Models and algorithms for an integrated vessel scheduling and tug assignment problem within a canal harbor

Matteo Petris, Paola Pellegrini, Raffaele Pesenti

Models and algorithms for an integrated vessel scheduling and tug assignment problem within a canal harbor

Matteo Petris*

INRIA Lille - Nord Europe, Villeneuve dAscq 59650, France
Department of Management, Università Ca' Foscari Venezia, Cannaregio 873, 30121 Venice, Italy

Paola Pellegrini

COSYS-LEOST, Univ Gustave Eiffel, IFSTTAR, Univ Lille, F-59650 Villeneuve d'Ascq, France

Raffaele Pesenti

Department of Management, Università Ca' Foscari Venezia, Cannaregio 873, 30121 Venice, Italy

Abstract

The in-Port vessel Scheduling and tug Assignment Problem (PSAP) aims at determining the schedule for a given set of vessel movements, and their escorting tugs within a port. In this paper, we propose, compare and discuss models and algorithms for determining solutions for the PSAP. Specifically, we introduce two mathematical programming models and we derive from them four heuristics: two based on the time limited execution of a commercial solver, and two on a receding horizon principle. Finally, we present the results of a computational study aiming at assessing the performance of the considered algorithms on problem instances obtained from the Port of Venice, a medium size Italian port. The receding horizon based heuristics show good performances. They provide good quality solutions for the majority of the instances within a reasonable computational time.

Keywords: `OR in maritime industry, Transportation, Vessel scheduling, Tug assignment.`

1. Introduction

The in-Port vessel Scheduling and tug Assignment Problem (PSAP) aims at determining the schedule for a given set of vessel movements and their escorting tugs within a port. Optimizing the schedule of vessel movements, specifically, vessel arrivals, departures, and berth to berth connections, is critical for the efficient management of a port, as many stakeholders have to coordinate their in-port activities accordingly. In this paper, we deal with the PSAP for ports consisting of canal harbors, where vessels have to keep a safety distance and cannot cross or pass each other. In this kind of ports, vessels are also subject to more stringent requirements than those of seaports as regard the usage of tugs, which have to help to navigate through narrow water canals.

*Corresponding author
Email address: `matteo.petris@inria.fr` (Matteo Petris)

The PSAP can be framed within the in-port or canal vessel scheduling and routing problem literature [8]. The literature on in-port scheduling problems considers vessels (typically container-ships) that have to be assigned to berths. Then quayside, yard and landside operations are scheduled accordingly. The berth allocation and quay crane scheduling problems [4, 6, 12, 35, 24] fall within this kind of literature. In recent years, the literature has also considered other topics, such as the problem of routing tankers that have to move between berths [34]. Other recent works focus on the problem of scheduling the access of port resources such as channels and berths [10, 37, 38]. In [19, 13], the problem of scheduling incoming and outgoing vessels through different waterways for accessing or leaving the port is dealt with. A framework that combines decision theory and stochastic optimization techniques to address tide routing is considered in [7]. Problems arising in managing traffic in navigation channels that join the terminal basin or different terminals are dealt with in [17, 9]. Canal scheduling literature considers vessels navigating waterways and/or locks [33, 25, 16], such as the Panama Canal [11, 14, 39], the Kiel Canal [23, 21], the Istanbul Strait [22, 31], the Yangtze River [20].

In this work, we apply the PSAP to canal harbors such as the Port of Venice in Italy. The Port of Venice is a medium Italian port of the north Adriatic Sea. It is situated within the shallow water of the Venetian Lagoon. Here, vessels can access or leave the port only navigating along narrow canals. For this problem, we propose a formal definition, two mathematical models and four algorithms based on these models.

A first work in this context is [26], which introduces the problem of scheduling vessel movements within a canal harbor (PSP), without taking tugs into consideration. The authors propose a mixed integer linear programming (MILP) model inspired by RECIFE-MILP [27], an algorithm for the railway traffic management problem. This choice is motivated by the observation that the navigation of vessels along narrow canals presents some similarities with the movement of trains along single track railway lines. One of the models that we consider in this work extends the one in [26] in order to include the assignment of tugs.

The problem of assigning tugs to a set of vessel movements shows similarities with another railway management problem, the Locomotive Scheduling Problem (LSP) [32]. In the LSP, locomotives have to be assigned to trains with the objective of minimizing the deviation of the trains' actual schedule from the planned one. We refer the interested reader to survey [28] on the different variants of LSP. However, PSAP and LSP present some main differences. Locomotives can move from a train to another only along free track, where there are no other trains. Trains typically need a single locomotive and can stop along their route. Differently, tugs may navigate also through canals that are occupied by vessels. Vessels often need more than one tug and cannot stop along their route.

The PSAP assumes that the schedule of vessels and the assignment of tugs is decided with an integrated approach. In recent years, integrated approaches have attracted increasing interest as the sequential ones have intuitive drawbacks, as pointed out in [5] in the context of railway management. An example of an integrated approach in railways can be found in [36], where train timetable and locomotive assignment is defined at once, possibly canceling trains that cannot be served. A minimum cost multi-commodity network flow model is proposed to this aim.

In this paper, we first define and formalize the PSAP. Then, we introduce two mathematical programming models for it: one based on time-indexed variables, the other on continuous time variables. We exploit each model to develop

two heuristic algorithms. Specifically, in one algorithm we execute a commercial solver on the considered model within a time limit; in the other one, we embed the considered model in a receding horizon framework. Finally, we assess the applicability and the performance of our algorithms by means of an extensive experimental analysis conducted on real and realistic randomly generated instances representing traffic at the Port of Venice, in Italy.

The remainder of this paper is organized as follows. In Section 2, we formally define the problem we deal with. In Section 3, we present the models for the PSAP that we propose in this paper. In Section 4, we detail the solution algorithms exploiting these models. In Section 5, we discuss our case study and the performance of the proposed algorithms. Finally, we draw conclusions in Section 6 and we discuss the applicability of some strengthening cuts for the models in the Appendix.

2. Problem statement

In this section, we provide the formal statement of the PSAP. We start by introducing the assumptions we make and the necessary notation. We then define the problem.

We model a port as a network of waterways $\mathcal{G} = (V, E)$. The vertex set V includes the *navigation points*, i.e., the points in the canals that have some interest for navigation:

- *berths* and *inlets*;
- *connection points* between two waterways or between a berth and a waterway;
- other *relevant points* such as turning basins, initial/final points of tug services, extremes of areas where some harbormaster constraints hold, e.g., particular speed limits.

The edge set E of $\mathcal{G}$ includes edges $e = (i, j)$, $i, j \in V$. Each edge corresponds to a canal segment joining two navigation points i and j and including no other navigation point in between. The particular layout of the case study motivating this research, i.e., the Port of Venice, (Figure 1, left) induces the so-defined graph to have a tree structure (see Figure 1, right). Given this definition of the network, vessel *movements* refer to vessels entering and exiting the port as well as sailing within the port from one berth to another. With a slight abuse of terminology, hereinafter we consider inlets as berths.

The first set of assumptions we consider pertains to safety and operational constraints of vessel movements.

Assumptions 1 (Vessel movements). *Each movement:*

1.1 has both route and sailing time fixed a-priori. *In particular, a vessel cannot stop along its route once it has started moving;*

1.2 has its starting time to be scheduled within a given time window, typically of few hours;

1.3 may have to respect a minimum and maximum separation time with respect to other movements, e.g., because performed by the same vessel or because a transshipment of cargo must occur;

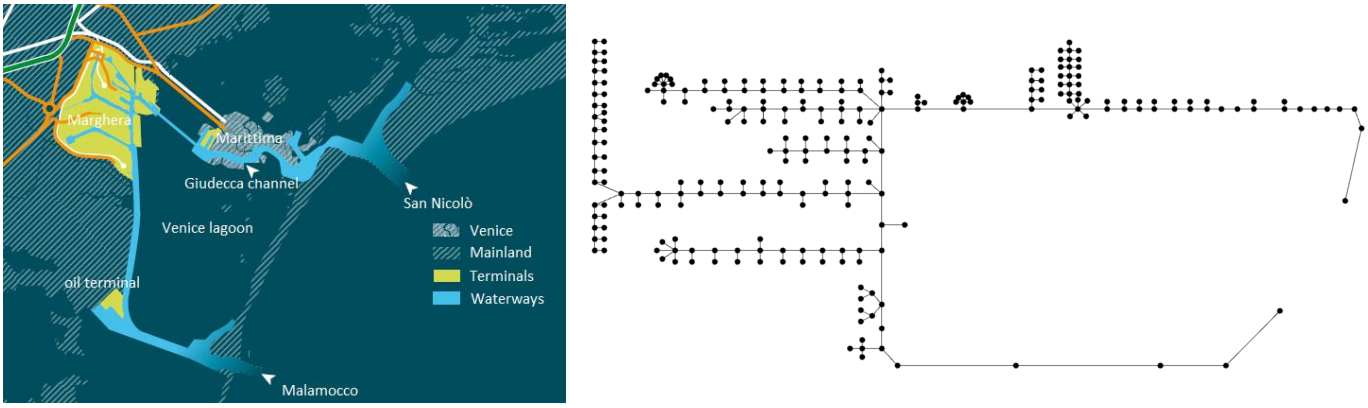

Figure 1: Layout (left) and topological structure (right) of the Port of Venice

1.4 has to keep a minimum headway with any other movement whose route may interfere with its own. We say that two routes interfere with each other if it exists a pair of points, one per route, at a distance smaller than or equal to the safety distance between vessels (we refer to these pairs of points as interference areas);

1.5 may need to be escorted by a given number of compatible tugs along its entire route or a portion of it;

1.6 may have to respect physical constraints, e.g., two vessels in general should not be moored at the same berth or the presence of a moored vessel in a narrow canal may prevent the movement of other vessels.

Assumption 1.1 implies that port operators finalize berth and route allocations before deciding movement schedules. To do so, they take into account the width and draft of the canals. Indeed, the mooring berths for the movements are usually chosen by shipping companies and terminal operators before the movement itself is submitted to the harbormaster, port pilots, and tug companies. In ports similar to the one of Venice, the allocation of berths may also be influenced by the relatively short length of terminals and by the size of canals. As previously mentioned, the Port of Venice has two inlets and a tree-like structure (Figure 1 (right)), hence, once the berth allocation is determined no rerouting is possible.

We remark that Assumptions 1.2 and 1.3 ensure that enough time is given to perform in-port operations, such as loading and unloading. Moreover, through the imposition of minimum separation times, it is also possible to guarantee the coherence of berth availability. In the Port of Venice, shallow depths may impose the setting of appropriate time windows for some movements, so that they do not occur during low tides. Furthermore, the limited room for manoeuvre of some canals may require operators to fix some precedence among movements planned at the same or at close terminals.

For example, consider two of the vessels pictured in Figure 2: the red one is moored at berth 14 and must leave it going north; the blue one must reach berth 14 from inlet 1. The red vessel is moored at the berth at the beginning of the time horizon. Indeed, it must pass through navigation points 13, 11 and 10 before the blue vessel. Hence, a minimum separation time has to be imposed between the movements of the red and blue vessels. Specifically, the minimum separation time is such that the blue vessel does not reach point 10 before the red one has passed it. Hence, the blue vessel must start sailing not earlier than the time at which the red one does, plus the latter's sailing time from 14 to 10 or to a point far

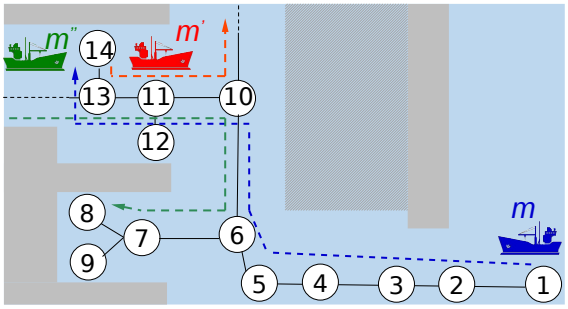

Figure 2: Toy example representing three movements in a simple port.

enough to ensure safety (Assumption 1.4), minus the former's sailing time from 1 to 10. As sailing times are fixed as for Assumption 1.1, no interference will occur if this separation is respected. Similarly, suppose that the narrow width of the canal prevents vessel movements through point 13 when berth 14 is occupied. Let the green vessel in Figure 2 aim to berth 8, arriving from the west. Then, a minimum separation time must be imposed: the green vessel can start sailing only at a time that will allow it to arrive at 13, 11 and 10 after the red one has passed them. Minimum separation times have to be imposed also between the blue and the green vessel to avoid interferences.

Assumption 1.4 ensures that each vessel maintains a safety distance from other vessels while sailing along canals. In particular, it prevents vessels from crossing or overtaking each other. Assumption 1.1 allows the expression of the headway between each pair of movements required by Assumption 1.4 in terms of a minimum separation between their starting times [26].

The second set of assumptions pertains to tugs.

Assumptions 2 (Tug operations). *Each tug:*

2.1 can serve a movement at a time;

2.2 can cross or overtake a vessel sailing in the same canal;

2.3 can stop and wait in any point of the harbor when it is not serving a vessel;

2.4 sails at a reference constant speed through the canals;

2.5 is always available, i.e., no working shift for the tug crew nor maintenance stops are taken into account.

We denote by $\mathscr{M}$ the set of vessel movements that have to be scheduled within a time horizon T. The starting time of movement $m \in \mathscr{M}$ that has to be determined is denoted by t_m. Moreover, for each movement $m \in \mathscr{M}$, we denote by:

- o_m its origin berth;
- a_m its arrival berth;
- $r_m = P[o_m, a_m]$ its route, i.e., the path in $\mathcal{G}$ connecting o_m and a_m;
- $T_m = [e_m, l_m]$ its starting time window, where e_m and l_m are respectively the earliest and latest starting time;
- w_m its desired starting time;

- $c_{m,t}$ the cost to be paid when $t_m = t$ rather than w_m;
- v_m its vessel;
- $s_{i,m}$ its sailing time from o_m to navigation point $i \in r_m$.
- $S_m = s_{a_m,m}$ its total sailing time;
- N_m the number of tug services it requires;
- U_m the set of its compatible tugs;
- i_m^k (f_m^k) the initial (final) navigation points in r_m of the k-th tug service;
- p_m (d_m) the time needed by a tug to pick up/connect (drop/disconnect) it.

