi}}) \geq \sum_{i\in\mathcal{I}}\sum_{t\in\mathcal{T}}\sum_{\omega\in\Omega}\bar{\chi}_\omega\alpha_{\omega,i,t}\delta^*_{i,t} + \sum_{\phi\in\Phi}\zeta^*_\phi$ due to constraints (54). Let $\boldsymbol{\mu}^*$ denote the optimal solution to the dual of the DBSP (i.e., the PBSP). Because of strong duality, we have $\sum_{\phi\in\Phi}\mu^*_\phi\bar{c}_\phi = \sum_{i\in\mathcal{I}}\sum_{t\in\mathcal{T}}\sum_{\omega\in\Omega}\bar{\chi}_\omega\alpha_{\omega,i,t}\delta^*_{i,t} + \sum_{\phi\in\Phi}\zeta^*_\phi$.

We construct the solution $(\boldsymbol{\nu'}, \boldsymbol{n'})$ for the variables in the $\mathrm{P}_{LBL}(\bar{\boldsymbol{\chi}})$, where

$$\nu'_{i,t,s} = \sum_{\phi\in\Phi_s} \mu^*_\phi \gamma_{\phi,i,t} \ \forall i\in\mathcal{I}, t\in\mathcal{T}_s, s\in\mathcal{S} \tag{EC.3}$$

$$n'_s = \sum_{\phi\in\Phi_s} \mu^*_\phi \ \forall s\in\mathcal{S}. \tag{EC.4}$$

Considering the feasibility of pilot routes and the feasibility of $\boldsymbol{\mu}^*$ for the PBSP, we have

$$\sum_{i\in\mathcal{I}} \sum_{t'=\max\{\underline{T}_s, t-d_i-\underline{q}'_{i,t,s}+1\}}^{t} \gamma_{\phi,i,t'} \leq 1, \ \forall t\in\mathcal{T}_s, \phi\in\Phi_s, s\in\mathcal{S} \tag{EC.5}$$

$$\sum_{s\in\mathcal{S}_t}\sum_{\phi\in\Phi_s} \mu^*_\phi\gamma_{\phi,i,t} = \sum_{\omega\in\Omega}\bar{\chi}_\omega\alpha_{\omega,i,t} \ \forall t\in\mathcal{E}_i, i\in\mathcal{I}. \tag{EC.6}$$

Because $\mu^*_\phi \geq 0$, $\forall\phi\in\Phi$, by multiplying each term in inequalities (EC.5) with μ^*_ϕ, and summing them over $\phi\in\Phi_s$ we have

$$\sum_{i\in\mathcal{I}} \sum_{t'=\max\{\underline{T}_s, t-d_i-\underline{q}'_{i,t,s}+1\}}^{t} \sum_{\phi\in\Phi_s} \mu^*_\phi\gamma_{\phi,i,t'} \leq \sum_{\phi\in\Phi_s}\mu^*_\phi, \ \forall t\in\mathcal{T}_s, s\in\mathcal{S}. \tag{EC.7}$$

Then, combining (EC.3), (EC.4), and (EC.7) gives

$$\sum_{i\in\mathcal{I}}\sum_{t'=\max\{\underline{T}_s,t-d_i-\underline{q}'_{i,t,s}+1\}}^{t}\nu'_{i,t',s}\le n'_s\ \forall t\in\mathcal{T}_s, s\in\mathcal{S}, \tag{EC.8}$$

and from (EC.3) and (EC.6) we have

$$\sum_{s\in\mathcal{S}_t}\nu'_{i,t,s}=\sum_{\omega\in\Omega}\bar{\chi}_\omega\alpha_{\omega,i,t}\ \forall t\in\mathcal{E}_i, i\in\mathcal{I} \tag{EC.9}$$

$$\nu'_{i,t,s}\le 1\ \forall i\in\mathcal{I}, t\in\mathcal{T}, s\in\mathcal{S} \tag{EC.10}$$

$$\nu'_{i,t,s}\ge 0\ \forall i\in\mathcal{I}, t\in\mathcal{T}, s\in\mathcal{S}. \tag{EC.11}$$

Finally, summarizing inequalities (EC.8)–(EC.11), one can conclude that the constructed solution $(\boldsymbol{\nu'}, \boldsymbol{n'})$ is feasible for the $\mathrm{P}_{LBL}(\bar{\boldsymbol{\chi}})$. Besides, we have $\sum_{s\in\mathcal{S}} c_s^3 n'_s=\sum_{s\in\mathcal{S}}\sum_{\phi\in\Phi_s}\bar{c}_\phi\mu^*_\phi=\sum_{\phi\in\Phi}\bar{c}_\phi\mu^*_\phi\le\eta^*(\bar{\boldsymbol{\chi}})$. Let $(\boldsymbol{\nu^*}, \boldsymbol{n^*})$ denote an optimal solution for the $\mathrm{P}_{LBL}(\bar{\boldsymbol{\chi}})$. Because the $\mathrm{P}_{LBL}(\bar{\boldsymbol{\chi}})$ forms a minimization problem, we have $\sum_{s\in\mathcal{S}} c_s^3 n^*_s\le\sum_{s\in\mathcal{S}} c_s^3 n'_s\le\eta^*(\bar{\boldsymbol{\chi}})$.

Therefore, the $\mathrm{P}_{LBL}(\bar{\boldsymbol{\chi}})$ generates a lower bound to $\eta^*(\bar{\boldsymbol{\chi}})$, given any feasible solution of the χ variables to the original BMP. This indicates that the LBL inequalities are valid for the BMP. □

EC.1.4. Proof of Lemma 1

Proof of Lemma 1. We provide a proof to this lemma based on the BMP with a full set of constraints (39)–(41) and without the LBL cuts proposed in Section 4.5.2, while the extension to the BMP with only a subset of constraints from (39)–(41) or with the LBL cuts is straightforward.

We start by formulating the Lagrangian dual of the BMP. Let $\bar{\boldsymbol{\pi}}=(\bar{\pi}_k|\bar{\pi}_k\in\mathbb{R}, k\in\mathcal{K})$, $\bar{\boldsymbol{\lambda}}=(\bar{\lambda}_{b,t}|\bar{\lambda}_{b,t}\le 0, t\in\mathcal{T}, b\in\mathcal{B})$, $\bar{\boldsymbol{\theta}}=(\bar{\theta}_{t,(i,j)}|\bar{\theta}_{t,(i,j)}\le 0, t\in\mathcal{T}, (i,j)\in\mathcal{V})$, $\bar{\boldsymbol{\kappa}}=(\bar{\kappa}_{t,(i,j)}|\bar{\kappa}_{t,(i,j)}\le 0, t\in\mathcal{T}, (i,j)\in\mathcal{U})$, $\bar{\boldsymbol{\varphi}}=(\bar{\varphi}_{(\boldsymbol{\delta},\boldsymbol{\zeta})}|\bar{\varphi}_{(\boldsymbol{\delta},\boldsymbol{\zeta})}\ge 0, (\boldsymbol{\delta},\boldsymbol{\zeta})\in\Gamma_\Delta)$, and $\bar{\boldsymbol{\rho}}=(\bar{\rho}_\omega|\bar{\rho}_\omega\le 0, \omega\in\Omega)$ be the vectors of the Lagrangian multipliers for constraints (38)–(41), (54), and (55), respectively. Besides, let $\Omega_{k,b}$ denote the set of columns corresponding to vessel k being handled at berth b, where $k\in\mathcal{K}$, $b\in\mathcal{B}_k$. As implied by constraints (38), (55), and (56), the following inequalities are valid for the BMP:

$$\sum_{\omega\in\Omega_{k,b}}\chi_\omega\le 1,\ \forall b\in\mathcal{B}_k, k\in\mathcal{K}. \tag{EC.12}$$

Given $\bar{\boldsymbol{\pi}}$, $\bar{\boldsymbol{\lambda}}$, $\bar{\boldsymbol{\theta}}$, $\bar{\boldsymbol{\kappa}}$, $\bar{\boldsymbol{\varphi}}$, and $\bar{\boldsymbol{\rho}}$, the Lagrangian dual of the BMP is as follows:

$$\begin{aligned}\mathcal{D}(\bar{\boldsymbol{\pi}},\bar{\boldsymbol{\lambda}},\bar{\boldsymbol{\theta}},\bar{\boldsymbol{\kappa}},\bar{\boldsymbol{\varphi}},\bar{\boldsymbol{\rho}})=\min_{\boldsymbol{\chi},\eta}\Bigg\{&\sum_{\omega\in\Omega}\chi_\omega\bar{c}_\omega+\eta+\sum_{k\in\mathcal{K}}(1-\sum_{\omega\in\Omega_k}\chi_\omega)\bar{\pi}_k\\&+\sum_{t\in\mathcal{T}}\sum_{b\in\mathcal{B}}(1-\sum_{\omega\in\Omega}\chi_\omega\sum_{t=1}^{\tau}\sum_{t'=\min\{\tau+1,\overline{T}\}}^{\overline{T}}\beta_{\omega,b,t,t'})\bar{\lambda}_{b,t}\end{aligned}$$

$$
\begin{aligned}
&+\sum_{t\in\mathcal{T}}\sum_{(i,j)\in\mathcal{V}}(1-\sum_{\omega\in\Omega}\chi_\omega\sum_{t'=\max\{t-f_{i,j}+1,1\}}^{t}\alpha_{\omega,i,t'}-\sum_{\omega\in\Omega}\chi_\omega\alpha_{\omega,j,t})\bar{\theta}_{t,(i,j)}\\
&+\sum_{t\in\mathcal{T}}\sum_{(i,j)\in\mathcal{U}}(1-\sum_{\omega\in\Omega}\chi_\omega\sum_{t'=\max\{t-d_i+1,1\}}^{t}\alpha_{\omega,i,t'}-\sum_{\omega\in\Omega}\chi_\omega\sum_{t'=\max\{t-d_j+1,1\}}^{t}\alpha_{\omega,j,t'})\bar{\kappa}_{t,(i,j)}\\
&-\sum_{(\boldsymbol{\delta},\boldsymbol{\zeta})\in\Gamma_\Delta}(\eta-\sum_{i\in\mathcal{I}}\sum_{t\in\mathcal{T}}\sum_{\omega\in\Omega}\chi_\omega\alpha_{\omega,i,t}\delta_{i,t}-\sum_{\phi\in\Phi}\zeta_\phi)\bar{\varphi}_{(\boldsymbol{\delta},\boldsymbol{\zeta})}\\
&+\sum_{\omega\in\Omega}(1-\chi_\omega)\bar{\rho}_\omega,\ \text{s.t. (56), (57), (EC.12)}\Bigg\}. \qquad \text{(EC.13)}
\end{aligned}
$$

Equation (EC.13) is equivalent to

$$
\begin{aligned}
\mathcal{D}(\bar{\boldsymbol{\pi}},\bar{\boldsymbol{\lambda}},\bar{\boldsymbol{\theta}},\bar{\boldsymbol{\kappa}},\bar{\boldsymbol{\varphi}},\bar{\boldsymbol{\rho}}) &= \sum_{k\in\mathcal{K}}\bar{\pi}_k+\sum_{t\in\mathcal{T}}\sum_{b\in\mathcal{B}}\bar{\lambda}_{b,t}+\sum_{t\in\mathcal{T}}\sum_{(i,j)\in\mathcal{V}}\bar{\theta}_{t,(i,j)}+\sum_{t\in\mathcal{T}}\sum_{(i,j)\in\mathcal{U}}\bar{\kappa}_{t,(i,j)}+\sum_{(\boldsymbol{\delta},\boldsymbol{\zeta})\in\Gamma_\Delta}\sum_{\phi\in\Phi}\zeta_\phi\bar{\varphi}_{(\boldsymbol{\delta},\boldsymbol{\zeta})}\\
&+\sum_{\omega\in\Omega}\bar{\rho}_\omega+\min_{\boldsymbol{\chi},\eta}\Bigg\{\sum_{\omega\in\Omega}\chi_\omega\bar{c}_\omega+(1-\sum_{(\boldsymbol{\delta},\boldsymbol{\zeta})\in\Gamma_\Delta}\bar{\varphi}_{(\boldsymbol{\delta},\boldsymbol{\zeta})})\eta-\sum_{k\in\mathcal{K}}\sum_{\omega\in\Omega_k}\chi_\omega\bar{\pi}_k\\
&-\sum_{t\in\mathcal{T}}\sum_{b\in\mathcal{B}}\sum_{\omega\in\Omega}\chi_\omega\sum_{t=1}^{\tau}\sum_{t'=\min\{\tau+1,\overline{T}\}}^{\overline{T}}\beta_{\omega,b,t,t'}\bar{\lambda}_{b,t}\\
&-\sum_{t\in\mathcal{T}}\sum_{(i,j)\in\mathcal{V}}(\sum_{\omega\in\Omega}\chi_\omega\sum_{t'=\max\{t-f_{i,j}+1,1\}}^{t}\alpha_{\omega,i,t'}\bar{\theta}_{t,(i,j)}+\sum_{\omega\in\Omega}\chi_\omega\alpha_{\omega,j,t}\bar{\theta}_{t,(i,j)})\\
&-\sum_{t\in\mathcal{T}}\sum_{(i,j)\in\mathcal{U}}(\sum_{\omega\in\Omega}\chi_\omega\sum_{t'=\max\{t-d_i+1,1\}}^{t}\alpha_{\omega,i,t'}\bar{\kappa}_{t,(i,j)}+\sum_{\omega\in\Omega}\chi_\omega\sum_{t'=\max\{t-d_j+1,1\}}^{t}\alpha_{\omega,j,t'}\bar{\kappa}_{t,(i,j)})\\
&+\sum_{(\boldsymbol{\delta},\boldsymbol{\zeta})\in\Gamma_\Delta}\sum_{i\in\mathcal{I}}\sum_{t\in\mathcal{T}}\sum_{\omega\in\Omega}\chi_\omega\alpha_{\omega,i,t}\delta_{i,t}\bar{\varphi}_{(\boldsymbol{\delta},\boldsymbol{\zeta})}-\sum_{\omega\in\Omega}\chi_\omega\bar{\rho}_\omega,\ \text{s.t. (56), (57), (EC.12)}\Bigg\}. \qquad \text{(EC.14)}
\end{aligned}
$$

Let Z^* be the optimal objective function value for the BMP. We have

$$
Z^*\geq\max_{\bar{\boldsymbol{\pi}},\bar{\boldsymbol{\lambda}},\bar{\boldsymbol{\theta}},\bar{\boldsymbol{\kappa}},\bar{\boldsymbol{\varphi}},\bar{\boldsymbol{\rho}}}\mathcal{D}(\bar{\boldsymbol{\pi}},\bar{\boldsymbol{\lambda}},\bar{\boldsymbol{\theta}},\bar{\boldsymbol{\kappa}},\bar{\boldsymbol{\varphi}},\bar{\boldsymbol{\rho}}). \qquad \text{(EC.15)}
$$

