The berth allocation problem in terminals with irregular layouts

Juan Francisco Correcher[a,*], Thomas Van den Bossche[b], Ramon Alvarez-Valdes[a], Greet Vanden Berghe[b]

[a] *Universitat de València, Dept. de Estadística i Investigació Operativa - Doctor Moliner 50, 46100 Burjassot, Spain*
[b] *KU Leuven, Department of Computer Science, CODeS & imec - Gebroeders De Smetstraat 1, 9000 Gent, Belgium*

Abstract

As international trade thrives, terminals attempt to obtain higher revenue while coping with an increased complexity with regard to terminal management operations. One of the most prevalent problems such terminals face is the Berth Allocation Problem (BAP), which concerns allocating vessels to a set of berths and time slots while simultaneously minimizing objectives such as total stay time or total assignment cost. Complex layouts of real terminals introduce spatial constraints which limit the mooring and departure of vessels. Although significant research has been conducted regarding the BAP, these real-world restrictions have not been taken into account in a general way. The present work proposes both a mixed integer linear programming formulation and a heuristic, which are capable of obtaining optimal or near-optimal solutions to this novel variant of the BAP. In order to assess the quality of the heuristic, which is being employed in a real tank terminal in Belgium, it is compared against the exact approach by way of randomly-generated instances and real-world benchmark sets derived from the tank terminal.

Keywords: Combinatorial optimization, port terminal, berth allocation, integer programming, iterated local search

1. Introduction

Ships have increasingly become an essential component within international trade. Every day cargo ships leave from and dock at port terminals, thereby providing consumers with a wide variety of goods. The development of naval engineering, the extension of container-based transport and the enhancement of bulk facilities now make it possible to carry huge quantities of resources from one side of the world to the other in very short time. This places pressure on port terminals, which compete against each other to offer the best service to customers primarily by seeking the shortest ship waiting times.

One of the most critical problems that terminals face when optimizing their operations is the Berth Allocation Problem (BAP). This problem concerns assigning a specific berth and a time slot to each

*Corresponding author.
Email addresses: juan.correcher@uv.es (Juan Francisco Correcher), thomas.vandenbossche@kuleuven.be (Thomas Van den Bossche), ramon.alvarez@uv.es (Ramon Alvarez-Valdes), greet.vandenberghe@cs.kuleuven.be (Greet Vanden Berghe)

vessel, while minimizing the total cost or service time. The berth characteristics, vessel dimensions and estimated arrival and departure times restrict the number of compatible berths for each vessel.

The BAP has been studied considering different characteristics and operational aspects. The surveys of seaside operations at container terminals published by Bierwirth and Meisel (2010, 2015) present the main variants addressed by academic researchers. In literature work, a classification is usually formed by considering temporal and spatial attributes. The temporal attribute describes the arrival process of vessels. According to Imai et al. (2001), *static* and *dynamic* temporal assumptions may restrict berthing times. The *static* variant occurs when ships arrive prior to berth allocation; it is assumed that vessels are already waiting in the port and can therefore berth immediately. By contrast, the *dynamic* case assumes that arrival times are provided for all vessels, meaning that mooring is only possible from a vessel's arrival. The existing temporal classification was extended by Bierwirth and Meisel (2015) with *cyclic* and *stochastic* because of their occurrence in recent papers. The cyclic variant assumes that vessels arrive at terminals according to a fixed liner schedule. Finally, stochastic arrival times are determined by either a certain random distribution or by real-life scenarios. Note that it is crucial to highlight how 'dynamic' in the context of established literature concerning the BAP has an entirely different meaning when compared against what it denotes throughout the operational research community more generally. Specifically, the dynamic BAP refers to the case where unique arrival times for vessels are known in advance rather than symbolising any change or response to new data.

With respect to the spatial attribute, the BAP may be classified as *discrete*, *continuous* or *hybrid* depending on the berthing layout. The *discrete* variant considers the quay as consisting of a finite set of berths or sections such that only one ship may be moored and served at each berth at any one time. In the *continuous* version, no quay partitioning exists and therefore vessels may berth at arbitrary positions within the boundaries of the quay. Finally, the *hybrid* variant considers the quay to be partitioned into a number of berths, with vessels capable of sharing a berth or occupying more than one under certain conditions.

Beyond these aspects, the geometrical disposition of the berths along the quay and the resulting operational implications have rarely been considered up to date. Important problems posed by irregular terminal layouts in which berths are adjacent or opposite to each other have thus remained unaddressed. This is often the case in terminals located at highly developed ports wherein artificial docks form indented quays. For example, in terminals such as the one depicted in Fig. 1, distances between berths together with their concurrent usage prevent vessels from docking at or departing from some berths subject to given rules. This also gives rise to special restrictions between berths which form virtual gates and thus may block the access to inner berths.

The present work addresses the discrete and dynamic BAP for the general case of terminals with irregular layouts involving adjacency, oppositional and blocking restrictions between berths. Permitted berthing operations are addressed by formulating them in a general fashion, thereby enabling both representing and solving the problem in wide variety of terminal layouts.

An original exact approach based on a Mixed Integer Linear Programming (MILP) model is proposed for solving small instances and a heuristic approach based on Iterated Local Search (ILS) and Ruin & Recreate strategies is proposed for larger ones. The heuristic approach is currently being used as a decision support tool for generating feasible schedules in a tank terminal with an irregular layout.