For all pairs of movements $m, m' \in \mathscr{M}$ we denote by:

- $\mathbb{1}(m, m')$ the indicator function which takes value 1 if a positive bound has to be imposed on the separation between the starting times of m and m', e.g., because they are operated by the same vessel, 0 otherwise;
- $\underline{s}_{m,m'}$, $\bar{s}_{m,m'}$ the minimum, respectively maximum, separation time between the end of a movement m and the start of the subsequent movement m', when $\mathbb{1}(m, m') = 1$;
- $h_{m,m'}$ the minimum headway between the starting times of m and m'. This time guarantees that v_m and $v_{m'}$ maintain a minimal safety distance during their movements. It is defined when $\mathbb{1}(m, m') = 0$. In particular, $h_{m,m'}$ is assumed equal to infinity if m cannot be scheduled before m'. It is equal to 0 if m and m' do not interfere with each other, that is if the vessels performing m and m' are always at a physical distance larger than the minimal safety distance while sailing their respective routes, whatever their starting times. This holds if the routes of the two movements have no interference areas or if the combination of their sailing time and allowed time windows implies that m and m' cannot be moving simultaneously.

Finally, we denote by U the set of available tugs. For all tugs $u \in U$, we denote by:

- $\mathscr{M}^u \subseteq \mathscr{M}$ the subset of $\mathscr{M}$ containing the movements that it can operate;
- $s_u^{i,j}$ its sailing time between navigation points i and j when it is not serving a vessel.

We are now ready to formally state the PSAP.

Problem 1 (in-Port vessel Scheduling and tug Assignment Problem).
Consider a set of vessel movements $\mathscr{M}$ within port $\mathcal{G}$ and a set of tugs U. For each movement $m \in \mathscr{M}$, determine a starting time $t_m \in T_m$ and assign a set of tugs in U_m that can carry on the N_m services required by m, such that Assumptions 1 and 2 hold and the weighted sum of the deviations between the actual and desired starting times of the movements is minimized.

Hereinafter, we denote a PSAP instance as $\pi(\mathscr{M}, U, \mathcal{G})$. In addition, we assume that both sets $\mathscr{M}$ and U are ordered sets of indices. Hence, we identify movements and tugs with positive integer numbers. Then, expression $m < m'$ indicates that the position of movement m in $\mathscr{M}$ precedes the position of m'.

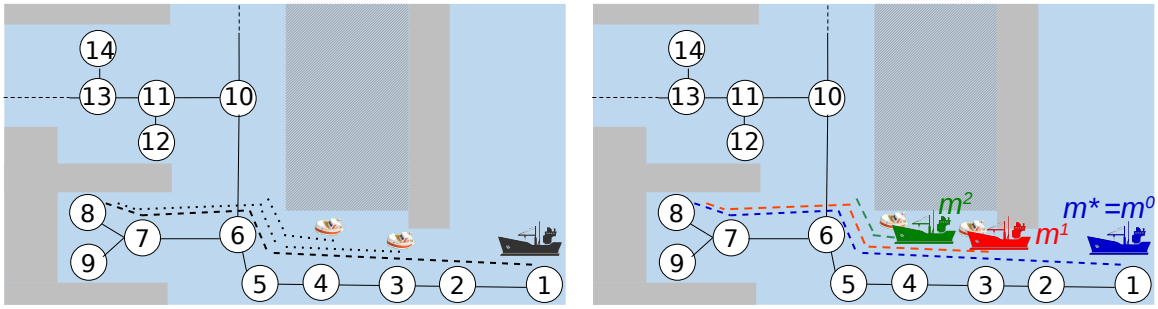

Figure 3: Graphical representation of a toy PSAP instance $\pi(\mathscr{M}, U, \mathcal{G})$ (left) and of its associated instance $\alpha(\bar{\mathscr{M}}, \bar{U}, \mathcal{G})$ (right).

In the rest of this section, we use as reference example the toy instance $\pi(\mathscr{M}, U, \mathcal{G})$ depicted in Figure 3 (left). Here, one movement m of vessel v_m is to be scheduled. Vessel v_m route $r_m = P[1, 8]$ is indicated as a dashed line in the figure. The sailing time of vessel v_m on each arc composing r_m is one time unit, hence $S_m = 7$. The movement time window, desired time and deviation cost are set respectively to $T_m = [5, 15]$, $w_m = 10$ and $c_{m,t} = |t - w_m|$ for all $t \in T_m$. In addition, m requires $N_m = 2$ tug services: tug service 1 to escort m from $i^1_m = 3$ to $f^1_m = 8$, and tug service 2 from $i^2_m = 4$ to $f^2_m = 6$. These partial routes are indicated as dotted lines.

Given a PSAP instance $\pi(\mathscr{M}, U, \mathcal{G})$, let us define its *associated* instance $\alpha(\bar{\mathscr{M}}, \bar{U}, \mathcal{G})$, where each movement in $\bar{\mathscr{M}}$ stands for a movement of $\mathscr{M}$ or a part of it, and requires to be escorted by a single tug along its entire route. The idea is to create dummy movements and tugs. Each movement is associated to a partial route of an original one, requiring one tug. If no tug is requested for a partial route in the original instance, the resulting movement is considered as requesting a dummy tug. By imposing constraints on the separation of starting times, we obtain that movements associated to the same original one actually occur simultaneously. We will exploit the concept of associated instance to define the mathematical programming models in Section 3.

In the following, we report the procedure followed to construct it, considering the example instance $\pi(\mathscr{M}, U, \mathcal{G})$ of Figure 3 (left) and creating its associated $\alpha(\bar{\mathscr{M}}, \bar{U}, \mathcal{G})$ represented in Figure 3 (right). As a start, for each $m \in \mathscr{M}$, we define the set of movements $\mathscr{M}_m$, each one served by a single tug. Specifically, for each tug service $k = 1, \ldots, N_m$, set $\mathscr{M}_m$ includes a movement m^k. It has: i) route going from navigation point i^k_m to navigation point f^k_m along the route of m ($r_{m^k} \subseteq r_m$); ii) sailing times coherent with those of m ($s_{i,m^k} = s_{i,m} - s_{i^k_m,m}$, for all $i \in r_{m^k}$); iii) necessity of a tug in U_m ($N_{m^k} = 1$), with same pick up/connect and drop/disconnect times as m ($p_{m^k} = p_m$ and $d_{m^k} = d_m$). In addition, if $N_m = 0$ or no service $k = 1, ..., N_m$ goes from the origin berth o_m to the arrival berth a_m of movement m, then $\mathscr{M}_m$ includes also an extra movement m^0 that requires a service by a dummy tug $u_m \notin U$ along the whole route of m, with $p_{m^*} = d_{m^*} = 0$. This dummy tug is dedicated to m and does not serve any other movement of the associated instance. Hereinafter, we indicate by m^* the movement $m^k \in \mathscr{M}_m$ with minimum index k whose route goes from o_m to a_m. Each movement's starting time window and desired starting time are computed considering the time window and starting time of m, and shifting them by the sailing time $s_{i^k_m,m}$ necessary for vessel v_m to reach the origin point of movement m^k: $T_{m^k} = \{e_m + s_{i^k_m,m}, \ldots, l_m + s_{i^k_m,m}\}$ and $w_{m^k} = w_m + s_{i^k_m,m}$. The cost to be paid when the

actual starting time differs from the desired one is set to 0 for all movements in $\mathscr{M}_m$ but m^*, which inherits the cost of m. Furthermore, the vessel associated to m^* is the same one associated to m ($v_{m^*} = v_m$). Instead, dummy vessels are associated to all $m^k \in \mathscr{M}_m \setminus \{m^*\}$: they are not subject to Assumption 1.6 which constrains possible movements and simultaneous presence.

In the example of Figure 3, set $\mathscr{M}_m$ is composed of 3 movements, m^0, m^1 and m^2. They are represented in blue, red and green, respectively. Movement m^0 is m^* as it covers the entire route of m. The sailing time, time window and desired time of m^* are exactly those of m. In addition, m^* requires $N_{m^*} = 1$ service of a dummy tug ($U_{m^*} = \{u_m\}$) along its entire route (i.e., from $i^1_{m^*} = 1$ to $f^1_{m^*} = 8$). Movement m^1 corresponds to tug service 1, so we set its route and sailing time to $r_{m^1} = [3, 8]$ and $S_{m^1} = 5$. Its time window, $T_{m^1} = [7, 17]$, and desired time, $w_{m^1} = 12$, are those of m shifted by the sailing time that vessel v_m requires to reach navigation point 3, that is by $s_{3,m} = 2$. Then, we impose that m^1 requires the service of a single tug ($N_{m^1} = 1$) along its entire route, that is from $i^1_{m^1} = 3$ to $f^1_{m^1} = 8$. The tug is to be chosen from those compatible with m ($U_{m^1} = U_m$). We observe that dummy tug u_m cannot be used to perform the service on m_1. The pick up and drop time of the tugs are those of m, i.e., $p_{m^1} = p_m$ and $d_{m^1} = d_m$. Similarly, movement m^2 corresponds to tug service 2: $r_{m^2} = [4, 6]$, $v_{m^2} = 2$, $T_{m^2} = [8, 18]$, $c_{m^2} = 13$, $N_{m^2} = 1$, $i^2_{m^1} = 4$, $f^1_{m^2} = 6$, $U_{m^2} = U_m$, $p_{m^2} = p_m$ and $d_{m^2} = d_m$. As for the costs of starting time deviation, $c_{m^*,t} = c_{m,t}$ for all $t \in T_{m^*}$, $c_{m^1,t} = 0$ for all $t \in T_{m^1}$, $c_{m^2,t} = 0$ for all $t \in T_{m^2}$.

Next, we impose that all movements in each set $\mathscr{M}_m$ are performed simultaneously to movement $m^* \in \mathscr{M}$. To impose constraints on the time separation between the arrival and starting times, we set $\mathbb{1}(m^k, m^*) = \mathbb{1}(m^*, m^k) = 1$ for each $m^k \in \mathscr{M}_m \setminus \{m^*\}$. On the one hand, the difference between the arrival of m^k and the start of m^* must be equal to the negative of the time necessary for v_m to reach the end of the route of m^k (f^k_m): $\bar{s}_{m^k,m^*} = \underline{s}_{m^k,m^*} = -s_{f^k_m,m}$. On the other hand, the difference between the end of m^* and the start of m^k must be equal to the difference between the time v_m needs to reach the beginning of r_{m^k} and the whole sailing time of S_m: $\bar{s}_{m^*,m^k} = \underline{s}_{m^*,m^k} = s_{i^k_m,m} - S_m$. Indeed, being v_{m^h} a dummy vessel for all $m^h \in \mathscr{M}_m \setminus \{m^*\}$, no minimum headway time exists between pairs of movements in $\mathscr{M}_m$: $h_{m^k,m^h} = h_{m^h,m^k} = 0$ for all $m^h \in \mathscr{M}_m$.

In the example of Figure 3 (right), movements m^1 and m^2 have to be performed simultaneously to m^*. Let us consider m^1. We set $\mathbb{1}(m^1, m^*) = \mathbb{1}(m^*, m^1) = 1$ and the separation times $\bar{s}_{m^1,m^*} = \underline{s}_{m^1,m^*} = -7$ and $\bar{s}_{m^*,m^1} = \underline{s}_{m^*,m^1} = -5$. Specifically, recalling that the navigation time on each arc is 1 time unit, to have m^1 simultaneous to m^*, m^1 needs to arrive at its destination 7 time units after m^* starts. Moreover, the latter needs to arrive 5 time units after the former starts ($S_m = 7$ and $s_{3,m} = 2$). If, for example, $t_{m^*} = 10$, the separation times $\bar{s}_{m^*,m^1} = \underline{s}_{m^*,m^1} = -5$ impose that $t_{m^1} = 12$. Similarly, to have m^2 simultaneous to m^*, we set $\mathbb{1}(m^2, m^*) = \mathbb{1}(m^*, m^2) = 1$ and separation times $\bar{s}_{m^2,m^*} = \underline{s}_{m^2,m^*} = -5$ and $\bar{s}_{m^*,m^2} = \underline{s}_{m^*,m^2} = -4$. Finally, we impose no minimum headway time between m^*, m^1 and m^2, that is $h_{m^*,m^1} = h_{m^1,m^*} = h_{m^*,m^2} = h_{m^2,m^*} = h_{m^1,m^2} = h_{m^2,m^1} = 0$.

As for the relations between the movements in $\mathscr{M}_m$ and in $\mathscr{M}_{m'}$, where $m' \in \mathscr{M} \setminus \{m\}$, we mirror the original relations between m and m' on m^* and m'^*. This holds for minimum headway times ($h_{m^*,m'^*} = h_{m,m'}$ and $h_{m'^*,m^*} = h_{m',m}$) and for minimum and maximum separation ($\mathbb{1}(m^*, m'^*) = \mathbb{1}(m, m')$, and $\bar{s}_{m^*,m'^*} = \bar{s}_{m,m'}$ and $\underline{s}_{m'^*,m^*} = \underline{s}_{m',m}$ if $\mathbb{1}(m, m') = 1$).

In conclusion, instance $\alpha(\bar{\mathscr{M}}, \bar{U}, \mathcal{G})$ associated to $\pi(\mathscr{M}, U, G)$ is such that $\bar{\mathscr{M}} = \bigcup_{m \in \mathscr{M}} \mathscr{M}_m$ and $\bar{U} = \bigcup_{m \in \mathscr{M}} (U_m \cup \{u_m\})$.