Since the BMP is solved by column and Benders cut generation, where columns and Benders cuts are added dynamically, we let $\tilde{\Omega}\subseteq\Omega$ and $\tilde{\Gamma}_\Delta\subseteq\Gamma_\Delta$ denote the current sets of columns and Benders cuts generated for solving the BMP, respectively. After solving the BMP with columns from $\tilde{\Omega}$ and Benders cuts from $\tilde{\Gamma}_\Delta$, one obtains the optimal solutions to the dual variables associated with constraints (38)–(41), (54), and (55) which are denoted by $\boldsymbol{\pi}^*=(\pi_k^*|k\in\mathcal{K})$, $\boldsymbol{\lambda}^*=(\lambda_{b,t}^*|t\in\mathcal{T},b\in\mathcal{B})$, $\boldsymbol{\theta}^*=(\theta_{t,(i,j)}^*|t\in\mathcal{T},(i,j)\in\mathcal{V})$, $\boldsymbol{\kappa}^*=(\kappa_{t,(i,j)}^*|t\in\mathcal{T},(i,j)\in\mathcal{U})$, $\boldsymbol{\varphi}^*=(\varphi_{(\boldsymbol{\delta},\boldsymbol{\zeta})}^*|(\boldsymbol{\delta},\boldsymbol{\zeta})\in\tilde{\Gamma}_\Delta)$, and $\boldsymbol{\rho}^*=(\rho_\omega^*|\omega\in\tilde{\Omega})$, respectively. We further define vectors $\boldsymbol{\varphi}^+$ and $\boldsymbol{\rho}^+$ such that $\boldsymbol{\varphi}^+=(\varphi_{(\boldsymbol{\delta},\boldsymbol{\zeta})}^+|\varphi_{(\boldsymbol{\delta},\boldsymbol{\zeta})}^+=\varphi_{(\boldsymbol{\delta},\boldsymbol{\zeta})}^*,(\boldsymbol{\delta},\boldsymbol{\zeta})\in\tilde{\Gamma}_\Delta,\varphi_{(\boldsymbol{\delta},\boldsymbol{\zeta})}^+=0,(\boldsymbol{\delta},\boldsymbol{\zeta})\in\Gamma_\Delta\setminus\tilde{\Gamma}_\Delta)$ and $\boldsymbol{\rho}^+=(\rho_\omega^+|\rho_\omega^+=\rho_\omega^*,\omega\in\tilde{\Omega},\rho_\omega^+=0,\omega\in\Omega\setminus\tilde{\Omega})$.

By definition, the BMP is constructed to be feasible and bounded. As a result, the dual of BMP is also feasible and bounded. By the feasibility of the dual of the BMP, one can easily verify that $(\boldsymbol{\pi}^*, \boldsymbol{\lambda}^*, \boldsymbol{\theta}^*, \boldsymbol{\kappa}^+, \boldsymbol{\varphi}^+, \boldsymbol{\rho}^+)$ forms a feasible solution for $\mathcal{D}(\bar{\boldsymbol{\pi}}, \bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\kappa}}, \bar{\boldsymbol{\varphi}}, \bar{\boldsymbol{\rho}})$. It follows that

$$Z^* \geq \mathcal{D}(\boldsymbol{\pi}^*, \boldsymbol{\lambda}^*, \boldsymbol{\theta}^*, \boldsymbol{\kappa}^*, \boldsymbol{\varphi}^+, \boldsymbol{\rho}^+), \tag{EC.16}$$

where

$$\begin{aligned}
\mathcal{D}(\boldsymbol{\pi}^*, \boldsymbol{\lambda}^*, \boldsymbol{\theta}^*, \boldsymbol{\kappa}^*, \boldsymbol{\varphi}^+, \boldsymbol{\rho}^+) = & \sum_{k\in\mathcal{K}} \pi_k^* + \sum_{t\in\mathcal{T}}\sum_{b\in\mathcal{B}} \lambda_{b,t}^* + \sum_{t\in\mathcal{T}}\sum_{(i,j)\in\mathcal{V}} \theta_{t,(i,j)}^* + \sum_{t\in\mathcal{T}}\sum_{(i,j)\in\mathcal{U}} \kappa_{t,(i,j)}^* \\
& + \sum_{(\boldsymbol{\delta},\boldsymbol{\zeta})\in\tilde{\Gamma}_\Delta}\sum_{\phi\in\Phi} \zeta_\phi \varphi_{(\boldsymbol{\delta},\boldsymbol{\zeta})}^* + \sum_{\omega\in\tilde{\Omega}} \rho_\omega^* + \min_{\boldsymbol{\chi},\eta} \Bigg\{ \sum_{\omega\in\Omega} \chi_\omega \bar{c}_\omega + (1 - \sum_{(\boldsymbol{\delta},\boldsymbol{\zeta})\in\tilde{\Gamma}_\Delta} \varphi_{(\boldsymbol{\delta},\boldsymbol{\zeta})}^*)\eta - \sum_{k\in\mathcal{K}}\sum_{\omega\in\Omega_k} \chi_\omega \pi_k^* \\
& - \sum_{t\in\mathcal{T}}\sum_{b\in\mathcal{B}}\sum_{\omega\in\Omega} \chi_\omega \sum_{t=1}^{\tau} \sum_{t'=\min\{\tau+1,\overline{T}\}}^{\overline{T}} \beta_{\omega,b,t,t'} \lambda_{b,t}^* \\
& - \sum_{t\in\mathcal{T}}\sum_{(i,j)\in\mathcal{V}} \Big(\sum_{\omega\in\Omega} \chi_\omega \sum_{t'=\max\{t-f_{i,j}+1,1\}}^{t} \alpha_{\omega,i,t'} \theta_{t,(i,j)}^* + \sum_{\omega\in\Omega} \chi_\omega \alpha_{\omega,j,t} \theta_{t,(i,j)}^* \Big) \\
& - \sum_{t\in\mathcal{T}}\sum_{(i,j)\in\mathcal{U}} \Big(\sum_{\omega\in\Omega} \chi_\omega \sum_{t'=\max\{t-d_i+1,1\}}^{t} \alpha_{\omega,i,t'} \kappa_{t,(i,j)}^* + \sum_{\omega\in\Omega} \chi_\omega \sum_{t'=\max\{t-d_j+1,1\}}^{t} \alpha_{\omega,j,t'} \kappa_{t,(i,j)}^* \Big) \\
& + \sum_{(\boldsymbol{\delta},\boldsymbol{\zeta})\in\tilde{\Gamma}_\Delta}\sum_{i\in\mathcal{I}}\sum_{t\in\mathcal{T}}\sum_{\omega\in\Omega} \chi_\omega \alpha_{\omega,i,t} \delta_{i,t} \varphi_{(\boldsymbol{\delta},\boldsymbol{\zeta})}^* - \sum_{\omega\in\tilde{\Omega}} \chi_\omega \rho_\omega^*, \text{ s.t. } (56), (57), (\text{EC.12}) \Bigg\}.
\end{aligned} \tag{EC.17}$$

Due to the feasibility of the dual of the BMP, we have $\sum_{(\boldsymbol{\delta},\boldsymbol{\zeta})\in\tilde{\Gamma}_\Delta} \varphi_{(\boldsymbol{\delta},\boldsymbol{\zeta})}^* \leq 1$ and $\rho_\omega^* \leq 0$, $\forall \omega \in \tilde{\Omega}$. Further, considering that $\eta \geq 0$ and $\chi_\omega \geq 0$, $\forall \omega \in \tilde{\Omega}$, we have $(1 - \sum_{(\boldsymbol{\delta},\boldsymbol{\zeta})\in\tilde{\Gamma}_\Delta} \varphi_{(\boldsymbol{\delta},\boldsymbol{\zeta})}^*)\eta \geq 0$ and $\sum_{\omega\in\tilde{\Omega}} \chi_\omega \rho_\omega^* \leq 0$. Recall that in this lemma, the optimal objective function value of the BMP is denoted by Υ. By strong duality, we have $\Upsilon = \sum_{k\in\mathcal{K}} \pi_k^* + \sum_{t\in\mathcal{T}}\sum_{b\in\mathcal{B}} \lambda_{b,t}^* + \sum_{t\in\mathcal{T}}\sum_{(i,j)\in\mathcal{V}} \theta_{t,(i,j)}^* + \sum_{t\in\mathcal{T}}\sum_{(i,j)\in\mathcal{U}} \kappa_{t,(i,j)}^* + \sum_{(\boldsymbol{\delta},\boldsymbol{\zeta})\in\tilde{\Gamma}_\Delta}\sum_{\phi\in\Phi} \zeta_\phi \varphi_{(\boldsymbol{\delta},\boldsymbol{\zeta})}^* + \sum_{\omega\in\tilde{\Omega}} \rho_\omega^*$. In addition, by definition, each $\omega \in \Omega$ is associated with a berth $b \in \mathcal{B}$ and a vessel $k \in \mathcal{K}$. Let $k(\omega)$ and $b(\omega)$ represent the vessel and the berth associated with ω, $\forall \omega \in \Omega$. Therefore, we have the following inequality:

$$\begin{aligned}
\mathcal{D}(\boldsymbol{\pi}^*, \boldsymbol{\lambda}^*, \boldsymbol{\theta}^*, \boldsymbol{\kappa}^*, \boldsymbol{\varphi}^+, \boldsymbol{\rho}^+) \geq \mathcal{D}' = \Upsilon + \min_{\boldsymbol{\chi}} \Bigg\{ & \sum_{\omega\in\Omega} \chi_\omega (\bar{c}_\omega - \pi_{k(\omega)}^* - \sum_{t\in\mathcal{T}}\sum_{t=1}^{\tau} \sum_{t'=\min\{\tau+1,\overline{T}\}}^{\overline{T}} \beta_{\omega,b(\omega),t,t'} \lambda_{b(\omega),t}^* \\
& - \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}_{k(\omega)}}\sum_{(i,j)\in\mathcal{V}} \sum_{t'=\max\{t-f_{i,j}+1,1\}}^{t} \alpha_{\omega,i,t'} \theta_{t,(i,j)}^* - \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}_{k(\omega)}}\sum_{(j,i)\in\mathcal{V}} \alpha_{\omega,i,t} \theta_{t,(j,i)}^* \\
& - \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}_{k(\omega)}}\sum_{(i,j)\in\mathcal{U}} \sum_{t'=\max\{t-d_i+1,1\}}^{t} \alpha_{\omega,i,t'} \kappa_{t,(i,j)}^* - \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}_{k(\omega)}}\sum_{(j,i)\in\mathcal{U}} \sum_{t'=\max\{t-d_i+1,1\}}^{t} \alpha_{\omega,i,t'} \kappa_{t,(j,i)}^* \\
& + \sum_{(\boldsymbol{\delta},\boldsymbol{\zeta})\in\tilde{\Gamma}_\Delta}\sum_{i\in\mathcal{I}_{k(\omega)}}\sum_{t\in\mathcal{T}}\sum_{\omega\in\Omega} \chi_\omega \alpha_{\omega,i,t} \delta_{i,t} \varphi_{(\boldsymbol{\delta},\boldsymbol{\zeta})}^*), \text{ s.t. } (56), (\text{EC.12}) \Bigg\},
\end{aligned} \tag{EC.18}$$

Considering that $\Omega = \bigcup_{k\in\mathcal{K}}\bigcup_{b\in\mathcal{B}_k}\Omega_{k,b}$ and by decomposing the minimization problem in (EC.18) for each combination of b and k, where $k\in\mathcal{K}$, $b\in\mathcal{B}_k$, $\mathcal{D}'$ can be equivalently calculated by:

$$\begin{aligned}\mathcal{D}' = \Upsilon + \sum_{k\in\mathcal{K}}\sum_{b\in\mathcal{B}_k}\sum_{\omega\in\Omega_{k,b}} \min_{\chi_\omega:\omega\in\Omega_{k,b}} \Bigg\{ & \chi_\omega(\bar{c}_\omega - \pi^*_k - \sum_{t\in\mathcal{T}}\sum_{t=1}^{\tau}\sum_{t'=\min\{\tau+1,\overline{T}\}}^{\overline{T}} \beta_{\omega,b,t,t'}\lambda^*_{b,t} \\ & - \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}_k}\sum_{(i,j)\in\mathcal{V}}\sum_{t'=\max\{t-f_{i,j}+1,1\}}^{t} \alpha_{\omega,i,t'}\theta^*_{t,(i,j)} - \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}_k}\sum_{(j,i)\in\mathcal{V}} \alpha_{\omega,i,t}\theta^*_{t,(j,i)} \\ & - \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}_k}\sum_{(i,j)\in\mathcal{U}}\sum_{t'=\max\{t-d_i+1,1\}}^{t} \alpha_{\omega,i,t'}\kappa^*_{t,(i,j)} - \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}_k}\sum_{(j,i)\in\mathcal{U}}\sum_{t'=\max\{t-d_i+1,1\}}^{t} \alpha_{\omega,i,t'}\kappa^*_{t,(j,i)} \\ & + \sum_{(\boldsymbol{\delta},\boldsymbol{\zeta})\in\tilde{\Gamma}_\Delta}\sum_{i\in\mathcal{I}_k}\sum_{t\in\mathcal{T}}\sum_{\omega\in\Omega}\chi_\omega\alpha_{\omega,i,t}\delta_{i,t}\varphi^*_{(\boldsymbol{\delta},\boldsymbol{\zeta})}),\ \text{s.t.}\sum_{\omega\in\Omega_{k,b}}\chi_\omega\le 1,\ 0\le\chi_\omega\le 1,\ \forall\omega\in\Omega_{k,b}\Bigg\}. \end{aligned} \tag{EC.19}$$

Let $r_{k,b}$ be the optimal objective function value of the pricing problem $\text{MPP}_{k,b}$, where $k\in\mathcal{K}$, $b\in\mathcal{B}_k$. It can be readily seen that

$$\mathcal{D}' = \Upsilon + \sum_{k\in\mathcal{K}}\sum_{b\in\mathcal{B}_k:r_{k,b}<0} r_{k,b}. \tag{EC.20}$$

Summarizing (EC.16), (EC.18), and (EC.20), we have $Z^* \ge \Upsilon + \sum_{k\in\mathcal{K}}\sum_{b\in\mathcal{B}_k:r_{k,b}<0} r_{k,b}$, which completes the proof. □

EC.1.5. Proof of Proposition 3

Proof of Proposition 3. Let $LB = \Upsilon + \sum_{k\in\mathcal{K}}\sum_{b\in\mathcal{B}_k:r_{k,b}<0} r_{k,b}$. From Lemma 1, we have that LB is a lower bound for the BMP at the current node. Given a pair of $b\in\mathcal{B}$ and $k\in\mathcal{K}$ that satisfies the conditions in this proposition, the lower bound (denoted by LB') of the BMP with the constraint $\sum_{\omega\in\Omega_{k,b}}\chi_\omega = 1$ at the current node can be calculated by

$$\begin{aligned} LB' = LB + \min_{\omega\in\Omega_{k,b}} \Bigg\{ & \bar{c}_\omega - \pi^*_k - \sum_{t\in\mathcal{T}}\sum_{t=1}^{\tau}\sum_{t'=\min\{\tau+1,\overline{T}\}}^{\overline{T}} \beta_{\omega,b,t,t'}\lambda^*_{b,t} \\ & - \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}_k}\sum_{(i,j)\in\mathcal{V}}\sum_{t'=\max\{t-f_{i,j}+1,1\}}^{t} \alpha_{\omega,i,t'}\theta^*_{t,(i,j)} - \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}_k}\sum_{(j,i)\in\mathcal{V}} \alpha_{\omega,i,t}\theta^*_{t,(j,i)} \\ & - \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}_k}\sum_{(i,j)\in\mathcal{U}}\sum_{t'=\max\{t-d_i+1,1\}}^{t} \alpha_{\omega,i,t'}\kappa^*_{t,(i,j)} - \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}_k}\sum_{(j,i)\in\mathcal{U}}\sum_{t'=\max\{t-d_i+1,1\}}^{t} \alpha_{\omega,i,t'}\kappa^*_{t,(j,i)} \\ & + \sum_{(\boldsymbol{\delta},\boldsymbol{\zeta})\in\tilde{\Gamma}_\Delta}\sum_{i\in\mathcal{I}_k}\sum_{t\in\mathcal{T}}\sum_{\omega\in\Omega}\chi_\omega\alpha_{\omega,i,t}\delta_{i,t}\varphi^*_{(\boldsymbol{\delta},\boldsymbol{\zeta})} \Bigg\}, \end{aligned}$$

which equals $LB + r_{k,b}$.

