https://doi.org/10.1016/j.ejor.2020.05.009

Pilotage Planning in Seaports

Lingxiao Wu, Shuai Jia, Shuaian Wang*

Department of Logistics & Maritime Studies, The Hong Kong Polytechnic University, Hung Hom, Hong Kong

Abstract

Vessel pilotage is compulsory in most seaports around the world. When traveling into or out of the terminals of a seaport, vessels should be navigated by sea pilots in order to follow correct and safe waterways. This paper studies a pilotage planning problem that involves decisions of scheduling the vessel traffic in a seaport, assigning work shifts to pilots, and scheduling the pilots in each work shift for vessel navigation. We formulate the problem as a linear mixed-integer programming (MIP) model that aims at minimizing the cost incurred by the pilotage operations, and show that the problem is strongly NP-hard. For solving the problem, we develop a branch-and-price (B&P) algorithm in which the pricing problem is solved by a novel dynamic programming algorithm. We further propose several acceleration techniques to improve the efficiency of the B&P algorithm. Computational performance of the B&P algorithm is evaluated in extensive numerical experiments. Computational results demonstrate that the B&P algorithm is able to solve problem instances of practical sizes, and that the algorithm outperforms a standard MIP solver and a solution method commonly used in practice.

Keywords: Scheduling, Port Operations, Sea Pilot, Pilotage Planning, Branch-and-Price, Column Generation

1. Introduction

Seaborne transportation forms the backbone of the global supply chain. According to the estimation of UNCTAD (2018), the volume of seaborne trade reached 10.7 billion tons in 2017, accounting for over 80 percent of the total volume of global trade. In the global supply chain, seaports play a critical role in cargo transportation, as cargos are transhipped between sea transportation and land transportation at seaports. Due to the growth of seaborne trade, the numbers and sizes of vessels that need to be served at seaports have been increasing continuously. As a result, vessel traffic in seaports has become more and more dense, imposing the challenge of congestion mitigation on vessel traffic service (VTS) operators.

Figure 1 shows the layout of a seaport, which can generally be divided into two parts, namely, the seaside part and the landside part (i.e., terminals). The seaside part of a seaport is composed of an outer anchorage, an inner anchorage, and a navigation channel. The navigation channel is a bidirectional waterway that is used by the vessels for entering or leaving the terminals of the

*Corresponding author.

E-mail addresses: lingxiaowu513@gmail.com (L. Wu), shuai.jia@connect.polyu.hk (S. Jia), wangshuaian@gmail.com (S. Wang).

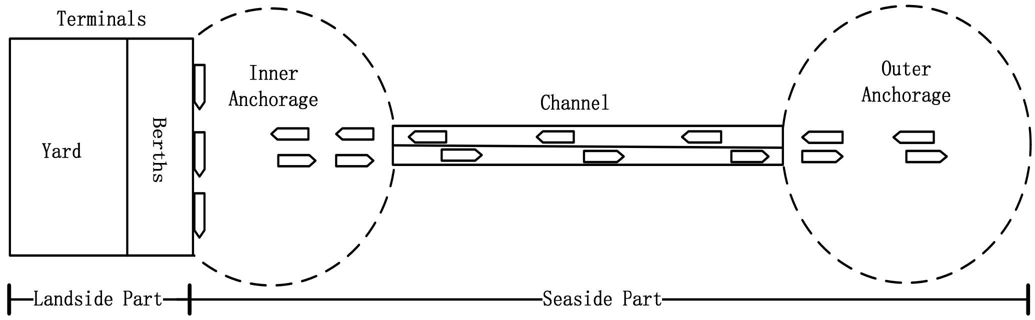

Figure 1: Layout of a seaport.

seaport. Vessels that need to enter the terminals from the open sea should first reach the outer anchorage, then travel through the channel and arrive at the inner anchorage, where vessels will wait for their berths to become available. After being served at the berths, the vessels will unberth and leave the terminals. Vessels that need to leave the terminals should first reach the inner anchorage, where the vessels will wait for the navigation channel to become available. The vessels will then arrive at the outer anchorage after passing through the navigation channel, and finally return to the open sea.

In most seaports, pilotage is compulsory for the majority of the calling vessels (especially for foreign vessels and vessels of large sizes) due to safety and security concerns. When entering or leaving the terminals of a seaport, vessels should be navigated by sea pilots. For each vessel that enters (or leaves) a terminal, a pilot will board the vessel to provide navigation service when the vessel starts entering or leaving the terminal. The pilot will provide assistance for the captain to maneuver the vessel so that the vessel will sail at a safe speed and keep clear of the other vessels that sail in the same direction in the channel. Pilotage is completed when the vessel arrives at the inner anchorage (or outer anchorage), upon which the pilot will leave the vessel. The pilot then either travels (by taking a pilot boat or a helicopter) to the depot where he/she rests and waits for the next task, or travels directly to the location of another vessel to provide navigation service.

To mitigate vessel congestion in the channel of a seaport, the vessel traffic and the pilots should be jointly scheduled so that the vessels can enter and leave the terminals on time. In practice, pilotage and vessel traffic in navigation channels are regulated by VTS operators of local maritime departments (International Maritime Organization, 1997), whereas the berthing and handling of each calling vessel are managed by terminal operators. After receiving berthing and unberthing plans of calling vessels from the terminals, the VTS is responsible for developing a pilotage plan, which controls the vessel traffic in the channel, assigns pilots to the vessels, and schedules the pilots for serving the vessels. However, pilotage planning can be very challenging because it involves joint management of vessel traffic, pilots, and the pilot transport vehicles (namely the pilot boats and the helicopters), as well as various operational restrictions, such as the tidal conditions of the channel, capacity of the channel, and the availability of the pilots.

This paper studies a pilotage planning problem (PPP) from the perspective of a VTS operator for congestion mitigation in a seaport. We develop a linear mixed-integer programming (MIP) model for this problem. Our model involves various decisions that arise in the practice of a VTS operator, including the decision of scheduling the vessel traffic in the navigation channel, the decision of assigning work shifts to pilots, and the decision of scheduling the pilots for vessel navigation in each work shift. The objective of the model is to minimize the cost incurred by the pilotage operations. We show that the problem is strongly NP-hard, and develop a branch-and-price (B&P) algorithm for solving the problem.

Maritime transportation has been a hot topic in operations research, e.g., Song and Carter (2009), Meng et al. (2012), Ng (2017), and Cruz et al. (2019). Studies on seaport operations date back to the 1950s (Eddison and Owen, 1953). Researchers that study seaport operations have mainly focused on problems that manage the utilization of the landside (terminal) resources. These problems include the berth allocation problem, e.g., Giallombardo et al. (2010) and Xu and Lee (2018), the quay crane scheduling problem, e.g., Choo et al. (2010), the yard planning problem, e.g., Zhen et al. (2016), the yard crane scheduling problem e.g., Galle et al. (2018), and the integration of two or more of these problems, e.g., Vacca et al. (2013) and Robenek et al. (2014). For comprehensive reviews of operations research applications in terminal operations management problems, we refer the readers to Stahlbock and Voß (2008), Bierwirth and Meisel (2010), Bierwirth and Meisel (2015), and Carlo et al. (2015).

While there exist extensive studies on the management of terminal operations, only a few studies focused on the vessel traffic management in the seaside. Zhang et al. (2016) considered a problem that schedules entering and leaving vessels for passing a navigation channel. A similar problem was investigated by Lalla-Ruiz et al. (2016). The authors formulated the problem as a variation of the vehicle routing problem, and proposed a metaheuristic for solving the problem. Jia et al. (2019) extended the problem studied by Zhang et al. (2016) and Lalla-Ruiz et al. (2016) by considering the utilization of the anchorages areas in the terminal basin. They showed that the traffic management problem with anchorage area utilization is strongly NP-hard, and proposed a Lagrangian relaxation heuristic for generating near-optimal solutions.

Besides vessel traffic scheduling, pilotage planning is another important issue in seaside operations management. However, studies on pilotage planning problems are rare. To our knowledge, only two works have studied problems related to pilotage planning. Wermus and Pope (1994) studied a pilot rostering problem that determines the start and end times of the work shift assigned to each pilot. While the problem involves the decision of scheduling of pilots, the scheduling of the vessel traffic is not part of the decisions of the problem. In another study, Edwards (2010) investigated a problem that assigns a group of pilots to a set of pilotage tasks. In this problem, each pilotage task corresponds to a vessel that enters or leaves a terminal through a channel and each task has a fixed start time. Capacity limitations in the channels were not considered, and all pilots were transported among tasks using pilot boats. Our work differs from the two studies in several aspects. First, our work schedules the vessel traffic and the pilots jointly, whereas the two studies only considered the assignment of pilots to a given set of pilotage tasks. Second, our work considers two options for transporting pilots among different locations, namely transporting pilots via pilot boats and transporting pilots via helicopters, whereas Edwards (2010) only considered the transportation of pilots by pilot boats. Note that helicopters have been widely used for transporting pilots in seaports around the world, to name a few, Ports of Newcastle and Hedland in Australia, Ports of Shanghai and Tianjin in China, Port of Le Havre in France and Port of Richards Bay in South Africa. Finally, in terms of solution methods, our work applies an exact solution method which is able to generate optimal solutions for the PPP, while Wermus and Pope (1994) and Edwards (2010) generated sub-optimal solutions using heuristics.

Our main contributions are highlighted below:

- We study the PPP that aims to schedule the vessel traffic in the seaside of a seaport and schedule the pilots for serving the vessels. The problem that we study is of practical importance but has not been well addressed in the literature. We develop a MIP model for the problem and show that the problem is strongly NP-hard.

- For solving the problem, we develop a tailored B&P algorithm, in which the pricing problem is solved by a novel and efficient dynamic programming (DP) algorithm. We also propose several acceleration strategies to improve the efficiency of the B&P algorithm.

- We conduct extensive numerical experiments to evaluate the computational performance of the proposed solution method. Computational results show that the proposed solution method can generate optimal or near-optimal solutions for instances of practical sizes within a short running time.

The remainder of the paper is organized as follows. Section 2 provides a detailed description of the PPP. Section 3 presents the MIP model. Section 4 discusses the computational complexity of the PPP. Section 5 describes our solution method. Section 6 describes the computational experiments and reports the computational results. Section 7 draws conclusions with our main findings.

2. Problem Description

In this section, we describe the pilotage operations and the costs incurred by the operations. We introduce some terminologies that we use throughout the paper, and give a formal definition of the PPP. We also present the assumptions that we made to facilitate the modeling and analysis of the PPP.

We refer to the service requests of vessels for channel navigation as pilotage tasks (or tasks). In particular, as shown in Figure 2, the service requests for entering the terminals are called the in-wharf tasks, while the service requests for leaving the terminals are called the out-wharf tasks. An in-wharf task requires a pilot to navigate a vessel from the outer anchorage to the inner anchorage, while an out-wharf task requires a pilot to navigate a vessel from the inner anchorage to the outer anchorage. In the PPP, we assume that pilotage is compulsory for each vessel, that is, each vessel must be served by one pilot when traveling between the outer anchorage and the inner anchorage. Therefore, each task must be assigned to one pilot. Furthermore, each task has a time window during which it must be executed by a pilot. In practice, the time window of each in-wharf task is determined based on estimated arrival time and the planned berthing time of the corresponding vessel, whereas the time window of each out-wharf task is determined based on the planned unberthing time and the planned port departure time of the corresponding vessel. For a seaport where the water depth in the navigation channel is affected by the tide such that some vessels can sail in the channel only during high-tide periods, the time window of a task also depends on the tide and the draft of the vessel (Ding et al., 2016; Zhen et al., 2017). Note that the planned berthing and unberthing times of vessels are provided by the terminal operators, while the pilotage plans are devised by the VTS. The decision maker of our model is the VTS, and the time windows of tasks are predetermined and are given as input data.

To ensure that in-wharf tasks and out-wharf tasks can be executed during their time windows, the VTS should assign work shifts to the pilots and schedule the pilot in each work shift for serving the tasks. We refer to a work shift of a pilot as a pilot shift. The length of each pilot shift is limited

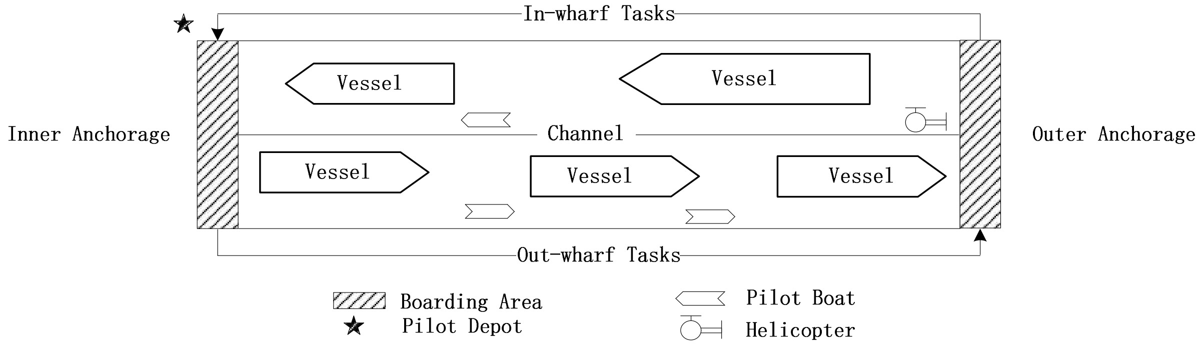

Figure 2: An illustration of in-wharf and out-wharf tasks.