Let us remark that, to solve the PSAP on $\alpha(\bar{\mathscr{M}}, \bar{U}, \mathcal{G})$, it is sufficient to characterize constraints, headways, and separation times between m^* and m'^*. Indeed, the respect of constraints, headways, and separation times between each pair of movements in the Cartesian product of the sets $\mathscr{M}_m$ and $\mathscr{M}_{m'}$ is guaranteed by the fact that all the movements in $\mathscr{M}_m$, respectively in $\mathscr{M}_{m'}$, are performed simultaneously with m^*, respectively m'^*.

Let two PSAP instances be *equivalent* if each feasible solution of one instance, in terms of movement schedule and tug assignment, allows to derive a feasible solution of the other instance with equal cost in at most $O(|\mathscr{M}||U|)$ operations. The following Lemma 1 guarantees that a PSAP instance $\pi(\mathscr{M}, U, \mathcal{G})$ is equivalent to its associated one $\alpha(\bar{\mathscr{M}}, \bar{U}, \mathcal{G})$.

Lemma 1. *A PSAP instance $\pi(\mathscr{M}, U, \mathcal{G})$ and its associated one $\alpha(\bar{\mathscr{M}}, \bar{U}, \mathcal{G})$ are equivalent. The optimal solution of $\pi(\mathscr{M}, U, \mathcal{G})$ can be derived in polynomial time from the optimal solution of $\alpha(\bar{\mathscr{M}}, \bar{U}, \mathcal{G})$.*

Proof. We first prove that, given a feasible solution for $\alpha(\bar{\mathscr{M}}, \bar{U}, \mathcal{G})$, we can derive a feasible solution for $\pi(\mathscr{M}, U, \mathcal{G})$ with identical costs, i.e., identical weighted sum of the deviations between the actual and desired starting times of the movements. To this aim, consider a feasible solution σ_α of α and the solution σ_π of π obtained by scheduling each movement $m \in \mathscr{M}$ as the corresponding movement $m^* \in \mathscr{M}_m \subseteq \bar{\mathscr{M}}$, and by assigning to m the tugs assigned to all movements in $\mathscr{M}_m$ with the exception of the dummy tugs. The number of passages to obtain σ_π from σ_α is trivially polynomial.

Next, we prove that solution σ_π is feasible for π. Indeed, by construction, movements m and all their associated movements in $\mathscr{M}_m$, such as m^*, are subject to the same constraints derived from Assumptions 1. Then, σ_π is feasible with respect to this set of constraints as movement m and m^* are simultaneous. Consider now the solution σ_α of α. All the movements in each subset $\mathscr{M}_m$ occur simultaneously and the scheduling of the tugs satisfies the constraints derived from Assumptions 2. As movements $m \in \mathscr{M}$ in σ_π are simultaneous to the corresponding m^* in σ_α, we can affirm that σ_π is also feasible with respect to these constraints. Indeed, in σ_π, the tugs assigned to $m \in \mathscr{M}$ serve it in exactly the same time interval and along exactly the same route as they served the movements $m^k \in \mathscr{M}_m$ in σ_α.

Finally, we observe that solutions σ_α and σ_π have the same costs. Indeed, movements m^* and the associated movements m occur simultaneously, have the same starting time deviation cost and have the same desired starting time; the remaining movements m_k of α have null costs by construction.

A symmetrical argument allows deriving a feasible solution for α given a feasible solution for π.

All π feasible solutions can be derived from all α ones using the same procedure. Hence, the optimal solution of the former can be derived in polynomial time from the optimal solution of the latter.

□

Hereinafter, we assume that each movement requires to be escorted by exactly one tug along its whole route in all PSAP instances. When this is not the case in the instances to be tackled, we will simply consider their associated ones.

3. PSAP models

In this section, we introduce and discuss two mathematical programming models of PSAP denoted respectively by $\mathcal{P}_1$ and $\mathcal{P}_2$: $\mathcal{P}_1$ is a time-indexed model, whereas $\mathcal{P}_2$ is a continuous time one. Model $\mathcal{P}_2$ can be considered as the extension of the RECIP-MILP model presented in [26]. In the appendix, we propose possible strengthening cuts for the two models.

3.1. Model $\mathcal{P}_1$

Model $\mathcal{P}_1$ assumes a discretized time horizon T, i.e., T is a set of successive periods, where each period t starts at time t and ends at time $t + \Delta t$. We assume that each movement may start only at the beginning of a period. In the practice of the Port of Venice, T covers 24 hours and can be discretized with a step Δt of five minutes. Indeed, the duration of movements and the inevitable uncertainties afflicting navigation may make a finer time resolution hardly significant, at least in the movements' planning phase considered in the PSAP.

Hereinafter, whenever dealing with Model $\mathcal{P}_1$, all time intervals $[\underline{t}, \bar{t}]$ have to be understood as sets of discrete time instants $\{\underline{t}, \underline{t} + \Delta t, \underline{t} + 2\Delta t, \ldots, \bar{t}\}$, the time step always being Δt.

Model $\mathcal{P}_1$ includes the following binary variables.

We introduce decision variables that define the movements $m \in \mathscr{M}$ starting time $t_m \in T_m$:

$$x_{m,t} = \begin{cases} 1 & \text{if } t_m = t \\ 0 & \text{otherwise} \end{cases} \qquad \forall m \in \mathscr{M},\ t \in T_m.$$

Then, we include decision variables that define the assignment of tugs $u \in U_m$ to movements $m \in \mathscr{M}$:

$$z_{m,u} = \begin{cases} 1 & \text{if } m \text{ is escorted by } u \\ 0 & \text{otherwise} \end{cases} \qquad \forall m \in \mathscr{M},\ u \in U_m.$$

Finally, we introduce service variables that register which pairs of movements $m, m' \in \mathscr{M}$ share the same tugs:

$$y_{m,m'} = \begin{cases} 1 & \text{if } m \text{ and } m' \text{ are escorted by the same tug} \\ 0 & \text{otherwise} \end{cases} \qquad \forall m, m' \in \mathscr{M} : U_m \cap U_{m'} \neq \emptyset.$$

Model $\mathcal{P}_1$ reads as follows.

The objective function minimizes the total weighted deviation which affects the movement schedules:

$$\min \sum_{m \in \mathscr{M}} \sum_{t \in T_m} c_{m,t} x_{m,t}. \tag{1}$$

The following constraints must be satisfied.

Movement operation constraints:

$$\sum_{t \in T_m} x_{m,t} = 1 \qquad \forall m \in \mathscr{M}, \tag{2}$$

$$\sum_{u \in U_m} z_{m,u} = 1 \qquad \forall m \in \mathscr{M}. \tag{3}$$

Constraints (2) impose that the starting time t_m of each movement m assumes a single value t that falls within time window T_m. Differently, Constraints (3) require that each movement m is escorted by exactly one compatible tug belonging to U_m.

Separation time constraints:

$$\sum_{t' \in Q_{m,m'}(t) \cap T_{m'}} x_{m',t'} \geq x_{m,t} \qquad \forall m, m' \in \mathscr{M} : \mathbb{1}(m,m') = 1,\ t \in T_m, \tag{4}$$

where $Q_{m,m'}(t) = [t + S_m + \underline{s}_{m,m'}, t + S_m + \bar{s}_{m,m'}]$.

Constraints (4) require that if two movements m and m' are operated by the same vessel, then the starting time of m' has to follow the completion time of m of at least $\underline{s}_{m,m'}$ and at most $\bar{s}_{m,m'}$.

Headway constraints:

$$\sum_{t' \in H_{m,m'}(t)} x_{m',t'} \geq x_{m,t} \qquad \forall m, m' \in \mathscr{M} : m < m',\ \mathbb{1}(m,m') = 0,\ t \in T_m, \tag{5}$$

where $H_{m,m'}(t) = [e_{m'}, t - h_{m',m}] \cup [t + h_{m,m'}, l_{m'}]$.

Constraints (5) require that the starting time of two movements m and m' operated by different vessels have to be separated by a minimal headway $h_{m,m'}$, if m is scheduled before m', or by $h_{m',m}$, otherwise.

Movement compatibility constraints:

$$x_{\underline{m},\underline{t}} \leq \sum_{\overline{t} \in [\underline{t} + h_{\underline{m},\overline{m}}, l_{\overline{m}}]} x_{\overline{m},\overline{t}} + \sum_{\overline{t}' \in [e_{\overline{m}'}, \underline{t} - h_{\overline{m}',\underline{m}}]} x_{\overline{m}',\overline{t}'} \qquad \forall (\underline{m}, \overline{m}, \overline{m}', \underline{t}) \in I^1, \tag{6}$$

$$x_{\underline{m}',\underline{t}'} \leq \sum_{\overline{t} \in [\underline{t}' + h_{\underline{m}',\overline{m}}, l_{\overline{m}}]} x_{\overline{m},\overline{t}} + \sum_{\overline{t}' \in [e_{\overline{m}'}, \underline{t}' - h_{\overline{m}',\underline{m}'}]} x_{\overline{m}',\overline{t}'} \qquad \forall (\underline{m}', \overline{m}, \overline{m}', \underline{t}') \in I^2 \tag{7}$$

$$x_{m,t} \leq \sum_{\overline{t} \in [t + h_{m,\overline{m}}, l_{\overline{m}}]} x_{\overline{m},\overline{t}} + \sum_{\overline{t}' \in [e_{\overline{m}'}, t - h_{\overline{m}',m}]} x_{\overline{m}',\overline{t}'} \qquad \forall (m, \overline{m}, \overline{m}', t) \in I^3 \tag{8}$$

where sets I^1, I^2, I^3 are defined as follows: $I^1 = \{(\underline{m}, \overline{m}, \overline{m}', \underline{t}) \in \mathscr{M}^3 \times T_{\underline{m}} : v_{\overline{m}} = v_{\overline{m}'}, a_{\underline{m}} = a_{\overline{m}} = o_{\overline{m}'}, \exists \underline{m}' \in \mathscr{M}$ s.t. $v_{\underline{m}} = v_{\underline{m}'}, a_{\underline{m}} = o_{\underline{m}'}\}$, $I^2 = \{(\underline{m}', \overline{m}, \overline{m}', \underline{t}') \in \mathscr{M}^3 \times T_{\underline{m}'} : v_{\overline{m}} = v_{\overline{m}'}, o_{\underline{m}'} = o_{\overline{m}'} = a_{\overline{m}}, \exists \underline{m} \in \mathscr{M}$ s.t. $v_{\underline{m}} = v_{\underline{m}'}, a_{\underline{m}} = o_{\underline{m}'}\}$ and $I^3 = \{(m, \overline{m}, \overline{m}', t) \in \mathscr{M}^3 \times T_m : v_{\overline{m}} = v_{\overline{m}'}, a_{\overline{m}} = o_{\overline{m}'}$ and $v_{\overline{m}}$ in $a_{\overline{m}}$ prevents the passage of $v_m\}$.

Specifically, sets I^1 and I^2 involve the movements of vessel pairs $(v_{\underline{m}}, v_{\overline{m}})$ which arrive and depart from the same berth during the considered time horizon: movements $\underline{m}$ and $\underline{m}'$ correspond to the arrival and departure of $v_{\underline{m}} (= v_{\underline{m}'})$ at berth $a_{\underline{m}} (= o_{\underline{m}'})$; $\overline{m}$ and $\overline{m}'$ correspond to the arrival and departure of $v_{\overline{m}} (= v_{\overline{m}'})$ at the same berth, i.e. at $a_{\overline{m}} = o_{\overline{m}'} = a_{\underline{m}} = o_{\underline{m}'}$. Then, Constraints (6) and (7) state that either vessel $v_{\underline{m}}$ uses the berth before $v_{\overline{m}}$, or vice-versa. Specifically, for each time $\underline{t} \in T_{\underline{m}}$, Constraints (6) impose that movement $\underline{m}$ can start at $\underline{t}$ if one of the following events occurs: $\overline{m}$ starts at a time that brings it to the interference area after $\underline{m}$ (between $\underline{t} + h_{\underline{m},\overline{m}}$ and the end of its starting time window $l_{\overline{m}}$); $\overline{m}'$ starts early enough to use the interference area before $\underline{m}$ (between the beginning of its starting time window $e_{\overline{m}'}$ and $\underline{t} - h_{\overline{m}',\underline{m}}$). Recall that Constraints (4) ensure that $\overline{m}$ precedes $\overline{m}'$; thus, the relations set in Constraints (6) are enough to have both movements of vessel $v_{\underline{m}}$ preceding or following $\underline{m}$. Constraints (7) impose similar conditions for the starting

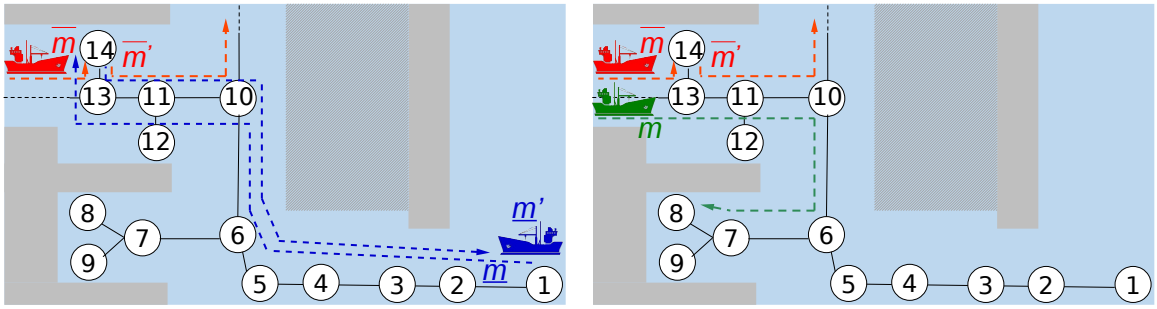

Figure 4: Toy example showing possible needs for compatibility constraints.

time of movement $\underline{m}'$ with respect to those of $\overline{m}$ and $\overline{m}'$: either both $\overline{m}$ and $\overline{m}'$ precede $\underline{m}'$ in the interference area, or they both follow it. Remark that a pair of these constraints may allow both movements of vessel $v_{\overline{m}}$ to occur within the ones of $v_{\underline{m}}$. However, by imposing Constraints (6) and (7) both for $v_{\underline{m}}$ w.r.t $v_{\overline{m}}$ and for $v_{\overline{m}}$ w.r.t $v_{\underline{m}}$, we forbid any overlap. Consider, for example, the red and the blue vessels in Figure 4 (left). Constraints (6) and (7) ensure that either both movements of the blue vessel ($\underline{m}$ and $\underline{m}'$) precede both movements of the red one ($\overline{m}$ and $\overline{m}'$) in the interference area of their routes, or the opposite holds.