Therefore, $LB + r_{k,b}$ is a lower bound for the BMP with the constraint $\sum_{\omega \in \Omega_{k,b}} \chi_\omega = 1$ at the current node. It follows easily that $LB + r_{k,b}$ is also a lower bound for the BMP with the constraint $\sum_{\omega \in \Omega_{k,b}} \chi_\omega = 1$ at any node descended from the current node. $\square$

EC.1.6. Proof of Proposition 4

Proof of Proposition 4. Given any solution to the BMP, denoted by $\bar{\boldsymbol{\chi}}$, the PBSP at node ι' is a relaxation of that at node ι. Accordingly, given $\bar{\boldsymbol{\chi}}$, the DBSP at node ι is a relaxation of that at node ι'. Hence, any point in the polyhedron defined by (50)–(52) of the DBSP at node ι' is also in the polyhedron defined by (50)–(52) of the DBSP at node ι. As a result, according to Benders (1962), the Benders cuts that are valid for the BMP at node ι' are also valid for the BMP at node ι. $\square$

EC.1.7. Proof of Corollary 1

Proof of Corollary 1. For the root node (denoted by ι_0), we have $\Theta_{\iota_0} = \varnothing$. Hence, we have that for any node $\iota \neq \iota_0$, $\Theta_{\iota_0} \subseteq \Theta_\iota$. Therefore, according to Proposition 4, Benders cuts that are valid for the BMP at the root node are also valid for the BMP at any other node in the branch-and-bound tree. $\square$

EC.2. The Label Correcting Algorithm for Solving the SPP$_s$

This section describes the label correcting algorithm for solving the pricing problems in the Dantzig-Wolfe decomposition of the PBSP. Consider the pricing problem corresponding to shift s (SPP$_s$). The label correcting algorithm aims to identify the path, denoted by $\mathscr{P}_s^*$ in the space-time network G_s that has the minimum cost (denoted by $\tilde{c}_s^*$).

In the algorithm, H_s is partitioned into two subsets: $H_s^1 = \{(i,t) | t + d_i \leq \overline{L}_s, (i,t) \in H_s\}$ and $H_s^2 = \{(i,t) | t \geq \underline{L}_s + g_s, (i,t) \in H_s\}$ (a node (i,t) can appear in both subsets). Observe that H_s^1 contains all tasks that can start before the rest period in shift s, whereas H_s^2 contains all tasks that can start after the rest period. Each $(i,t) \in H_s^1$ is associated with a label $l_{(i,t)}^1 = (\tilde{c}_{(i,t)}^1, V_{(i,t)}^1)$, where $V_{(i,t)}^1$ is the list of nodes that are visited before or at time t in the label and $\tilde{c}_{(i,t)}^1$ is the associated cost. Similarly, each $(i,t) \in H_s^2$ is associated with a label $l_{(i,t)}^2 = (\tilde{c}_{(i,t)}^2, V_{(i,t)}^2)$, where $V_{(i,t)}^2$ is the list of nodes that are visited after or at time t in the label and $\tilde{c}_{(i,t)}^2$ is the associated cost.

The algorithm generates $\mathscr{P}_s^*$ in three steps. The first step, which is called forward label correcting, generates the $l_{(i,t)}^1$ with the minimum cost $\tilde{c}_{(i,t)}^1$ for each $(i,t) \in H_s^1$. The second step, which is called backward label correcting, generates the $l_{(i,t)}^2$ with the minimum cost $\tilde{c}_{(i,t)}^2$ for each $(i,t) \in H_s^2$. Finally, $\mathscr{P}_s^*$ is constructed in the last step, which is called label joining and route construction. The details of these steps are presented in Sections EC.2.1, EC.2.2 and EC.2.3, respectively. We prove the correctness of the label correcting algorithm for solving the SPP$_s$ and discuss its time complexity in Proposition EC.1 which is presented in Section EC.2.3.

EC.2.1. Forward Label Correcting

In forward label correcting, for each node $(i,t) \in H_s^1$, we find the minimum-cost (partial) path that ends at this node. To this end, we first sequence all nodes in set H_s^1 in non-decreasing order according to their start times (t), where ties are broken arbitrarily. Let (i_h, t_h), $h = 1, ..., |H_s^1|$ denote the h-th node in the sequence. The details of this procedure are presented in Algorithm EC.1.

Algorithm EC.1 Forward Label Correcting.

Initialization: $l^1_{(i,t)} = (-\delta_{i,t}, [(i,t)])$, $\forall (i,t) \in H_s^1$.
Sequence $(i,t) \in H_s^1$ in non-decreasing order of t.
for $h = 1, ..., |H_s^1| - 1$ **do**
  for $h' = h+1, ..., |H_s^1|$ **do**
    if $i_h \neq i_{h'}$ & $t_h + d_{i_h} + q_{i_h, i_{h'}} \leq t_{h'}$ **then**
      if $\tilde{c}^1_{(i_h,t_h)} - \delta_{i_{h'},t_{h'}} \leq \tilde{c}^1_{(i_{h'},t_{h'})}$ **then**
        $\tilde{c}^1_{(i_{h'},t_{h'})} = \tilde{c}^1_{(i_h,t_h)} - \delta_{i_{h'},t_{h'}}$.
        $V^1_{(i_{h'},t_{h'})} = V^1_{(i_h,t_h)} \leftarrow (i_{h'}, t_{h'})$. ▷ Operator "$\leftarrow$" represents adding an element to the rear of a list.
      end if
    end if
  end for
end for
Return $l^1_{(i,t)}$, $\forall (i,t) \in H_s^1$.

EC.2.2. Backward Label Correcting

In backward label correcting, for each node $(i,t) \in H_s^2$, we find the minimum-cost (partial) path that starts from this node. To this end, we first sequence all nodes in set H_s^2 in non-decreasing order according to their start times (t), where ties are broken arbitrarily. Let (i_h, t_h), $h = 1, ..., |H_s^2|$ denote the h-th node in the sequence. The details of this procedure are presented in Algorithm EC.2.

Algorithm EC.2 Backward Label Correcting.

Initialization: $l^2_{(i,t)} = (-\delta_{i,t}, [(i,t)])$, $\forall (i,t) \in H^2_s$.
Sequence $(i,t) \in H^2_s$ in non-decreasing order of t.
for $h = |H^2_s|, ..., 2$ **do**
    for $h' = h-1, ..., 1$ **do**
        if $i_h \neq i_{h'}$ & $t_{h'} + d_{i_{h'}} + q_{i_{h'}, i_h} \leq t_h$ **then**
            if $\tilde{c}^2_{(i_h,t_h)} - \delta_{i_{h'},t_{h'}} \leq \tilde{c}^2_{(i_{h'},t_{h'})}$ **then**
                $\tilde{c}^2_{(i_{h'},t_{h'})} = \tilde{c}^2_{(i_h,t_h)} - \delta_{i_{h'},t_{h'}}$.
                $V^2_{(i_{h'},t_{h'})} = [(i_{h'}, t_{h'})] \leftarrow V^2_{(i_h,t_h)}$.
            end if
        end if
    end for
end for
Return $l^2_{(i,t)}$, $\forall (i,t) \in H^2_s$.

EC.2.3. Label Joining and Route Construction

In the last step, we identify the pilot route with the minimum cost ($\mathscr{P}^*_s$) by constructing and searching among three sets of pilot routes, including (i) the set of routes that terminate at the rest period, (ii) the set of routes that start from the rest period, and (iii) the set of routes where the rest period serves as an intermediate node. Algorithm EC.3 shows the pseudo-code of this procedure, where we let j denote the rest period.

Algorithm EC.3 Label Joining and Pilot Route Construction.

Initialization: $\tilde{c}_s^* = M$. ▷ M is a sufficiently large constant.
for $(i,t) \in H_s^1$ **do**
  if $\tilde{c}_s^* > \tilde{c}_{(i,t)}^1 + c_s^3$ **then**
    $\tilde{c}_s^* = \tilde{c}_{(i,t)}^1 + c_s^3$.
    $\mathscr{P}_s^* = V_{i,t}^1 \leftarrow (j, \overline{L}_s)$. ▷ $(j, \overline{L}_s)$ represents starting the rest period at time step $\overline{L}_s$.
  end if
end for
for $(i,t) \in H_s^2$ **do**
  if $\tilde{c}_s^* > \tilde{c}_{(i,t)}^2 + c_s^3$ **then**
    $\tilde{c}_s^* = \tilde{c}_{(i,t)}^2 + c_s^3$.
    $\mathscr{P}_s^* = [(j, \underline{L}_s)] \leftarrow V_{i,t}^2$. ▷ $(j, \underline{L}_s)$ represents starting the rest period at time step $\underline{L}_s$.
  end if
end for
for $(i,t) \in H_s^1$ **do**
  for $(i',t') \in H_s^2$ **do**
    if $i \neq i'$ & $t + d_i + q_{i,i'} \leq t'$ **then**
      $t_j = \min\{t + d_i, \underline{L}_s\}$. ▷ Find the earliest feasible start time for the rest period.
      if $t_j + g_s \leq t'$ **then** ▷ Recall that g_s denote the minimum duration for a rest period in shift s.
        if $\tilde{c}_s^* > \tilde{c}_{(i,t)}^1 + \tilde{c}_{(i',t')}^2 + c_s^3$ **then**
          $\tilde{c}_s^* = \tilde{c}_{(i,t)}^1 + \tilde{c}_{(i',t')}^2 + c_s^3$.
          $\mathscr{P}_s^* = V_{(i,t)}^1 \leftarrow (j,t_j) \leftarrow V_{(i',t')}^2$. ▷ (j,t_j) represents starting the rest period at time step t_j.
        end if
      end if
    end if
  end for
end for
Return $\mathscr{P}_s^*$ and $\tilde{c}_s^*$.

PROPOSITION EC.1. *The label correcting algorithm solves the SPP$_s$ in $\mathcal{O}(|H_s|^2)$ time.*

Proof. We first prove the correctness of the algorithm. In the first step of the algorithm, we aim to find, for each node $(i,t) \in H_s^1$, the minimum-cost (partial) path that ends at this node. All nodes in H_s^1 have been sequenced in non-decreasing order of t, and it is only possible for a path to travel from node (i,t) to another node (i',t'), if $t' \geq t$. Therefore, for the first node in the sequence, i.e., (i_1,t_1), the minimum cost and the minimum-cost path have been contained in the label $l_{(i,t)}^1$ after initialization. Next, consider the two-layer for-loop in Algorithm EC.1. Because routes can

only travel through the nodes chronologically, when $h=1$ in the first layer, the algorithm finds the optimal label $l^1_{(i_2,t_2)}$ for node (i_2,t_2) in the second layer. By induction, we can prove that the algorithm finds the best label for node $(i_{\bar{h}+1},t_{\bar{h}+1})$ when $h=\bar{h}$ in the for-loop. This indicates that Algorithm EC.1 can identify the minimum-cost (partial) path that ends at each node $(i,t)\in H^1_s$ after executing the for-loop.

Following a similar logic, one can prove that Algorithm EC.2 generates the minimum-cost (partial) path that starts from each node $(i,t)\in H^2_s$.

Because H^1_s (resp. H^2_s) contains all nodes that can be visited before (resp. after) the rest period starts in a pilot route, by comparing the labels corresponding to all nodes in H^1_s (resp. H^2_s), Algorithm EC.3 finds the pilot route with the minimum cost that terminates at (resp. starts from) the rest period. Finally, by concatenating all compatible pairs of labels corresponding to nodes from H^1_s and H^2_s, Algorithm EC.3 finds the pilot route with the minimum cost where the rest period is an intermediate node.

As for the time complexity of the algorithm, in Algorithms EC.1 and EC.2, it takes $\mathcal{O}(|H_s|\log|H_s|)$ time and $\mathcal{O}(|H_s|^2)$ time, respectively, to sequence the nodes in H^1_s and H^2_s and correct their labels. In Algorithms EC.3, the optimal pilot routes from the three sets are generated in $\mathcal{O}(|H_s|)$, $\mathcal{O}(|H_s|)$, and $\mathcal{O}(|H_s|^2)$ time, respectively. It can be readily seen that the overall time complexity for the label correcting algorithm is $\mathcal{O}(|H_s|^2)$. $\square$

EC.3. The C&BCG Algorithm with Dynamic Constraint Generation

In this section, we describe the C&BCG algorithm with dynamic constraint generation for solving the BMP.

Let Q_1, Q_2, and Q_3 be the sets of all constraints (39), (40), and (41), respectively. The pseudo-code for describing the algorithm is presented in Algorithm EC.4. In the pseudo-code, we let $Q^0_1\subseteq Q_1$, $Q^0_2\subseteq Q_2$, and $Q^0_3\subseteq Q_3$ be sets of initial constraints from (39), (40), and (41), respectively. We also use $\tilde{Q}_1\subseteq Q_1$, $\tilde{Q}_2\subseteq Q_2$, and $\tilde{Q}_3\subseteq Q_3$ to represent the sets of currently used constraints from (39), (40), and (41) in the algorithm, respectively.