The remainder of this paper is structured as follows. Section 2 presents a literature review aimed towards highlighting the main academic contributions and related work. Next, the problem is formally introduced and defined (Section 3). In Section 4 an MILP model for the problem is proposed, while a heuristic approach is described throughout Section 5. Computational experiments and results are discussed in Section 6. Finally, Section 7 presents conclusions and future lines of research which may result from this work.

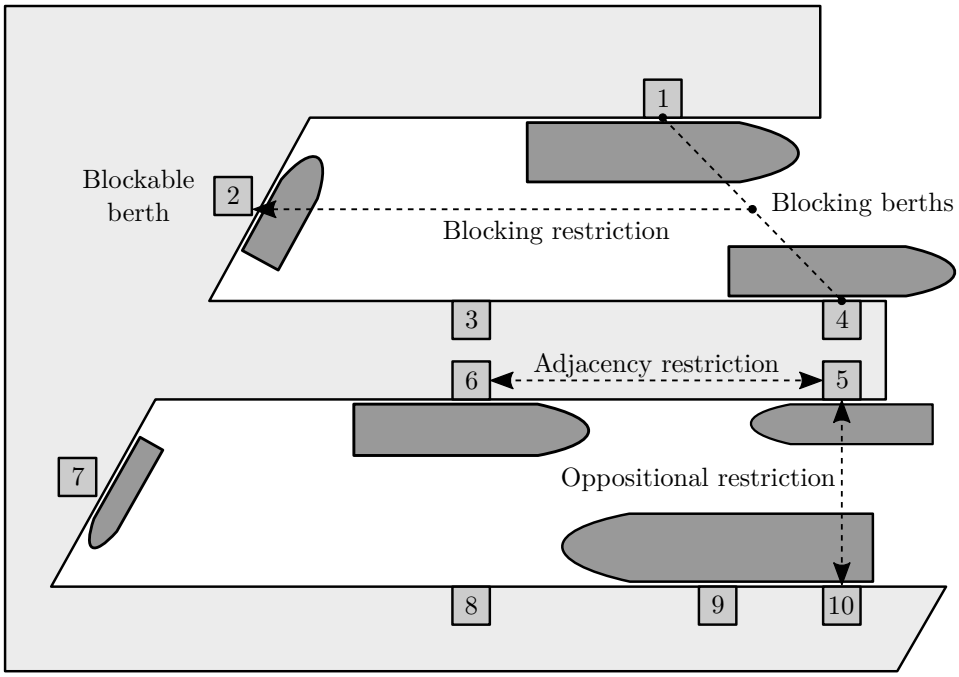


Figure 1: Terminal with irregular layout. Only some of the occurring spatial restrictions are shown.

2. Literature review

Berth allocation was first studied as an optimization problem in the late 1990s. The static variant of the discrete BAP was first formulated by Imai et al. (1997), while the dynamic variant was addressed by Imai et al. (2001) and Hansen and Oğuz (2003). Regarding the continuous version, the static and dynamic variants were first formulated by Li et al. (1998) and Lim (1998), respectively. Since then, the BAP has attracted ever-increasing attention in the combinatorial optimization community, and many new variants and characteristics of this problem have been addressed. The most recent reviews of the academic literature regarding seaside operations, including the BAP, were published by Bierwirth and Meisel (2010, 2015) and Carlo et al. (2015). In recent years, researchers have proposed new solution methods for addressing the BAP which respect more realistic conditions and characteristics (Table 1).

BAP Characteristics	References
Stochastic vessel arrival and handling times	Zhen, 2015; Ursavas and Zhu, 2016; Umang et al., 2017; Liu et al., 2017; Xiang et al., 2017
Fuel consumption and emissions	Hu et al., 2014; He, 2016; Venturini et al., 2017
Terminals with multiple continuous quays	Frojan et al., 2015; Ma et al., 2017
Bulk terminals	Bridi et al., 2016; Ernst et al., 2017; Pratap et al., 2017, de León et al., 2017
Water depth and tidal restrictions	Lalla-Ruiz et al., 2016; Qin et al., 2016; Ernst et al., 2017; Zhen et al., 2017
Quay crane assignment problem	Hsu, 2016; Lalla-Ruiz and Voß, 2016; He, 2016; Karam and Eltawil, 2016; Türkoğullari et al., 2016; Shang et al., 2016; Iris et al., 2017; Correcher and Alvarez-Valdes, 2017; Agra and Oliveira, 2018

Table 1: Recent research directions regarding the BAP.

Academic work has primarily focused on terminals consisting of linear quays where the berthing of vessels is only restricted by the berths' spatial compatibility and their temporal occupation. Adjacency, oppositional and blocking restrictions between berths have rarely been considered and those studies which did address these restrictions were limited to specific terminal layouts.

Consider, for example, Imai et al. (2007), who tackled a discrete BAP in which a dedicated indented berth is capable of serving either one mega-containership or up to four small vessels. This indented berth, located in the port of Amsterdam, has cranes located on both opposite quays, thereby enabling larger vessels to be served faster than at an ordinary berth. When there are no such large vessels in the port, terminal operators use this berth to serve smaller vessels. In the specific case addressed by

Imai et al. (2007), two small vessels are capable of docking either side of the berth if their combined lengths do not exceed the total length of the indented berth. This introduced the problem of accessing or leaving the inner sections of the berth when the outer sections were occupied by other vessels. The berth planner consequently had to take into account the constraints resulting from this situation, such as the need to postpone vessel departures from the inner sections until the outer ones had been vacated. Imai et al. (2007) first proposed an MILP for a hybrid dynamic BAP with berth-dependent vessel handling time and then extended this model to address the case of several indented berths.