(typically eight hours) due to safety and fatigue considerations. If a pilot is assigned a pilot shift, then the pilot will be available for the tasks during the pilot shift. We assume that all pilots are initially deployed at the depot, which is located at the inner anchorage (see Figure 2). We also assume that all pilots must return to the depot by the end of each pilot shift assigned to the pilot.

To serve a task, a pilot may need to move from the depot to the target vessel, or move from a vessel that has completed navigation service to the target vessel. Two types of vehicles are used for transporting pilots among different locations, namely pilot boats and helicopters. Pilot transport operations are classified into two types: (i) through-channel transport; and (ii) non-through-channel transport. Through-channel transport refers to the transport operation in which a pilot needs to pass through the channel in order to arrive at the location of the target vessel. Non-through-channel transport refers to the transport operation in which a pilot does not need to pass through the channel when traveling to the location of the target vessel. Hence, in a through-channel transport operation, a pilot serves two tasks of the same type (i.e., both are in-wharf tasks or both are out-wharf tasks) consecutively, while in a non-through-channel transport operation, a pilot serves two tasks of different types consecutively. Note that while the time required for performing non-through-channel transport is usually quite short, the time required for performing through-channel transport can be much longer, since the distance between the outer anchorage and the inner anchorage of a seaport is generally very long. In our model, both pilot boats and helicopters are used for through-channel transport, but only pilot boats are used for non-through-channel transport.

Pilotage operations incur various operational costs. In our model, four cost components are considered. These cost components are described as follows: (i) Task delay cost incurred in the situation where the service of a task starts later than the earliest possible start time. (ii) Pilot dispatching cost incurred by assigning a pilot shift to a pilot. (iii) Pilotage cost incurred by executing the in-wharf and out-wharf tasks. (iv) Pilot transport cost incurred by transporting a pilot (either by a pilot boat or by a helicopter) between two locations.

We now provide a formal definition of the PPP as follows: Given a planning horizon, a set of tasks to be executed during the planning horizon, and the times and costs involved in the pilotage operations, determine a plan that decides the number of pilot shifts to be assigned to the pilots, assigns start and end times to each pilot shift, assigns service start times to the tasks, and schedules the pilot in each pilot shift so that all tasks can complete service within their time windows, and that the operational requirements on vessel pilotage are respected. The objective is to minimize the total operational costs.

To facilitate the modeling and analysis of the PPP, we make the following assumptions:

A1. The planning horizon is discretized using a sequence of time points so that the time period between any two consecutive time points has a unit length.

A2. Pilots are identical and are capable of executing each of the tasks.

A3. The number of pilot boats and the number of helicopters are sufficiently large.

A4. The time spent in each non-through-channel transport operation and the cost incurred by each non-through-channel transport operation are trivial and are assumed to be zero.

Before presenting the model, we explain several important aspects in our problem. First, discretized planning horizon is used in port management practices, so that vessels start entering the channel at certain time points and pilot shifts also start and end at certain time points. For port authorities, using a discretized planning horizon can better coordinate the activities of different parties (e.g., the terminals, the VTS center, and vessels). Similar assumptions have also been utilized in a number of studies on port operations (e.g., Giallombardo et al., 2010; Zhen et al., 2016; Jia et al., 2019). Second, in this paper, we do not consider the rostering problem of pilots (i.e., assign particular pilots to the shifts). In fact, the rostering problem can be solved based on the output of our problem by considering various regulations regarding assigning workload to the pilots. Third, pilot boats and helicopters can travel at much higher speeds than vessels in the channel, therefore, a relatively small number of pilot boats and helicopters (compared with the number of pilots and vessels) are sufficient to transport pilots between tasks without delay. Hence, to simplify the analysis, we do not consider the detailed scheduling of pilot boats and helicopters in the PPP.

3. Mathematical Model

In this section, we present a discrete-time MIP model for the PPP. In our model, the planning horizon is discretized by a sequence of time points, and the time period between any two consecutive time points has a unit length. All notations used in our model are explained in Table 1.

Table 1: Notation.

Indices:	
i, j	Indices for tasks and the depot (0 denotes the depot).
p	Index for pilot shifts.
t, t_1, t_2	Indices for time points in a planning horizon.
k	Index for types of pilot transport vehicles; $k = 1$ represents pilots boats, $k = 2$ represents helicopters.
h	Index for task types; $h = 1$ represents in-wharf tasks, $h = 2$ represents out-wharf tasks.
Sets:	
Ω_1	Set of in-wharf tasks.
Ω_2	Set of out-wharf tasks.
Ω	Set of all tasks, $\Omega = \Omega_1 \bigcup \Omega_2$.

$\bar{\Omega}$	An extended task set, $\bar{\Omega} = \Omega \bigcup \{0\}$ (0 denotes the depot).
Φ	Set of task types, $\Phi = \{1, 2\}$.
Ψ	Set of pilot shifts. We set $\Psi = \{1, 2, ..., \lvert\Omega\rvert\}$ to ensure that there are sufficiently many pilot shifts for the tasks. In addition, if a pilot shift is not assigned to any pilot, then the pilot shift is empty.
Θ	Set of types of pilot transport vehicles, $\Theta = \{1, 2\}$.
T	Set of time points in the planning horizon, $T = \{1, 2, ..., \max_{i\in\Omega}\{l_i\}\}$.
$\hat{T}$	Set of time points in an extended planning horizon, $\hat{T} = T \bigcup \{\lvert T\rvert + 1, \lvert T\rvert + 2, ..., \lvert T\rvert + \max_{i\in\Omega}\{d_i\} + \max_{k\in\Theta}\{r_k\}\}$.
Parameters:	
c_1^i	Unit delay cost of task i.
c_2	Cost for assigning a pilot shift.
c_3	Cost for providing pilotage service to a task.
c_4^k	Cost for performing a through-channel transport operation using a type-k vehicle.
e_i	Earliest allowed start time of task i.
l_i	Latest allowed start time of task i.
d_i	Time required for serving task i.
$\sigma_{i,j}$	$= 1$ if through-channel transport is required to transport a pilot to task j after he/she finishes task i, and 0, otherwise.
$\sigma_{0,j}$	$= 1$ if through-channel transport is required to transport a pilot from the depot to task j, and 0, otherwise.
$\sigma_{i,0}$	$= 1$ if through-channel transport is required to transport a pilot to the depot after he/she finishes task i, and 0, otherwise.
r_k	Time required for performing through-channel transport using a type-k vehicle.
u_h	Capacity of the navigation channel for vessels that correspond to type-h tasks.
$\bar{D}$	Maximum length of duration of each pilot shift.
M	A sufficiently large positive constant.
Decision Variables:	
α_t^p	$= 1$, if pilot shift p starts at time t, and 0, otherwise.
β_t^p	$= 1$, if pilot shift p ends at time t, and 0, otherwise.
x_{ij}^p	$= 1$, if a pilot executes task j immediately after task i during pilot shift p, and 0, otherwise.
x_{0j}^p	$= 1$, if task j is the first task executed during pilot shift p, and 0, otherwise.
x_{i0}^p	$= 1$, if task i is the last task executed during pilot shift p, and 0, otherwise.
y_{it}	$= 1$, if task i starts at time t, and 0, otherwise.
z_{ij}^k	$= 1$, if a type-k vehicle is used in a through-channel transport operation that transports a pilot to task j after he/she finishes task i, and 0, otherwise.
z_{0j}^k	$= 1$, if a type-k vehicle is used in a through-channel transport operation that transports a pilot from the depot to task j, and 0, otherwise.
z_{i0}^k	$= 1$, if a type-k vehicle is used in a through-channel transport operation that transports a pilot to the depot after he/she finishes task i, and 0, otherwise.

The mathematical formulation (denoted by M1) for the PPP is presented below:

$$\text{(M1)}\quad \min F = \sum_{i\in\Omega} c_1^i \sum_{t\in T} (t - e_i) y_{it} + c_2 \sum_{p\in\Psi} \sum_{t\in T} \alpha_t^p + c_3 \sum_{p\in\Psi} \sum_{i\in\Omega} \sum_{j\in\bar{\Omega}\setminus\{i\}} x_{ij}^p + \sum_{k\in\Theta} c_4^k \sum_{i\in\bar{\Omega}} \sum_{j\in\bar{\Omega}\setminus\{i\}} z_{ij}^k, \tag{1}$$

$$\sum_{t \in T} \alpha_t^p \leq 1, \;\; p \in \Psi, \tag{2}$$

$$\sum_{t \in \hat{T}} \beta_t^p \leq \sum_{t \in T} \alpha_t^p, \;\; p \in \Psi, \tag{3}$$

$$\sum_{t_2 = t_1}^{\min\{t_1 + \bar{D}, \hat{T}\}} \beta_{t_2}^p \geq \alpha_{t_1}^p, \;\; t_1 \in T, \tag{4}$$

$$\sum_{j \in \Omega} x_{0j}^p = \sum_{t \in T} \alpha_t^p, \;\; p \in \Psi, \tag{5}$$

$$\sum_{j \in \bar{\Omega} \setminus \{i\}} x_{ij}^p - \sum_{j \in \bar{\Omega} \setminus \{i\}} x_{ji}^p = 0, \;\; i \in \Omega, p \in \Psi, \tag{6}$$

$$\sum_{p \in \Psi} \sum_{j \in \bar{\Omega} \setminus \{i\}} x_{ij}^p = 1, \;\; i \in \Omega, \tag{7}$$

$$\sum_{t \in T} y_{it} = 1, \;\; i \in \Omega, \tag{8}$$

$$\sum_{t_1 \in T} t_1 \alpha_{t_1}^p + \sum_{k \in \Theta} r_k z_{0j}^k - \sum_{t_2 \in T} t_2 y_{jt_2} + (x_{0j}^p - 1)M \leq 0, \;\; j \in \Omega, p \in \Psi, \tag{9}$$

$$\sum_{t_1 \in T} t_1 y_{it_1} + d_i + \sum_{k \in \Theta} r_k z_{ij}^k - \sum_{t_2 \in T} t_2 y_{jt_2} + (\sum_{p \in \Psi} x_{ij}^p - 1)M \leq 0, \;\; i, j \in \Omega, \tag{10}$$

$$\sum_{t_1 \in T} t_1 y_{it_1} + d_i + \sum_{k \in \Theta} r_k z_{i0}^k - \sum_{t_2 \in \hat{T}} t_2 \beta_{t_2}^p + (x_{i0}^p - 1)M \leq 0, \;\; i \in \Omega, p \in \Psi, \tag{11}$$

$$y_{it} = 0, \;\; t \in T \setminus [e_i, l_i], i \in \Omega, \tag{12}$$

$$\sum_{k \in \Theta} z_{ij}^k \geq \sum_{p \in \Psi} x_{ij}^p, \;\; i \in \bar{\Omega}, j \in \bar{\Omega} \setminus \{i\}, \sigma_{ij} = 1, \tag{13}$$

$$\sum_{i \in \Omega_h} \sum_{t_2 = \max\{1, t_1 - d_i + 1\}}^{t_1} y_{it_2} \leq u_h, \;\; t_1 \in T, h \in \Phi, \tag{14}$$

$$\alpha_t^p \in \{0, 1\}, \;\; t \in T, p \in \Psi, \tag{15}$$

$$\beta_t^p \in \{0, 1\}, \;\; t \in \hat{T}, p \in \Psi, \tag{16}$$

$$x_{ij}^p \in \{0, 1\}, \;\; p \in \Theta, i \in \bar{\Omega}, j \in \bar{\Omega} \setminus \{i\}, \tag{17}$$

$$z_{ij}^k \in \{0,1\}, \quad k \in \Psi, i \in \bar{\Omega}, j \in \bar{\Omega} \setminus \{i\}, \tag{18}$$

$$y_{it} \in \{0,1\}, \quad i \in \Omega, t \in T. \tag{19}$$

The objective function (1) minimizes the total cost of the pilotage operations, including the task delay costs (the first term), the pilot dispatching costs (the second term), the pilotage costs (the third term), and the pilot transport costs (the fourth term). We note that the third term is a constant. Constraint (2) ensures that each pilot shift is assigned at most one start time. If a pilot shift is not assigned any start time, then the pilot shift is empty and will not be assigned to any pilot. Constraint (3) ensures that each pilot shift is assigned at most one finishing time. Constraint (4) ensures that the length of each pilot shift does not exceed $\bar{D}$. Constraint (5) ensures that each pilot shift has at most one starting task. Constraint (6) ensures flow balances. Constraint (7) ensures that each task is exected by one pilot. Constraint (8) ensures that each task is assigned one start time. Constraint (9) ensures that the start time of the first task in a pilot shift is no earlier than the start time of the pilot shift, and that the first task cannot start until a pilot has arrived at the location of the corresponding vessel. Similarly, Constraint (10) specifies the relationship between the start times of two tasks conducted consecutively in the same pilot shift; and Constraint (11) specifies the relationship between the start time of the last task in a pilot shift and the time at which the pilot returns to the depot. Constraint (12) imposes a time window for each of the tasks. Constraint (13) ensures that a transport vehicle is used to transport a pilot through the channel if the pilot needs to consecutively execute two tasks of the same type. Constraint (14) imposes upper limits on the number of entering vessels and the number of leaving vessels that sail in the channel at the same time. Constraints (15)–(19) define binary variables.