For initializing the sets of constraints, we let Q^0_1, Q^0_2, and Q^0_3 include all constraints that have been separated (or generated) so far in the BPBC approach. Specifically, when the C&BCG is run for the first time in the BPBC approach, we set $Q^0_1=\bigcup_{b\in\mathcal{B}}\hat{Q}_{1b}$ and $Q^0_2=Q^0_3=\varnothing$. Here, for each $b\in\mathcal{B}$, $\hat{Q}^0_{1b}=\{(b,t)|t=\underline{t}_b+N\cdot W, t\le\bar{t}_b, N\in\mathbb{Z}^+, t\in\mathcal{T}\}$, where $\underline{t}_b$ and $\bar{t}_b$ represent the earliest and the latest time steps when any vessel can reach berth b, respectively, and $W>1$ is a predetermined step size.

Algorithm EC.4 C&BCG with Dynamic Constraint Generation.

Initialize the set of Benders cuts as $\tilde{\Gamma}_\Delta = \Gamma_\Delta^0$; initialize the sets of constraints from (39), (40), and (41) as $\tilde{Q}_1 = Q_1^0$, $\tilde{Q}_2 = Q_2^0$, and $\tilde{Q}_3 = Q_3^0$, respectively.
while 1 **do** ▷ While-loop for Benders cut generation.
  while 1 **do** ▷ While-loop for separating constraints from (39), (40), and (41).
    Solve the BMP with Benders cuts in $\tilde{\Gamma}_\Delta$ and constraints in $\tilde{Q}_1$, $\tilde{Q}_2$, and $\tilde{Q}_3$ by column generation; let Z_{BMP}^* and $(\boldsymbol{\chi}^*, \eta^*)$ denote the optimal objective function value and the optimal solution to the BMP, respectively.
    Call sub-procedure $(Q_1^+, Q_2^+, Q_3^+) = \mathbf{S}(\boldsymbol{\chi}^*)$ in Algorithm EC.5. ▷ $\mathbf{S}(\boldsymbol{\chi}^*)$ is for constraint separation and Q_1^+, Q_2^+, Q_3^+ represent the sets of constraints newly separated from (39), (40), and (41), respectively.
    if $Q_1^+ = Q_2^+ = Q_3^+ = \varnothing$ **then**
      Break. ▷ $\boldsymbol{\chi}^*$ is feasible with respect to constraints (39), (40), and (41).
    end if
    $\tilde{Q}_1 = \tilde{Q}_1 \cup Q_1^+$; $\tilde{Q}_2 = \tilde{Q}_2 \cup Q_2^+$; $\tilde{Q}_3 = \tilde{Q}_3 \cup Q_3^+$.
  end while
  Let $\bar{\boldsymbol{\chi}} = \boldsymbol{\chi}^*$, and solve the PBSP by column generation; let Z_{PBSP}^* and $\boldsymbol{\mu}^*$ denote the optimal objective function value and the optimal solution to the PBSP, respectively; let $(\boldsymbol{\delta}^*, \boldsymbol{\zeta}^*)$ denote the vectors of the values of the associated optimal dual variables.
  if $Z_{PBSP}^* > \eta^*$ **then**
    Update $\tilde{\Gamma}_\Delta = \tilde{\Gamma}_\Delta \cup (\boldsymbol{\delta}^*, \boldsymbol{\zeta}^*)$.
  else
    Break.
  end if
end while
Return Z_{BMP}^*, $\boldsymbol{\chi}^*$, and $\boldsymbol{\mu}^*$.

The procedures for separating constraints from (39)–(41) are illustrated in Algorithm EC.5. In the pseudo-code, we use variables $o_{b,t}$, $b \in \mathcal{B}, t \in \mathcal{T}$ to record the number of vessels occupying berth b at time step t and variables $\varsigma_{i,t}$, $i \in \mathcal{I}$, $t \in \mathcal{T}$ to record whether task i starts at time step t. Let also $\Omega^+ = \{\omega | \chi_\omega^* > 0, \omega \in \Omega\}$, which represents the set of columns with positive solution values in the optimal solution $\boldsymbol{\chi}^*$. In addition, we make use of the following additional notation in the pseudo-code:

$b(\omega)$: The berth $b \in \mathcal{B}$ used in vessel route ω.

$\mathcal{I}(\omega)$: The set of tasks $i \in \mathcal{I}$ covered in vessel route ω.

$\tau_i(\omega)$: The time step when task $i \in \mathcal{I}(\omega)$ starts in vessel route ω.

$t^{in}(\omega)$: The time step $t \in \mathcal{T}$ when the vessel in ω reaches its berth.

$t^{out}(\omega)$: The time step $t \in \mathcal{T}$ when the vessel in ω leaves its berth.

Algorithm EC.5 Constraint Separation $[\mathbf{S}(\boldsymbol{\chi}^*)]$.

for $t \in \mathcal{T}$ **do** ▷ Initialize $o_{b,t}$ and $\varsigma_{i,t}$.
  for $b \in \mathcal{B}$ **do**
    $o_{b,t} = 0$.
  end for
  for $i \in \mathcal{I}$ **do**
    $\varsigma_{i,t} = 0$.
  end for
end for
for $\omega \in \Omega^+$ **do** ▷ Update $o_{b,t}$ and $\varsigma_{i,t}$.
  for $t \in \mathbb{Z}_{[t^{in}(\omega), t^{out}(\omega)]}$ **do**
    $o_{b(\omega),t} = o_{b(\omega),t} + \chi^*_\omega$.
  end for
  for $i \in \mathcal{I}(\omega)$ **do**
    $\varsigma_{i,\tau_i(\omega)} = \varsigma_{i,\tau_i(\omega)} + \chi^*_\omega$.
  end for
end for
Call sub-procedures $Q_1^+ = \mathbf{S_1}(\boldsymbol{\chi}^*)$, $Q_2^+ = \mathbf{S_2}(\boldsymbol{\chi}^*)$, and $Q_3^+ = \mathbf{S_3}(\boldsymbol{\chi}^*)$. ▷ $\mathbf{S_1}(\boldsymbol{\chi}^*)$, $\mathbf{S_2}(\boldsymbol{\chi}^*)$, and $\mathbf{S_3}(\boldsymbol{\chi}^*)$ separate constraints from (39), (40), and (41), respectively (see Algorithms EC.6–EC.8).
Return Q_1^+, Q_2^+, Q_3^+.

We explain the procedures $\mathbf{S_1}(\boldsymbol{\chi}^*)$, $\mathbf{S_2}(\boldsymbol{\chi}^*)$, and $\mathbf{S_3}(\boldsymbol{\chi}^*)$ in Algorithms EC.6, EC.7, and EC.8, respectively. Note that the separation of constraints (39) is based on the following observation:

Observation EC.1 *An integer solution of the BMP is feasible with respect to constraints* (39) *if and only if it is feasible for any berth at any time step when a vessel reaches or leaves the berth.*

Based on this observation, we separate constraints from (39) only at time steps when a vessel reaches or leaves a berth. Our preliminary experiments showed that this strategy is very efficient.

Algorithm EC.6 Separation of Constraints (39) [$\mathbf{S_1}(\boldsymbol{\chi}^*)$].

$Q_1^+ = \varnothing$. ▷ We check the feasibility of $\boldsymbol{\chi}^*$ with respect to constraints (39) at time steps when a vessel reaches or leaves a berth.
for $\omega \in \Omega^+$ **do**
  if $o_{b(\omega),t^{in}(\omega)} > 1$ **then**
    $Q_1^+ = Q_1^+ \cup \{(b(\omega), t^{in}(\omega))\}$.
  end if
  if $o_{b(\omega),t^{out}(\omega)} > 1$ **then**
    $Q_1^+ = Q_1^+ \cup \{(b(\omega), t^{out}(\omega))\}$.
  end if
end for
Return Q_1^+.

Algorithm EC.7 Separation of Constraints (40) [$\mathbf{S_2}(\boldsymbol{\chi}^*)$].

$Q_2^+ = \varnothing$. ▷ We check the feasibility of $\boldsymbol{\chi}^*$ with respect to constraints (40) at each time step when task j in a task pair $(i,j) \in \mathcal{V}$ starts.
for $\omega \in \Omega^+$ **do**
  for $\omega' \in \Omega^+$ **do**
    for $(i,j) \in \{(i,j) | (i,j) \in \mathcal{V}, i \in \mathcal{I}(\omega), j \in \mathcal{I}(\omega')\}$ **do**
      if $\sum_{t \in \mathbb{Z}_{[\tau_j(\omega') - f_{i,j}+1, \tau_j(\omega')]} \cap \mathcal{E}_i} \varsigma_{i,t} + \varsigma_{j,\tau_j(\omega')} > 1$ **then**
        $Q_2^+ = Q_2^+ \cup \{((i,j), \tau_j(\omega'))\}$.
      end if
    end for
  end for
end for
Return Q_2^+.

Algorithm EC.8 Separation of Constraints (41) [$\mathbf{S_3}(\boldsymbol{\chi}^*)$].

$Q_3^+ = \varnothing$. ▷ We check the feasibility of $\boldsymbol{\chi}^*$ with respect to constraints (41) at each time step when task i in a task pair $(i,j) \in \mathcal{U}$ is active.
for $\omega \in \Omega^+$ **do**
  for $\omega' \in \Omega^+$ **do**
    for $(i,j) \in \{(i,j) | (i,j) \in \mathcal{U}, i \in \mathcal{I}(\omega), j \in \mathcal{I}(\omega')\}$ **do**
      for $t \in \mathbb{Z}_{[\tau_i(\omega), \tau_i(\omega)+d_i-1]}$ **do**
        if $\sum_{t' \in \mathbb{Z}_{[t-d_i+1,t]} \cap \mathcal{E}_i} \varsigma_{i,t'} + \sum_{t' \in \mathbb{Z}_{[t-d_j+1,t]} \cap \mathcal{E}_j} \varsigma_{j,t'} > 1$ **then**
          $Q_3^+ = Q_3^+ \cup \{((i,j),t)\}$.
        end if
      end for
    end for
  end for
end for
Return Q_3^+.

EC.4. The Pricing Problems for the BMP with the LBL cuts

In this section, we explain the pricing problems for the BMP with the LBL cuts.

Let $\boldsymbol{\varpi} = (\varpi_{s,t} | t \in \mathcal{T}_s, s \in \mathcal{S})$ denote vector of dual variables associated with constraints (70). New vessel routes are generated by solving a pricing problem (denoted by $\text{MPP}'_{k,b}$) for each $k \in \mathcal{K}$ and $b \in \mathcal{B}_k$. The $\text{MPP}'_{k,b}$ can be defined as a problem that identifies the arc with the minimum cost in a three-dimensional task-time-shift network $G'_{k,b} = (N'_{k,b}, A'_{k,b})$, where $N'_{k,b}$ and $A'_{k,b}$ denote the sets of nodes and arcs in the network, respectively. Here, $N'_{k,b} = \{(i,t,s) | s \in \mathcal{S}_t, t \in \mathcal{E}_i, i \in \mathcal{I}_k\}$ and $A'_{k,b} = \{[(i,t,s),(i',t',s')] | t' - t \geq d_i + h_{k,b}, (i,t,s),(i',t',s') \in N'_{k,b}, i \in \mathcal{I}^{in}, i \in \mathcal{I}^{out}\}$. The cost of sending a unit flow through arc $[(i,t,s),(i',t',s')]$, denoted by $\hat{c}'_{[(i,t,s),(i',t',s')]}$, is calculated by

$$\hat{c}'_{[(i,t,s),(i',t',s')]} = \hat{c}_{[(i,t),(i',t')]} + \sum_{\tau=t}^{\min\{t+d_i+\underline{q}'_{i,t,s}-1, \overline{T}_s\}} \varpi_{s,\tau} + \sum_{\tau=t'}^{\min\{t'+d_{i'}+\underline{q}'_{i',t',s'}-1, \overline{T}_{s'}\}} \varpi_{s',\tau}, \quad \text{(EC.21)}$$

where $\hat{c}_{[(i,t),(i',t')]}$ is calculated by (58).

EC.5. The Heuristic for Generating Integer Solutions

In this section, we explain the heuristic that tries to construct a feasible integer solution for M2 from a fractional solution. Let $\boldsymbol{\chi}^0$ and $\boldsymbol{\mu}^0$ denote the optimal (fractional) solution to the BMP and PBSP at a node in the branch-and-bound tree, respectively. The heuristic consists of three stages. In the first stage, we try to construct a feasible integer solution (denoted by $\boldsymbol{\chi}^1$) for the BMP (without the Benders cuts (54)). If such a solution can be constructed, then in the second step, we solve the associated PBSP using column generation. Let $\boldsymbol{\mu}^1$ denote the solution to the PBSP. If $\boldsymbol{\mu}^1$ is fractional, we construct an integer solution (denoted by $\boldsymbol{\mu}^2$) based on $\boldsymbol{\mu}^1$ in the last step. Note that if $\boldsymbol{\chi}^0$ is integral then we skip the first stage and let $\boldsymbol{\chi}^1 = \boldsymbol{\chi}^0$ and if $\boldsymbol{\mu}^1$ is integral then we skip the third stage and let $\boldsymbol{\mu}^2 = \boldsymbol{\mu}^1$. Corresponding to each feasible integer solution for M2 generated by the heuristic, we obtain a valid upper bound which equals $\sum_{\omega \in \Omega} \chi^1_\omega \bar{c}_\omega + \sum_{\phi \in \Phi} \mu^2_\phi \bar{c}_\phi$. Details of the first and the third stages in the heuristic are explained as follows.

EC.5.1. Constructing Vessel Routes

Given a fractional solution ($\boldsymbol{\chi}^0$) to the BMP, we apply a greedy algorithm to construct a feasible integer solution ($\boldsymbol{\chi}^1$) for the BMP without the Benders cuts (54). The algorithm leverages information from the current fractional solution. Let $\Omega^+ = \{\omega | \chi^0_\omega > 0, \omega \in \Omega\}$ and let $\mathbb{S}_{\Omega^+}$ denote a sequence which lists elements in Ω^+ in non-decreasing order of χ^0_ω. We let ω_ϱ, $\varrho = 1, 2, ..., |\Omega^+|$ denote the ϱ-th element in $\mathbb{S}_{\Omega^+}$. Constructing $\boldsymbol{\chi}^1$ is equivalent to generating a set $\Omega^1 \subseteq \Omega$ such that $\chi^1_\omega = 1$, $\forall \omega \in \Omega^1$ and $\chi^1_\omega = 0$, $\forall \omega \in \Omega \setminus \Omega^1$. The procedure for constructing Ω^1 is presented in

Algorithm EC.9, and we make use of the following additional notation in the pseudo-code:

$k(\omega)$:	The vessel $k \in \mathcal{K}$ associated with vessel route ω.
$b(\omega)$:	The berth $b \in \mathcal{B}$ used in vessel route ω.
$i^{in}(\omega)$:	The task $i \in \mathcal{I}^{in}$ covered in vessel route ω.
$i^{out}(\omega)$:	The task $i \in \mathcal{I}^{out}$ covered in vessel route ω.
$t_1(\omega)$:	The time step $t \in \mathcal{T}$ when the vessel in ω starts sailing into its berth.
$t_2(\omega)$:	The time step $t \in \mathcal{T}$ when the vessel in ω reaches its berth.
$t_3(\omega)$:	The time step $t \in \mathcal{T}$ when the vessel in ω leaves its berth.
$\bar{\mathcal{K}}$:	Set of vessels that have associated vessel routes in Ω^1.
$\tilde{\mathcal{T}}_i$:	Set of time steps at which task $i \in \mathcal{I}$ cannot start.
$\hat{\mathcal{T}}_b$:	Set of time steps at which berth $b \in \mathcal{B}$ is occupied by vessels.