In a later work, Imai et al. (2013) studied a variant which addresses the case when two parallel quays form a channel, which they refer to as *channel berths*. This case was modelled as an open-ended indented berth in which, again, only four small vessels or one mega-containership may be served. A genetic algorithm was proposed for solving this problem and several computational experiments were conducted to compare the service times of vessels in various scenarios with indented and channel berths. Another study in which channel berths play a significant role was presented by Zhen et al. (2017). In their work, the BAP is addressed together with quay crane assignment considering feasible mooring and departing time windows. These time windows depend on tidal cycles and the congestion of the port access channel. Unlike the work by Imai et al. (2007), only a single channel is taken into account, with no berths located along it. However, the maximum number of vessels allowed to sail through the channel towards or from a berthing area is time-dependent. Zhen et al. (2017) proposed an integer programming model and a column generation approach in order to solve this variant of the problem.

The aforementioned studies may be considered important regarding the present work, since they include some of the spatial restrictions that ports impose upon berth planning. Nevertheless, none of these studies addresses the general problem consisting of an arbitrary number of complex restrictions between berths regarding vessel mooring and departure. Consequently, one may conclude that the version of the BAP tackled throughout the present work, despite its real world relevance, has not yet been addressed by academia. The following section describes the BAP in terminals with irregular layouts and its particular characteristics in detail.

3. Problem description

3.1. Overview

The problem outlined in this study is a discrete dynamic BAP in which mooring at one or more sets of berths can be limited according to the relations between the berths or their particular characteristics (Figure 1). The objective is to minimize the total assignment cost, which is calculated by summing

the cost of waiting prior to berthing and the delay cost associated with each vessel when leaving. In terms of the scheme proposed by Bierwirth and Meisel (2010), this problem can be classified as $discr, draft \,|\, dyn \,|\, pos \,|\, \sum(w_1\, wait + w_2\, tard)$. This classification identifies a discrete dynamic BAP where the handling time of a vessel depends on its berthing position and the objective being the minimization of waiting time and tardiness against given due dates. There is however no means of adequately describing the various special relations present in the current problem by using the existing formulation, thereby necessitating the introduction of new categories.

It is possible to render a problem solution as a space-time diagram in which berths are represented on the vertical axis and time is represented on the horizontal axis (Figure 2). A berth's schedule is rendered as a horizontal line with the schedule of an individual vessel constituting a horizontal segment whose left-most point represents its *berthing time* and whose right-most point represents its *departure time*. A vessel may be forced to wait before berthing due to its occupation by another vessel or as a consequence of blocking restrictions. Similarly, a vessel's departure time may be delayed due to blocking restrictions.

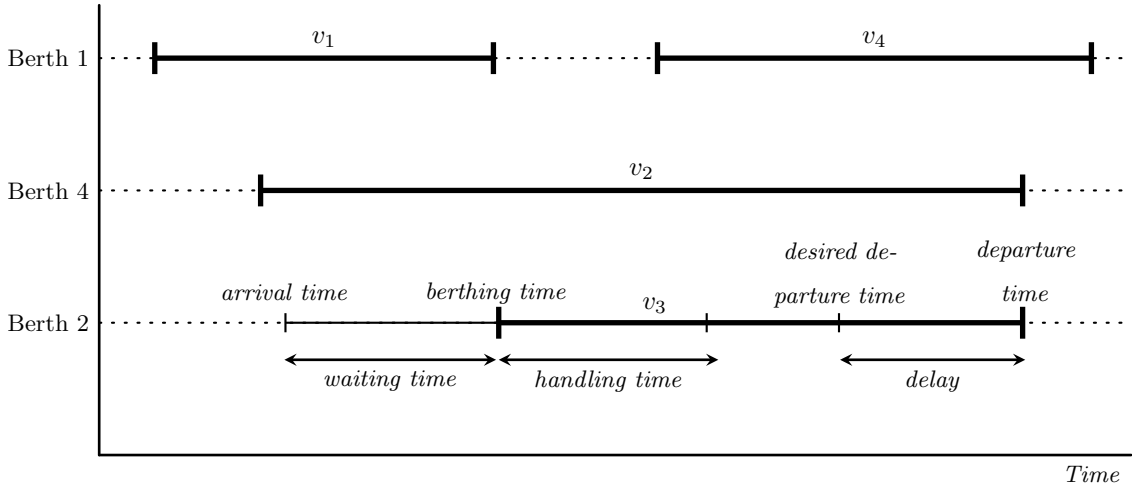


Figure 2: A berth schedule consisting of three berths, which correspond with those depicted in Figure 1. Four vessels are assigned to these berths.

3.2. Assumptions

The particular assumptions of this problem are as follows:

- A berth is considered as a specific point on the quay (referred to by number, as per Figure 1).
- A berth can accommodate at most one vessel at a time.

- A vessel is moored lengthwise parallel to the quay with its midpoint coinciding with its assigned berth.
- Once a vessel is moored, its position cannot be changed, nor may its handling be interrupted.
- A vessel moored at a berth may extend into an adjacent berth if its length exceeds the berth. Consequently, that berth will not be available for other vessels. For example, in Figure 1 this applies to Berths 9 and 10. The same may occur between oppositional berths, depending on the width of the vessels. This contrasts with most discrete BAPs in the literature, in which a single vessel may not occupy multiple berths.
- Vessel priorities may be reflected by setting coefficients in the objective function for the waiting and delay costs.
- The time for docking and undocking maneuvers is considered to be included in the vessel handling time.