4. Complexity of the Problem

In this section, we show that the PPP is NP-hard in the strong sense. To do so, we show the decision version of the PPP is strongly NP-hard. That is, given settings of tasks, pilot shifts, and pilot transport, it cannot be determined in polynomial time or even in pseudo-polynomial time whether the objective value F of the problem is no larger than a given constant λ unless P=NP.

We prove the strong NP-hardness of the PPP by reducing a well-known strongly NP-hard problem—the Bin Packing Problem (BPP)—to a decision version of the PPP.

Theorem 1. *The PPP is strongly NP-hard.*

Proof. We transform the BPP to the decision version of the PPP. The BPP can be stated as follows. There is a set Ω of items, and the volume of item $i \in \Omega$ is v_i. There is also a set B of identical bins, each of which has a capacity of $\bar{C}$. The BPP asks whether there is a packing strategy (denoted by $\mathbf{S}$) such that a set $\Omega_b \subseteq \Omega$ of i's are packed in bin $b \in B$ and the following conditions hold:

$$\sum_{i \in \Omega_b} v_i \leq \bar{C}, \quad b \in B, \tag{20}$$

$$\bigcup_{b \in B} \Omega_b = \Omega. \tag{21}$$

Given an arbitrary instance of BPP, we construct a corresponding instance of the PPP as follows. There is a set Ω of tasks (i's) to be executed in a planning horizon. Let $T = \{1, 2, ..., |T|\}$ denote the set of time points in the planning horizon. Specifically, we set other parameters as follows.

$$u_h = |\Omega_h|, \;\; h \in \Phi, \tag{22}$$

$$d_i = v_i, \;\; i \in \Omega, \tag{23}$$

$$e_i = 1, \;\; i \in \Omega, \tag{24}$$

$$l_i = |T|, \;\; i \in \Omega, \tag{25}$$

$$r_k = 0, \;\; k \in \Theta, \tag{26}$$

$$\bar{D} = \bar{C}, \tag{27}$$

$$|T| \geq \bar{C} + 1, \tag{28}$$

$$c_1^i = 0, \;\; i \in \Omega, \tag{29}$$

$$c_2 = 1, \tag{30}$$

$$c_3 = 0, \tag{31}$$

$$c_4^k = 0, \;\; k \in \Theta, \tag{32}$$

$$\lambda = |B|. \tag{33}$$

Clearly, this transformation is pseudo-polynomial. We will show that there exists a feasible solution to the constructed instance of PPP if and only if the answer to the BPP is "yes".

Suppose the answer to the BPP is "yes". Let Ω_b^* denote the items packed in bin $b \in B$ in $\mathbf{S}$. Then consider the following solution ($\mathbb{S}$) to the constructed instance of the PPP. First, corresponding to each $b \in B$, generate a pilot shift p_b, such that p_b starts from time 1 and ends at time $\bar{D} + 1$. Then, corresponding to each item $i \in \Omega_b^*$, assign task i to pilot shift p_b. Let P and Ω_{p_b} denote the set of generated pilot shifts and the set of tasks executed in shift p_b, respectively. Obviously, $|P| = |B|$. The feasibility of $\mathbb{S}$ to the PPP instance can be verified as follows. To begin with, Equations (22), (24), and (25) indicate that each task can start at any time within the planning horizon, since the time window of each task covers the entire planning horizon and the channel always has spare capacities. In addition, Equation (23) and (26) indicate that the time to complete all tasks $i \in \Omega_{p_b}$ (denoted by T_{p_b}) can be calculated by: $T_{p_b} = \sum_{i \in \Omega_{p_b}} d_i = \sum_{i \in \Omega_b^*} v_i$, which follows that $T_{p_b} \leq \bar{C} = \bar{D}$. Hence, all tasks assigned to shift p_b can be completed within the shift. Besides, Equation (21) implies that $\bigcup_{p_b \in P} \Omega_{p_b} = \bigcup_{b \in B} \Omega_b^* = \Omega$. Hence, all tasks in Ω are executed in the shifts in set P. As for the objective value F, since $c_1^i = 0$, $c_3 = 0$, and $c_4^k = 0$, we have $F = c_2 \sum_{p \in \Psi} \sum_{t \in T} \alpha_t^p = c_2|P| = |B| = \lambda$. Therefore, $\mathbb{S}$ is feasible to the constructed instance of the PPP.

Conversely, suppose that there exists a feasible solution to the constructed instance of the PPP such that $F \leq \lambda$. Let P^* denote the set of pilot shifts assigned in the solution. Further, for each $p \in P^*$, let Ω_p be the set of tasks executed in shift p. It is easy to infer that (i) $\sum_{i \in \Omega_p} d_i \leq \bar{D}, p \in P^*$, (ii) $\bigcup_{p \in P^*} \Omega_p = \Omega$, and (iii) $|P^*| \leq |B|$. Considering that $\bar{D} = \bar{C}$ and $v_i = d_i, i \in \Omega$, we can construct a feasible solution to the BPP by packing all items $i \in \Omega$ to the set B of bins. This completes the proof. □

Remark 1. *In the proof of Theorem 1, the constructed instance of the PPP has channel capacities* $u_1 = |\Omega_1|$ *and* $u_2 = |\Omega_2|$. *Therefore, the PPP is NP-hard in the strong sense even if there are no capacity limitations in the navigation channel.*

5. The Branch and Price Algorithm

In view of the complexity of the considered problem, this paper proposes a tailored B&P algorithm to solve the problem. In this section, we first reformulate model M1 developed in Section 3 as a set covering model in Section 5.1. In Section 5.2, we introduce the framework of the algorithm. The pricing problem and the branching strategy are described in Section 5.3 and Section 5.4, respectively. Finally, in Section 5.5, we propose several acceleration techniques to improve the efficiency of the algorithm.

5.1. A Set Covering Reformulation

In this section, we provide a set covering formulation for the PPP. To formulate the model, we define columns used in the model as follows. A column q corresponds to a route which records the sequence of a pilot for conducting a set of tasks in a pilot shift. The route begins and ends at the depot. Associated with column q are two parameters c_q and s_{it}^q. In particular, c_q denotes the cost of the column, that is, the cost for assigning the pilot shift, starting tasks with delays, providing pilotage services and transporting pilots in the corresponding route. Besides, s_{it}^q is set to be 1 if task i starts at time t in column q and 0, otherwise.

Let Q denote the set of all feasible columns, and let ω_q be the decision variable which equals 1, if column q is selected and 0, otherwise. The set covering model (MP) for the PPP can be formulated as follows.

$$\text{(MP)} \ \min \sum_{q \in Q} \omega_q c_q, \tag{34}$$

$$\sum_{q \in Q} \omega_q \sum_{t \in T} s_{it}^q \geq 1, \quad i \in \Omega, \tag{35}$$

$$-\sum_{q \in Q} \omega_q \sum_{i \in \Omega_h} \sum_{t_1 = \max\{1, t - d_i + 1\}}^{t} s_{it_1}^q \geq -u_h, \quad t \in T, h \in \Phi, \tag{36}$$

$$\omega_q \in \{0,1\}, \quad q \in Q. \tag{37}$$

The objective function (34) aims at minimizing the total cost. Constraint (35) ensures each task is covered by at least one column (i.e., the task is conducted in at least one pilot shift). Constraint (36) enforces channel capacity restrictions. Constraint (37) defines the decision variables to be binary.

5.2. The B&P Scheme

The B&P algorithm solves the MP in a Branch-and-Bound (B&B) framework. At each node of the B&B tree, we solve a linear relaxation of a restricted MP (denoted by LRMP) by using column generation. Column generation has been widely used to solve complicated integer programming problems (refer to Lübbecke and Desrosiers, 2005 and Desaulniers et al., 2006). In column generation, columns are generated for an LRMP at a node by solving a pricing problem (see in Section 5.3). If columns with negative reduced costs are identified in the pricing problem, these columns will be added into the LRMP, which will then be solved again, otherwise, the LRMP has been solved to its optimum. Then, if the solution to the LRMP is integral, a valid upper bound is obtained for the MP, otherwise, a branch is made by partitioning the feasible space of the integer solution in a way that eliminates the current fractional solution (see Section 5.4).

5.3. The Pricing Problem

This section introduces the pricing sub-problem (SP) for generating columns with negative reduced costs when solving an LRMP. We first formulate the SP as a MIP model and then propose a dynamic programming (DP) algorithm to solve the problem.

5.3.1. A MIP Model for the Pricing Problem

To begin with, the new variables and parameters used in the MIP model for the SP are listed in Table 2.

Table 2: Notations for the SP.

Parameters:	
π_i	Dual value for the ith constraint in Constraint (35).
μ_{th}	Dual value for t, hth constraint in Constraint (36).
e'_i	(Truncated) earliest start time for task i. $e'_i \geq e_i$, due to the time-window branch (see Section 5.4).
l'_i	(Truncated) latest start time for task i. $l'_i \leq l_i$, due to the time-window branch (see Section 5.4).
Decision Variables:	
χ_{ij}	1, if task j starts immediately after task i finishes in the column (route) and 0, otherwise.
χ_{0j}	1, if task j is the first task in the column (route) and 0, otherwise.
χ_{i0}	1, if task i is the last task in the column (route) and 0, otherwise.
γ^k_{ij}	1, if a type-k vehicle is used in a through-channel transport operation to transport the pilot to j after he/she finishes task i and 0, otherwise.
γ^k_{0j}	1, if a type-k vehicle is used in a through-channel transport operation to transport the pilot from the depot to task j and 0, otherwise.
γ^k_{i0}	1, if a type-k vehicle is used in a through-channel transport operation to transport the pilot to the depot after he/she finishes task i and 0, otherwise.
ν_{it}	1, if task i starts at time t and 0, otherwise.
a	Time when the corresponding pilot shift starts.
b	Time when the corresponding pilot shift ends.

The problem can be formulated as follows.