Algorithm EC.9 Vessel Route Construction.

Initialization: $\Omega^1 = \varnothing$; $\hat{\mathcal{T}}_b = \varnothing,\ b \in \mathcal{B}$; $\tilde{\mathcal{T}}_i = \varnothing,\ i \in \mathcal{I}$.
for $\varrho = 1, 2, ..., |\Omega^+|$ **do**
  if $k(\omega_\varrho) \notin \bar{\mathcal{K}}$ & $t_1(\omega_\varrho) \notin \tilde{\mathcal{T}}_{i^{in}(\omega_\varrho)}$ & $t_3(\omega_\varrho) \notin \tilde{\mathcal{T}}_{i^{out}(\omega_\varrho)}$ & $\mathbb{Z}_{[t_2(\omega_\varrho), t_3(\omega_\varrho)-1]} \cap \hat{\mathcal{T}}_{b(\omega_\varrho)} = \varnothing$ **then**
    $\bar{\mathcal{K}} = \bar{\mathcal{K}} \cup \{k(\omega_\varrho)\}$. ▷ Update the set of vessels that have been included in the solution.
    $\Omega^1 = \Omega^1 \cup \{\omega_\varrho\}$. ▷ Update the set of vessel routes that have been included in the solution.
    Call sub-procedure $\mathbf{P}(\omega_\varrho)$ in Algorithm EC.10. ▷ Update the constraints.
  end if
  if $|\bar{\mathcal{K}}| = |\mathcal{K}|$ **then**
    Break.
  end if
end for
for $k \in \mathcal{K} \setminus \bar{\mathcal{K}}$ **do** ▷ Construct routes for vessels not included in the current solution; k are enumerated chronologically according to their earliest ready times for entering the channel.
  for $t_1 \in \mathcal{E}_{i^{in}}$ **do** ▷ $i^{in} \in \mathcal{I}_k \cap \mathcal{I}^{in}$. t_1 are enumerated chronologically.
    for $t_3 \in \mathcal{E}_{i^{out}}$ **do** ▷ $i^{out} \in \mathcal{I}_k \cap \mathcal{I}^{out}$. t_3 are enumerated chronologically.
      if $t_1 \notin \tilde{\mathcal{T}}_{i^{in}}$ & $t_3 \notin \tilde{\mathcal{T}}_{i^{out}}$ **then**
        for $b \in \mathcal{B}_k$ **do** ▷ b are enumerated in non-increasing order of $c^2_{k,b}$.
          $t_2 = t_1 + d_{i^{in}}$.
          if $t_2 \geq a_b$ & $t_2 + h_{k,b} \leq t_3$ & $\mathbb{Z}_{[t_2, t_3-1]} \cap \hat{\mathcal{T}}_b = \varnothing$ **then**
            Construct a vessel route ω such that $k(\omega) = k$, $b(\omega) = b$, $t_1(\omega) = t_1$, $t_2(\omega) = t_2$, $t_3(\omega) = t_3$, and $\bar{c}_\omega = c^1_{i^{in}, t_1} + c^1_{i^{out}, t_3} + c^2_{k,b}$.
            $\bar{\mathcal{K}} = \bar{\mathcal{K}} \cup \{k\}$.
            $\Omega^1 = \Omega^1 \cup \{\omega\}$.
            Call sub-procedure $\mathbf{P}(\omega)$ in Algorithm EC.10.
            Break and **go to** Line 12.
          end if
        end for
      end if
    end for
  end for
end for
Return Ω^1.

One can easily verify that if $|\bar{\mathcal{K}}| = |\mathcal{K}|$, then by letting $\chi^1_\omega = 1$, $\forall \omega \in \Omega^1$, and $\chi^1_\omega = 0$, $\forall \omega \in \Omega \setminus \Omega^1$, $\boldsymbol{\chi}^1$ becomes a feasible integer solution for the BMP (without Benders cuts). Meanwhile, observe that it is also possible that Algorithm EC.9 fails to generate a feasible integer solution. In this case, the primal heuristic fails to find a valid upper bound for M2.

Algorithm EC.10 Constraint Update [$\mathbf{P}(\omega)$].

1: $\hat{\mathcal{T}}_{b(\omega)} = \hat{\mathcal{T}}_{b(\omega)} \cup \mathbb{Z}_{[t_2(\omega), t_3(\omega)-1]}$. ▷ Update time steps at which the berth is occupied.
2: **for** $(i^{in}(\omega), j) \in \mathcal{V}$ **do** ▷ Update time steps at which a task cannot start due to the headway constraints (40).
3: $\quad \tilde{\mathcal{T}}_j = \tilde{\mathcal{T}}_j \cup \mathbb{Z}_{[t_1(\omega), t_1(\omega)+f_{i^{in}(\omega),j}-1]}$.
4: **end for**
5: **for** $(i^{out}(\omega), j) \in \mathcal{V}$ **do**
6: $\quad \tilde{\mathcal{T}}_j = \tilde{\mathcal{T}}_j \cup \mathbb{Z}_{[t_3(\omega), t_3(\omega)+f_{i^{out}(\omega),j}-1]}$.
7: **end for**
8: **for** $(i^{in}(\omega), j) \in \mathcal{U}$ or $(j, i^{in}(\omega)) \in \mathcal{U}$ **do** ▷ Update time steps at which a task cannot start due to the non-simultaneity constraints (41).
9: $\quad \tilde{\mathcal{T}}_j = \tilde{\mathcal{T}}_j \cup \mathbb{Z}_{[t_1(\omega), t_1(\omega)+d_{i^{in}(\omega)}-1]}$.
10: **end for**
11: **for** $(i^{out}(\omega), j) \in \mathcal{U}$ or $(j, i^{out}(\omega)) \in \mathcal{U}$ **do**
12: $\quad \tilde{\mathcal{T}}_j = \tilde{\mathcal{T}}_j \cup \mathbb{Z}_{[t_3(\omega), t_3(\omega)+d_{i^{out}(\omega)}-1]}$.
13: **end for**
14: Return $\hat{\mathcal{T}}_b,\ b \in \mathcal{B}$, $\tilde{\mathcal{T}}_i,\ i \in \mathcal{I}$.

EC.5.2. Constructing Pilot Routes

Given a feasible integer solution ($\boldsymbol{\chi}^1$) to the BMP, we solve the associated PBSP using column generation. Suppose the delivered solution to the PBSP, denoted by $\boldsymbol{\mu}^1$, is fractional. In this case, to construct a feasible integer solution for the PBSP (denoted by $\boldsymbol{\mu}^2$), we use an algorithm that is similar to the algorithm for generating vessel routes in the previous section. The algorithm can generate a feasible integer solution for the PBSP given any feasible integer solution for the BMP ($\boldsymbol{\chi}^1$). The details are explained as follows.

Constructing $\boldsymbol{\mu}^2$ is equivalent to generating a set $\Phi^2 \subseteq \Phi$ such that $\mu^2_\phi = 1$, $\forall \phi \in \Phi^2$ and $\mu^2_\phi = 0$, $\forall \phi \in \Phi \setminus \Phi^2$. Algorithm EC.11 presents the details of the procedure. In the algorithm, we let $\Phi^+ = \{\phi | \mu^1_\phi > 0, \phi \in \Phi\}$, and let $\mathbb{S}_{\Phi^+}$ denote a sequence that lists the elements in Φ^+ in non-decreasing order of μ^1_ϕ. Besides, let ϕ_ϱ, $\varrho = 1, 2, ..., |\Phi^+|$ denote the ϱ-th element in $\mathbb{S}_{\Phi^+}$. The following notation is also used in the pseudo-code: $s(\phi)$ which denotes the shift associated with pilot route ϕ, $\mathcal{I}(\phi)$ which denotes the set of tasks covered in pilot route ϕ, and $\bar{\mathcal{I}}$ which denotes the set of tasks that have been covered by pilot routes in Φ^2.

Algorithm EC.11 Pilot Route Construction.

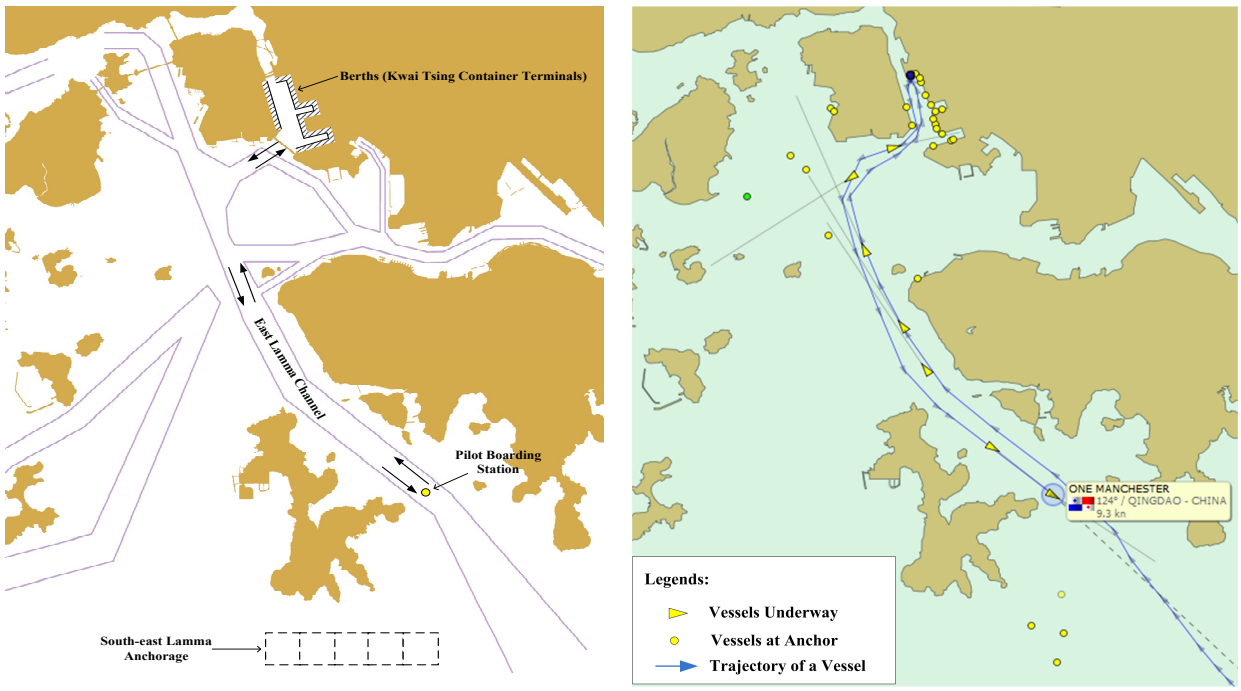
Initialization: $\Phi^2 = \varnothing$; $\bar{\mathcal{I}} = \varnothing$.
for $\varrho = 1, 2, ..., |\Phi^+|$ **do**
  $\mathcal{I}' = \varnothing$.
  for $i \in \mathcal{I}(\phi_\varrho)$ **do**
    if $i \notin \bar{\mathcal{I}}$ **then**
      $\mathcal{I}' = \mathcal{I}' \cup \{i\}$.
    end if
  end for
  if $\mathcal{I}' \neq \varnothing$ **then**
    $\bar{\mathcal{I}} = \bar{\mathcal{I}} \cup \mathcal{I}'$.
    Construct a pilot route $\phi' \in \Phi_{s(\phi)}$ that covers all tasks in $\mathcal{I}'$; ▷ Route ϕ' can be generated by letting all tasks in $\mathcal{I}'$ and the rest period start at the same times as they do in route ϕ_ϱ; The cost of the route equals $\bar{c}_{\phi'} = c^3_{s(\phi)}$.
    $\Phi^2 = \Phi^2 \cup \{\phi'\}$.
  end if
  if $|\bar{\mathcal{I}}| = |\mathcal{I}|$ **then**
    Break.
  end if
end for
Return Φ^2.

EC.6. Data of the Hong Kong Container Port and Parameter Settings in the Instances

EC.6.1. Data of the Port

We generated the instances based on real operational data from the Hong Kong Container Port. The physical layout of the port is depicted in Figure EC.1(a). There are 24 berths in the Kwai Tsing Container Terminals. More than 85% of all vessel arrivals in this port travel through the East Lamma Channel, which is a bidirectional channel with two traffic lanes: one for incoming vessels and the other for outgoing vessels. Due to its sufficient water depth, vessels of all sizes can sail in the channel at any time of the day (i.e., there is no tidal window for vessels in the channel). There is a pilot boarding station at the entrance of the channel where pilots can board or deboard vessels. The port also has an anchorage located near the East Lamma Channel, where arriving vessels can wait before sailing into the channel.

To better explain the movements of vessels in the port, in Figure EC.1(b), we illustrate the locations of vessels in the port at a given time and the trajectory of a vessel that was handled in the port. This figure was adapted from a screenshot of a software application that monitors real-time vessel positions and movements in the Hong Kong Container Port.



(a) Port Layout. (b) Vessel Traffic.

Figure EC.1 Layout of the Hong Kong Container Port and an Illustration of Vessel Traffic in the Port.

In 2017, 10,633 vessel arrivals were handled in the port (The Marine Department of Hong Kong 2018), which translates to approximately 1.21 vessel arrivals per berth per day on average. To derive the vessel service data, we collected messages sent by the onboard Automatic Identification System (AIS) of the vessels handled in the Hong Kong Container Port in 2017. The AIS is to vessels what GPS is to cars. The reader is referred to Yang et al. (2019) for more information on the AIS and AIS messages. By leveraging the AIS data, we obtained 7,062 complete vessel arrival records. Each record depicts the movements of a vessel (accurate to one minute) during its entire port stay. For each of the arrivals, we identified (i) the capacity of the associated vessel, (ii) the handling time at the berths, and (iii) the transit time between the pilot boarding station and the berths.

Table EC.1 reports the distribution of the vessel capacities associated with the 7,062 vessel arrivals. We classify vessels into five groups according to their capacities (in twenty-foot equivalent units, or TEU): feeder, intermediate (S), intermediate (L), large, and jumbo. We present the distribution of handling times (in hours) of different vessels in Figure EC.2. Among the 7,062 vessel arrivals, 6,425 and 5,655 arrivals entered and departed the port through the East Lamma Channel, respectively. The average traveling time between the berths and the pilot boarding station in the channel was 64 minutes, and nearly 80% of transit times lay within the range of [54,72] minutes.