Finally, set I^3 involves the movements of vessel pairs $(v_m, v_{\overline{m}})$ such that v_m cannot perform its movement m between the arrival and departure movements of $v_{\overline{m}}$ ($\overline{m}$ and $\overline{m}'$). We use this set to manage movements in narrow canals, where the presence of a vessel at a berth prevents the transit of other vessels. Constraints (8) impose that vessel $v_{\underline{m}}$ can start its movement m either early enough to transit in the narrow canal before the arrival of $v_{\overline{m}}(= v_{\overline{m}'})$ (movement $\overline{m}$) or after $v_{\overline{m}}(= v_{\overline{m}'})$ has left the canal (movement $\overline{m}$). Consider, for example, the red and the green vessels of Figure 4 (right): Constraints (8) impose that either both red movements $\overline{m}$ and $\overline{m}'$ precede the green m, or they both follow it.

Analogous constraints can be defined for other similar incompatibility situations.

Tug usage constraints:

$$y_{m,m'} \geq z_{m,u} + z_{m',u} - 1 \quad \forall u \in U,\, m, m' \in \mathscr{M}^u : m < m', \tag{9}$$

$$\sum_{t' \in K_{m,m'}(t)} x_{m',t'} \geq x_{m,t} + y_{m,m'} - 1 \quad \forall m, m' \in \mathscr{M} : m < m',\, U_m \cap U_{m'} \neq \emptyset,\, t \in T_m, \tag{10}$$

where $K_{m,m'}(t) = [e_{m'}, t - t^u_{m',m}] \cup [t + t^u_{m,m'}, l_{m'}]$, $t^u_{m',m} = s_{f^k_{m'},m'} - s_{i^k_{m'},m'} + d_{m'} + s_u^{a_{m'},o_m} + p_m$ and $t^u_{m,m'} = s_{f^k_m,m} - s_{i^k_m,m} + d_m + s_u^{j_m,o_{m'}} + p_{m'}$.

Constraints (9) force variable $y_{m,m'}$ to assume value 1 if the two movements m and m' share the same tug.

Constraints (10) impose that if two movements m and m' are served by the same tug, and hence $y_{m,m'} = 1$, then their starting times have to be separated so that the tug can complete its service with one movement before starting its service with the other movement. Indeed, the term $t^u_{m',m}$ (and similarly $t^u_{m,m'}$) includes: i) the service time, $s_{f^k_{m'},m'} - s_{i^k_{m'},m'}$, and drop off time, $d_{m'}$, of the tug on the first movement, ii) the sailing time of the tug from the first to the second service, $s_u^{a_{m'},o_m}$, and iii) the second movement pick up time, p_m. We observe that if movement m is scheduled at t and there are

no time instants $t' \in T_{m'}$ before $t - t^u_{m',m}$ or after $t + t^u_{m,m'}$, then the constraints impose that m' and m are escorted by different tugs.

To conclude this section, we remark that Constraints (4), (5) and (10) may be alternatively reformulated as follows:

$$x_{m,t} + x_{m',t'} \leq 1 \quad \forall m, m' \in \mathscr{M} : \mathbb{1}(m, m') = 1,\ t \in T_m, t' \in T_{m'} \setminus Q_{m,m'}(t),$$
$$x_{m,t} + x_{m',t'} \leq 1 \quad \forall m, m' \in \mathscr{M} : \mathbb{1}(m, m') = 1,\ t \in T_m, t' \in T_{m'} \setminus H_{m,m'}(t),$$
$$x_{m,t} + x_{m',t'} \leq 2 - y_{m,m'} \quad \forall m, m' \in \mathscr{M} : m < m',\ U_m \cap U_{m'} \neq \emptyset,\ t \in T_m, t' \in T_{m'} \setminus K_{m,m'}(t).$$

In a preliminary experimental analysis, we tested two different reformulations of Model $\mathcal{P}_1$ (2)-(10), when solved through a commercial solver. The former considers all the reformulated constraints. The remarkably high number of constraints in the reformulation caused poor performance when compared with $\mathcal{P}_1$. The latter considers only the reformulation of Constraints (4) and (5), keeping the original Constraints (10). No significant difference has been observed when comparing it with $\mathcal{P}_1$. In the rest of the paper, we will refer to Model $\mathcal{P}_1$ to indicate the model including Constraints (2)-(10).

3.2. Model $\mathcal{P}_2$

Model $\mathcal{P}_2$ rephrases the constraints on the movement times of Model $\mathcal{P}_1$ in terms of non-negative continuous variables representing the movement starting times, and binary variables establishing the precedences among pairs of movements.

It includes the following non-negative continuous variables for all $m \in \mathscr{M}$: t_m is the actual starting time; and D^+_m and D^-_m are the positive and negative deviations of the actual starting time t_m from the desired one w_m.

In addition, $\mathcal{P}_2$ includes three sets of binary variables introduced for Model $\mathcal{P}_1$: $z_{m,u}$ for all $m \in \mathscr{M}$, and $u \in U$ and $y_{m,m'}$ for all $m, m' \in \mathscr{M}$. Finally, for each pair of movements $m, m' \in \mathscr{M}$ with interfering routes, the model includes schedule binary variables $p_{m,m'}$ that register which one reaches the interference area first

$$p_{m,m'} = \begin{cases} 1 & \text{if } m \text{ reaches before } m' \text{ the interference area of their routes} \\ 0 & \text{otherwise} \end{cases} \qquad \forall m, m' \in \mathscr{M} : \mathbb{1}(m, m') = 0.$$

Model $\mathcal{P}_2$ reads as follows.

The objective function minimizes the total weighted deviation of the movement schedules:

$$\min \sum_{m \in \mathscr{M}} (c^+_m D^+_m + c^-_m D^-_m) \tag{11}$$

where c^-_m and c^+_m are the costs associated to a unit of time of earliness and lateness of movement m, respectively.

The following constraints must be satisfied.

Movement operation constraints:

$$e_m \leq t_m \leq l_m \qquad \forall m \in \mathscr{M}, \tag{12}$$
$$\sum_{u \in U_m} z_{m,u} = 1 \qquad \forall m \in \mathscr{M}, \tag{13}$$

$$D_m^- \geq w_m - t_m \qquad \forall m \in \mathscr{M}, \tag{14a}$$

$$D_m^+ \geq t_m - w_m \qquad \forall m \in \mathscr{M}. \tag{14b}$$

Constraints (12) impose that each movement m is scheduled within time window T_m. Constraints (13) require that each movement m is assigned to a tug in U_m. Constraints (14a) and (14b) define respectively the earliness and lateness of movement m actual starting time t_m with respect to desired starting time w_m.

Separation time constraints:

$$t_m + S_m + \underline{s}_{m,m'} \leq t_{m'} \leq t_m + S_m + \bar{s}_{m,m'} \qquad \forall m, m' \in \mathscr{M} : \mathbb{1}(m, m') = 1. \tag{15}$$

Constraints (15) require that two movements m and m' operated by a vessel are scheduled so that the starting time of m' follows the completion time of m of at least $\underline{s}_{m,m'}$ and at most $\bar{s}_{m,m'}$.

Headway constraints:

$$t_{m'} \geq t_m + h_{m,m'} - M_{m,m'}(1 - p_{m,m'}) \qquad \forall m, m' \in \mathscr{M} : m < m', \mathbb{1}(m, m') = 0, \tag{16a}$$

$$t_m \geq t_{m'} + h_{m',m} - M_{m,m'} p_{m,m'} \qquad \forall m, m' \in \mathscr{M} : m < m', \mathbb{1}(m, m') = 0. \tag{16b}$$

Constraints (16) impose that two movements m and m' operated by different vessels are separated by a minimal headway. The "big-Ms" in these constraints are necessary to impose the disjunctive condition that either $t_{m'} \geq t_m + h_{m,m'}$ or $t_m \geq t_{m'} + h_{m',m}$ must hold, depending on the value of $p_{m,m'}$. The value of $M_{m,m'}$ should be set as tight as possible to improve the performances of solvers. Specifically, in (16a), $M_{m,m'}$ is set equal to $l_m + h_{m,m'} - e_{m'}$, as t_m and $t_{m'}$ are bounded by Constraints (12). Symmetrical argument applies to justify $M_{m,m'} = l_{m'} + h_{m',m} - e_m$ in (16b).

Movement compatibility constraints:

$$p_{\underline{m},\bar{m}} = p_{\underline{m},\bar{m}'} = p_{\underline{m}',\bar{m}} = p_{\underline{m}',\bar{m}'} \qquad \forall (\underline{m}, \underline{m}', \bar{m}, \bar{m}') \in J_1 \tag{17}$$

$$p_{m,\bar{m}} = p_{m,\bar{m}'} \qquad \forall (m, \bar{m}, \bar{m}') \in J_2, \tag{18}$$

where $J_1 = \{(\underline{m}, \underline{m}', \bar{m}, \bar{m}') \in \mathscr{M}^4 : v_{\underline{m}} = v_{\underline{m}'}, v_{\bar{m}} = v_{\bar{m}'}, a_{\underline{m}} = o_{\underline{m}'} = a_{\bar{m}} = o_{\bar{m}'}\}$ and $J_2 = \{(m, \bar{m}, \bar{m}') \in \mathscr{M}^3 : v_{\bar{m}} = v_{\bar{m}'}, a_{\bar{m}} = o_{\bar{m}'},$ the presence of $v_{\bar{m}}$ in $a_{\bar{m}}$
interferes with the transit of $v_m\}$.

Constraints (17) and (18) are equivalent to Constraints (6) to 8): the former state that, for all pairs of vessels arriving at and departing from the same berth during the considered time horizon, either the first arrives and departs (movements $\underline{m}$ and $\underline{m}'$) before the second (movements $\bar{m}$ and $\bar{m}'$), or the opposite holds. The latter ensures the compatibility of vessel transits in narrow canals when a berth needs to be occupied by another vessel for some time during the considered time horizon. As for Model $\mathcal{P}_1$, analogous constraints can be defined for other similar incompatibility situations.

Tug usage constraints:

$$y_{m,m'} \geq z_{m,u} + z_{m',u} - 1 \qquad \forall u \in U,\ m, m' \in M_u : m < m', \tag{19}$$

$$t_{m'} \geq t_m + t^u_{m,m'} - M_{m,m'}(2 - p_{m,m'} - y_{m,m'}) \qquad \forall m, m' \in \mathscr{M} : m < m', U_m \cap U_{m'} \neq \emptyset, \tag{20a}$$

$$t_m \geq t_{m'} + t^u_{m',m} - M_{m,m'}(1 + p_{m,m'} - y_{m,m'}) \qquad \forall m, m' \in \mathscr{M} : m < m', U_m \cap U_{m'} \neq \emptyset, \tag{20b}$$

where $t^u_{m',m}$ and $t^u_{m,m'}$ are defined as for Constraints (10) of Model $\mathcal{P}_1$.

Constraints (19) set variables $y_{m,m'}$ equal to 1 if movements m and m' are served by the same tug. Constraints (20) impose that, if two movements m and m' are served by the same tug, then their starting times have to be separated so that the tug can complete its service with the former before starting its service with the latter. Here again, "big-Ms" are necessary to impose the disjunctive condition that a tug either serves m before m' or vice-versa. Value $M_{m,m'}$ is set equal to $l_m + t^u_{m,m'} - e_{m'}$ in (20a) and to $l_{m'} + t^u_{m',m} - e_m$ in (20b).

3.3. Comparison

We next propose a comparison between Models $\mathcal{P}_1$ and $\mathcal{P}_2$.

First, we note that the objective function of Model $\mathcal{P}_1$ is more general than the one of $\mathcal{P}_2$: it also allows the modeling of costs that are not directly proportional to the deviations of the actual starting times from the desired ones. On the other hand, the discretization step affects the solution of Model $\mathcal{P}_1$ with respect to Model $\mathcal{P}_2$.

Model $\mathcal{P}_1$ involves $O\left(|\mathscr{M}|\left(|T| + |U| + |\mathscr{M}|\right)\right)$ binary variables and $O\left(|\mathscr{M}|^2\left(|T| + |U|\right)\right)$ constraints. Differently, Model $\mathcal{P}_2$ involves $O\left(|\mathscr{M}|\left(|U| + |\mathscr{M}|\right)\right)$ binary variables, $3|\mathscr{M}|$ non negative continuous variables and $O\left(|\mathscr{M}|^2|U|\right)$ constraints. Obviously, since Model $\mathcal{P}_1$ makes use of time-indexed variables, the number of its variables and constraints depends on the size of the starting time windows T_m's and on the choice of the discretization step Δt.

In Model $\mathcal{P}_2$, it is possible to relax Constraints (12) on the starting time windows, by simply removing them. Differently, in $\mathcal{P}_1$, the relaxation of this constraints implies the definition of variables $x_{m,t}$ over a set T_m covering the whole time horizon T, for all $m \in \mathscr{M}$. The drawback of Model $\mathcal{P}_2$ is that it involves four sets of "big-M" constraints.