$$
\begin{aligned}
\text{(SP)}\ \ \min \hat{c} = & \sum_{i\in\Omega} c_1^i \sum_{t\in T}(t-e_i)\nu_{it} + c_2 + c_3 \sum_{i\in\Omega}\sum_{j\in\bar{\Omega}\setminus\{i\}} \chi_{ij} + \sum_{k\in\Theta} c_4^k \sum_{i\in\bar{\Omega}}\sum_{j\in\bar{\Omega}\setminus\{i\}} \gamma_{ij}^k \\
& - \sum_{i\in\Omega}\pi_i \sum_{t\in T}\nu_{it} + \sum_{t\in T}\sum_{h\in\Phi}\mu_{th}\sum_{i\in\Omega_h}\sum_{t_1=\max\{1,t-d_i+1\}}^{t}\nu_{it_1},
\end{aligned} \tag{38}
$$

$$
b - a \le \bar{D}, \tag{39}
$$

$$
\sum_{j\in\bar{\Omega}\setminus\{i\}} \chi_{ij} - \sum_{j\in\bar{\Omega}\setminus\{i\}} \chi_{ji} = 0, \quad i\in\Omega, \tag{40}
$$

$$
\sum_{t\in T}\nu_{it} = \sum_{j\in\bar{\Omega}\setminus\{i\}} \chi_{ij}, \quad i\in\Omega, \tag{41}
$$

$$
a + \sum_{k\in\Theta} r_k\gamma_{0j}^k - \sum_{t\in T} t\nu_{jt} + (\chi_{0j}-1)M \le 0, \quad j\in\Omega, \tag{42}
$$

$$
\sum_{t_1\in T} t_1\nu_{it_1} + d_i + \sum_{k\in\Theta} r_k\gamma_{ij}^k - \sum_{t_2\in T} t_2\nu_{jt_2} + (\chi_{ij}-1)M \le 0, \quad i,j\in\Omega, \tag{43}
$$

$$
\sum_{t\in T} t\nu_{it} + \sum_{k\in\Theta} r_k\gamma_{i0}^k - b + (\chi_{i0}-1)M \le 0, \quad i\in\Omega, \tag{44}
$$

$$
\nu_{it} = 0, \quad t\in T\setminus[e_i', l_i'], \quad i\in\Omega, \tag{45}
$$

$$
\sum_{k\in\Theta}\gamma_{ij}^k \ge \chi_{ij}, \quad i\in\bar{\Omega}, j\in\bar{\Omega}\setminus\{i\}, \sigma_{ij}=1, \tag{46}
$$

$$
a \in T, \tag{47}
$$

$$
b \in \hat{T}, \tag{48}
$$

$$
\chi_{ij} \in \{0,1\}, \quad i\in\bar{\Omega}, j\in\bar{\Omega}\setminus\{i\}, \tag{49}
$$

$$
\gamma_{ij}^k \in \{0,1\}, \quad k\in\Psi, i\in\bar{\Omega}, j\in\bar{\Omega}\setminus\{i\}, \tag{50}
$$

$$
\nu_{it} \in \{0,1\}, \quad i\in\Omega, t\in T. \tag{51}
$$

The objective function (38) is to minimize the reduced cost of the column, which is obtained by deducting the summation of the values of the involved dual variables from the cost of the corresponding route. Constraint (39) ensures that the route must cover a period with no more than $\bar{D}$ time units. Constraint (40) ensures the flow balance. Constraint (41) ensures each task has at most one start time. Constraint (42) describes the relationship between the time when the pilot shift starts and the time when the first task in the column starts. Constraint (43) describes the relationship between the start times of two tasks that are conducted consecutively in the column. Constraint (44) describes the relationship between the time when the pilot shift ends and the time when the last task in the column starts. Constraint (45) enforces the time windows constraints. Constraint (46) specifies when through-channel transport is needed by clarifying the relationship among σ_{ij}, χ_{ij}, and γ_{ij}^{k}. Finally, the last five constraints define integer and binary variables.

5.3.2. A DP for Solving the SP

In this section, we propose a DP for solving the SP. The algorithm first decomposes the SP into a minimum distance routing problem in a three-dimensional task–time–pilot-transport network. In addition, to speed up the DP, we propose several dominance rules to narrow down the searching space of the algorithm.

Task–time–pilot-transport Network Construction. To solve the SP by a DP algorithm, we first construct a task–time–pilot-transport network as follows. To begin with, each node in the task–time–pilot-transport network can be denoted by a three-dimensional tuple (i, t, z), where i is the position [including the depot ($i = 0$) and tasks ($i \neq 0$)], t stands for the time the pilot starts working on the task or arrives at the depot, and z reflects the transport mode the pilot utilizes to get to position i. In particular, z has three states, which are 0 denoting "no through-channel transport", 1 denoting "through-channel transport using pilot boats", and 2 denoting "through-channel transport using helicopters".

Then, consider an extension from a node (i_n, t_n, z_n) to another node $(i_{n'}, t_{n'}, z_{n'})$. Suppose i_n, t_n, $i_{n'}$, and $t_{n'}$ are given. Since the cost of transport-by-boat is always lower than that of transport-by-helicopter $z_{n'}$ can also be fixed according to the following property.

Property 1. *$z_{n'}$ can be obtained by Equation (52) when a node (i_n, t_n, z_n) is extended to another node $(i_{n'}, t_{n'}, z_{n'})$.*

$$z_{n'} = \begin{cases} 0, & \text{if } \sigma_{i_n,i_{n'}} = 0, \\ 1, & \text{if } \sigma_{i_n,i_{n'}} = 1, t_{n'} - t_n \geq d_{i_n} + r_1, \\ 2, & \text{if } \sigma_{i_n,i_{n'}} = 1, t_{n'} - t_n < d_{i_n} + r_1, \end{cases} \tag{52}$$

where d_{i_n} is set to be 0, specifically, if $i_n = 0$.

The cost of extending node (i_n, t_n, z_n) to $(i_{n'}, t_{n'}, z_{n'})$, denoted by $\bar{c}(i_n, t_n, z_n \to i_{n'}, t_{n'}, z_{n'})$, is defined as follows (note that we only need to consider arcs between nodes with different positions, i.e., $i_n \neq i_{n'}$):

1. $\bar{c}(0, t_n, 0 \to i_{n'}, t_{n'}, z_{n'}) = c_2 + c_3 + c_1^{i_{n'}}(t_{n'} - e'_{i_{n'}}) + \mathrm{f}(z_{n'}),\ \ i_{n'} \in \Omega;$
2. $\bar{c}(i_n, t_n, z_n \to i_{n'}, t_{n'}, z_{n'}) = c_3 + c_1^{i_{n'}}(t_{i_{n'}} - e'_{i_{n'}}) + \mathrm{f}(z_{n'}) - \pi_{i_n} + \varphi_{i_n,t_n},\ \ i_n, i_{n'} \in \Omega;$
3. $\bar{c}(i_n, t_n, z_n \to 0, t_{n'}, z_{n'}) = \mathrm{f}(z_{n'}) - \pi_{i_n} + \varphi_{i_n,t_n},\ \ i_n \in \Omega.$

In the above equations, $\mathrm{f}(z_{n'})$ and φ_{i_n,t_n} are calculated using Equations (53) and (54), respectively.

$$\mathrm{f}(z_{n'}) = \begin{cases} 0, & \text{if } z_{n'} = 0, \\ c_4^1, & \text{if } z_{n'} = 1, \\ c_4^2, & \text{if } z_{n'} = 2, \end{cases} \tag{53}$$

$$\varphi_{i_n,t_n} = \sum_{h\in\Phi} \alpha_{i_n}^h \sum_{t_1=t_n}^{\min\{|T|,t_n+d_{i_n}-1\}} \mu_{t_1 h}, \tag{54}$$

where, $\alpha_{i_n}^h = 1$, if $i_n \in \Omega_h$ and 0, otherwise.

The DP Algorithm. We now present the DP algorithm. The algorithm aims at finding the set of routes in the network that have negative reduced costs. A route in the network can be denoted by an ordered array $\mathbf{R}$ that starts from and ends at the pilot depot. $\mathbf{R} = [(i_0,t_0,z_0),(i_1,t_1,z_1),...,(i_n,t_n,z_n),...,(i_N,t_N,z_N),(i_{N+1},t_{N+1},z_{N+1})]$, where tuple (i_n,t_n,z_n) stands for the nth visited node. Specifically, $n=0$ represents starting from the depot and $n=N+1$ represents returning to the depot. In the algorithm, we only consider elementary routes, that is, $i_n \neq i_{n'}, n, n' \in \{1,2,...,N\}, n \neq n'$.

Given a set of partial routes, the DP works by extending each partial route task by task in the task–time–pilot-transport network. To this end, corresponding to each partial route, we define a state (s_n). $s_n = [i_n,t_n,pt_n,\tilde{c}_n,\Omega_n^r]$, where i_n stands for the current position (depot or tasks) of the partial route, t_n stands for the time the pilot arrives at the position (gets ready to leave the depot or starts working on the task), pt_n is the duration of the pilot working time when the pilot arrives at the position (i.e., $pt_n = t_n - t_0$), $\tilde{c}_n$ represents the reduced cost of the partial route, and Ω_n^r is the set of tasks that can be reached after the pilot leaves i_n. To ensure the elementarity of the route, Ω_n^r only includes tasks that have not been visited in the current route. In addition, each task $j \in \Omega_n^r$ of state s_n is associated with a vector (denoted by $\boldsymbol{rst}_n^j$) such that $\boldsymbol{rst}_n^j = [het_n^j, hlt_n^j, bet_n^j, blt_n^j]$, where het_n^j and hlt_n^j denote the earliest and latest start times of task j using helicopters as the transport vehicle between i_n and j, and bet_n^j and blt_n^j are the earliest and latest start times of task j using pilot boats as the transport vehicle between i_n and j. It can be easily inferred that $het_n^j = bet_n^j$, and $hlt_n^j = blt_n^j$ if $\sigma_{i_n,j} = 0$.

The algorithm starts by creating $|T|$ routes each of which corresponds to a state at the depot which can be denoted by $[0,t_0,0,0,\Omega_0]$, where $t_0 = 1,2,...,|T|$, and Ω_0 is the set of tasks that can be reached after the pilot leaves the depot at time t_0. Each time, s_n can be extended by adding tasks from Ω_n^r. The detailed extension procedure for a given stage s_n is presented in Algorithm 1.

When a new state s_n is generated, we check whether a column with negative reduced cost can be identified by extending the corresponding partial route (from node (i_n,t_n,z_n)) to the depot. It is noted that the cost of extending (i_n,t_n,z_n) to the depot $(0,t_{n'},z_{n'})$, which is calculated by $\bar{c}(i_n,t_n,z_n \rightarrow 0,t_{n'},z_{n'}) = \mathrm{f}(z_{n'}) - \pi_{i_n} + \varphi_{i_n,t_n}$, is independent of the time $t_{n'}$, if $z_{n'}$ is given. Therefore, we only need to check the reduced cost of the column that enables the pilot to return to the depot at the lowest transport cost (if through-channel transport is needed) and at the correspondingly earliest time. Further, it is easy to infer that when through-channel transport is needed to send the pilot back to the depot, if time permits, using a pilot boat is always more favorable than using a helicopter (cost of using a pilot boat is always lower than using a helicopter). Knowing this, we design the procedure shown in Algorithm 2 to calculate the (minimum) reduced cost of extending a partial route to the depot.

Suppose the reduced cost corresponding to a route $\mathbf{R} = [(i_0,t_0,z_0),(i_1,t_1,z_1),...,(i_n,t_n,z_n),...,(i_N,t_N,z_N),(i_{N+1},t_{N+1},z_{N+1})]$ returned by Algorithm 2 is negative (i.e., $RC < 0$). Then, we generate a column q such that $c_q = c_2 + \sum_{n=1}^{N} c_1^{i_n}(t_n - e_{i_n}) + \sum_{n=1}^{N+1} \mathrm{f}(z_n)$, and $s_{it}^q = 1$, if $i = i_n$ and $t = t_n$, $n = 1,...,N$ and $s_{it}^q = 0$, otherwise. q is then added into the corresponding LRMP, which will be solved again.

Algorithm 1 State extension procedure.

Input: (1) The current state s_n; (2) Vector $\boldsymbol{rst}_n^j$ for each task $j \in \Omega_n^r$;
Output: The set of states extended from s_n (denoted by ES);

$ES = \varnothing$;
for $j \in \Omega_n^r$ **do**
  if $\sigma_{i_n,j} = 0$ **then**
    for $t = het_j$ to hlt_j **do**
      $pt_{n'} = pt_n + (t - t_n)$;
      $\tilde{c}_{n'} = \tilde{c}_n + \bar{c}(i_n, t_n, z_n \to j, t, 0)$;
      $\Omega_{n'}^r = \mathrm{U}(\Omega_n^r, j, t, pt_{n'})$; ▷ The procedure $\mathrm{U}(\Omega_n^r, j, t, pt_{n'})$ for updating $\Omega_{n'}^r$ from Ω_n^r is presented in Algorithm 3 in Appendix A.
      $s_{n'} = [j, t, pt_{n'}, \tilde{c}_{n'}, \Omega_{n'}^r]$;
      $ES = ES \bigcup \{s_{n'}\}$;
    end for
  end if
  if $\sigma_{i_n,j} = 1$ **then**
    for $t = bet_j$ to blt_j **do**
      $pt_{n'} = pt_n + (t - t_n)$;
      $\tilde{c}_{n'} = \tilde{c}_n + \bar{c}(i_n, t_n, z_n \to j, t, 1)$;
      $\Omega_{n'}^r = \mathrm{U}(\Omega_n^r, j, t, pt_{n'})$;
      $s_{n'} = [j, t, pt_{n'}, \tilde{c}_{n'}, \Omega_{n'}^r]$;
      $ES = ES \bigcup \{s_{n'}\}$;
    end for
    for $t = het_j$ to hlt_j **do**
      $pt_{n'} = pt_n + (t - t_n)$;
      $\tilde{c}_{n'} = \tilde{c}_n + \bar{c}(i_n, t_n, z_n \to j, t, 2)$;
      $\Omega_{n'}^r = \mathrm{U}(\Omega_n^r, j, t, pt_{n'})$;
      $s_{n'} = [j, t, pt_{n'}, \tilde{c}_{n'}, \Omega_{n'}^r]$;
      $ES = ES \bigcup \{s_{n'}\}$;
    end for
  end if
end for

Algorithm 2 Column generation procedure.