Table EC.1 Distribution of Vessel Capacities.

Type	Size (TEU)	Number	Ratio (%)
Feeder	[500, 3000]	3815	54%
Intermediate (S)	[3001, 6000]	1710	24%
Intermediate (L)	[6001, 9000]	988	14%
Large	[9001, 12000]	229	3%
Jumbo	[12001, 18500]	320	5%

Pilotage is compulsory in Hong Kong; all vessels over 3,000 gross tons must have a pilot on board when navigating in the port (the smallest container vessel in our record is over 8,900 gross tons). There are approximately 100 pilots licensed to serve vessels in Hong Kong (The Marine Department of Hong Kong 2020). To board vessels, pilots travel in the port using pilot boats. Pilot boats are significantly smaller than container vessels; the typical length of a pilot boat is between 12 and 14 meters, whereas the smallest feeder container vessel in our record is 138 meters long. While the speed of container vessels in the channel is normally between 10–15 knots, pilot boats can travel at much higher speeds (up to a maximum of 30 knots) in port waters. Accordingly, pilot boats can overtake container vessels when sailing in the channel. A pilot boat can travel between any two berths in the port within 5 minutes and between berths and the pilot boarding station within 20 minutes.

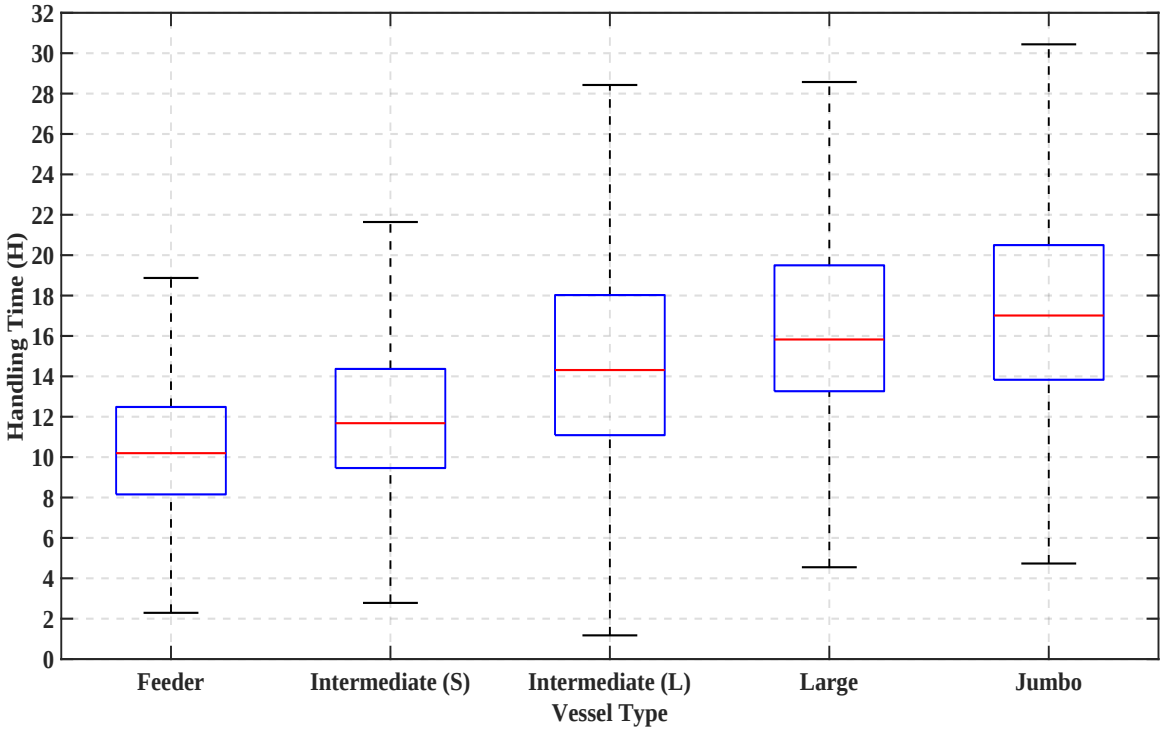


Figure EC.2 Distribution of Vessel Handling Times.

EC.6.2. Parameter Settings in the Instances

Based on the operational data of the Hong Kong Container Port, we generate parameters in our testing instances as follows.

First, corresponding to the five vessel types listed in Table EC.1, we classify vessels in the instances into five grades: Grades I, II, III, IV, and V. A higher grade indicates larger capacities. Among all vessels calling at the seaport in an instance, the probabilities of the vessels being of Grades I, II, III, IV, and V are set to 54%, 24%, 14%, 3%, and 5%, respectively. Vessels arrive at the seaport dynamically and uniformly throughout the planning horizon. In the instances, a vessel can be handled at any berth in the seaport, and the cargo handling times for vessel k at all berths are set to be identical and equal to $\hat{h}_k$. Here, $\hat{h}_k$ is randomly generated using the uniform distributions $U(8,12)$, $U(10,14)$, $U(11,18)$, $U(13,20)$, and $U(14,21)$ (in hours) for vessels of Grades I, II, III, IV, and V, respectively.

Second, requirements regarding channel traffic management in the instances are set as follows. The time for a vessel to travel between the anchorage and berths (including sailing in the channel and berthing into or unberthing from a berth), denoted by $\hat{d}$, is set to 70 minutes. There should be a minimum headway between any two vessels that sail in the same direction in the channel. Meanwhile, large-sized vessels cannot sail in opposite directions in the channel simultaneously.

However, the exact minimum headway and non-simultaneity requirements in the Hong Kong Container Port are not available to us. In the instances, we set the minimum headway to be 10 minutes, which is in accordance with the setting used by Jia et al. (2019). We also stipulate that vessels of Grade IV or V cannot sail in opposite directions in the channel simultaneously.

Third, the time windows of the pilotage tasks are set in such a way as to fulfill the following two requirements: (i) any incoming vessel should start sailing into the berths no more than eight hours after arrival, and (ii) the total turnaround time of vessel k should not exceed the sum of the sailing time between the anchorage and the berths and the handling time $(2\hat{d} + \hat{h}_k)$, plus a total buffer time of 16 hours.

Fourth, given that seaports operate around the clock, we simulate the initial condition in the seaport as follows. We set the probability of a berth being occupied by a vessel to $0.5R$ (R denotes the vessel arrival rate [vessel arrivals per berth per day]). For an occupied berth, the probabilities of it being occupied by a Grade I, II, III, IV, and V vessel are set to 54%, 24%, 14%, 3%, and 5%, respectively. If a berth is occupied, its available time (counting from the beginning of the planning horizon) is generated from $U(1, 10)$ (in hours); otherwise, we set the available time to zero. Meanwhile, to impose the minimum headway requirement, no vessels in the instances can leave their berths within the first 10 minutes after an initially occupied berth becomes available. Moreover, to impose the non-simultaneity requirements between large vessels, if a berth is occupied by a Grade IV or V vessel, no Grade IV or V vessels in the instances can start sailing into the berths within the first 70 minutes after the berth becomes available. To impose these requirements, the time windows of the pilotage tasks for the vessels are adjusted accordingly.

Fifth, in each instance, each pilot working in a shift is awarded a rest period that should last at least an hour and cannot start within the first 30% or the last 30% period in the shift. A pilot can maneuver multiple vessels in a shift, and a minimum setup time is required to reposition a pilot between two tasks. If the repositioning does not necessitate travel through the channel, we set the minimum setup time to 10 minutes; otherwise, we set the minimum setup time to 30 minutes.

Sixth, because we consider a discrete-time problem, the time points (for defining various time windows) and time durations (of tasks, rest periods, and vessel handling) generated in continuous time are translated into parameters depicted by discrete time steps. Such translation is conducted by applying the appropriate ceiling or flooring operators on the continuous-time parameters to ensure the feasibility of the obtained discrete-time solutions for the original continuous-time instances.

Finally, we set the cost components as follows. First, tasks that start later than their earliest possible start times are punished. The unit-time delay cost for tasks corresponding to vessels of

Grades I, II, III, IV, and V sailing into berths are set to 2, 3, 4, 5, and 6, respectively. The unit-time delay cost for tasks corresponding to vessels of Grades I, II, III, IV, and V leaving berths are set to 4, 6, 8, 10, and 12, respectively. Meanwhile, the cost for handling vessel k at berth b is set to $c^2_{k,b} = \lceil 5e_{b,k}\hat{h}_k \rceil$, where $e_{b,k}$ is generated from $U(1, 1.5)$. Finally, the cost paid by the port authority for assigning a pilot to work in a shift is set to 48 (i.e., 1 per unit time).

EC.7. Additional Computational Results

In this section, we report the additional computational results associated with Sections 5.5 and 5.6 in the main text of the paper.

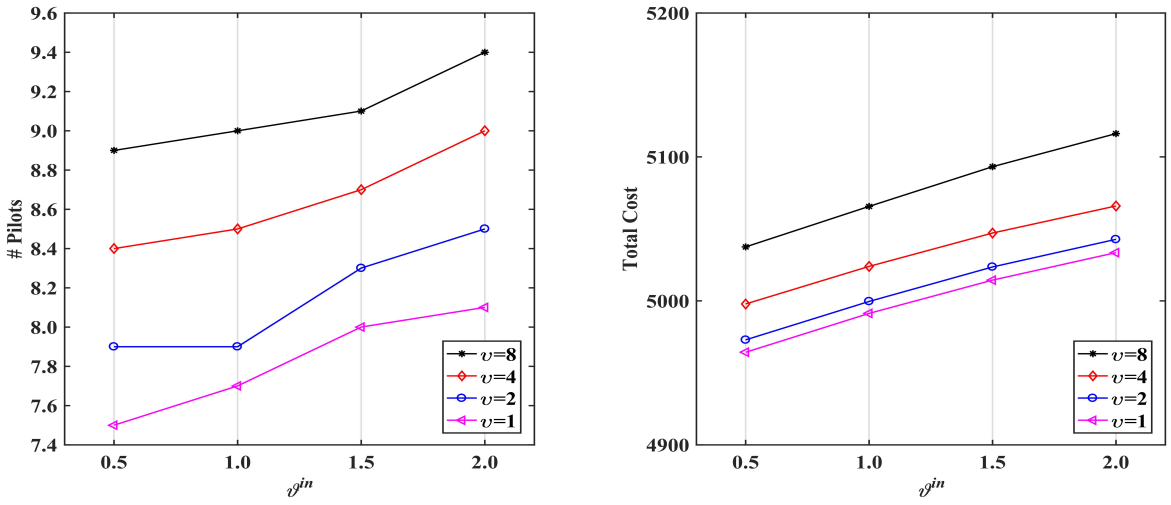
EC.7.1. Results of Sensitivity Analyses

This subsection presents the detailed computational results in the sensitivity analyses.

EC.7.1.1. Impacts of Varying Penalty Costs for Cargo Dispatch Delays The computational results on the instances with different shift arrangements and cargo dispatch delay costs are shown in Table EC.2. In this table, υ denotes the interval (hours) between the start times of two consecutive shifts. The second column shows the settings of (L, B, V) in a group of five instances. For each group of instances under each setting (with respect to υ and the cost modifier ϑ^{in}), we report the number of instances solved to optimality, the average optimality gap (in percentage terms), and the average computational time (in CPU seconds) obtained by the BPBC approach in the corresponding columns *Opt*, *Gap*, and *CPU*, respectively. To further examine the impacts of varying the settings of υ and ϑ^{in} on the solution structures, we compare the average numbers of working pilots and the average objective function values of the instances under different settings. These results are shown in Figure EC.3.

Table EC.2 Computational Results on Instances with Different Cargo Dispatch Delay Costs.

		$\vartheta^{in}=0.5$			$\vartheta^{in}=1.0$			$\vartheta^{in}=1.5$			$\vartheta^{in}=2.0$		
	(L,B,V)	Opt	Gap	CPU	Opt	Gap	CPU	Opt	Gap	CPU	Opt	Gap	CPU
$\upsilon=1$	$(1,10,10)$	5	0.0	49.0	5	0.0	24.4	5	0.0	18.0	5	0.0	19.0
	$(1,10,12)$	5	0.0	114.5	5	0.0	74.8	5	0.0	42.4	5	0.0	33.6
	$(1,10,14)$	5	0.0	313.6	5	0.0	208.4	5	0.0	113.8	5	0.0	84.3
	Total	15	0.0	159.0	15	0.0	102.5	15	0.0	58.1	15	0.0	45.6
$\upsilon=2$	$(1,10,10)$	5	0.0	12.3	5	0.0	7.8	5	0.0	8.1	5	0.0	6.9
	$(1,10,12)$	5	0.0	21.5	5	0.0	12.0	5	0.0	11.9	5	0.0	8.5
	$(1,10,14)$	5	0.0	76.1	5	0.0	52.8	5	0.0	46.9	5	0.0	31.0
	Total	15	0.0	36.7	15	0.0	24.2	15	0.0	22.3	15	0.0	15.5
$\upsilon=4$	$(1,10,10)$	5	0.0	13.8	5	0.0	9.6	5	0.0	8.0	5	0.0	6.5
	$(1,10,12)$	5	0.0	19.9	5	0.0	10.6	5	0.0	6.4	5	0.0	4.7
	$(1,10,14)$	5	0.0	22.8	5	0.0	17.9	5	0.0	12.5	5	0.0	9.1
	Total	15	0.0	18.8	15	0.0	12.7	15	0.0	9.0	15	0.0	6.8
$\upsilon=8$	$(1,10,10)$	5	0.0	9.2	5	0.0	5.0	5	0.0	5.3	5	0.0	4.1
	$(1,10,12)$	5	0.0	5.0	5	0.0	2.9	5	0.0	2.6	5	0.0	2.1
	$(1,10,14)$	5	0.0	31.2	5	0.0	14.9	5	0.0	13.0	5	0.0	9.6
	Total	15	0.0	15.2	15	0.0	7.6	15	0.0	7.0	15	0.0	5.3



(a) Number of Working Pilots. (b) Objective Function Value.

Figure EC.3 Average Number of Working Pilots and Objective Function Value Under Different Cargo Dispatch Delay Costs.