The characteristics of the berths and the spatial relations between berths may give rise to various restrictions:

- Restrictions on mooring
  - *Availability restrictions* account for maintenance or unavailability of a certain berth, this restriction forbids vessels being moored at a given berth from the beginning of the planning interval until a given *release time* of the berth.
  - *Structural restrictions* forbid a given vessel from being moored at a given berth at any time. Structural restrictions often arise from physical berth limitations, such as the draft or the type of cargo handled. When a vessel satisfies all of a berth's structural relations, this berth is considered *compatible* with the vessel.
  - *Adjacency restrictions* prevent vessels from occupying a pair of adjacent berths concurrently if a given inter-ship clearance length is not satisfied. For example, in Figure 1 pairs of berths $\{3, 4\}$, $\{5, 6\}$, $\{8, 9\}$ and $\{9, 10\}$ are adjacent and are thus affected by this restriction type.
  - *Oppositional restrictions* prevent vessels from occupying a pair of oppositional berths concurrently if a given inter-ship clearance width is not satisfied. For example, in Figure 1 pairs of berths $\{1, 3\}$, $\{1, 4\}$, $\{5, 9\}$, $\{5, 10\}$, $\{6, 8\}$ and $\{6, 9\}$ are opposite to each other and are thus affected by this restriction type.

– *General mooring restrictions* prevent a given set of vessels from occupying a given set of berths concurrently. This restriction type represents decisions made by terminal operators seeking to address special situations, such as those involving berths that process hazardous cargo.

Figure 3 shows how these mooring restrictions are enforced in practice. In the case depicted, based on Figure 1, Berth 6 and Berth 8 are opposite to each other with vessels v_1 and v_2 already scheduled at Berth 6. Assuming that the widths of v_1 and v_2 are such that this oppositional restriction is operative, the time at which vessel v_3 can be scheduled at Berth 8 is restricted by the assignments at Berth 6. As mooring restrictions prevent overlap in time for vessels during their stay time, v_3 can only be scheduled when there is neither total nor partial concurrency with assignments at Berth 6. If we assume that v_3 arrives at some point during the handling of v_1 and the objective is to reduce both waiting and demurrage costs, v_3 must be scheduled immediately after the departure of vessel v_1.

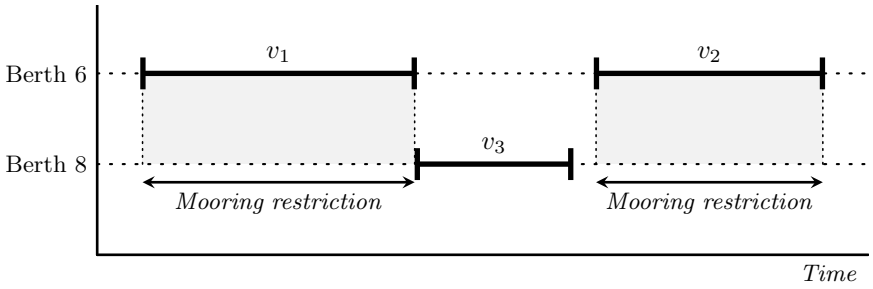


Figure 3: Example of a mooring restriction corresponding to the berthing situation depicted in Figure 1. Berths 6 and 8 are opposite to each other, with vessels v_1 and v_2 scheduled at Berth 6. The widths of v_1 and v_2 are such that the oppositional restriction is operative. Vessel v_3 cannot be moored at Berth 8 when vessels v_1 or v_2 are moored at Berth 6.

- Restrictions on both berthing and departure

  – *Blocking restrictions* prevent a given vessel from berthing at or departing from a berth, called a *blockable berth*, when a set of other berths, called *blocking berths*, are simultaneously occupied. A vessel at the blockable berth will be blocked by other vessels during the period in which those vessels coincide in time.

  For example, in Figure 1 Berths 1, 2, and 4, and the vessels moored at these berths are involved in this kind of relationship. Figure 4 depicts this blocking situation in time.

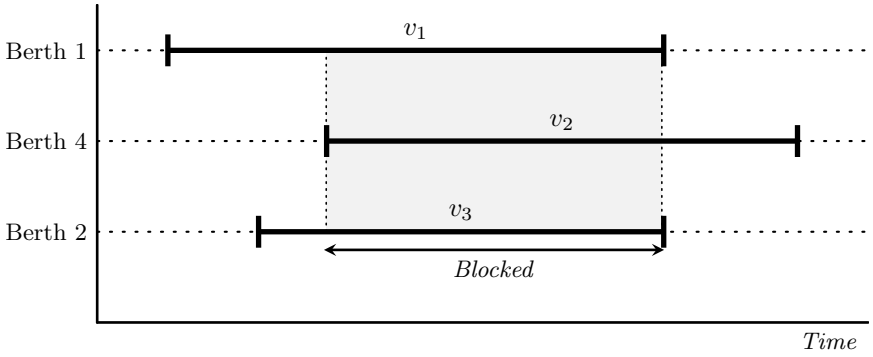


Figure 4: Example of a blocking restriction, corresponding to the berthing situation depicted in Figure 2. Berth 2 is *blockable* by Berths 1 and 4 for this combination of vessels. Vessel v_3 cannot berth at or depart from Berth 2 when vessel v_1 and v_2 are simultaneously moored.