Input: State s_n.
Output: The reduced cost of the column generated by extending the corresponding partial route (from node (i_n, t_n, z_n)) to the depot (denoted by RC);

if $\sigma_{i_n,0} = 0$ **then**
  $z_{n'} = 0$;
  $t_{n'} = t_n + d_{i_n}$;
end if
if $\sigma_{i_n,0} = 1$ **then**
  $z_{n'} = 1$;
  $t_{n'} = t_n + d_{i_n} + r_1$;
  if $pt_n + d_{i_n} + r_1 > \bar{D}$ **then**
    $z_{n'} = 2$;
    $t_{n'} = t_n + d_{i_n} + r_2$;
  end if
end if
$RC = \tilde{c}_n + \bar{c}(i_n, t_n, z_n \to 0, t_{n'}, z_{n'})$;

Dominance Rules. Two dominance rules are applied in the DP to speed up the searching process. We introduce them in the following two statements.

Proposition 1. *State s_{n_1} dominates state s_{n_2} if the following Conditions (55)-(61) hold:*

$$i_{n_1} = i_{n_2} \neq 0, \tag{55}$$

$$\tilde{c}_{n_1} \leq \tilde{c}_{n_2}, \tag{56}$$

$$\Omega^r_{n_1} \supseteq \Omega^r_{n_2}, \tag{57}$$

$$het^j_{n_1} \leq het^j_{n_2}, \quad j \in \Omega^r_{n_2}, \tag{58}$$

$$hlt^j_{n_1} \geq hlt^j_{n_2}, \quad j \in \Omega^r_{n_2}, \tag{59}$$

$$bet^j_{n_1} \leq bet^j_{n_2}, \quad j \in \Omega^r_{n_2}, \tag{60}$$

$$blt^j_{n_2} \geq blt^j_{n_2}, \quad j \in \Omega^r_{n_2}, \tag{61}$$

and Condition (62) or (63) holds:

$$\begin{cases} pt_{n_2} - t_{n_2} + het^j_{n_2} > \bar{D} - (2\hat{d}), \quad j \in \Omega^r_{n_2}, \quad \text{if } \sigma_{i_{n_2},0} = 0 & (a), \\ pt_{n_2} - t_{n_2} + het^j_{n_2} > \bar{D} - (\hat{d} + r_1), \quad j \in \Omega^r_{n_2}, \quad \text{if } \sigma_{i_{n_2},0} = 1 & (b), \end{cases} \tag{62}$$

$$pt_{n_1} - t_{n_1} \leq pt_{n_2} - t_{n_2}, \tag{63}$$

where $\hat{d} = \min_{i \in \Omega}\{d_i\}$.

Proof. To begin with, Equations (55) to (61) ensure that corresponding to any state $s_{n'_2} = [i_{n'_2}, t_{n'_2}, pt_{n'_2}, \tilde{c}_{n'_2}, \Omega^r_{n'_2}]$ that is directly extended from s_{n_2}, there exists a state $s_{n'_1} = [i_{n'_1}, t_{n'_1}, pt_{n'_1}, \tilde{c}_{n'_1}, \Omega^r_{n'_1}]$ that is directly extended from s_{n_1} such that:

$$i_{n'_1} = i_{n'_2}, \tag{64}$$

$$t_{n'_1} = t_{n'_2}, \tag{65}$$

$$\tilde{c}_{n'_1} \leq \tilde{c}_{n'_2}. \tag{66}$$

Besides, as for the relationship between $\Omega^r_{n'_1}$ and $\Omega^r_{n'_2}$, we distinguish the following cases:

Case 1. Equation (62a) holds. Note that the left-hand side of Equation (62a) calculates the lower bound of the pilot working time when task i_{n_2} starts, and the right-hand side of the equation calculates the upper bound of pilot working time after completing two additional tasks (including task i_{n_2}). Hence, in this case, $s_{n'_2}$ can no longer be extended to any new task due to Constraint (39). Besides, as $\sigma_{i_{n_2},0} = 0$, we have $\bar{c}(i_{n'_2}, t_{n'_2}, z_{n'_2} \rightarrow 0, t_{n''_2}, 0) = \bar{c}(i_{n'_1}, t_{n'_1}, z_{n'_1} \rightarrow 0, t_{n''_1}, 0) = 0$, where $t_{n''_1}$, $t_{n''_2}$ are obtained using Algorithm 2;

Case 2. Equation (62b) holds. Note that the left-hand side of Equation (62b) calculates the lower bound of the pilot working time when task i_{n_2} starts, and the right-hand side of the equation calculates the upper bound of pilot working time for completing task i_{n_2} and returning the depot via pilot boats. Hence, in this case, $s_{n'_2}$ can no longer be extended to any new task due to Constraint (39) (note that $r_2 < \hat{d}$). Besides, Equation (62b) indicates that $z_{n''_2} = 2$ in $\bar{c}(i_{n'_2}, t_{n'_2}, z_{n'_2} \rightarrow 0, t_{n''_2}, z_{n''_2})$ for any feasible $t_{n''_2}$. It easily follows that $\bar{c}(i_{n'_2}, t_{n'_2}, z_{n'_2} \rightarrow 0, t_{n''_2}, z_{n''_2}) \geq \bar{c}(i_{n'_1}, t_{n'_1}, z_{n'_1} \rightarrow 0, t_{n''_1}, z_{n''_1})$, where $t_{n''_1}$, $z_{n''_1}$, $t_{n''_2}$, and $z_{n''_2}$ are obtained using Algorithm 2;

Case 3. Equation (63) holds. According to Algorithm 3 in Appendix A (which generates $\Omega^r_{n'_1}$, $\Omega^r_{n'_2}$, $\boldsymbol{rst}^j_{n'_1}$'s, and $\boldsymbol{rst}^j_{n'_2}$'s), Equations (57) to (61) and (63) easily lead to:

$$\Omega^r_{n'_1} \supseteq \Omega^r_{n'_2}, \tag{67}$$

$$het^j_{n'_1} \leq het^j_{n'_2}, \ \ j \in \Omega^r_{n'_2}, \tag{68}$$

$$hlt^j_{n'_1} \geq hlt^j_{n'_2}, \ \ j \in \Omega^r_{n'_2}, \tag{69}$$

$$bet^j_{n'_1} \leq bet^j_{n'_2}, \ \ j \in \Omega^r_{n'_2}, \tag{70}$$

$$blt^j_{n'_2} \geq blt^j_{n'_2}, \ \ j \in \Omega^r_{n'_2}. \tag{71}$$

In addition, Algorithm 2 indicates that $\bar{c}(i_{n'_2}, t_{n'_2}, z_{n'_2} \rightarrow 0, t_{n''_2}, z_{n''_2}) \geq \bar{c}(i_{n'_1}, t_{n'_1}, z_{n'_1} \rightarrow 0, t_{n''_1}, z_{n''_1})$ for any $t_{n''_2}$.

Therefore, it now easily follows that for any state $s_{n^*_2} = [i_{n^*_2}, t_{n^*_2}, pt_{n^*_2}, \tilde{c}_{n^*_2}, \Omega^r_{n^*_2}]$ with $\boldsymbol{rst}^j_{n^*_2} = [het^j_{n^*_2}, hlt^j_{n^*_2}, bet^j_{n^*_2}, blt^j_{n^*_2}]$ for each $j \in \Omega^r_{n^*_2}$ that is either directly or indirectly extended from s_{n_2}, there exists a state $s_{n^*_1} = [i_{n^*_1}, t_{n^*_1}, pt_{n^*_1}, \tilde{c}_{n^*_1}, \Omega^r_{n^*_1}]$ that is either directly or indirectly extended from s_{n_1} such that the following conditions hold:

$$i_{n^*_1} = i_{n^*_2}, \tag{72}$$

$$t_{n^*_1} = t_{n^*_2}, \tag{73}$$

$$\tilde{c}_{n^*_1} \leq \tilde{c}_{n^*_2}, \tag{74}$$

$$\Omega^r_{n^*_1} \supseteq \Omega^r_{n^*_2}, \tag{75}$$

$$het^j_{n^*_1} \leq het^j_{n^*_2}, \ \ j \in \Omega^r_{n^*_2}, \tag{76}$$

$$hlt^j_{n_1^*} \geq hlt^j_{n_2^*}, \quad j \in \Omega^r_{n_2^*}, \tag{77}$$

$$bet^j_{n_1^*} \leq bet^j_{n_2^*}, \quad j \in \Omega^r_{n_2^*}, \tag{78}$$

$$blt^j_{n_2^*} \geq blt^j_{n_2^*}, \quad j \in \Omega^r_{n_2^*}, \tag{79}$$

$$\bar{c}(i_{n_1^*}, t_{n_1^*}, z_{n_1^*} \to 0, t_{n_1^{**}}, z_{n_1^{**}}) \leq \bar{c}(i_{n_2^*}, t_{n_2^*}, z_{n_2^*} \to 0, t_{n_2^{**}}, z_{n_2^{**}}), \tag{80}$$

where $t_{n_1^{**}}$, $z_{n_1^{**}}$, $t_{n_2^{**}}$, and $z_{n_2^{**}}$ in the last equation are obtained using Algorithm 2. Therefore, we have $s_{n_1^*}$ dominates $s_{n_2^*}$, and further, s_{n_1} dominates s_{n_2}. This completes the proof. □

Corollary 1. *For two states s_{n_1} and s_{n_2} that are extended from the same state, suppose the following conditions hold:*

$$i_{n_1} = i_{n_2} \neq 0, \tag{81}$$

$$t_{n_1} \leq t_{n_2}, \tag{82}$$

$$\tilde{c}_{n_1} \leq \tilde{c}_{n_2}, \tag{83}$$

then, s_{n_1} dominates s_{n_2}.

Proof. We only need to prove that Conditions (55)–(61) and Condition (62) or (63) hold for s_{n_1} and s_{n_2} in this corollary. First, Equations (81) and (83) are equivalent to Equations (55) and (56), respectively. Second, based on Algorithm 1, Equation (82) easily leads to

$$pt_{n_1} - t_{n_1} = pt_{n_2} - t_{n_2}, \tag{84}$$

which indicates Condition (63) holds. Third, based on Algorithm 3 and Equations (82) and (84), it is easy to infer that Conditions (57)–(61) hold as well. Therefore, the dominance relationship is proved. □

5.4. Branching Strategy

This section introduces how the B&B tree is explored and extended. To begin with, the B&B tree is explored in a balanced manner, i.e., we alternatively select the node with the overall minimum lower bound and the node with the minimum lower bound at the highest level to branch. Such an exploration method helps strike a balance between the speed of finding new integer solutions and the quality of them, both of which are important for finding tighter upper bounds.

After solving the LRMP at a node, if the obtained objective is no smaller the incumbent upper bound, the node is fathomed, otherwise, a branching operation is applied on the node. In particular, when generating new nodes, this paper adopts a two-level branching strategy, which combines a 0-1–arc branch at the primary level and a time-window branch at the secondary level. Details of the branching strategy are presented as follows.

Let $Q_s := \{q|\omega_q > 0, q \in Q\}$. Then, for each arc (i, j) (which denotes that position j is visited immediately after position i in the columns), we calculate:

$$\lambda_{ij} = \sum_{q \in Q_s} \varsigma_{ij}^q \omega_q, \ \ i \in \bar{\Omega}, j \in \bar{\Omega} \setminus \{i\}, \tag{85}$$

where $\varsigma_{ij}^q = 1$, if arc (i, j) is included in column q and 0, otherwise.

We distinguish the following three conditions regarding the solutions (ω_q's) in Q_s:

Condition (i) ω_q's are all integral;

Condition (ii) All or part of the ω_q's are fractional, and there are fractional λ_{ij}'s;

Condition (iii) All or part of the ω_q's are fractional, but all the λ_{ij}'s are integral.

Note that **Condition (iii)** is possible when and only when there exists at least two columns (q_1 and q_2) that belong to Q_s such that $0 < \omega_{q_1}, \omega_{q_2} < 1$ and q_1 and q_2 share the same task sequences (i.e., arcs) while the start times of certain tasks in them are different. Then, according to different conditions, we make different branching decisions.

If **Condition (i)** happens (i.e., $\omega_q = 0$ or 1, $q \in Q_s$), a valid upper bound to MP can be identified and the node is thus fathomed. Note that in practice, the time and cost to conduct any pilotage task are considerably larger than the time and cost required in any through-channel transport operation (i.e., in our models, $d_i > r_k$ and $c_3 > c_4^k$, $i \in \Omega$, $k \in \Theta$). Therefore, in any optimal integer solution to any LRMP each task is conducted only once in the columns with $\omega_q = 1$. This indicates that the solution is also feasible to M1.