We can see from Table EC.2 that the solution times decrease when υ increases. This is because a smaller υ represents more pricing problems to be solved in the column generation for solving the PBSP and a higher probability of generating fractional solutions for the PBSP. The major trade-off in the VSPP is between the delays in vessel movements (related to decisions in the BMP) and the costs in pilot dispatching (related to decisions in the PBSP). When ϑ^{in} increases, more weights are put in the objective function value of the BMP (against the objective function value of the PBSP). In this case, the trade-off between the two objectives becomes easier and the Benders

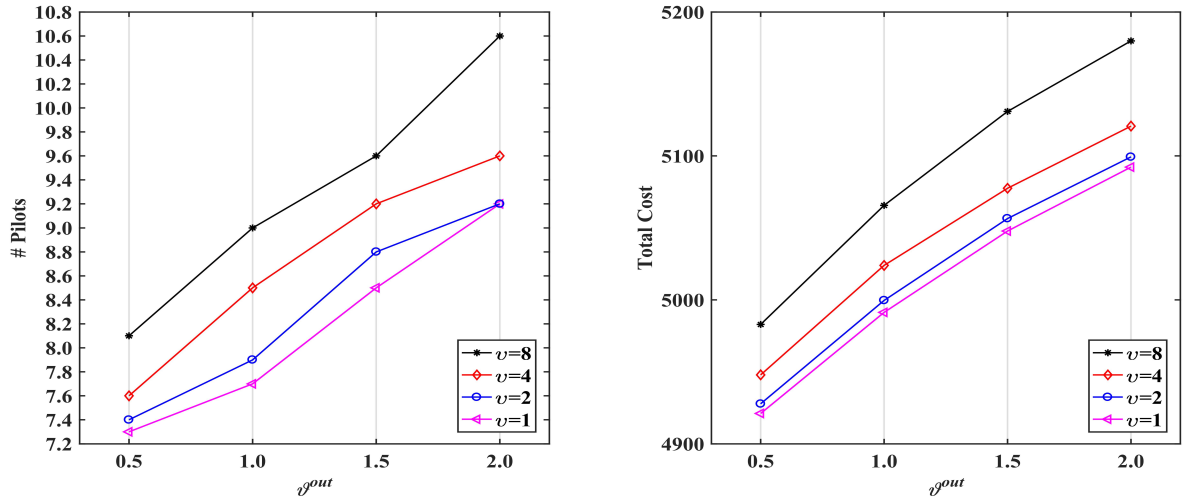
decomposition for solving the BMP can converge more quickly. As a result, the solution times decrease with the growth in ϑ^{in}.

From Figure EC.3, we can find that more pilots are assigned to work if a seaport values more fast cargo dispatch and that flexible shift arrangements lead to fewer pilots on-duty, which reduces the total costs.

EC.7.1.2. Impacts of Varying Penalty Costs of Vessel Turnaround Delays We further examined the impacts of varying the costs associated with vessel turnaround delays on the performance of the BPBC approach and the solution structures. The computational results are reported in Table EC.3, where ϑ^{out} denotes the values of the modifier applied to the relevant costs. The changes of solution structures caused by varying ϑ^{out} are demonstrated in Figure EC.4.

Table EC.3 Computational Results on Instances with Different Vessel Turnaround Delay Costs.

		$\vartheta^{out}=0.5$			$\vartheta^{out}=1.0$			$\vartheta^{out}=1.5$			$\vartheta^{out}=2.0$		
	(L,B,V)	Opt	Gap	CPU	Opt	Gap	CPU	Opt	Gap	CPU	Opt	Gap	CPU
$\upsilon=1$	$(1,10,10)$	5	0.0	339.9	5	0.0	24.4	5	0.0	21.1	5	0.0	9.1
	$(1,10,12)$	4	0.0	1202.9	5	0.0	74.8	5	0.0	30.3	5	0.0	17.7
	$(1,10,14)$	2	0.2	2244.2	5	0.0	208.4	5	0.0	67.6	5	0.0	38.6
	Total	11	0.1	1262.3	15	0.0	102.5	15	0.0	39.7	15	0.0	21.8
$\upsilon=2$	$(1,10,10)$	5	0.0	175.7	5	0.0	7.8	5	0.0	5.6	5	0.0	2.4
	$(1,10,12)$	5	0.0	367.8	5	0.0	12.0	5	0.0	7.9	5	0.0	3.8
	$(1,10,14)$	4	0.1	2099.6	5	0.0	52.8	5	0.0	14.9	5	0.0	7.1
	Total	14	0.0	881.0	15	0.0	24.2	15	0.0	9.5	15	0.0	4.4
$\upsilon=4$	$(1,10,10)$	5	0.0	186.4	5	0.0	9.6	5	0.0	3.4	5	0.0	1.9
	$(1,10,12)$	5	0.0	161.9	5	0.0	10.6	5	0.0	3.4	5	0.0	2.0
	$(1,10,14)$	5	0.0	226.3	5	0.0	17.9	5	0.0	5.0	5	0.0	2.1
	Total	15	0.0	191.5	15	0.0	12.7	15	0.0	3.9	15	0.0	2.0
$\upsilon=8$	$(1,10,10)$	5	0.0	127.0	5	0.0	5.0	5	0.0	2.2	5	0.0	1.1
	$(1,10,12)$	5	0.0	43.0	5	0.0	2.9	5	0.0	2.1	5	0.0	1.0
	$(1,10,14)$	5	0.0	640.8	5	0.0	14.9	5	0.0	7.5	5	0.0	5.5
	Total	15	0.0	270.3	15	0.0	7.6	15	0.0	3.9	15	0.0	2.5



(a) Number of Working Pilots. (b) Objective Function Value.

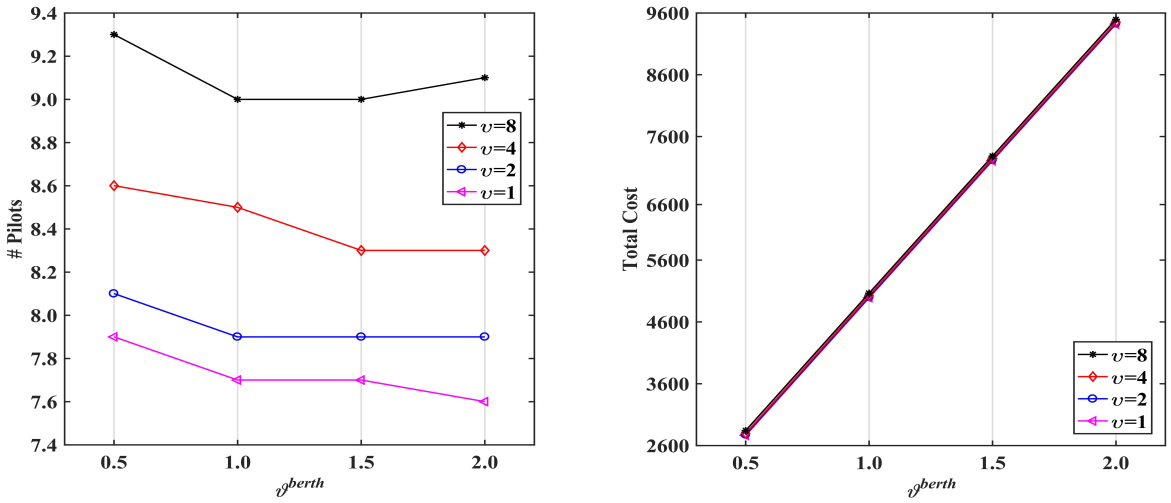
Figure EC.4 Average Number of Working Pilots and Objective Function Value Under Different Vessel Turnaround Delay Costs.

Table EC.3 demonstrates that the solution times increased dramatically when ϑ^{out} was set to 0.5. This is because a small ϑ^{out} complicates the trade-off between the delays in vessel movements and the costs in pilot dispatching. In particular, reducing ϑ^{out} makes the objective function value of the BMP less important in the overall objective function value. This hinders the convergence of the Benders decomposition for the BMP, resulting in longer solution times. Besides, in the instances, we put relatively higher penalties on delays in vessel turnaround than on delays in cargo dispatch. Hence, compared with the value of ϑ^{in}, the value of ϑ^{out} has a greater impact on the objective function value of the BMP and thus also has a greater impact on the solution procedure. Figure EC.4 presents similar patterns as in Figure EC.3.

EC.7.1.3. Impacts of Varying Berth Handling Costs In this part, we analyze the impacts of changing berth handling costs on the performance of the BPBC approach and the solution structures. The corresponding computational results are presented in Table EC.4, where ϑ^{berth} indicates the values of the modifier applied to the relevant costs. The changes in solution structures are presented in Figure EC.5.

Table EC.4 Computational Results on Instances with Different Berth Handling Costs.

		$\vartheta^{berth}=0.5$			$\vartheta^{berth}=1.0$			$\vartheta^{berth}=1.5$			$\vartheta^{berth}=2.0$		
	(L,B,V)	Opt	Gap	CPU	Opt	Gap	CPU	Opt	Gap	CPU	Opt	Gap	CPU
$v=1$	$(1,10,10)$	5	0.0	21.0	5	0.0	24.4	5	0.0	24.7	5	0.0	11.6
	$(1,10,12)$	5	0.0	69.4	5	0.0	74.8	5	0.0	46.6	5	0.0	83.3
	$(1,10,14)$	5	0.0	220.8	5	0.0	208.4	5	0.0	141.8	5	0.0	307.7
	Total	15	0.0	103.7	15	0.0	102.5	15	0.0	71.1	15	0.0	134.2
$v=2$	$(1,10,10)$	5	0.0	11.4	5	0.0	7.8	5	0.0	7.2	5	0.0	10.0
	$(1,10,12)$	5	0.0	14.2	5	0.0	12.0	5	0.0	9.9	5	0.0	16.4
	$(1,10,14)$	5	0.0	52.0	5	0.0	52.8	5	0.0	50.1	5	0.0	49.7
	Total	15	0.0	25.9	15	0.0	24.2	15	0.0	22.4	15	0.0	25.4
$v=4$	$(1,10,10)$	5	0.0	10.6	5	0.0	9.6	5	0.0	7.1	5	0.0	6.6
	$(1,10,12)$	5	0.0	8.4	5	0.0	10.6	5	0.0	8.6	5	0.0	13.0
	$(1,10,14)$	5	0.0	18.2	5	0.0	17.9	5	0.0	14.4	5	0.0	11.3
	Total	15	0.0	12.4	15	0.0	12.7	15	0.0	10.1	15	0.0	10.3
$v=8$	$(1,10,10)$	5	0.0	5.9	5	0.0	5.0	5	0.0	4.9	5	0.0	3.9
	$(1,10,12)$	5	0.0	2.3	5	0.0	2.9	5	0.0	2.7	5	0.0	3.5
	$(1,10,14)$	5	0.0	14.1	5	0.0	14.9	5	0.0	20.3	5	0.0	27.4
	Total	15	0.0	7.4	15	0.0	7.6	15	0.0	9.3	15	0.0	11.6



(a) Number of Working Pilots. (b) Objective Function Value.

Figure EC.5 Average Number of Working Pilots and Objective Function Value Under Different Berth Handling Costs.

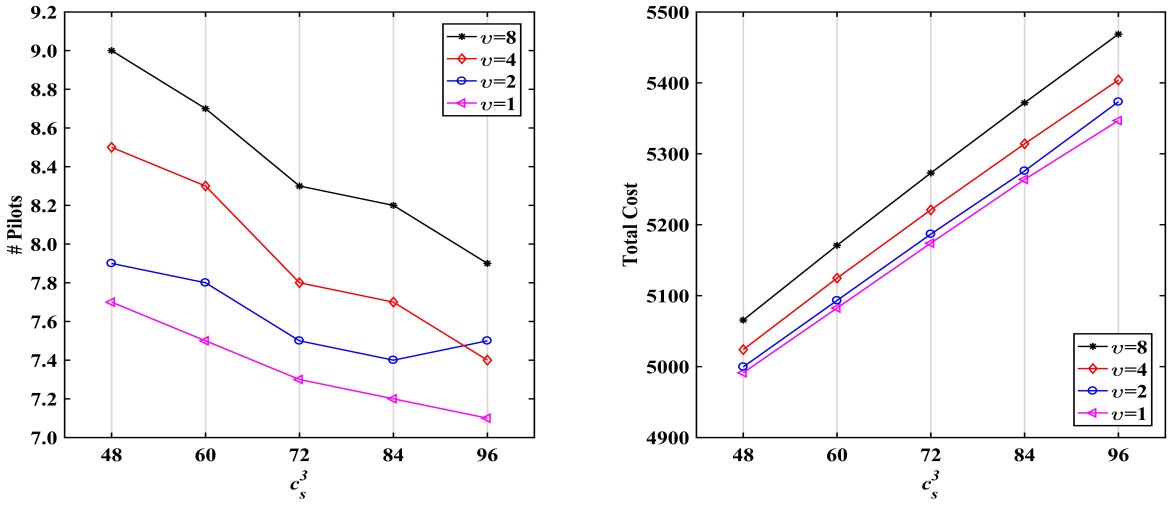
We can see from Table EC.4 that the changes in berth handling costs did not have obvious impacts on the performance of the approach. This can be explained by the fact that the main difficulty for solving the VSPP comes from the trade-off between the timing decisions of vessel movements and pilot dispatching decisions and thus different berth handling costs have very limited impacts on the performance of the approach. Figure EC.5 demonstrates that there are increases in the numbers of working pilots when berth handling costs are lower. This is possibly caused by the fact that seaports are less likely to let vessels wait for particular berths because of lower handling

costs when these costs are less important and hence more pilots are required to move the vessels more quickly. As shown in Figure EC.5(b), the value of ϑ^{berth} greatly affects the objective function values such that the total costs grow almost linearly with the value of ϑ^{berth}. Note that in order to plot all data points in one figure, the range of the y-axis in Figure EC.5(b) is set between 2,600 and 9,600, and the numbers are marked at intervals of 1,000. Meanwhile, in other similar figures (i.e., EC.3(b), EC.4(b), and EC.6(b)), the spans of y-axes are within 600, and the numbers are marked at intervals of 100. As a result, while the numerical differences in the objective function values between different settings of v in Figure EC.5(b) are similar to those in Figures EC.3(b), EC.4(b), and EC.6(b), they are not as visually noticeable as in those figures with finer scales.

EC.7.1.4. Impacts of Varying Pilot Dispatching Costs The last part of the sensitivity analyses focuses on the impacts of varying pilot dispatching costs. Table EC.5 and Figure EC.6 present the computational results and the solution structures obtained in the tests, respectively.

Table EC.5 Computational Results on Instances with Different Pilot Dispatching Costs.