To conclude the section, let us mention that we tested several symmetry-breaking constraints to strengthen both models Some are based on literature papers [30], [15] and [3]. For some others, we derived some cuts to strengthen Model $\mathcal{P}_2$ inspired by the facet inducing inequalities proposed in [29, Section 2] for the *single machine scheduling problem*. The introduction of these cuts yields no discernible benefit in terms of solution quality or computational time. In the Appendix, we report the details regarding the tested cuts.

4. Solution algorithms

In this paper, we propose four algorithms to tackle the PSAP. The first pair of them consist in solving the models proposed in Section 3 using a commercial solver within a predefined time limit. They are denoted Ω_1 and Ω_2: the former solves Model $\mathcal{P}_1$, the latter Model $\mathcal{P}_2$. The second pair of algorithms still exploit the models but are characterized by an overlaying receding horizon heuristic, which we introduce in this section. As in Section 2, we denote by σ_π a feasible solution for a PSAP instance $\pi(\mathscr{M}, U, \mathcal{G})$. Specifically, σ_π is a set of starting times of movements $m \in \mathscr{M}$ and of assignments of tugs $u \in U$ to these movements. We write $\sigma_\pi = \emptyset$ when a feasible solution is not available.

Algorithm 1: Heuristic structure

Data: An instance $\pi(\mathscr{M}, U, \mathcal{G})$ of PSAP, a Model $\mathcal{P}_i$, a time limit τ

Result: A feasible solution $S(\pi)$

```
1 begin
    // constructive phase
2   σ_π := (M)ILP_SOLVER(π, P_i, τ);
3   if σ_π is optimal then return σ_π ;
4   else if σ_π = ∅ then σ_π := SOLUTION_GENERATING_PROCEDURE(π) ;
    // local search phase
5   if σ_π ≠ ∅ then σ_π := LOCAL_SEARCH_PROCEDURE(σ_π, π) ;
6   return σ_π;
```

The receding horizon heuristic includes a constructive phase and a local search phase, as described in Algorithm 1. Initially, the heuristic solves its corresponding model running a commercial (M)ILP solver (see Algorithm 1 line 2) with a time limit τ. Then, three situations may occur. If the solver returns an optimal solution (line 3), the heuristic stops. If the solver returns no feasible solution, the heuristic calls SOLUTION_GENERATING_PROCEDURE (line 4) that tries to build one. If it does not manage to do so, the heuristic stops. Otherwise, it moves to the local search phase. Finally, if the solver returns a feasible solution, which is not proven optimal, the heuristic goes directly to the local search phase. Here, it calls the procedure LOCAL_SEARCH_PROCEDURE (line 5) to perform a local search to determine at least a locally optimal solution.

The pseudocodes of SOLUTION_GENERATING_PROCEDURE and LOCAL_SEARCH_PROCEDURE are presented respectively in Algorithms 2 and 3.

SOLUTION_GENERATING_PROCEDURE is based on a receding horizon approach and relies on the fact that both Models $\mathcal{P}_1$ and $\mathcal{P}_2$ become more computationally tractable once the precedence relations among movements are fixed. In addition to the considered instance and model, it takes as input an integer value l and a computational time τ'. This procedure exploits two subsets of movements F and $\mathscr{M}'$ and a set *prec* of predetermined precedence relations between pairs of movements in $F \cup \mathscr{M}'$, where at least one of the two belongs to F. Initially, the elements of set $\mathscr{M}$ are ordered by increasing desired starting times, through function SORT() (see Algorithm 2 line 2). The procedure defines $\mathscr{M}'$ as the set of the first l movements through a call to function FIRST() (line 3), and F and *prec* as empty sets (lines 4 and 5). Then, at each iteration, the procedure considers instance $\pi'(F \cup \mathscr{M}', U, \mathcal{G})$ (line 7). Now, a call to the (M)ILP solver attempts to obtain solution $\sigma_{\pi'}$ for instance π' within time limit τ', when precedence relations in set *prec* are imposed (line 8). If a feasible solution cannot be found, the procedure returns an empty set (line 9). Otherwise, function EARLIEST() identifies movement $\bar{m} \in \mathscr{M}'$ that has minimum starting time in $\sigma_{\pi'}$ (line 11). If several movements have equal minimum starting time, $\bar{m}$ is a randomly selected one. Function SET_PRECEDENCES adds to set *prec* all the precedence relations between $\bar{m}$ and all other movements of $F \cup \mathscr{M}'$ according to $\sigma_{\pi'}$ schedule (line 12). Finally, $\bar{m}$ is included in F (line 13). Before starting the next iteration, $\mathscr{M}'$ is redefined as the set of the first l movements of $\mathscr{M} \setminus F$ built through a call to function

Algorithm 2: SOLUTION_GENERATING_PROCEDURE

Data: An instance $\pi(\mathscr{M}, U, \mathcal{G})$ of PSAP, a Model $\mathcal{P}_i$, an integer $l \leq |\mathscr{M}|$, a time limit τ'

Result: A (possibly empty) solution σ_π

1 **begin**

// inizialization

2 $\mathscr{M} = \text{SORT}(\mathscr{M}, w_m)$; // movements are ordered by increasing desired starting time

3 $\mathscr{M}' \coloneqq \text{FIRST}(\mathscr{M}, l)$; // current subset of movements to be scheduled

4 $F \coloneqq \emptyset$; // subset of movements whose relative precedences have already been fixed

5 $prec = \emptyset$; // set of predetermined precedence relations

// iteration

6 **while** $F \neq \mathscr{M}$ **do**

7 build instance $\pi'(F \cup \mathscr{M}', U, \mathcal{G})$;

8 $\sigma_{\pi'} \coloneqq$ (M)ILP_SOLVER$(\pi', \mathcal{P}_i, \tau', prec)$;

9 **if** $\sigma_{\pi'} = \emptyset$ **then return** $\sigma_{\pi'}$;

10 **else**

11 $\bar{m} \coloneqq \text{EARLIEST}(\mathscr{M}', \sigma_{\pi'})$; // $\bar{m}$: movement in $\mathscr{M}'$ with the earliest starting time in $\sigma_{\pi'}$

12 SET_PRECEDENCES$(\bar{m}, F \cup \mathscr{M}' \setminus \{\bar{m}\}, \sigma_{\pi'})$; // add in $prec$ precedences of $\sigma_{\pi'}$ between $\bar{m}$ and $m \in F \cup \mathscr{M}'$

13 $F \coloneqq F \cup \{\bar{m}\}$;

14 $\mathscr{M}' \coloneqq (\mathscr{M}' \setminus \{\bar{m}\}) \cup \text{FIRST}(\mathscr{M} \setminus F, 1)$;

15 **return** $\sigma_{\pi'}$;

FIRST() (line 14).

LOCAL_SEARCH_PROCEDURE is also based on a receding horizon approach and mimics the ideas presented for SOLUTION_GENERATING_PROCEDURE to reoptimize solution σ_π received in input together with an integer value k and a time limit τ''. Indeed, two subsets of movements F and $\mathscr{M}'$ and a set of predetermined precedences *prec* are considered. First, the procedure sorts the elements of set $\mathscr{M}$ by increasing starting times of σ_π (see Algorithm 3 line 2) and initializes F and *prec* as empty sets (lines 3 and 5). Here, set $\mathscr{M}'$ contains the first k elements of $\mathscr{M} \setminus F$ (see lines 4 and 14). Then, at each iteration, the (M)ILP solver is run, within time limit τ'', to obtain a solution $\hat{\sigma}_\pi$ that possibly improves σ_π (line 7). The solver considers the precedence relations of set *prec* as fixed and takes solution σ_π as a warm start. Now, a movement $\bar{m}$ is selected from $\mathscr{M}'$ with the same criterion as in SOLUTION_GENERATING_PROCEDURE and its precedence relations with all the other movements in $\mathscr{M}$ are included in set *prec* (lines 9 and 10). Finally, $\bar{m}$ is inserted in F (line 11). The procedure stops if a solution $\hat{\sigma}_\pi$ is proven to be optimal or if F coincides with $\mathscr{M}$ (lines 8 and 13).

We remark that the way of imposing the precedence relations of set *prec* in the (M)ILP solver in both SOLUTION_GENERATING_PROCEDURE and LOCAL_SEARCH_PROCEDURE depends on the model considered. Indeed, when it is Model $\mathcal{P}_1$, the precedence relation between two movements m, m' are imposed by setting $h_{m,m'} = +\infty$ if m' must be scheduled before m or $h_{m',m} = +\infty$ otherwise. Whereas, when the considered one is Model $\mathcal{P}_2$, the same precedence relations are imposed by setting the value of variable $p_{m,m'}$ either equal to 0 or to 1.

Hereinafter, Algorithm 1 is denoted by $\mathcal{E}_1$ ($\mathcal{E}_2$) if it uses Model $\mathcal{P}_1$ ($\mathcal{P}_2$) in the (M)ILP_SOLVER procedure.

Algorithm 3: LOCAL_SEARCH_PROCEDURE

Data: An instance $\pi(\mathscr{M}, U, \mathcal{G})$ of PSAP, a Model $\mathcal{P}_i$, an integer $k \leq |\mathscr{M}|$, a feasible solution σ_π, a time limit τ''

Result: A feasible solution σ_π

1 **begin**

// inizialization

2 $\mathscr{M} \coloneqq$ SORT$(\mathscr{M}, t_m)$; // movements are ordered by increasing starting time according to σ_π

3 $F \coloneqq \emptyset$; // subset of movements whose relative precedences have already been fixed

4 $\mathscr{M}' \coloneqq$ FIRST$(\mathscr{M}, k)$; // subset of movements whose relative precedences can be reoptimized

5 $prec =$ {precedence relations of σ_π that involve at least one movement in $\mathscr{M} \setminus \mathscr{M}'$} ;

// iteration

6 **while** $F \neq \mathscr{M}$ **do**

7 $\hat{\sigma}_\pi \coloneqq$ (M)ILP_SOLVER$(\pi, \mathcal{P}_i, \tau'', prec, \sigma_\pi)$;

8 **if** $\hat{\sigma}_\pi$ *is optimal* **then return** $\hat{\sigma}_\pi$;

9 $\bar{m} \coloneqq$ EARLIEST$(\mathscr{M}', \hat{\sigma}_\pi)$; // $\bar{m}$: movement in $\mathscr{M}'$ with the earliest starting time in $\hat{\sigma}_\pi$

10 SET_PRECEDENCES$(\bar{m}, \mathscr{M} \setminus \{\bar{m}\}, \hat{\sigma}_\pi)$; // add in *prec* precedences of $\hat{\sigma}_\pi$ between $\bar{m}$ and $m \in \mathscr{M}$

11 $F \coloneqq F \cup \{\bar{m}\}$;

12 $\sigma_\pi \coloneqq \hat{\sigma}_\pi$;

13 **if** $F = \mathscr{M}$ **then return** σ_π ;

14 **else** $\mathscr{M}' \coloneqq (\mathscr{M}' \setminus \{\bar{m}\}) \cup$FIRST$(\mathscr{M} \setminus F, 1)$;

5. Computational experiments

In this section, we present the results obtained by applying the proposed algorithms on two clusters of instances for the Port of Venice, our case study. We start with the presentation of the case study, and we continue with the results.

All the tests we present in this section are run on a laptop PC Dell XPS 15 9560, with an Intel Core i7-7700HQ at 2.80GHz and 16.0GB of installed RAM and with the solver XPRESS v8.5.6 64 bit. The following parameter setting is used for Algorithms $\mathcal{E}_1$ and $\mathcal{E}_2$. The time limits for the (M)ILP solver in the constructive phase of the heuristic, in SOLUTION_GENERATING_PROCEDURE and in LOCAL_SEARCH_PROCEDURE are set, respectively, to $\tau = 5$ minutes (Algorithm 1 line 2), $\tau' = 30$ seconds (Algorithm 2 line 8) and $\tau'' = 30$ seconds (Algorithm 3 line 7). In addition, the cardinality of set $\mathscr{M}'$ in SOLUTION_GENERATING_PROCEDURE (Algorithm 2 line 3) and in LOCAL_SEARCH_PROCEDURE (Algorithm 3 line 4) is set to five (i.e., $l = k = 5$).

5.1. Case study description

The Port of Venice is a medium-size Italian port (about 3 300 calls in 2019, for a total 78 000 000 gross tonnage, 593 000 containers–in TEU, and 1 600 000 passengers) situated in the Venetian Lagoon. As discussed in Section 2 and shown in Figure 1, its layout induces the graph structure depicted in the same figure. Two inlets, respectively named San Nicolò (north) and Malamocco (south), guarantee the access to the lagoon. Canals connect them with the passenger terminals in Marittima (old town center) and with the commercial terminals in Marghera (mainland), respectively. The harbormaster regulation imposes that all non-passenger vessels enter the port from the Malamocco inlet, whereas passenger vessels

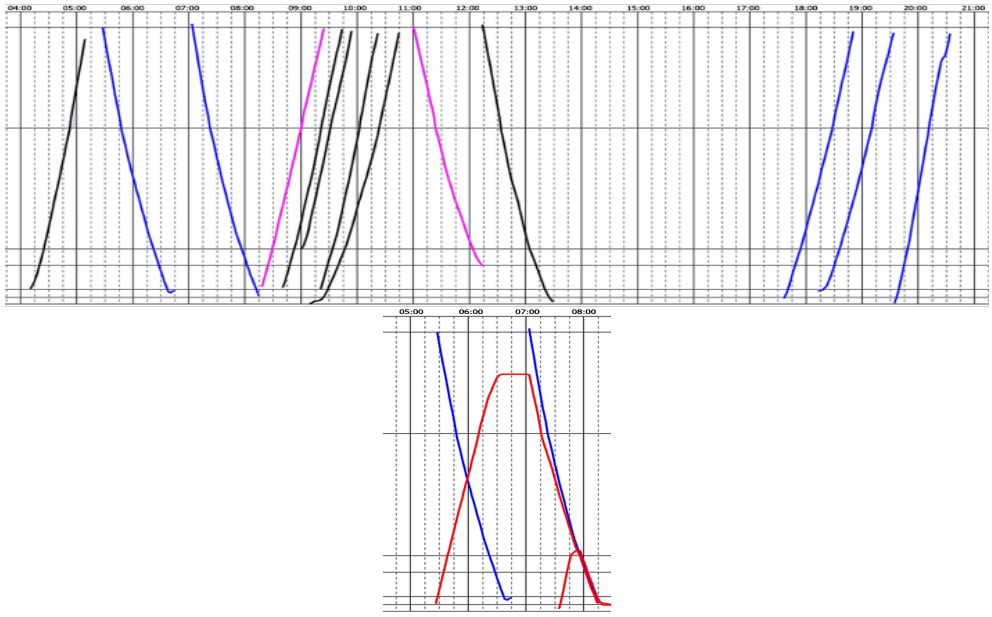

Figure 5: AIS kinetic diagram of the movements of vessels along the Port of Venice main canal (top) and of two tugs escorting a cruise vessel (bottom)

enter from the San Nicolò inlet if they are directed to the Marittima terminals and from the Malamocco inlet otherwise. Currently, the canal joining the Marittima and the Marghera terminals is too shallow for medium size passenger vessels.