If **Condition (ii)** happens, the node is branched on the λ_{ij} that is closest to 0.5, where ties are broken arbitrarily. Two new nodes will be generated in the branch. In particular, for the node generated by 0-branch, the LRMP of the node is imposed with an implicit constraint $\lambda_{ij} = 0$ (i.e., position j cannot be visited immediately after position i in any column). In this case, in the DP for solving the SP on the node (and its offsprings), we stop generating any route that travels from position i to position j. For the node generated by 1-branch, the LRMP of the node is imposed with an implicit constraint $\lambda_{ij} = 1$ (i.e., position j must be visited immediately after position i in at least one column). In this case, in the DP for solving the SP on the node (and its offsprings), we stop generating any route that travels from position i to position $j' \in \bar{\Omega} \setminus \{j\}$ or from $i' \in \bar{\Omega} \setminus \{i\}$ to position j.

If **Condition (iii)** happens we branch the node using the time-window branch. As mentioned above, in this condition, there are tasks with more than one start time in Q_s. Among these tasks, we identify the one with the minimum number of start times (denoted by i^*). Let $[st_{i^*}^1, st_{i^*}^2, ..., st_{i^*}^N]$ be an array of a total of N start times of task i^* in Q_s which are listed chronologically. Further, let $[\omega_q^1, \omega_q^2, ..., \omega_q^N]$ be an array of the solution values for columns corresponding to each start time in the former array. Then the node will be branched on $st_{i^*}^k$ such that $\sum_{u=1}^{k-1} \omega_q^u < 0.5$ and $\sum_{u=1}^{k} \omega_q^u \geq 0.5$, and we generate two new nodes where the time windows of task i^* are truncated to be $[e_{i^*}^0, st_{i^*}^k)$ and $[st_{i^*}^k, l_{i^*}^0]$ in the corresponding new LRMPs, respectively. Note that $[e_{i^*}^0, l_{i^*}^0]$ is the time window of task i^* in the node selected to branch. The time windows of task i^* are also adjusted accordingly when we solve the SP on the nodes and their offsprings using the DP.

5.5. Acceleration Techniques

To improve the efficiency of the B&P algorithm, we propose to accelerate the algorithm by using the following methods. When we are solving an LRMP at a node, it is unnecessary to solve the pricing problem (Section 5.3) to optimum in each iteration. Hence, two heuristical DP algorithms [denoted by SDP (Simple DP) and GDP (Greedy DP)] are designed to solve the SP. The two algorithms, at their cores, are truncated versions of the DP proposed in Section 5.3.2 [for notational simplicity, we denote it by FDP (Full DP) hereafter].

The SDP is reduced from the FDP by (a) only generating initial states each of which enables a task i to start exactly at its earliest possible start time(s) (i.e., e'_i, if $\sigma_{0,i} = 0$ or $\max\{e'_i, r_1\}$ and $\max\{e'_i, r_2\}$, if $\sigma_{0,i} = 1$) and (b) extending any state s_n only to states $s_{n'}$'s where task $i_{n'}$'s start exactly at their earliest feasible start times using pilot boats or helicopters as the transport vehicles between i_n and $i_{n'}$, i.e., het'_i and bet'_i (note that $het'_i = bet'_i$ if no through-channel transport is required between i_n and $i_{n'}$).

The GDP reduces the searching space by only generating a subset of the initial states. In particular, corresponding to each task $i \in \Omega$, let Δ_i denote the set of all feasible start times of i. Then, for each $t \in \Delta_i$, (1) if $\sigma_{0,i} = 0$ we generate initial states $[0, t, 0, 0, \Omega_0]$ only when there exist no $t' \in \Delta_i$ such that $t' < t$ and $\bar{c}(0, t', 0 \rightarrow i, t', 0) < \bar{c}(0, t, 0 \rightarrow i, t, 0)$, and (2) if $\sigma_{0,i} = 1$ we generate initial states $[0, t - r_1, 0, 0, \Omega_0]$ only when there exist no $t' \in \Delta_i$ such that $t' < t$ and $\min\{\bar{c}(0, t' - r_1, 0 \rightarrow i, t', 1), \bar{c}(0, t' - r_2, 0 \rightarrow i, t', 2)\} < \bar{c}(0, t - r_1, 0 \rightarrow i, t, 1)$ and initial states $[0, t - r_2, 0, 0, \Omega_0]$ only when there exist no $t' \in \Delta_i$ such that $t' < t$ and $\min\{\bar{c}(0, t' - r_1, 0 \rightarrow i, t', 1), \bar{c}(0, t' - r_2, 0 \rightarrow i, t', 2)\} < \bar{c}(0, t - r_2, 0 \rightarrow i, t, 2)$. The extension procedure of the GDP is the same as that of the FDP.

To further accelerate the algorithm, we allow the SDP, GDP, and FDP to stop prematurely, after a total of MRC_s, MRC_g, and MRC_f columns with negative reduced costs are generated by the three DPs, respectively. In particular, when solving the SP, SDP is firstly used, and GDP is only used when SDP fails to provide MRC_s columns with negative reduced costs. Then, FDP is only used when the other two fail to deliver MRC_g columns with negative reduced costs. Finally, the FDP stops when MRC_f columns with negative reduced costs have been found by the DPs.

6. Numerical Experiments

In this section, we perform extensive computational experiments to verify the applicability and effectiveness of our proposed models and solution methods. To do so, we generated 210 groups of instances based on real cases faced by the VTS operators of Port of Xiamen in China (the 14th largest container port in the world). The instance groups have different planning horizons, numbers of tasks, and tidal conditions. Each group contains five randomly generated instances, resulting in a total of 1050 instances.

We solve all instances with the proposed B&P method and a heuristic that simulates the First-Come-First-Served (FCFS) policy that is commonly used in practice. Details of the heuristic will be elaborated in Section 6.1. In addition, the instances with the smallest scales are also solved by the well-known commercial MIP solver CPLEX using model M1. The algorithms are coded in C++ language, and CPLEX 12.6 is used as the LP solver in the B&P method and as the MIP solver for M1. The experiments are conducted on an Intel Core $i7$ 3.60 GHz PC with 16 GB RAM.

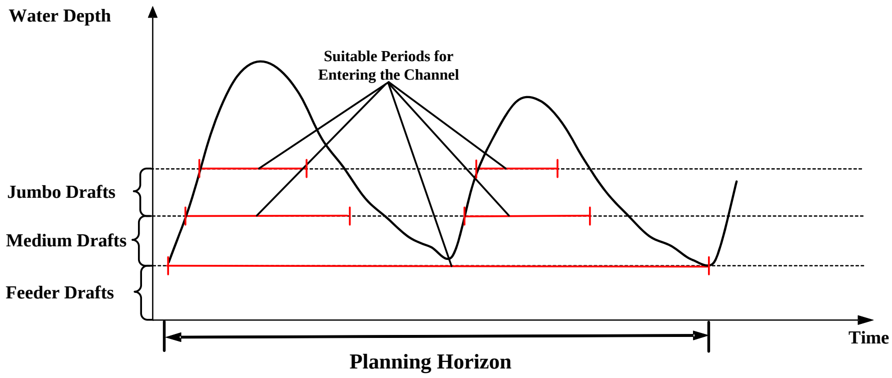

Figure 3: Water depth in the channel of a tidal port.

6.1. Instance Generation and Algorithm Settings

To test the performance of the model and the algorithm, 210 groups of instances were generated. Each group can be denoted using a three-field notation "PL|TN|TC", where "PL" stands for the length of the planning horizon (days), "TN" stands for the number of tasks and "TC" indicates whether the port is tidal or not. In these instance groups, PL changes from 1 to 5, representing 1 to 5 days, TN changes from 6 to 100 (the average number of tasks in a day is set to be within [6, 20]), and TC takes two values, where "T" represents tidal cases and "U" stands for untidal cases.

In all instances, the unit time (i.e., the interval between two consecutive time points) is set as 15 minutes, hence, a day is divided into 96 unit times, and we suppose the arrival times of entering vessels and the handling completion times of the leaving vessels are uniformly distributed in the planning horizon. We consider three classes of vessels: Jumbo, Medium, and Feeder. In tidal cases, as shown in Figure 3, Jumbo and Medium vessels can only enter the channel during high-tide periods due to their large drafts, while Feeders can enter the channel at any time. Table 3 shows the settings for vessels of different sizes. Note that in the table, "Delay cost" refers to the per unit time costs caused by the delays to start the tasks for vessels of a certain class. Besides, for Jumbo and Medium vessels, the lengths of time windows to start the in-wharf or out-wharf tasks are dependent on the tidal conditions of the channels and the times when the vessels arrive or complete handling.

Table 3: Settings of vessels.

Class	Ratio	Delay cost	Time Window Length	
			Tidal Cases	Untidal Cases
Jumbo	1/3	$[25, 34]$	1–5 hours	5 hours
Medium	1/3	$[13, 22]$	2–5 hours	5 hours
Feeder	1/3	$[1, 10]$	5 hours	5 hours

It is further supposed that the durations of all tasks are identical in an instance. The duration (d) is generated from the uniform distribution on $[4, 10]$ (unit times). The capacities u_1 and u_2 of a channel are set according to the duration of a task, which are calculated by

$$u_1 = u_2 = \lfloor \theta d \rfloor, \tag{86}$$

where θ is generated from the uniform distribution on $[0.75, 1.5]$.

The maximum length of a pilot shift is set as eight hours (32 unit times) in all instances. The costs of appointing an additional pilot shift and assigning additional pilotage service are generated randomly in the ranges of $[200, 300]$ and $[40, 60]$, respectively. As for the two pilot transport vehicles, the times for transporting a pilot through the channel using pilot boats and helicopters are set to be $\lfloor 0.5d \rfloor$, and $\lceil 0.1d \rceil$, respectively. In addition, the costs for transporting pilots are generated randomly from the ranges of $[1, 6]$ for using pilot boats and $[10, 15]$ for using helicopters, respectively.

Parameters in the algorithms are set as follows. First, for the B&P algorithm, when solving the SP, the SDP is first used. Then, the GDP is used when no more than 5 columns with negative reduced costs are found by the SDP. The FDP is only used when no more than 10 such columns are found by the SDP and GDP. Finally, we let the FDP stop when 20 columns with negative reduced costs are found (i.e., we set $MRC_s = 5$, $MRC_g = 10$, and $MRC_f = 20$). We also set the time limits for both the B&P algorithm and CPLEX to be 3600 seconds.

In practice, FCFS policy is commonly used by the VTS operators (e.g., the VTS operators in Port of Xiamen) to schedule traffic in the channel and assign pilots to serve the vessels. To simulate the policy, we design a heuristic algorithm in which tasks are executed in an order that sequences their earliest start times chronologically, where ties are broken arbitrarily. The algorithm selects through-channel transport vehicles myopically. That is, when a transport operation is needed, the algorithm first calculates for each feasible type of vehicles (k) a cost (vc_k) by adding the transport cost to the delay cost of the following task (if the transport operation leads the pilot to a new task), then the one with lower vc_k is selected to be type of vehicles used in the transport operation. Note that the pilot should be able to return to the depot within the allowable working period after the transport operation and completing the following task (if any). In each pilot shift (denoted by PS), the first task (denoted by FT) starts at its earliest feasible start time (denoted by ET) by considering capacity limits of the channels, and under the transport-vehicle-selection rule given above. Accordingly, the start time of the pilot shift PS is set to be the latest time that enables the task FT to start at time ET under the transport-vehicle-selection rule. In PS, the following tasks start at their earliest feasible start times under channel capacity constraints and the transport-vehicle-selection rule, as well. Besides, during the scheduling process, pilot shifts are utilized in such a way to enable as many as possible tasks to be conducted in one pilot shift. In the remaining part of the paper, we refer to this heuristic as the FCFS heuristic.

6.2. Computational Results

This section reports the computational results of the numerical experiments. For a better illustration, we report the results by dividing the instances into four parts. In the first part, we report the results of instances with the smallest sizes (i.e., instances with PL= 1). The remaining instances (i.e., instances with PL$\geq$ 2) are divided into three parts according to the traffic densities of the instances, including the "Low Densities" ($6 \leq$ TN/PL ≤ 10), "Medium Densities" ($11 \leq$ TN/PL ≤ 15), and "High Densities" ($16 \leq$ TN/PL ≤ 20). The performances of the solution methods are reported in group-aggregated values according to their partition into the 210 groups. Each partition contains five instances. Finally, we summarize and discuss the main findings from the experiments at the end of this section.

6.2.1. *Computational Results of Instances with the Smallest Sizes*

The first part covers instances with the smallest sizes (PL=1). Instances in this part are solved using the proposed B&P method, CPLEX using M1, and the FCFS heuristic. Table 4 compares the performances of these methods for solving these instances. Note that because the B&P method managed to obtain optimal solutions for all instances in this part, we only report its average solution time for different instance groups. "GAP_c" reports average gaps (in percentage) of solutions obtained by CPLEX against the best lower bounds delivered by CPLEX for instances in a group. "GAP_cb" and "GAP_fb" report average gaps (in percentage) of solutions delivered by CPLEX and the FCFS heuristic for instances in a group against those obtained by the B&P method (the optimal solutions), respectively. "US" reports the number of instances in a group that cannot be solved to optimum and "NS" reports the number of instances in a group for which no feasible solutions are reported. "Average (U)" and "Average (T)" report average values for all untidal and all tidal instances, respectively.