3.3. Parameters

The following data concerning the terminal, vessels and restrictions define an instance of the BAP in terminals with irregular quays:

Berths:

- Set of berths: B. $N_B = |B|$
- Set of berths which are opposite to each other:
  $B^o = \{(f,k) \in B \times B \mid \text{berths } f \text{ and } k \text{ are opposite to each other}\}$
- Set of adjacent berths: $B^a = \{(f,k) \in B \times B \mid \text{berths } f \text{ and } k \text{ are adjacent}\}$
- For each berth $k \in B$, the release time is rel_k
- For each pair of berths $(f,k) \in B^o$, the oppositional distance is d^o_{fk} and the required inter-ship clearance c^o_{fk}
- For each pair of berths $(f,k) \in B^a$, the adjacency distance is d^a_{fk} and the required inter-ship clearance c^a_{fk}

Vessels:

- Set of vessels: V. $N_V = |V|$
- For each vessel $i \in V$, the following information is known:
  - Length: l_i
  - Beam (width): w_i
  - Expected arrival time: a_i
  - Desired departure time: s_i
  - Set of compatible berths: $B_i \subseteq B$
  - Estimated handling time at berth $k \in B_i$: h^k_i
  - Waiting cost per unit time for berthing after the expected arrival time: C^w_i
  - Delay cost per unit time after the desired departure time: C^d_i

Set of vessel and berth tuples representing mooring restriction decisions by the terminal operator:
$D = \{(b_1, v_1, \ldots, b_n, v_n) \mid v_1, \ldots, v_n \in V; (b_1, \ldots, b_n) \in B_{v_1} \times \cdots \times B_{v_n}$:
the operator forbids the simultaneous occupation of b_1 by v_1, b_2 by $v_2, \ldots, b_n$ by $v_n\}$
These relations are not necessarily symmetrical.

Set of vessel and berth tuples representing blocking restrictions:
$F = \{(b_1, v_1, \ldots, b_n, v_n) \mid v_1 \ldots v_n \in V; (b_1, \ldots, b_n) \in B_{v_1} \times \cdots \times B_{v_n}$:
b_1 can be blocked for v_1 by the simultaneous occupation of b_2 by $v_2, \ldots, b_n$ by $v_n\}$

4. A mixed integer linear model

The following MILP is proposed to formulate the BAP in terminals with irregular layouts previously described.

4.1. Precalculated sets

The following sets are defined, based on the input data, in order to avoid generating unnecessary variables and constraints:

Set of tuples of vessel and adjacent berths incompatible with respect to their length:
$A = \{(f, i, k, j) \mid i, j \in V; f \in B_i; k \in B_j; i < j; f \neq k; \frac{l_i}{2} + \frac{l_j}{2} + c^a_{fk} > d^a_{fk}\}$

Set of tuples of vessels and oppositional berths incompatible with respect to their width:
$O = \{(f, i, k, j) \mid i, j \in V; f \in B_i; k \in B_j; i < j; f \neq k; w_i + w_j + c^o_{fk} > d^o_{fk}\}$

Set of tuples employed to avoid pairs of vessels being served concurrently at the same berth: $C = \{(k, i, k, j) \mid i, j \in V; k \in B_i; k \in B_j; i \neq j\}$

Set of all the tuples representing the incompatible mooring of specific ships on specific berths: $I = C \cup A \cup O \cup D$

Set of pairs of vessels at which vessel i can be moored on a berth blockable by another berth which in turn can admit ship j:
$P = \{\, (i, j) \in V \times V \mid \exists (b_1, v_1, \ldots, b_n, v_n) \in F, \exists k \in \{2, \ldots, n\} : i = v_1, j = v_k \,\}$

4.2. Variables

A berth, a berthing time and a departure time are to be assigned to each calling vessel. Thus the decision variables are:

t_i = berthing time of vessel i

r_i = departure time of vessel i

u_i = delay incurred by vessel i relative to its desired departure time

$$m_i^k = \begin{cases} 1 & \text{if vessel } i \text{ is moored on berth } k \\ 0 & \text{otherwise} \end{cases}$$

$$\sigma_{ij} = \begin{cases} 1 & \text{if } r_i \leq t_j \mid i, j \in V \\ 0 & \text{otherwise} \end{cases}$$

$$\gamma_{ij} = \begin{cases} 1 & \text{if } t_i \leq t_j \mid i, j \in V \\ 0 & \text{otherwise} \end{cases}$$

$$\phi_{ij} = \begin{cases} 1 & \text{if } r_i \geq r_j \mid i, j \in V \\ 0 & \text{otherwise} \end{cases}$$