		$c_s^3=48$			$c_s^3=60$			$c_s^3=72$			$c_s^3=84$			$c_s^3=96$		
	(L,B,V)	Opt	Gap	CPU	Opt	Gap	CPU	Opt	Gap	CPU	Opt	Gap	CPU	Opt	Gap	CPU
$v=1$	$(1,10,10)$	5	0.0	24.4	5	0.0	37.9	5	0.0	275.5	5	0.0	707.1	3	0.2	1613.2
	$(1,10,12)$	5	0.0	74.8	5	0.0	460.6	5	0.0	539.5	4	0.1	1265.3	3	0.2	2065.3
	$(1,10,14)$	5	0.0	208.4	5	0.0	474.9	5	0.0	1084.9	2	0.3	2354.5	3	0.3	2424.6
	Total	15	0.0	102.5	15	0.0	324.5	15	0.0	633.3	11	0.1	1442.3	9	0.2	2034.3
$v=2$	$(1,10,10)$	5	0.0	7.8	5	0.0	17.3	5	0.0	52.4	5	0.0	211.1	4	0.0	1194.9
	$(1,10,12)$	5	0.0	12.0	5	0.0	33.7	5	0.0	140.7	5	0.0	814.4	4	0.1	1768.3
	$(1,10,14)$	5	0.0	52.8	5	0.0	277.8	4	0.1	1124.7	3	0.2	1832.5	1	0.7	2882.9
	Total	15	0.0	24.2	15	0.0	109.6	14	0.0	439.3	13	0.1	952.7	9	0.3	1948.7
$v=4$	$(1,10,10)$	5	0.0	9.6	5	0.0	30.0	5	0.0	112.3	5	0.0	363.3	4	0.0	1462.0
	$(1,10,12)$	5	0.0	10.6	5	0.0	27.8	5	0.0	85.8	5	0.0	360.2	5	0.0	1349.3
	$(1,10,14)$	5	0.0	17.9	5	0.0	43.2	5	0.0	222.5	5	0.0	640.1	4	0.1	1377.0
	Total	15	0.0	12.7	15	0.0	33.7	15	0.0	140.2	15	0.0	454.5	14	0.0	1396.1
$v=8$	$(1,10,10)$	5	0.0	5.0	5	0.0	15.3	5	0.0	57.3	5	0.0	154.6	5	0.0	355.4
	$(1,10,12)$	5	0.0	2.9	5	0.0	7.6	5	0.0	19.4	5	0.0	76.1	5	0.0	422.4
	$(1,10,14)$	5	0.0	14.9	5	0.0	30.3	5	0.0	122.0	5	0.0	435.6	5	0.0	1272.5
	Total	15	0.0	7.6	15	0.0	17.7	15	0.0	66.3	15	0.0	222.1	15	0.0	683.4



(a) Number of Working Pilots. (b) Objective Function Value.

Figure EC.6 Average Number of Working Pilots and Objective Function Value Under Different Pilot Dispatching Costs.

The results in Table EC.5 demonstrate that the solution times grow with the dispatching costs. This is because the Benders decomposition algorithm converges slower when the decisions in the subproblem have a greater impact on the overall objective function value. We can also see from Figure EC.6 that in general, when the pilot dispatching costs increase, fewer pilots are assigned to serve the vessels. However, because our approach could not deliver optimal solutions for some instances with $\upsilon = 2$ and $c_s^3 = 96$, there is an abnormal increase in the average number of working pilots under this setting.

EC.7.2. Results of the Tests for the Value of Integration

Tables EC.6–EC.9 compare the results obtained by the integrated optimization method (the BPBC approach) and the sequential optimization method in the tests for evaluating the value of integration in the VSPP. Columns *#P(S)* show the average numbers of working pilots in the solutions generated by the sequential optimization method. The average numbers of working pilots in the solutions generated by the BPBC approach are presented in columns *#P(I)*. For a certain instance under a certain setting, let ov_S and ov_I be the objective function values associated with the solutions delivered by the sequential optimization method and the BPBC approach, respectively. We calculated the percentage gap between ov_S and ov_I by $100(ov_S - ov_I)/ov_S$. Columns *Gap(%)* report the average of these gaps for each instance group under each setting.

Table EC.6 Value of Integration Under Different Cargo Dispatch Delay Costs.

		$\vartheta^{in}=0.5$			$\vartheta^{in}=1.0$			$\vartheta^{in}=1.5$			$\vartheta^{in}=2.0$		
	(L,B,V)	#P(S)	#P(I)	Gap(%)	#P(S)	#P(I)	Gap(%)	#P(S)	#P(I)	Gap(%)	#P(S)	#P(I)	Gap(%)
$v=1$	$(1,10,10)$	9.8	6.8	1.9	9.8	6.8	1.7	10.0	7.2	1.8	10.0	7.2	1.7
	$(1,10,12)$	10.4	7.6	1.6	10.6	8.0	1.6	10.6	8.0	1.5	10.6	8.0	1.4
	$(1,10,14)$	11.6	8.2	1.6	11.8	8.2	1.5	11.8	8.8	1.3	11.8	9.0	1.3
	Total	10.6	7.5	1.7	10.7	7.7	1.6	10.8	8.0	1.6	10.8	8.1	1.5
$v=2$	$(1,10,10)$	9.8	7.0	1.8	9.8	7.0	1.6	10.0	7.6	1.7	10.0	7.8	1.6
	$(1,10,12)$	10.8	8.0	1.8	10.8	8.0	1.7	10.8	8.2	1.6	10.8	8.2	1.5
	$(1,10,14)$	12.2	8.6	1.9	12.4	8.6	1.8	12.4	9.0	1.6	12.4	9.6	1.5
	Total	10.9	7.9	1.8	11.0	7.9	1.7	11.1	8.3	1.6	11.1	8.5	1.5
$v=4$	$(1,10,10)$	10.2	7.8	1.6	10.2	8.0	1.5	10.4	8.0	1.6	10.4	8.4	1.5
	$(1,10,12)$	11.4	8.6	1.7	11.4	8.6	1.6	11.4	8.8	1.4	11.4	8.8	1.3
	$(1,10,14)$	12.6	8.8	2.0	12.8	8.8	1.9	12.8	9.2	1.7	12.8	9.8	1.6
	Total	11.4	8.4	1.8	11.5	8.5	1.6	11.5	8.7	1.6	11.5	9.0	1.5
$v=8$	$(1,10,10)$	11.2	7.8	2.2	11.2	8.0	2.0	11.4	8.0	2.1	11.4	8.2	1.9
	$(1,10,12)$	12.0	8.8	1.9	12.2	8.8	1.9	12.4	8.8	1.8	12.4	9.2	1.7
	$(1,10,14)$	14.8	10.2	2.3	15.0	10.2	2.3	15.0	10.4	2.0	15.0	10.8	1.8
	Total	12.7	8.9	2.1	12.8	9.0	2.0	12.9	9.1	2.0	12.9	9.4	1.8

Table EC.7 Value of Integration Under Different Vessel Turnaround Delay Costs.

		$\vartheta^{out}=0.5$			$\vartheta^{out}=1.0$			$\vartheta^{out}=1.5$			$\vartheta^{out}=2.0$		
	(L,B,V)	#P(S)	#P(I)	Gap(%)	#P(S)	#P(I)	Gap(%)	#P(S)	#P(I)	Gap(%)	#P(S)	#P(I)	Gap(%)
$v=1$	$(1,10,10)$	10.0	6.6	2.7	9.8	6.8	1.7	9.8	7.8	1.2	9.8	8.2	0.9
	$(1,10,12)$	10.4	7.2	2.0	10.6	8.0	1.6	10.6	8.0	1.1	10.6	9.2	0.7
	$(1,10,14)$	11.4	8.2	1.8	11.8	8.2	1.5	11.8	9.6	1.1	11.8	10.2	0.9
	Total	10.6	7.3	2.2	10.7	7.7	1.6	10.7	8.5	1.1	10.7	9.2	0.8
$v=2$	$(1,10,10)$	10.0	6.6	2.7	9.8	7.0	1.6	9.8	8.2	1.1	9.8	8.2	0.9
	$(1,10,12)$	10.8	7.4	2.2	10.8	8.0	1.7	10.8	8.4	1.2	10.8	9.2	0.9
	$(1,10,14)$	11.8	8.2	2.0	12.4	8.6	1.8	12.4	9.8	1.2	12.4	10.2	1.0
	Total	10.9	7.4	2.3	11.0	7.9	1.7	11.0	8.8	1.2	11.0	9.2	0.9
$v=4$	$(1,10,10)$	10.4	7.0	2.4	10.2	8.0	1.5	10.2	8.6	1.1	10.2	8.6	0.9
	$(1,10,12)$	11.4	7.6	2.3	11.4	8.6	1.6	11.4	9.0	1.0	11.4	9.8	0.6
	$(1,10,14)$	12.4	8.2	2.4	12.8	8.8	1.9	12.8	10.0	1.4	12.8	10.4	1.2
	Total	11.4	7.6	2.4	11.5	8.5	1.6	11.5	9.2	1.2	11.5	9.6	0.9
$v=8$	$(1,10,10)$	11.4	7.4	3.1	11.2	8.0	2.0	11.2	8.2	1.4	11.2	9.4	1.2
	$(1,10,12)$	12.0	7.8	2.5	12.2	8.8	1.9	12.4	9.4	1.4	12.4	9.8	1.0
	$(1,10,14)$	14.4	9.2	2.7	15.0	10.2	2.3	15.0	11.2	1.5	15.0	12.6	1.1
	Total	12.6	8.1	2.8	12.8	9.0	2.0	12.9	9.6	1.4	12.9	10.6	1.1

Table EC.8 Value of Integration Under Different Berth Handling Costs.

		$\vartheta^{berth}=0.5$			$\vartheta^{berth}=1.0$			$\vartheta^{berth}=1.5$			$\vartheta^{berth}=2.0$		
	(L,B,V)	#P(S)	#P(I)	Gap(%)	#P(S)	#P(I)	Gap(%)	#P(S)	#P(I)	Gap(%)	#P(S)	#P(I)	Gap(%)
$v=1$	$(1,10,10)$	9.8	6.8	3.1	9.8	6.8	1.7	9.8	6.8	1.2	9.6	6.8	0.9
	$(1,10,12)$	10.6	8.0	2.7	10.6	8.0	1.6	10.4	8.0	1.0	10.4	7.8	0.8
	$(1,10,14)$	11.8	8.8	2.4	11.8	8.2	1.5	11.4	8.2	0.9	11.4	8.2	0.6
	Total	10.7	7.9	2.7	10.7	7.7	1.6	10.5	7.7	1.1	10.5	7.6	0.8
$v=2$	$(1,10,10)$	9.8	7.0	2.8	9.8	7.0	1.6	9.8	7.0	1.1	9.6	7.0	0.8
	$(1,10,12)$	10.8	8.0	2.8	10.8	8.0	1.7	10.8	8.0	1.2	10.8	8.0	0.9
	$(1,10,14)$	12.4	9.2	2.8	12.4	8.6	1.8	11.8	8.6	1.0	11.8	8.6	0.8
	Total	11.0	8.1	2.8	11.0	7.9	1.7	10.8	7.9	1.1	10.7	7.9	0.8
$v=4$	$(1,10,10)$	10.2	8.0	2.6	10.2	8.0	1.5	10.2	7.8	1.1	10.0	7.8	0.7
	$(1,10,12)$	11.4	8.4	2.6	11.4	8.6	1.6	11.4	8.6	1.1	11.4	8.6	0.8
	$(1,10,14)$	12.6	9.4	2.8	12.8	8.8	1.9	12.4	8.6	1.2	12.2	8.6	0.8
	Total	11.4	8.6	2.7	11.5	8.5	1.6	11.3	8.3	1.1	11.2	8.3	0.8
$v=8$	$(1,10,10)$	11.2	8.0	3.4	11.2	8.0	2.0	11.2	8.0	1.4	11.0	8.0	1.0
	$(1,10,12)$	12.4	8.8	3.5	12.2	8.8	1.9	12.0	8.8	1.2	12.0	8.8	0.9
	$(1,10,14)$	15.0	11.0	3.5	15.0	10.2	2.3	14.4	10.2	1.3	13.8	10.4	0.7
	Total	12.9	9.3	3.5	12.8	9.0	2.0	12.5	9.0	1.3	12.3	9.1	0.9

Table EC.9 Value of Integration Under Different Pilot Dispatching Costs.

			$c_s^3=48$		$c_s^3=60$		$c_s^3=72$		$c_s^3=84$		$c_s^3=96$	
	(L,B,V)	#P(S)	#P(I)	Gap(%)	#P(I)	Gap(%)	#P(I)	Gap(%)	#P(I)	Gap(%)	#P(I)	Gap(%)
$v=1$	$(1,10,10)$	9.8	6.8	1.7	6.8	2.5	6.6	3.3	6.6	4.0	6.6	4.7
	$(1,10,12)$	10.6	8.0	1.6	7.6	2.2	7.2	2.9	7.0	3.5	6.8	4.2
	$(1,10,14)$	11.8	8.2	1.5	8.0	2.2	8.0	2.8	8.0	3.3	7.8	4.2
	Total	10.7	7.7	1.6	7.5	2.3	7.3	3.0	7.2	3.6	7.1	4.4
$v=2$	$(1,10,10)$	9.8	7.0	1.6	6.8	2.4	6.6	3.2	6.6	3.9	6.6	4.6
	$(1,10,12)$	10.8	8.0	1.7	8.0	2.3	7.8	2.9	7.4	3.6	7.2	4.3
	$(1,10,14)$	12.4	8.6	1.8	8.6	2.5	8.2	3.1	8.2	3.8	8.6	4.1
	Total	11.0	7.9	1.7	7.8	2.4	7.5	3.1	7.4	3.8	7.5	4.3
$v=4$	$(1,10,10)$	10.2	8.0	1.5	7.8	2.1	7.4	2.7	7.4	3.4	6.8	4.1
	$(1,10,12)$	11.4	8.6	1.6	8.4	2.2	7.6	2.9	7.4	3.7	7.4	4.5
	$(1,10,14)$	12.8	8.8	1.9	8.8	2.7	8.4	3.4	8.2	4.1	8.0	4.9
	Total	11.5	8.5	1.6	8.3	2.3	7.8	3.0	7.7	3.7	7.4	4.5
$v=8$	$(1,10,10)$	11.2	8.0	2.0	7.6	2.9	7.4	3.7	7.4	4.6	7.4	5.4
	$(1,10,12)$	12.2	8.8	1.9	8.8	2.6	7.8	3.5	7.8	4.3	7.4	5.2
	$(1,10,14)$	15.0	10.2	2.3	9.8	3.2	9.8	4.1	9.4	5.0	8.8	5.8
	Total	12.8	9.0	2.0	8.7	2.9	8.3	3.8	8.2	4.6	7.9	5.5

Note. The sequential optimization method delivered the same optimal number of working pilots for an instance under the same v and different c_s^3's.

References

Jia S, Li CL, Xu Z (2019) Managing navigation channel traffic and anchorage area utilization of a container port. *Transportation Science* 53(3):728–745.

The Marine Department of Hong Kong (2018) Port of Hong Kong statistical tables 2017. Technical report, URL `https://www.mardep.gov.hk/en/fact/pdf/portstat_ast_2017.pdf`, Accessed May 1, 2021.

The Marine Department of Hong Kong (2020) Pilotage in Hong Kong. `https://www.mardep.gov.hk/en/pub_services/ocean/pilot.html`, Accessed May 1, 2021.

Yang D, Wu L, Wang S, Jia H, Li KX (2019) How big data enriches maritime research–a critical review of automatic identification system (AIS) data applications. *Transport Reviews* 39(6):755–773.