Table 4 shows that the B&P method can obtain optimal solutions for all instances in very short times. In comparison, CPLEX takes much longer to solve the instances and cannot solve more than half of the instances to optimum. It even fails to find feasible solutions for a considerable number of instances within the time limit. As for the FCFS heuristic, it provides solutions that are much worse than those obtained by the B&P method (the optimal solutions). The FCFS heuristic also fails to obtain feasible solutions for several tidal instances. Therefore, the B&P method outperforms CPLEX and the FCFS heuristic when solving small-sized instances.

6.2.2. *Computational Results of Instances with Low Traffic Densities*

In this part, we report the results of instances with the planning horizons of 2 to 5 days and 6 to 10 tasks a day on average. The instances are solved using the B&P method and the FCFS heuristic. The results are shown in Table 5. Similar to the previous section, the B&P method manages to obtain optimal solutions for all instances in this part. Hence, we only report the average solution time for different instance groups. "GAP_fb" reports average gaps (in percentage) of solutions delivered by the FCFS heuristic for instances in a group against those obtained by the B&P method (the optimal solutions). "NS" reports the number of instances in a group for which no feasible solutions are reported. "Average (U)" and "Average (T)" report average values for all untidal and all tidal instances, respectively.

As shown in Table 5, for all instances with low traffic densities, the B&P method can obtain optimal solutions in short computational time, and the solutions are much better than those delivered by the FCFS heuristic. Note that when the traffic densities are low, the FCFS can find feasible solutions for all instances under both tidal and untidal conditions.

6.2.3. *Computational Results of Instances with Medium Traffic Densities*

This section reports the computational results of instances where $2 \leq \text{PL} \leq 5$ and $11 \leq \text{TN/PL} \leq 15$. Table 6 demonstrates the results obtained using the B&P method and the FCFS heuristic to solve these instances. The B&P method can provide feasible solutions for all instances; however, the optimality of some solutions cannot be proved within the time limit. "GAP_b" reports average gaps (in percentage) of solutions delivered by the B&P method for instances in a group against the minimum lower bounds of the unfathomed nodes in the B&B trees. "GAP_fb" reports average gaps (in percentage) of solutions delivered by the FCFS heuristic for instances in a group against those obtained by the B&P method. "US" reports the number of instances in a group that cannot be solved to optimum and "NS" reports the number of instances in a group for which no feasible

Table 4: Results of small-sized instances.

Group (PL|TN|TC)	B&P Time(s)	CPLEX GAP$_{c}$	GAP$_{cb}$	Time(s)	US	NS	FCFS GAP$_{fb}$	NS
1|6|U	0.03	0.00	0.00	4.51	0	0	20.71	0
1|7|U	0.03	0.00	0.00	3.66	0	0	11.09	0
1|8|U	0.06	0.00	0.00	311.81	0	0	20.22	0
1|9|U	0.06	2.76	0.00	1393.61	1	0	11.99	0
1|10|U	0.05	0.08	0.00	765.23	1	0	18.31	0
1|11|U	0.12	18.06	0.08	3086.70	4	0	37.04	0
1|12|U	0.10	0.48	0.00	1980.57	1	0	28.16	0
1|13|U	0.08	17.23	0.84	3073.73	3	0	20.61	0
1|14|U	0.06	31.43	5.14	3600.00	5	0	46.55	0
1|15|U	0.21	40.97	6.90	3600.00	5	0	37.42	0
1|16|U	0.43	35.19	2.53	3600.00	5	2	45.85	0
1|17|U	19.73	34.69	9.02	3600.00	5	3	46.23	0
1|18|U	6.13	36.53	6.18	3600.00	5	1	39.96	0
1|19|U	0.88	–	–	3600.00	5	5	41.22	0
1|20|U	0.78	–	–	3600.00	5	5	55.67	0
1|6|T	0.03	0.00	0.00	7.37	0	0	10.95	0
1|7|T	0.05	0.00	0.00	22.16	0	0	12.45	0
1|8|T	0.04	0.00	0.00	31.76	0	0	22.14	0
1|9|T	0.04	0.00	0.00	68.00	0	0	21.50	0
1|10|T	0.15	0.00	0.00	1698.12	0	0	16.54	0
1|11|T	0.06	9.56	0.00	3008.52	4	0	22.43	0
1|12|T	0.06	16.35	0.05	3600.00	5	0	23.58	0
1|13|T	0.14	25.18	0.13	3600.00	5	0	28.82	0
1|14|T	0.16	30.27	2.27	3600.00	5	0	32.29	0
1|15|T	0.27	34.56	5.50	3600.00	5	0	33.63	0
1|16|T	0.09	36.77	4.29	3600.00	5	0	27.46	1
1|17|T	0.17	34.25	5.00	3600.00	5	1	32.51	0
1|18|T	0.48	26.80	2.69	3600.00	5	4	34.60	0
1|19|T	0.51	33.75	6.42	3600.00	5	4	36.33	0
1|20|T	10.54	35.34	7.39	3600.00	5	4	45.54	1
Average (U)	1.92	16.72	2.36	2387.99	3.00	1.07	32.07	0
Average (T)	0.85	18.86	2.25	2482.40	3.27	0.87	26.72	0.13

Note 1. Values in columns “CPLEX” or “FCFS” are calculated based on the instances for which CPLEX or the FCFS heuristic is able to deliver feasible solutions.

Note 2. Values in “CPLEX” are displayed as “–” if CPLEX fails to deliver feasible solutions for all instances in a group.

Table 5: Results of instances with low traffic densities.

Group	B&P	FCFS		Group	B&P	FCFS	
(PL|TN|TC)	Time(s)	GAP_{fb}	NS	(PL|TN|TC)	Time(s)	GAP_{fb}	NS
2|12|U	0.14	15.53	0	2|12|T	0.09	9.83	0
2|14|U	0.29	19.03	0	2|12|T	0.06	15.46	0
2|16|U	0.18	19.71	0	2|12|T	0.16	23.89	0
2|18|U	0.31	22.43	0	2|12|T	0.36	16.68	0
2|20|U	0.66	31.77	0	2|12|T	0.34	21.46	0
3|18|U	0.14	15.88	0	2|12|T	0.16	18.26	0
3|21|U	0.11	18.82	0	2|12|T	1.48	17.34	0
3|24|U	0.38	22.64	0	2|12|T	0.48	20.72	0
3|27|U	0.67	20.82	0	2|12|T	0.27	24.91	0
3|30|U	1.77	19.63	0	2|12|T	1.83	21.87	0
4|24|U	0.98	18.38	0	2|12|T	0.60	11.00	0
4|28|U	1.58	16.23	0	2|12|T	0.53	16.01	0
4|32|U	1.13	15.13	0	2|12|T	8.88	14.81	0
4|36|U	15.39	19.14	0	2|12|T	6.13	19.08	0
4|40|U	1.53	25.49	0	2|12|T	23.64	23.31	0
5|30|U	0.47	11.53	0	2|12|T	0.75	13.34	0
5|35|U	1.75	15.82	0	2|12|T	0.46	17.78	0
5|40|U	4.36	16.46	0	2|12|T	1.21	15.18	0
5|45|U	87.42	20.70	0	2|12|T	26.84	18.97	0
5|50|U	703.84	28.60	0	2|12|T	458.88	20.48	0
Average (U)	41.16	19.69	0.00	Average (T)	26.66	18.02	0.00

solutions are reported. “Average (U)” and “Average (T)” report average values for all untidal and all tidal instances, respectively.

Table 6 shows that the B&P method can obtain optimal or near-optimal solutions for all instances in this part within reasonable time. In particular, most instances were solved to the proven optimum. The instances that could not be solved to optimum are also with small optimality gaps. In comparison, the solutions obtained by the FCFS heuristic are much worse. The results also indicate the B&P can better solve instances under tidal conditions such that more optimal solutions are found within shorter computational times. Moreover, under medium traffic densities, the FCFS heuristic can still find feasible solutions for all instances without tidal constraints but failed to provide feasible solutions for several instances with tidal constraints.

6.2.4. Computational Results of Instances with High Traffic Densities

In this section, we report the computational results of instances with high traffic densities (i.e., instances that cover 2 to 5 days of planning horizons and have 15 to 20 tasks a day on average). The solutions obtained using the B&P method and the FCFS heuristic are shown in Table 7. “GAP_{b}” reports average gaps (in percentage) of solutions delivered by the B&P method for instances in a group against the minimum lower bounds of the unfathomed nodes in the B&B trees. “GAP_{fb}” reports average gaps (in percentage) of solutions delivered by the FCFS for instances in a group against those obtained by the B&P method. “US” reports the number of instances in a group that cannot be solved to optimum and “NS” reports the number of instances in a group for which no feasible solutions are reported. “Average (U)” and “Average (T)” report average values for all untidal and all tidal instances, respectively.

As shown in Table 7, feasible solutions were found by the B&P method for all instances although

Table 6: Results of instances with medium traffic densities.

Group	B&P			FCFS		Group	B&P			FCFS	
(PL|TN|TC)	GAP_b	Time(s)	US	GAP_{fb}	NS	(PL|TN|TC)	GAP_b	Time(s)	US	GAP_{fb}	NS
2|22|U	0.00	0.32	0	30.89	0	2|22|T	0.00	0.20	0	27.66	0
2|24|U	0.00	0.70	0	23.34	0	2|24|T	0.00	0.85	0	19.43	0
2|26|U	0.00	0.49	0	42.09	0	2|26|T	0.00	1.70	0	25.45	0
2|28|U	0.00	2.99	0	28.74	0	2|28|T	0.00	0.54	0	28.47	0
2|30|U	0.00	1.32	0	41.87	0	2|30|T	0.00	430.13	0	28.72	0
3|33|U	0.00	6.86	0	20.27	0	3|33|T	0.00	31.93	0	19.98	0
3|36|U	0.00	18.62	0	33.19	0	3|36|T	0.00	27.16	0	21.79	0
3|39|U	0.09	751.09	1	30.29	0	3|39|T	0.00	0.98	0	23.94	0
3|42|U	0.00	206.37	0	29.98	0	3|42|T	0.00	4.20	0	29.34	0
3|45|U	0.00	37.57	0	35.26	0	3|45|T	0.00	24.85	0	39.15	1
4|44|U	0.05	746.54	1	27.85	0	4|44|T	0.00	3.00	0	21.70	0
4|48|U	0.03	815.48	1	28.57	0	4|48|T	0.00	26.57	0	24.18	0
4|52|U	0.00	646.52	0	30.14	0	4|52|T	0.16	799.06	1	26.00	1
4|56|U	0.00	114.50	0	33.59	0	4|56|T	0.00	90.56	0	33.10	0
4|60|U	0.08	954.50	1	31.74	0	4|60|T	0.00	90.15	0	30.44	0
5|55|U	0.00	407.70	0	25.13	0	5|55|T	0.18	770.91	1	28.03	0
5|60|U	0.13	796.98	1	31.20	0	5|60|T	0.00	224.34	0	23.63	0
5|65|U	0.08	900.63	1	31.64	0	5|65|T	0.00	1091.42	0	26.42	0
5|70|U	0.43	1672.09	2	31.84	0	5|70|T	0.04	2036.05	2	27.61	1
5|75|U	0.35	2138.80	2	33.89	0	5|75|T	0.06	1618.22	1	27.60	2
Average (U)	0.06	511.00	0.50	31.08	0.00	Average (T)	0.02	364.64	0.25	26.63	0.25

Table 7: Results of instances with high traffic densities.