4.3. Objective and constraints

Objective function (1) minimizes the total assignment cost, which is the sum of the cost of waiting before berthing and the delay cost for each vessel. For simplicity, the objective is referred to as *cost* throughout the remainder of the paper. Constraints (2) ensure that the berthing of each vessel occurs upon or after its arrival, while Constraints (3) ensure it is assigned to a compatible berth. Vessels are only allowed of mooring after the berths' release times, which is enforced by Constraints (4). Constraints (5) ensure the departure of each vessel must occur at the time its handling finishes or later, while Constraints (6) define the delays. Constraints (7)–(9) define σ, γ and ϕ auxiliary variables enabling to describe precedences. Constraints (10) prevent overlap in time for vessels assigned to the same berth (set C), adjacent berths (set A), oppositional berths (set O) or berths and vessels subject to special considerations (set D). In particular, given a tuple that represents a mooring restriction, if vessels are assigned to their corresponding berths (right-hand side of the inequality), at least one pair of vessels cannot be served concurrently (left-hand side). The same applies to tuples representing blocking restrictions in Constraints (11) and (12), although in this case blocking is avoidable if the vessel assigned to blockable berth (v_1) is moored earlier (summation of γ) and departs later (summation of ϕ). Thus, by enforcing Constraint (8), the berthing time of v_1 must be less than or equal to the berthing time of at least one of the vessels being concurrently processed at the blocking berths and likewise its departure time must be greater than or equal to the departure time of at least one of those vessels, as per Constraint (9). Finally, Constraints (13)–(16) define the variable types. In this formulation, constant M constitutes an upper bound on the total time required to service all vessels calculated by means of the procedure proposed in the following section (Algorithm 2).

$$Min \sum_{i \in V} (C_i^w (t_i - a_i) + C_i^d u_i) \tag{1}$$

s. t.

$$t_i \geq a_i, \qquad \forall i \in V \tag{2}$$

$$\sum_{k \in B_i} m_i^k = 1, \qquad \forall i \in V \tag{3}$$

$$t_i \geq \sum_{k \in B_i} m_i^k rel_k, \qquad \forall i \in V \tag{4}$$

$$r_i \geq t_i + \sum_{k \in B_i} m_i^k h_i^k, \qquad \forall i \in V \tag{5}$$

$$u_i \geq r_i - s_i, \qquad \forall i \in V \tag{6}$$

$$t_j \geq r_i - M(1 - \sigma_{ij}), \qquad \forall i, j \in V, i \neq j \tag{7}$$

$$t_i \leq t_j + M(1 - \gamma_{ij}), \qquad \forall (i,j) \in P \tag{8}$$

$$r_i \geq r_j - M(1 - \phi_{ij}), \qquad \forall (i,j) \in P \tag{9}$$

$$\sum_{i=1}^{n} \sum_{j=1, i \neq j}^{n} \sigma_{v_i v_j} \geq \sum_{i=1}^{n} m_{v_i}^{b_i} - n + 1, \qquad \forall (b_1, v_1, \ldots, b_n, v_n) \in I \tag{10}$$

$$\sum_{j=2}^{n} \gamma_{v_1 v_j} + \sum_{i=1}^{n} \sum_{j=1, i \neq j}^{n} \sigma_{v_i v_j} \geq \sum_{i=1}^{n} m_{v_i}^{b_i} - n + 1, \qquad \forall (b_1, v_1, \ldots, b_n, v_n) \in F \tag{11}$$

$$\sum_{j=2}^{n} \phi_{v_1 v_j} + \sum_{i=1}^{n} \sum_{j=1, i \neq j}^{n} \sigma_{v_i v_j} \geq \sum_{i=1}^{n} m_{v_i}^{b_i} - n + 1, \qquad \forall (b_1, v_1, \ldots, b_n, v_n) \in F \tag{12}$$

$$\sigma_{ij} \in \{0, 1\}, \qquad \forall i, j \in V, i \neq j \tag{13}$$

$$\gamma_{ij}, \phi_{ij} \in \{0, 1\}, \qquad \forall (i,j) \in P \tag{14}$$

$$m_i^k \in \{0, 1\}, \qquad \forall i \in V, \forall k \in B_i \tag{15}$$

$$t_i, r_i, u_i \geq 0, \qquad \forall i \in V \tag{16}$$

Furthermore, following valid inequalities are also included:

$$\gamma_{ij} \geq \sigma_{ij}, \quad \forall (i,j) \in P \tag{17}$$

$$\phi_{ij} \geq \sigma_{ji}, \quad \forall (i,j) \in P \tag{18}$$

5. Heuristic approach

All the aforementioned restrictions between berths result in a highly constrained BAP which may be difficult to solve in time-constrained circumstances. Indeed, terminals often have a limited amount of time available for computing feasible operational schedules and, therefore, the availability of a heuristic

which is capable of quickly providing good quality solutions is of paramount importance. This section presents an Iterated Local Search heuristic for this novel BAP.

5.1. ILS

The ILS procedure, the pseudocode of which is provided in Algorithm 1, operates at the highest level. The algorithm begins by obtaining an initial solution s_0 using a constructive heuristic. A local search is conducted on s_0 until a predefined termination criterion (number of iterations) is met. The obtained solution s^* is subsequently perturbed into $s^{'}$. Afterwards, the local search procedure is again applied to solution $s^{'}$ and candidate solution $s^{*'}$ is accepted based on the acceptance criterion. The complete procedure is run until ILS' stopping criterion is reached, which is a computation time limit in seconds.

Algorithm 1 ILS

$s_0 \leftarrow$ Constructive heuristic
$s^* \leftarrow$ LocalSearch(s_0)
while ILS termination condition unsatisfied **do**
  $s' \leftarrow$ Perturbation(s*)
  $s^{*'} \leftarrow$ LocalSearch(s')
  $s^* \leftarrow$ Acceptance criterion(s*, s*')
end while
Output: s*

5.1.1. Constructive heuristic

An initial feasible solution is obtained as follows. First, all vessels are sorted by increasing arrival time and sequentially assigned to its most efficient berth after the previous vessel's departure time, taking into account berth release times (Algorithm 2). Consequently, no pair of vessels is served concurrently at the same berth and all restrictions concerning mooring (availability, structural, adjacency, oppositional and general mooring restrictions), berthing and departure (blocking restrictions) are trivially satisfied. Due to active restrictions, a vessel's berthing and/or departure time may be delayed, resulting in a higher assignment cost. The cost of this initial solution may be employed as an upper bound to speed up branch-and-bound based strategies. Algorithm 2 also enables the calculation of a precise value M for each instance. M corresponds to the maximum departure time among the vessels in the solution constructed.