The tree-like topology of the port (Figure 1 (right)) counts 281 vertices: 163 active berths, 2 inlets, 97 connection and 19 other relevant points. A more detailed description of the port structure can be found in the harbormaster ordinances [2, 1]. In addition, we provide the vertex and the edge lists of $\mathcal{G}$ as supplementary material. A set U of 13 tugs is always available.

The first cluster of instances includes nine real instances, henceforth denoted $\pi_i := \pi_i(M_i, U, \mathcal{G})$, $i = 1, \ldots, 9$. They correspond to the movements requested in the most congested days in the period 2011–2019. These data were deduced either from the Port Authority records or from the analysis of the data transmitted by the Automatic Identification Systems (AIS) of the vessels and tugs involved in the considered movements. As an example, Figure 5 (top) represents a kinetic diagram obtained by elaborating the AIS data of some vessels sailing along the main canal of the Port of Venice. Specifically, it shows the movements of three cruise vessels (blue lines), five freight vessels (black lines) and one tanker (magenta lines). The horizontal axis represents time, while the vertical one is space, going from Malamocco inlet (top) to Marghera terminals (bottom). By considering the cruise vessels, we deduce that two of them enter the lagoon and reach their berths in the morning and all of them leave the port in the evening. Moreover, from Figure 5 (bottom) we infer how two tugs (red lines) escort the second cruise vessel entering the port.

The second cluster of instances includes 100 random instances generated on the basis of the data of 608 real movements, by randomly changing the desired starting times. Movements may also differ for initial and final berth: we have sometimes changed the destination of a movement to a close berth on the same terminal when not doing so would have resulted in a trivially infeasible instance. For example, we did so when two vessels would have required to be moored to the same berth

at the end of the time horizon T. Each instance includes a random number of movements, between 35 and 60, chosen among the 608 at disposal. During the instance generation, we run XPRESS on each instance with a time limit of ten minutes. If an instance is proven infeasible in this time, it is discarded and a new one is generated. Random instances appear in general more complicated than real ones. They are available in the supplementary material.

In the Port of Venice, movements are generally served by one or two tugs (on average 1.32 tugs), rarely by zero or more than two. All tugs available in the Port can serve any movement. In the following, we consider the associated instances related to the original ones, in the sense described in Section 2: each movement requires to be served by exactly one tug. Hereinafter, the cardinality of the set of movements $|\mathscr{M}|$ is equal to the number of equivalent movements served by a single tug. This value varies between 54 and 70 for the real instances, and between 55 and 117 for the random ones. Movements have sailing times that vary within a range between 5 and 160 minutes, with an average of 102 minutes. We assume a starting time window T_m of four hours for all movements m of the real instances, and of six hours for those of the randomly generated ones. The deviation cost is equal to five and ten for freight and passenger vessels, respectively. This is the cost for deviating a movement actual starting time from its desired one of a time unit equal to the discretization step. We assume the weighted deviation cost to increases linearly as a function of the deviation itself. For example, if m is a movement of a passenger vessel $c_{m,t} = 10|t - w_m|$ in Model $\mathcal{P}_1$ and $c_m^+ = c_m^- = 10$ in Model $\mathcal{P}_2$.

In case of Model $\mathcal{P}_1$, which uses time-indexed variables, we need to round all times to a multiple of the discretization step Δt. To be able to compare algorithm performance, we use the same instances for all algorithms. In particular, relevant times are fixed as follows. For each movement m we round the desired starting time w_m to the closest Δt multiple. Differently, we round the movement sailing time S_m to the smallest Δt multiple greater than S_m. The same is done for the tugs sailing times, for the pick up/drop time required to connect/disconnect tugs to vessels and for the separation and headway times between movements. In the analysis, we consider three different discretization steps $\Delta t = 5, 10, 15$ minutes. Their choice is motivated by the context: indeed the uncertanties which afflict navigation make a finer time resolution meaningless. Given how the time related data are rounded, five minutes is the most pertinent value because it allows the best exploitation of the capacity of the Port in terms of number of vessel movements scheduled per day. The other steps are mostly used to understand whether the performance of the proposed algorithms is sensitive to this step. Indeed, as instance data vary, solution values related to different discretization steps cannot always be compared with each other. For example, an original desired starting time at 7:08 will be set to 7:10 with $\Delta t = 5$, and to 7:15 with $\Delta t = 15$. Suppose the optimization schedules the corresponding movement at 7:15. With $\Delta_t = 5$, this will cost 50 in both models. With $\Delta_t = 15$, it will cost 0.

5.2. Experimental results

We first compare the performance of Algorithms Ω_1 and Ω_2 on the real instances. For each instance, Table 1 reports the number of movements and the value of the solution returned by each algorithm for each discretization step, together with the percentage optimality gap when a time limit of five minutes is considered. We use symbol "−" if an algorithm detects that an instance is infeasible.

Table 1: Solutions of the real instances obtained by the Ω_1 and Ω_2 algorithms within a five minutes time limit with $\Delta t = 5, 10, 15$.

Δt		5				10				15			
Algorithm		Ω_1		Ω_2		Ω_1		Ω_2		Ω_1		Ω_2	
Inst	$\lvert\mathscr{M}\rvert$	val	gap%	val	gap%	val	gap%	val	gap%	val	gap%	val	gap%
π_1	69	2505	67	1155	24	730	1	810	41	500	2	520	32
π_2	70	1640	63	1165	47	1020	57	785	55	400	13	520	41
π_3	69	175	0	175	0	145	0	145	0	145	0	145	0
π_4	58	1790	13	1790	13	–	–	–	–	–	–	–	–
π_5	59	2735	55	2035	54	1120	31	1295	59	830	13	815	46
π_6	61	100	0	100	0	85	0	85	0	145	0	145	0
π_7	57	490	6	520	12	350	6	350	6	260	8	260	8
π_8	65	55	0	55	0	70	14	70	14	45	0	45	11
π_9	54	1180	0	1180	0	705	0	705	0	525	0	525	0

Both algorithms provide at least a feasible solution for all instances except for π_4 which is proven to be infeasible with the discretization step equal to 10 and 15 minutes. However, Ω_1 and Ω_2 manage to find a feasible solution for π_4 if $\Delta t = 5$, which is in line with the fact that larger discretization steps reduce the capacity of the port. The two algorithms solve four instances to optimality with $\Delta t = 5$. This number decreases to three with $\Delta t = 10, 15$. Both algorithms always solve to optimality the same instances, apart from π_8 with $\Delta t = 15$: Ω_1 proves the optimality of the solution whereas Ω_2 finds the optimal value but cannot close the gap in the time limit. The percentage gap of the two algorithms for the instances not solved to optimality is always larger than 5%, but for instance π_1 when solved by Ω_1 with $\Delta t = 10, 15$.

Despite the different models, for some instances the solver returns the same optimality gap: π_4 with $\Delta t = 5$, π_7 and π_8 with $\Delta_t = 10$ and π_7 with $\Delta t = 15$. The XPRESS log files report a different evolution of the lower bounds throughout the runs. However, the returned solutions are the same and they are found deep in the branch-and-bound search tree. The same holds for the lower bounds. We believe that these lower bounds are obtained considering the same movement precedences and tug assignments in the linear relaxations of the two models. Indeed, these linear relaxations provide the same values for the starting times t_m (defined in Model $\mathcal{P}_1$ as $t_m = \sum_{t \in T_m} t x_{m,t}$) when either the movement precedences and tug assignments are fixed, or the corresponding disjunctive constraints are not active.

In order to further assess the performances of Algorithms Ω_1 and Ω_2 we replicate the tests on the real instances imposing a time limit of one hour. Under this setting, no dominance relation can be inferred for the two algorithms and no trend can be identified for the impact of the discretization step on their performance. With respect to the results obtained within a five minutes time limit, we observe a good improvement of the percentage optimality gap. This improvement is equal to 32%, on average. However, the two algorithms are not able to prove the optimality of any additional solution.

Table 2 reports the results of $\mathcal{E}_1$ and $\mathcal{E}_2$ on the real instances. Columns *val* indicate the final solution value returned by the two algorithms, while *impr%* is the percentage improvement brought by the constructive phase with respect to the local search one. Finally, columns *time* report the overall solution times. In all these experiments, a feasible solution is found

Table 2: Solutions of the real instances obtained by the $\mathcal{E}_1$ and $\mathcal{E}_2$ algorithms with $\Delta t = 5, 10, 15$.

Δt	5						10					
Algorithm	$\mathcal{E}_1$			$\mathcal{E}_2$			$\mathcal{E}_1$			$\mathcal{E}_2$		
Inst	val	impr%	time [s]	val	impr%	time [s]	val	impr%	time [s]	val	impr%	time [s]
π_1	1660	54	1075	1135	2	456	730	0	384	735	10	397
π_2	945	74	1287	785	48	1568	570	79	731	575	37	1249
π_3	175	0	14	175	0	13	145	0	14	145	0	25
π_4	1790	0	600	1790	0	356	–	–	–	–	–	–
π_5	1980	38	1218	1935	5.17	584	1010	11	491	1080	20	497
π_6	100	0	8	100	0	2	85	0	2	85	0	3
π_7	490	0	617	520	0	431	350	0	547	350	0	1154
π_8	55	0	9	55	0	3	70	0	393	70	0	1358
π_9	1180	0	271	1180	0	21	705	0	24	705	0	30

Δt	15					
Algorithm	$\mathcal{E}_1$			$\mathcal{E}_2$		
Inst	val	impr%	time [s]	val	impr%	time [s]
π_1	500	0	507	515	1	443
π_2	400	0	980	485	7	431
π_3	145	0	7	145	0	31
π_4	–	–	–	–	–	–
π_5	830	0	342	800	2	362
π_6	145	0	3	145	0	4
π_7	260	0	376	260	0	443
π_8	45	0	3	45	0	772
π_9	525	0	10	525	0	8

in the constructive phase by solving Model $\mathcal{P}_1$ or $\mathcal{P}_2$ with XPRESS. Hence, the *impr%* columns show the improvement of the heuristic with respect to corresponding Ω algorithm presented in Table 1.

First of all, let us remark that when the solution time is smaller than 300 seconds, the optimal solution is returned by the (M)ILP solver and no local search needs to be started: the $\mathcal{E}$ and Ω algorithms are completely equivalent. For the remaining instances, the local search phase manages to improve the solution returned after five minutes in 14 out of 29 experiments, regardless of the discretization step. The average percentage improvements observed with Algorithms $\mathcal{E}_1$ and $\mathcal{E}_2$ w.r.t. Ω_1 and Ω_2 are equal to 18% and 7%, respectively. As expected, the best improvements are attained where the optimality gaps returned by Algorithms Ω_1 and Ω_2 (Table 1) are large: e.g., instances π_1 and π_2 with $\mathcal{E}_1$ and $\Delta t = 5$, and instance π_2 with $\mathcal{E}_2$ and $\Delta t = 5$.

On average, Algorithm $\mathcal{E}_2$ is faster than $\mathcal{E}_1$ with $\Delta t = 5$, the contrary happens with $\Delta t = 10$ and 15. The average solution times of $\mathcal{E}_1$ and $\mathcal{E}_2$ respectively are 567 and 382 seconds with $\Delta t = 5$, 323 and 592 seconds with $\Delta t = 10$ and

278 and 312 seconds with $\Delta t = 15$. The instance-wise comparison of the computational time respects the trend observed on average, although some exceptions occur: $\mathcal{E}_1$ is faster than $\mathcal{E}_2$ on instance π_2 with $\Delta t = 5$ and $\mathcal{E}_2$ is faster than $\mathcal{E}_1$ on instances π_1, π_2 and π_9 with $\Delta t = 15$.

From the results on the real instances presented in Tables 1 and 2 a correlation may be inferred between the discretization step and the performance of the algorithms that solve Model $\mathcal{P}_1$ against the ones that solve Model $\mathcal{P}_2$. Indeed, in general, it appears that Ω_1 and $\mathcal{E}_1$ perform better than Ω_2 and $\mathcal{E}_2$ when the discretization step is set to 10 and 15 minutes. The contrary happens if this step is set to five minutes.