Group	B&P			FCFS		Group	B&P			FCFS	
(PL|TN|TC)	GAP_b	Time(s)	US	GAP_{fb}	NS	(PL|TN|TC)	GAP_b	Time(s)	US	GAP_{fb}	NS
2|32|U	0.00	10.55	0	32.87	0	2|32|T	0.00	0.99	0	36.19	0
2|34|U	0.00	16.98	0	38.38	0	2|34|T	0.00	1.38	0	31.79	1
2|36|U	0.00	28.07	0	54.88	0	2|36|T	0.00	0.36	0	37.86	1
2|38|U	0.00	97.02	0	47.49	0	2|38|T	0.00	10.59	0	31.81	0
2|40|U	0.12	730.24	1	58.13	0	2|40|T	0.00	212.29	0	39.09	0
3|48|U	0.00	17.48	0	39.63	0	3|48|T	0.00	14.84	0	31.26	0
3|51|U	0.00	116.03	0	39.89	0	3|51|T	0.01	725.23	1	36.06	0
3|54|U	0.25	1159.08	1	38.29	0	3|54|T	0.00	56.12	0	31.42	0
3|57|U	0.00	1156.10	0	52.73	0	3|57|T	0.07	829.26	1	29.95	1
3|60|U	0.00	348.50	0	48.64	0	3|60|T	0.00	207.56	0	31.89	0
4|64|U	0.10	1460.79	2	39.72	0	4|64|T	0.49	1722.63	2	30.86	0
4|68|U	0.59	2588.48	3	38.44	0	4|68|T	0.35	1478.61	2	33.24	1
4|72|U	0.04	1467.27	1	42.83	0	4|72|T	0.42	1458.70	2	41.07	1
4|76|U	0.13	1298.83	1	48.75	0	4|76|T	0.00	166.78	0	37.87	0
4|80|U	0.16	1593.76	2	53.21	0	4|80|T	0.07	1214.88	1	34.91	2
5|80|U	0.50	1183.37	1	40.25	0	5|80|T	0.75	2171.57	3	28.56	1
5|85|U	0.14	2489.50	2	35.37	0	5|85|T	0.27	949.83	1	30.88	0
5|90|U	0.00	574.47	0	45.81	0	5|90|T	0.33	2189.52	3	34.13	1
5|95|U	0.55	2246.27	3	46.25	0	5|95|T	0.32	2520.26	3	32.98	0
5|100|U	0.90	1766.41	2	45.28	0	5|100|T	0.00	332.50	0	36.10	0
Average (U)	0.17	1017.46	0.95	44.34	0.00	Average (T)	0.15	815.19	0.95	33.90	0.45

some of them may not be optimal. The B&P method can solve most of the instances to optimum and obtain near-optimal solutions for the remaining ones and thus, it obviously outperforms the FCFS heuristic. In addition, the shorter average computational times and smaller average gaps indicate the B&P method can better solve tidal cases than untidal cases. Under high traffic densities, the FCFS can still provide feasible solutions for all untidal instances. However, it failed to find feasible solutions for more tidal instances when compared with results of the instances with medium traffic densities.

6.2.5. Summary and Discussion

In the experiments, we have tested the performances of the B&P algorithm on a large set of instances with different parameter settings. The results demonstrate that the algorithm is able to obtain optimal or near-optimal solutions for all instances within short computational time. The B&P algorithm outperforms the well-known optimization software CPLEX and a scheduling policy that is commonly used in practice. Hence, the proposed B&P algorithm is able to provide high-quality solutions to support decision-making in real applications in reasonable times.

The B&P algorithm significantly outperforms the FCFS policy in almost all instances. In particular, gaps between the solutions of the two methods are between around 20% to over 40%, and even in low-traffic instances where the traffic in the channels is relatively sparse the gaps between the two methods still reach 18%. This indicates that our method which tackles the channel traffic management and the pilotage arrangement in a joint manner generates considerable benefits to seaports when compared with the method that handles the two decisions sequentially (as in the FCFS heuristic).

It is also mentionable that the gaps between the solutions of the B&P algorithm and the FCFS heuristic grow with the densities in the channels. Besides, B&P algorithm gains greater superiority against the FCFS heuristic in terms of the solution quality when solving untidal instances. This indicates the B&P algorithm brings more values to the port in busy periods and when there are more flexibilities in channel traffic control (e.g., longer time windows to start pilotage tasks).

We have also found that when solving an instance, the B&P algorithm typically generates fewer pilot shifts than the FCFS heuristic does. Therefore, by using the B&P algorithm, we can lessen the work burdens for the pilots by enabling them to have longer continuous resting periods, which is important for mitigating fatigue of the pilots (see the discussion on the importance of the continuity of resting time by International Maritime Organization, 2001). Moreover, with fewer pilot shifts, it will be easier for the VTS to roster pilots to work in the shifts.

In addition to providing a better pilotage arrangement for a seaport, the B&P algorithm also helps improve the efficiency of other operations in the port. For example, as shown in Tables 4, 6, and 7, the FCFS policy cannot find feasible solutions to some instances. In these cases, the port may have to revise the vessel handling plan, leading to losses in the productivity of the port and prolonged turnaround times of vessels.

By solving the PPP, we obtain the (sub)optimal number of pilot shifts to be set in a planning horizon and their start and end times. Using this information, the VTS operators are able to generate a rostering plan that assigns particular pilots to the shifts by taking into account practical considerations (e.g., relevant governmental and industrial regulations regarding workforce management).

7. Conclusions

Pilotage is compulsory for vessels that need to pass through the navigation channel in most seaports around the world. Managing the vessel traffic and scheduling the pilots for vessel navigation in the seaside of a port is important for congestion mitigation and vessel service enhancement. This paper studies the PPP that arise in the practical operations in seaports. The problem aims to schedule the vessel traffic in the navigation channel, assign work shifts to pilots, and schedule the pilot in each work shift for vessel navigation. We formulate the problem as a MIP model, and show that the problem is strongly NP-hard. For solving the problem, we provide a set covering reformulation of the model and develop a B&P algorithm. Our algorithm makes use of a two-level branching scheme which performs 0-1 arc branch at the primary level and performs time-window branch at the secondary level, and solves each node of the branch-and-bound tree using a column generation algorithm in which the pricing problem is solved by a tailored DP. We also develop acceleration strategies for improving the efficiency of the B&P algorithm. We perform extensive numerical experiments to evaluate the performances of the proposed model and the algorithm. Computational results show that our solution method can efficiently solve problem instances of practical sizes, and that our solution method outperforms a standard MIP solver and a method used in practice in terms of solution quality and computation efficiency.

Although our model is designed for pilotage planning in the navigation channel of a seaport, the model may also be applicable to pilotage planning in waterways such as canals and straits. Hence, future research could focus on extending our model and solution method for solving pilotage planning problems in a variety of waterways. In a seaport, the service of vessels not only involves the utilization of seaside resources (e.g., navigation channels and pilots), but also involves the utilization of landside facilities, such as the berths, the quay cranes, and the yard. Another interesting research direction would be to investigate the possibilities of integrating the PPP into terminal resource planning problems to improve the efficiency of port operations and enhance the service levels of vessels.

References

Bierwirth, C., Meisel, F., 2010. A survey of berth allocation and quay crane scheduling problems in container terminals. European Journal of Operational Research 202, 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, 675–689.

Carlo, H.J., Vis, I.F., Roodbergen, K.J., 2015. Seaside operations in container terminals: literature overview, trends, and research directions. Flexible Services and Manufacturing Journal 27, 224–262.

Choo, S., Klabjan, D., Simchi-Levi, D., 2010. Multiship crane sequencing with yard congestion constraints. Transportation Science 44, 98–115.

Cruz, R., Mendes, A.B., Bahiense, L., Wu, Y., 2019. Integrating berth allocation decisions in a fleet composition and periodic routing problem of platform supply vessels. European Journal of Operational Research 275, 334–346.

Desaulniers, G., Desrosiers, J., Solomon, M.M., 2006. Column Generation. volume 5. Springer Science & Business Media.

Ding, Y., Jia, S., Gu, T., Li, C.L., 2016. SGICT builds an optimization-based system for daily berth planning. Interfaces 46, 281–296.

Eddison, R.T., Owen, D.G., 1953. Discharging iron ore. Journal of the Operational Research Society 4, 39–50.

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

Galle, V., Barnhart, C., Jaillet, P., 2018. Yard crane scheduling for container storage, retrieval, and relocation. European Journal of Operational Research 271, 288–316.

Giallombardo, G., Moccia, L., Salani, M., Vacca, I., 2010. Modeling and solving the tactical berth allocation problem. Transportation Research Part B: Methodological 44, 232–245.

International Maritime Organization, 1997. Guidelines for vessel traffic services, in: Resolution A.857(20). URL: http://www.maritime-vts.co.uk/A857.pdf.

International Maritime Organization, 2001. Guidelines on fatigue mitigation and management. URL: http://www.imo.org/en/OurWork/HumanElement/VisionPrinciplesGoals/Documents/1014.pdf.

Jia, S., Li, C.L., Xu, Z., 2019. Managing navigation channel traffic and anchorage area utilization of a container port. Transportation Science 53, 728–745.

Lalla-Ruiz, E., Shi, X., Voß, S., 2016. The waterway ship scheduling problem. Transportation Research Part D: Transport and Environment 60, 191–209.

Lübbecke, M.E., Desrosiers, J., 2005. Selected topics in column generation. Operations Research 53, 1007–1023.

Meng, Q., Wang, T., Wang, S., 2012. Short-term liner ship fleet planning with container transshipment and uncertain container shipment demand. European Journal of Operational Research 223, 96–105.

Ng, M.W., 2017. Revisiting a class of liner fleet deployment models. European Journal of Operational Research 257, 773–776.

Robenek, T., Umang, N., Bierlaire, M., Ropke, S., 2014. A branch-and-price algorithm to solve the integrated berth allocation and yard assignment problem in bulk ports. European Journal of Operational Research 235, 399–411.

Song, D.P., Carter, J., 2009. Empty container repositioning in liner shipping. Maritime Policy & Management 36, 291–307.

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

UNCTAD, 2018. Review of Maritime Transportation 2018, in: Paper presented at the United Nations Conference on Trade and Development, New York and Geneva. URL: http://unctad.org/en/PublicationsLibrary/rmt2018_en.pdf.

Vacca, I., Salani, M., Bierlaire, M., 2013. An exact algorithm for the integrated planning of berth allocation and quay crane assignment. Transportation Science 47, 148–161.

Wermus, M., Pope, J.A., 1994. Scheduling harbor pilots. Interfaces 24, 44–52.

Xu, Z., Lee, C.Y., 2018. New lower bound and exact method for the continuous berth allocation problem. Operations Research 66, 778–798.

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, L.H., Chew, E.P., 2017. Daily berth planning in a tidal port with channel flow control. Transportation Research Part B: Methodological 106, 193–217.

Zhen, L., Xu, Z., Wang, K., Ding, Y., 2016. Multi-period yard template planning in container terminals. Transportation Research Part B: Methodological 93, 700–719.

Appendix A. The Reachable Task Identification Procedure

The following pseudo code in Algorithm 3 demonstrates how the $\Omega^r_{n'}$ and the corresponding $\boldsymbol{rst}^j_{n'}$'s are obtained when extending a state (s_n) to another state ($s_{n'}$).

Algorithm 3 Reachable task identification procedure.

Input: (1) Parameters in the current state (s_n): Ω^r_n, and the (truncated) earliest and latest start times (e'_j and l'_j for each task $j \in \Omega^r_n$); (2) Parameters in the extended state ($s_{n'}$): $i_{n'}, t_{n'}, pt_{n'}$; (3) Times of through-channel transport using two types of vehicles (r_1 and r_2);

Output: Set of reachable tasks $\Omega^r_{n'}$ and for each task $j \in \Omega^r_{n'}$, vector $\boldsymbol{rst}^j_{n'}$;

$\Omega^r_{n'} = \varnothing$;
for $j \in \Omega^r_n \backslash \{i_{n'}\}$ **do**
 $trat = 0$;
 if $\sigma_{j,0} = 1$ **then**
  $trat = r_2$;
 end if
 if $\sigma_{i_{n'},j} = 0$ **then**
  $het^j_{n'} = \max\{e'_j, t_{n'} + d_{i_{n'}}\}$;
  $hlt^j_{n'} = \min\{l'_j, t_{n'} + \bar{D} - pt_{n'} - d_j - trat\}$;
  if $het^j_{n'} \leq hlt^j_{n'}$ **then**;
   $\Omega^r_{n'} = \Omega^r_{n'} \bigcup \{j\}$;
   $\boldsymbol{rst}^j_{n'} = [het^j_{n'}, hlt^j_{n'}, het^j_{n'}, hlt^j_{n'}]$;
  end if
 end if
 if $\sigma_{i_{n'},j} = 1$ **then**
  $het^j_{n'} = \max\{e'_j, t_{n'} + d_{i_{n'}} + r_2\}$;
  $hlt^j_{n'} = \min\{l'_j, t_{n'} + \bar{D} - pt_{n'} - d_j - r_2 - trat\}$;
  $bet^j_{n'} = \max\{e'_j, t_{n'} + d_{i_{n'}} + r_1\}$;
  $blt^j_{n'} = \min\{l'_j, t_{n'} + \bar{D} - pt_{n'} - d_j - r_1 - trat\}$;
  if $het^j_{n'} \leq hlt^j_{n'}$ **then**
   $\Omega^r_{n'} = \Omega^r_{n'} \bigcup \{j\}$;
   $\boldsymbol{rst}^j_{n'} = [het_j, hlt_j, |T| + 1, -1]$; ▷ We initialize bet_j and blt_j to be $|T| + 1$ and -1, respectively, which indicates that using pilot boats is not a feasible option for transporting the pilot.
   if $bet^j_{n'} \leq blt^j_{n'}$ **then**
    $\boldsymbol{rst}^j_{n'} = [het_j, hlt_j, bet_j, blt_j]$;
   end if
  end if
 end if
end for