Algorithm 2 Construction of an initial feasible solution

Input: instance data

$M \leftarrow 0$ ▷ Maximum departure time
Sort V by increasing arrival time
for $i \in V$, according to the ordering **do**
  Sort B_i lexicographically by 1) increasing release time and 2) increasing $h_i^{k \in B_i}$
  $b \leftarrow$ first element in B_i ▷ Auxiliary variable
  $m_i^b \leftarrow 1$
  $t_i \leftarrow \max\{rel_b, a_i, M\}$
  $r_i \leftarrow t_i + h_i^b$
  $M \leftarrow r_i$
end for

Output: feasible solution

5.2. Local search neighborhoods

Three relatively straightforward *Shift*, *Swap* and *Ruin & Recreate* neighborhood operators are proposed for the local search phase. Each neighborhood considers both the removal and the reinsertion of one or more vessels. At each local search iteration, one of the proposed operators is selected according an adaptive selection method. All details required for enabling implementation are provided throughout the following subsections.

5.2.1. Shift

One berth $b \in B$ is randomly selected, after which a random vessel i assigned to b is removed from the solution. A feasible compatible berth b' is randomly selected from $B_i \setminus b$ and then vessel i is reinserted into the lowest-cost position.

For example, the berthing schedule of two berths, Berth 1 and Berth 2, is shown in Figure 5. One berth is randomly selected (Berth 1) and then one vessel in its schedule is randomly selected and removed (Vessel 2). Next, Berth 2 is randomly selected from the set of compatible berths of Vessel 2. Finally, Vessel 2 is reinserted at the earliest feasible position in Berth 2. Note that vessels may not be inserted at times prior to their arrival at the terminal. Therefore, Vessel 2 may only be inserted after the completion time of Vessel 5.

5.2.2. Swap

Similar to the shift neighborhood, one berth and one of its allocated vessels are randomly selected. Next, a different berth and corresponding vessel are selected such that both vessels' berths are compatible. Note that employing this neighborhood does not guarantee berthing at the swapped vessel's

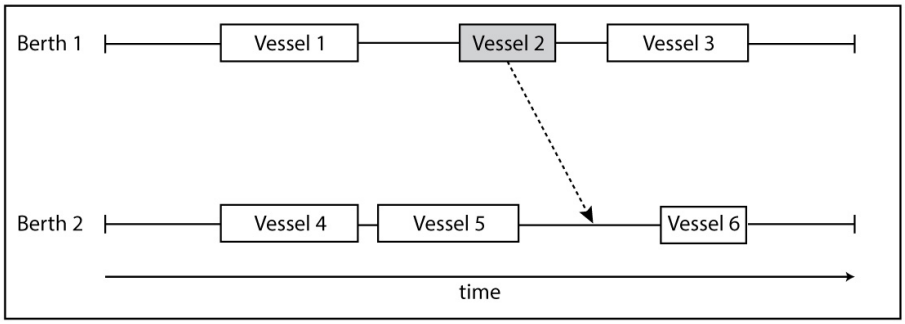


Figure 5: Shift neighborhood.

berthing time. The earliest feasible berthing time must be calculated based on the vessel's arrival time in addition to its mooring and blocking restrictions.

The swap operation is illustrated in Figure 6, where Vessel 3 and Vessel 5 are removed from their original berths and reassigned to each other's berth. Although the handling times are similar for both vessels and it appears possible to fit the assignment of Vessel 3 between those of Vessels 4 and 6, Vessel 3's arrival time prevents its insertion into the original position of Vessel 5. Therefore, it will be inserted after the assignment of Vessel 6.

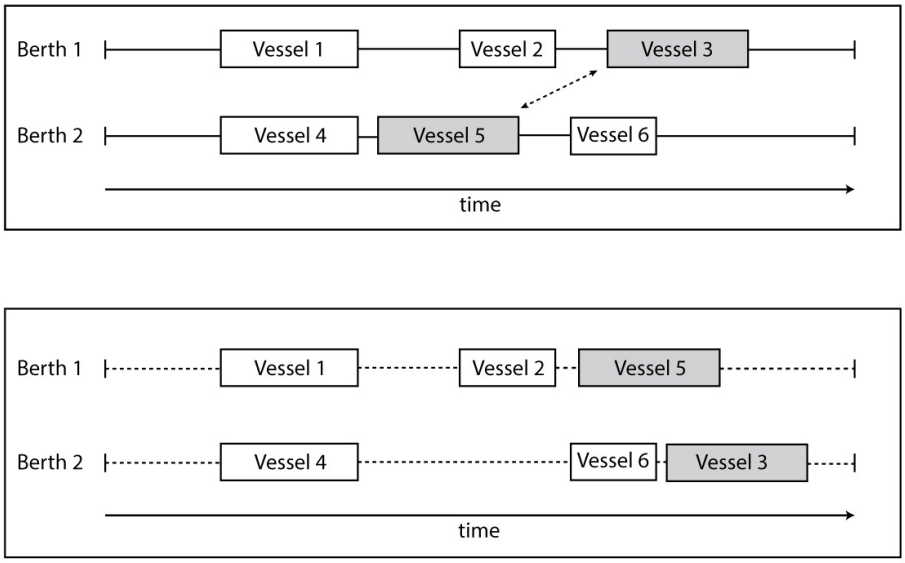


Figure 6: Swap neighborhood.