We end the discussion over the real instances by remarking that our algorithms quickly produce reasonable solutions which can be used by the harbormaster of the Port of Venice as a support tool to determine the schedule of the vessel movements to be implemented. Indeed, building the movement schedules and tug assignments from scratch requires the skills of experienced port operators and, typically, long times: the solutions provided by our algorithms can be used as a starting point by the operators, who then may perform some refinements to consider additional criteria. In addition, algorithms can be used to quickly assess different scenarios in presence, e.g., of adverse weather and sea conditions that force to reconsider previous schedule decisions. As an example, in Figure 6 we compare the actual schedules of vessel movements (top diagram) with those obtained by Algorithm Ω_2 (bottom diagram) in an representative day. Specifically, the kinetic diagrams show the vessel movements (or the portion of them) occurring along the main canals of the Port of Venice. They are those connecting the two inlets of the Lagoon and passing through the commercial and passenger terminals. The upper part of the diagrams show the movements toward and from the commercial terminals and the lower part those toward and from the passenger terminals. As for Figure 5, the actual movements' schedules (top diagram) are determined through the AIS data transmitted by the vessels, which are publicly available. The solutions displayed in the two kinetic diagrams are very similar: vessels are arranged in convoys and the precedence relations among them are in general the same in the two diagrams, but for few exceptions (e.g., the magenta lines). The small differences in the movement schedules of the two solutions are due to the rounding of the movements' desired starting time applied when the optimization approach is considered. Finally, the Gantt diagram shown in Figure 7 reports the vessel movements of the same representative day considered in Figure 6, when scheduled by Algorithm Ω_2: here also the inner canals of the Port are taken into account. Each movement is represented by a horizontal segment. If a movement requires tug services, lower and upper ticks on its segment represent their start and end time. Due to the pick up/drop time, tugs must be in the area close to the beginning/completion of their service 15 minutes before/after their start and end time. The diagram shows also the services provided by two tugs, named, respectively, E and I. The time at which the tugs start their first service is indicated above the upper limit of the graph. Tug E starts its service on vessel S around 05:00 am, then it joins tug I at 07:15 am to escort vessel M from one inlet to the passenger terminals. Finally, it escorts again the latter vessel when it leaves the port, at 05:15 pm. Differently, tug I has no service prior to the one on vessel M and, after that, it escorts three cargo vessels in the afternoon.

Table 3 summarizes the results obtained by solving the 100 random instances by means of $\mathcal{E}_1$ and $\mathcal{E}_2$, and implicitly

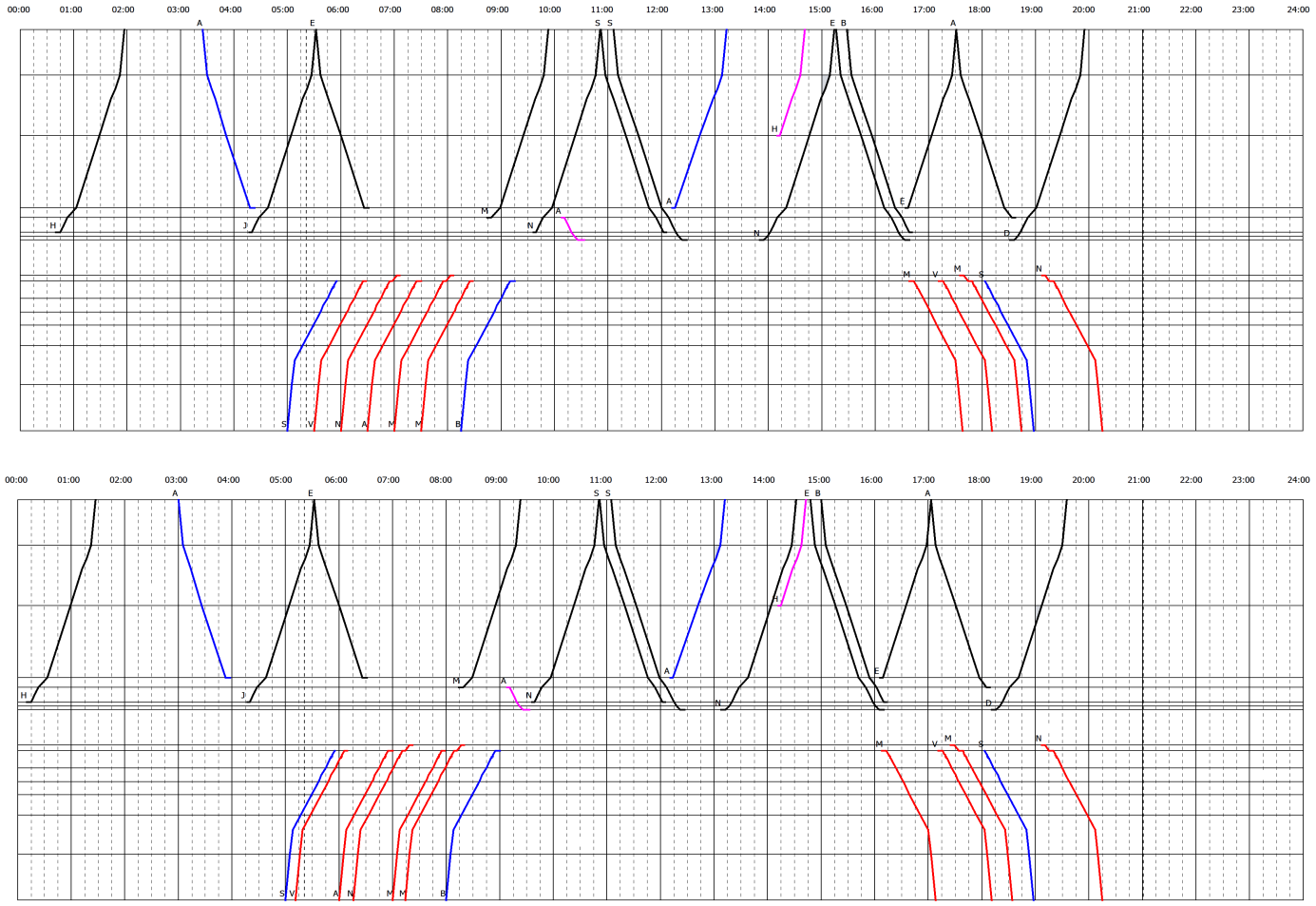

Figure 6: AIS kinetic diagram (top) and kinetic diagram arising from the optimization algorithm (bottom) of vessel movements along the port of Venice main canals in an average day.

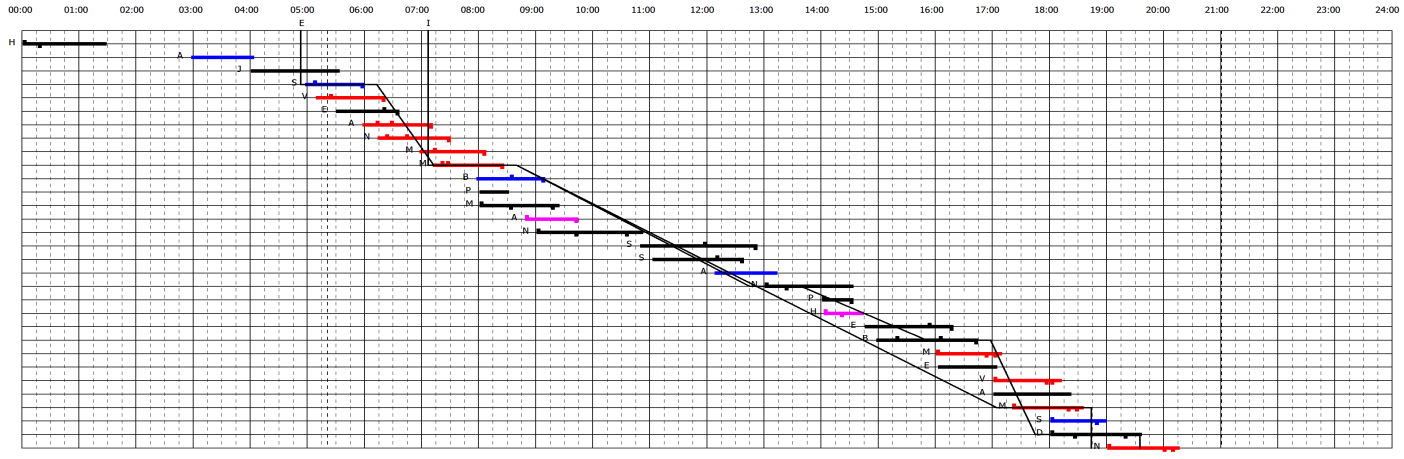

Figure 7: Gantt diagram of vessel movements along the port of Venice canals in an average day. It also shows the services provided by two tugs.

Table 3: Computational results obtained by solving the random instances by means of algorithms $\mathcal{E}_1$ and $\mathcal{E}_2$ with $\Delta t = 5, 10, 15$ min.

Δt	5		10		15	
Algorithm	$\mathcal{E}_1$	$\mathcal{E}_2$	$\mathcal{E}_1$	$\mathcal{E}_2$	$\mathcal{E}_1$	$\mathcal{E}_2$
# solved inst.	63	94	72	90	79	90
# not solved inst.	37	6	28	10	21	10
# inst. solved to optimality in constr. phase with (M)ILP	7	17	18	14	22	13
avg. optimality time	194	66	79	103	52	75
# inst. solved to feasibility in constr. phase with (M)ILP	44	77	48	73	54	75
# inst. solved to feasibility in constr. phase with SGP	12	0	6	3	3	2
avg. improvement through local search	30%	37%	28%	36%	19%	34%
avg. constr. phase time	490	258	326	319	279	297
avg. local search phase time	1267	1089	1030	1029	910	949
avg. solution time	1619	1157	1088	1194	940	1114
# inst. won (value)	11	71	41	32	52	20
# inst. won (time)	1	93	36	54	58	32

Ω_1 and Ω_2, with the three considered discretization steps. The rows of the table report respectively: the number of solved and not solved instances (for which no feasible solution is found); the number of instances for which the constructive phase builds an optimal solution by means of the (M)ILP solver ((M)ILP); the average solution time of these instances; the number of instances for which the constructive phase builds a feasible solution by means of the (M)ILP solver and by means of SOLUTION_GENERATING_PROCEDURE (SGP); the average percentage improvement achieved by the local search phase; the average computational time of the constructive and of the local search phase, and their sum; the number of instances for which the algorithm beats the competitor in terms of objective function value or computational time (the tie cases are not considered). As discussed for Table 2, the results obtained by the (M)ILP solver correspond to the ones that Algorithms Ω_1 and Ω_2 obtain in five minutes.

From Table 3, we observe that although Algorithm $\mathcal{E}_1$ struggles to solve as many instances as $\mathcal{E}_2$, they both provide a solution for the majority of the instances (at least 63 out of 100) regardless of the discretization step. Moreover, in the constructive phase, Algorithms Ω_1 and Ω_2 prove the optimality for at least 11% of the corresponding solved instances (Ω_1 with $\Delta t = 5$), with a peak of 28% (Ω_1 with $\Delta t = 15$). Now, considering the instances for which the optimality of a solution is not proven within five minutes, $\mathcal{E}_1$ $\mathcal{E}_2$ provide a remarkable improvement over Ω_1 and Ω_2. In the same table we report the average percentage improvement brought by the local search phase: it spans from 19% to 30% for $\mathcal{E}_1$ and from 34% to 36% for $\mathcal{E}_2$.

To guarantee a fair comparison, we also run Algorithms Ω_1 and Ω_2 setting as time limit on each instance the computational time used by $\mathcal{E}_1$ and $\mathcal{E}_2$. The discretization step does not appear to have a particular impact on the obtained

results. Thus, we report here only the results obtained with $\Delta t = 5$. Here, Ω_1 beats $\mathcal{E}_1$ in 43 out of the 100 instances: it finds a solution to 18 instances for which the latter fails to do so, and it returns solution values 30% better, in average. The opposite happens for 24 instances: $\mathcal{E}_1$ finds solutions to 3 instances where Ω_1 does not, and the average improvement is 18%. The comparison between Ω_2 and $\mathcal{E}_2$ goes in a different direction. In 11 out of the 100 instances, Ω_2 beats $\mathcal{E}_2$: it finds 4 additional solutions and it improves the solution values of 15%, in average. Instead, $\mathcal{E}_2$ wins in 71 instances by improving their solution value of an average of 29%.

We end our analysis by discussing the dominance relation between the algorithms that solve the two different models. The trend emerged in the analysis of the results of the real instances is confirmed by the results of Table 3: the smaller the discretization step, the better the algorithms that solve Model $\mathcal{P}_2$ perform over the ones that solve $\mathcal{P}_1$. Indeed, if the discretization step is set to $\Delta t = 5$, Algorithm $\mathcal{E}_2$ (Ω_2) outperforms $\mathcal{E}_1$ (Ω_1) according to all performance indicators considered: in general it obtains better solutions and it does it faster. If $\Delta t = 10, 15$, the average computational times of Algorithms $\mathcal{E}_2$ (Ω_2) and $\mathcal{E}_1$ (Ω_1) are comparable. However, in the instance-wise comparison (last row of the table) $\mathcal{E}_2$ prevails when $\Delta t = 10$, while the opposite happens when $\Delta t = 15$. Regarding solution quality, when the discretization step is larger than five minutes the algorithms solving $\mathcal{P}_1$ outperform the other ones: Ω_1 provides more optimal solutions than Ω_2 and $\mathcal{E}_1$ beats $\mathcal{E}_2$ in the instance-wise comparison (second last row of the table).

The explanation of the trend is related to the high sensitivity of the time-indexed model to the discretization step: the smaller this step, the bigger the size of Model $\mathcal{P}_1$ in terms of number of variables and constraints. Specifically, although the order of magnitude of the number of variables and constraints is not largely different between the two models, as discussed in Section 3.3, we observe that $\mathcal{P}_1$ ends up having about six times more constraints than $\mathcal{P}_2$ in our instance. After XPRESS pre-solve, this ratio decreases to about three. The same pre-solve manages to reduce the numbers of variables more for $\mathcal{P}_2$ than for $\mathcal{P}_1$, the final ratio being about 2.

6. Conclusions

In this paper, we defined and formulated the in-Port vessel Scheduling and tug Assignment Problem, and we proposed four algorithms for tackling it. In this problem, vessel movements and tugs must be scheduled to optimize the access to a port infrastructure. In particular, we focused on the case of canal harbors. We formally showed that instances in which vessels require the use of multiple tugs can be mapped into equivalent instances in which a one-to-one relation holds. The four proposed algorithms are based on the solution of mathematical programming models. Specifically, we proposed a time-indexed model, and a continuous time one. In two algorithms, we tackled these models considering a receding horizon framework within a local search approach, where a subset of variables are set as in the solution at the center of the neighborhood explored. Furthermore, we presented additional cuts to strengthen the models. We ran experiments on real and realistic randomly generated instances representing traffic at the Port of Venice, in Italy.

The results of the experimental analysis show that all our algorithms can find good, often optimal, solutions to all real instances considered in an acceptable computational time. On the more difficult realistic randomly generated instances, the choice of the discretization step has a notable impact on the dominance of the algorithms solving the continuous time model

over those solving the time-indexed one. If $\Delta t = 5$ the algorithms based on the continuous time model outperform the others, but the contrary happens if $\Delta t = 15$. Moreover, the receding horizon algorithms bring a remarkable improvement over the other ones. On the contrary, no benefit emerges when adding cuts to the models. As a discretization step larger than five minutes does not allow to exploit capacity in a satisfactory way, we conclude that the continuous time model is more appropriate to tackle this problem in a port at least as large and busy as the Venice one. In smaller ports, the difference probably becomes smaller, but we consider that the application of this model remains an appropriate and conservative choice.

Further research will be committed to the integration of the algorithms proposed in an actual decision support tool. Indeed, in principle, our algorithms may not be able to find a feasible solution to an instance in the available computational time, either because it does not exist or because it is simply very difficult to spot. Several possibilities can be explored to support port authorities in their scheduling decisions when this happens. For example, the size of the starting time windows allowed for each movement, or for some of them, may be progressively enlarged. Another option may consists in allowing movements to be served by additional virtual tugs at higher costs, or allowing cancellation of movements. The selection of the best approach to deal with particularly complex cases will have to be deeply investigated, to properly balance complexity and benefit of the various alternatives.

Future work will be devoted to the exploration of further modeling possibilities for the PSAP. An interesting research direction is inspired by [36], who proposes a minimum cost multi-commodity network flow model for the locomotive scheduling problem. As we discussed in the paper, the two problems are strictly related. In principle, a model for the PSAP inspired by the one for locomotive scheduling may be derived. However, its size in terms of number of variables and constraints makes the solution of the PSAP instances considered in this paper hardly possible using a commercial MILP solver. An *ad-hoc* solution algorithm will thus have to be designed.

Acknowledgments

This study was partially funded by the "Smart PORt Terminals - SPORT" MIUR-PRIN project (grant number: 2015XAPRKF) financed by the Italian Government.

Bibliography

References

[1] Access from the sea. `https://www.port.venice.it/en/access-from-the-sea.html`. Accessed: 2021-06-19.

[2] Ordinanze. `https://www.guardiacostiera.gov.it/venezia/Pages/ordinanze.aspx`. Accessed: 2021-06-19.

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

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

[5] Bussieck, M. R., Winter, T., and Zimmermann, U. T. (1997). Discrete optimization in public rail transport. *Mathematical Programming*, 79(1):415–444.

[6] Carlo, H. J., Vis, I. F. A., and Roodbergen, K. J. (2015). Seaside operations in container terminals: literature overview, trends, and research directions. *Flexible Services and Manufacturing Journal*, 27(2):224–262.

[7] Carrer, N. L., Ferson, S., and Green, P. L. (2020). Optimising cargo loading and ship scheduling in tidal areas. *European Journal of Operational Research*, 280(3):1082 – 1094.

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

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

[10] Du, Y., Chen, Q., Lam, J. S. L., Xu, Y., and Cao, J. X. (2015). Modeling the impacts of tides and the virtual arrival policy in berth allocation. *Transportation Science*, 49(4):939–956.

[11] Franzese, L. A. G., Abdenur, L. O., Botter, R. C., Starks, D., and Cano, A. R. (2004). Simulating the panama canal: present and future. In *Proceedings of the 2004 Winter Simulation Conference, 2004.*, volume 2, pages 1835–1838 vol.2.

[12] Gharehgozli, A. H., Roy, D., and de Koster, R. (2016). Sea container terminals: New technologies and or models. *Maritime Economics & Logistics*, 18(2):103–140.

[13] Hill, A., Lalla-Ruiz, E., Voss, S., and Goycoolea, M. (2019). A multi-mode resource-constrained project scheduling reformulation for the waterway ship scheduling problem. *Journal of Scheduling*.

[14] Jackman, J., de Castillo, Z. G., and Olafsson, S. (2011). Stochastic flow shop scheduling model for the panama canal. *Journal of the Operational Research Society*, 62(1):69–80.

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

[16] Ji, B., Yuan, X., Yuan, Y., Lei, X., Fernando, T., and Iu, H. H. (2019). Exact and heuristic methods for optimizing lock-quay system in inland waterway. *European Journal of Operational Research*, 277(2):740 – 755.

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

[18] Jünger, M., Liebling, T. M., Naddef, D., Nemhauser, G. L., Pulleyblank, W. R., Reinelt, G., Rinaldi, G., and Wolsey, L. A. (2009). *50 Years of integer programming 1958-2008: From the early years to the state-of-the-art*. Springer Science & Business Media.

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

[20] Li, F., Yang, D., Wang, S., and Weng, J. (2019). Ship routing and scheduling problem for steel plants cluster alongside the yangtze river. *Transportation Research Part E: Logistics and Transportation Review*, 122:198 – 210.

[21] Lübbecke, E., Lübbecke, M. E., and Möhring, R. H. (2019). Ship traffic optimization for the kiel canal. *Operations Research*, 67(3):719–812.

[22] Mavrakis, D. and Kontinakis, N. (2008). A queueing model of maritime traffic in bosporus straits. *Simulation Modelling Practice and Theory*, 16(3):315 – 328.

[23] Meisel, F. and Fagerholt, K. (2019). Scheduling two-way ship traffic for the kiel canal: Model, extensions and a matheuristic. *Computers & Operations Research*, 106:119 – 132.

[24] Nishi, T., Okura, T., Lalla-Ruiz, E., and Voß, S. (2020). A dynamic programming-based matheuristic for the dynamic berth allocation problem. *Annals of Operations Research*, 286:391–410.

[25] Passchyn, W., Coene, S., Briskorn, D., Hurink, J. L., Spieksma, F. C., and Berghe, G. V. (2016). The lockmasters problem. *European Journal of Operational Research*, 251(2):432 – 441.

[26] Pellegrini, P., di Tollo, G., and Pesenti, R. (2019). Scheduling ships movements within a canal harbor. *Soft Computing*, 23:2923–2936.

[27] Pellegrini, P., Marlire, G., Pesenti, R., and Rodriguez, J. (2015). Recife-milp: An effective milp-based heuristic for the real-time railway traffic management problem. *IEEE Transactions on Intelligent Transportation Systems*, 16(5):2609–2619.

[28] Piu, F. and Speranza, M. G. (2014). The locomotive assignment problem: a survey on optimization models. *International Transactions in Operational Research*, 21(3):327–352.

[29] Queyranne, M. and Shulz, A. S. (1994). Polyhedral approaches to machine scheduling. Technical Report 408, Technische Universitat Berlin.

[30] Sherali, H. D. and Smith, J. C. (2001). Improving discrete model representations via symmetry considerations. *Management Science*, 47(10):1396–1407.

[31] Ulusçu, Ö. S., Özbaş, B., Altıok, T., Or, I., and Yılmaz, T. (2009). Transit vessel scheduling in the strait of istanbul. *Journal of Navigation*, 62(1):5977.

[32] Vaidyanathan, B., Ahuja, R. K., Liu, J., and Shughart, L. A. (2008). Real-life locomotive planning: New formulations and computational results. *Transportation Research Part B: Methodological*, 42(2):147 – 168.

[33] Verstichel, J., Causmaecker, P. D., Spieksma, F., and Berghe, G. V. (2014). The generalized lock scheduling problem: An exact approach. *Transportation Research Part E: Logistics and Transportation Review*, 65:16 – 34.

[34] Wang, X., Arnesen, M. J., Fagerholt, K., Gjestvang, M., and Thun, K. (2018). A two-phase heuristic for an in-port ship routing problem with tank allocation. *Computers & Operations Research*, 91:37 – 47.

[35] Wawrzyniak, J., Drozdowski, M., and ric Sanlaville (2020). Selecting algorithms for large berth allocation problems. *European Journal of Operational Research*, 283(3):844 – 862.

[36] Xu, X., Li, C.-L., and Xu, Z. (2018). Integrated train timetabling and locomotive assignment. *Transportation Research Part B: Methodological*, 117:573 – 593.

[37] Zhang, X., Lin, J., Guo, Z., and Liu, T. (2016). Vessel transportation scheduling optimization based on channelberth coordination. *Ocean Engineering*, 112:145 – 152.

[38] Zhen, L., Liang, Z., Zhuge, D., Lee, L. H., and Chew, E. P. (2017). Daily berth planning in a tidal port with channel flow control. *Transportation Research Part B: Methodological*, 106:193 – 217.

[39] Zheng, J., Zhang, W., Qi, J., and Wang, S. (2019). Canal effects on a liner hub location problem. *Transportation Research Part E: Logistics and Transportation Review*, 130:230 – 247.

Appendix: Cuts

In this section we report the details regarding the cuts that we consider to strengthen our two models.

Let us observe that, independently from how it is modeled, the PSAP has a characteristic that makes it particularly difficult to solve. Specifically, it presents multiple optimal solutions when set U contains subsets of equivalent tugs that can swap their service assignments, as it usually occurs in the practice. Moreover, when the continuous time model is used, it includes disjunctive constrains. Indeed, the PSAP can be solved in polynomial time when these characteristics do not hold, that is, if the sequencing of movements is *a-priori* fixed and each movement can be served by a single tug, i.e., $|U_m| = 1$ for all $m \in \mathscr{M}$. Under these circumstances, Model $\mathcal{P}_2$ becomes a linear programming model as the values of the binary variables $p_{m,m'}$ and $z_{u,m}$, and hence of variables $y_{m,m'}$, are fixed.

The following sets of symmetry-breaking constraints reduce the number of symmetric solutions if the available tugs are compatible with all vessels, that is $U_m = U$, for all $m \in \mathscr{M}$:

$$\sum_{r \in U: r \leq m} z_{m,r} = 1 \qquad \forall m \in \mathscr{M} : m \leq |U|, \tag{21a}$$

$$z_{m,r} \leq \sum_{m' \in \mathscr{M}: r-1 \leq m' \leq m-1} z_{m',r-1} \qquad \forall r \in U \setminus \{1\}, m \in \mathscr{M} : r \leq m. \tag{21b}$$

Cuts (21) allow tug r to serve movement m only if tug $r-1$ has served at least a previous movement m', with $r-1 \leq m' \leq m-1$. In particular, Cuts (21a) impose that movement $m \in \mathscr{M}$ is served by one of the first m tugs, until $m \leq |U|$. Cuts (21b) state that a tug can be used to serve a movement m only if the one with previous index is used to serve a movement with index smaller than m. Cuts (21) can be alternatively written as lexicographic ordering constraints [30]:

$$\sum_{m' \in \mathscr{M}: m' \leq m} 2^{(m-m')} z_{m',r} \leq \sum_{m' \in \mathscr{M}: m' \leq m} 2^{(m-m')} z_{m',r-1} \qquad \forall r \in U \setminus \{1\},\, m \in \mathscr{M}.$$

Coefficients $2^{(m-m')}$ make these cuts numerically unstable when m is large [18]. Due to the size of our instances we cannot include them in our computational study, although they show good results in [15] and [3] (m up to 6 and 15 respectively).

An alternative set of cuts can be formulated as:

$$\sum_{m \in \mathscr{M}} \beta_m z_{m,r} \leq \sum_{m \in \mathscr{M}} \beta_m z_{m,r-1} \qquad \forall r \in U \setminus \{1\}, \tag{22}$$

where $\beta_m > 0$ for all $m \in \mathscr{M}$. Cuts (22), inspired by those presented in [15] and [3], break the symmetry by ordering the tugs. They impose that the weighted sum of the movements served by tug r must be less than the weighted sum of the movements served by tug $r-1$. In particular, if $\beta_m = 1$ for all $m \in \mathscr{M}$, Cuts (22) simply impose that a tug with lower index should not serve fewer movements.

The interested reader is referred to the seminal paper [30] and to [15] and [3] for similar cuts applied to scheduling problems.

We also consider the following cuts to strengthen Model $\mathcal{P}_2$, to better deal with disjunctive Constraints (16). They are inspired by some facet inducing inequalities proposed in [29, Section 2] for the *single machine scheduling problem* and read as follows:

$$\sum_{m \in \mathscr{M}'} S_m(t_m + S_m) \geq f(\mathscr{M}') \quad \forall \mathscr{M}' \subseteq \mathscr{M}, |\mathscr{M}'| = 2, \tag{23}$$

where $f(\mathscr{M}') = \frac{1}{2}\left(\left(\sum_{m \in \mathscr{M}'} S_m\right)^2 + \sum_{m \in \mathscr{M}'} S_m^2\right)$.

Unfortunately, the non-trivial facet inducing inequalities proposed in [29, Section 3] for models with time indexed variables cannot be generalized for Model $\mathcal{P}_1$. Then, we did not test additional cuts to strengthen Model $\mathcal{P}_1$.

To assess the performance of the proposed algorithms when the cuts are considered, we run a set of experiments using a discretization step $\Delta t = 5$ minutes. We test all the four algorithms proposed in the paper, but the cuts do not provide any benefit for either Ω_1 or $\mathcal{E}_1$. For algorithms Ω_2 or $\mathcal{E}_2$, worsenings appear here more relevant than improvements.