5.2.3. Ruin & Recreate

Ruin phase. Multiple vessels are removed from one or more berth schedules. Ruin-factor R determines the total number of assignments to be removed. This parameter R is proportional to the instance size, with its value set before employing the neighborhood. For ease of notation, auxiliary variable R' is initially set as $R' = R$ and updated throughout the employment of this neighborhood. The ruin phase is implemented by iteratively executing the following step until R vessels have been removed: randomly select a berth and randomly select $r = U[1, R']$ vessels from its schedule. U denotes a random number selected from a uniform distribution within the specified interval. The step is repeated while the value of R' is greater than zero. At each step R' is updated by subtracting the number of vessels removed ($R' = R' - r$). Note that r is always limited to the maximum number of vessels currently assigned to the considered berth.

In the given example (Figure 7), the ruin-factor $R = 4$ implies that a total of four vessels are removed from all berths' schedules. The adaptive ruin phase operates as follows:

1. Berth 1 is randomly selected and the number of vessels to remove at this iteration is calculated as $r = U[1, R']$, with $R' = R = 4$. By way of example it is assumed that r takes a value of 1, meaning only one vessel (Vessel 3) is randomly selected and removed from the schedule of Berth 1 (Figure 7a). Since one vessel is removed from the schedule, the value 1 is subtracted from R', and thus R' becomes 3.

2. As R' is still non-zero, another berth is randomly selected (Berth 2) and r is recalculated as $r = U[1, 3]$ (assume $r = 1$). Consequently, a single vessel is removed from the schedule of Berth 2 and R' is updated ($R' = 2$).

3. With the value of R' being 2, another berth, Berth 3, is selected and, assuming $r = 2$, two vessels are removed from its schedule. The ruin phase terminates as now R' equals zero and a total of four vessels have been removed from their corresponding schedules (Figure 7b).

Recreate phase. Vessels removed in the ruin phase are reinserted into the schedules of one of their compatible berths. The method *Greedy insertion with blinks* (Christiaens and Vanden Berghe, 2016) was employed here, so that instead of always reinserting vessels at their best position, a slightly worse position may be selected. Feasible insertion positions are iterated over in cost-ascending order and each one is only selected according to a given probability of $1 - \beta$, considering β as a parameter. In cases where the insertion position is not selected, it is skipped as though the algorithm 'blinks'. If, for example, β takes a value of 0, this implies that vessels are always inserted at their best position. In the

present work however, the primary bottleneck is the costly feasibility check for insertions due to various dependency constraints which must be satisfied. Given that for example $\beta = 0.01$, the probability of selecting the best insertion position is 0.99, while the second best is selected with a probability of 0.0099 $(0.01 \cdot 0.99)$, the third best with one of 0.000099 $(0.01 \cdot 0.01 \cdot 0.99)$ and so forth.

Therefore, the algorithm proposed by Christiaens and Vanden Berghe (2016) is adjusted such that the k-th best insertion position, denoted as rank k, is derived in advance using β:

$$k = \log_\beta \left(\frac{U[0,1] \cdot \beta}{1 - \beta} \right) \tag{19}$$

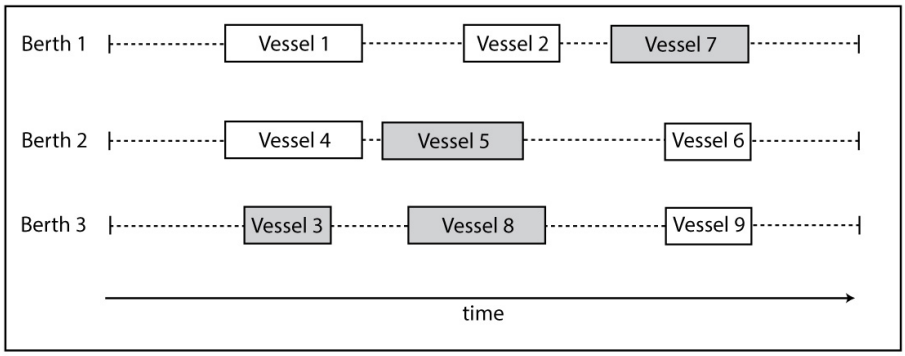
Consequently, for each vessel at most k insertion positions must be evaluated at any berth. Algorithm 3 formally describes the recreate phase. The set of removed vessels is first sorted by increasing arrival time. Next, rank k is determined based on Equation (19). For each compatible berth, the feasible insertion positions in its schedule are iterated over by increasing berthing time (trivially by increasing assignment cost) until the k-th feasible option is identified. These positions are subsequently added to a list of feasible insertion positions denoted by S. Once this process has finished, list S is sorted by increasing assignment cost. Finally, vessel i is placed at the k-th insertion position in this list. If the number of feasible insertion positions in S is less than k, vessel i is placed at the last insertion position in the list. This neighborhood is employed until all vessels are reinserted into one of their compatible berths.

Algorithm 3 Recreate phase

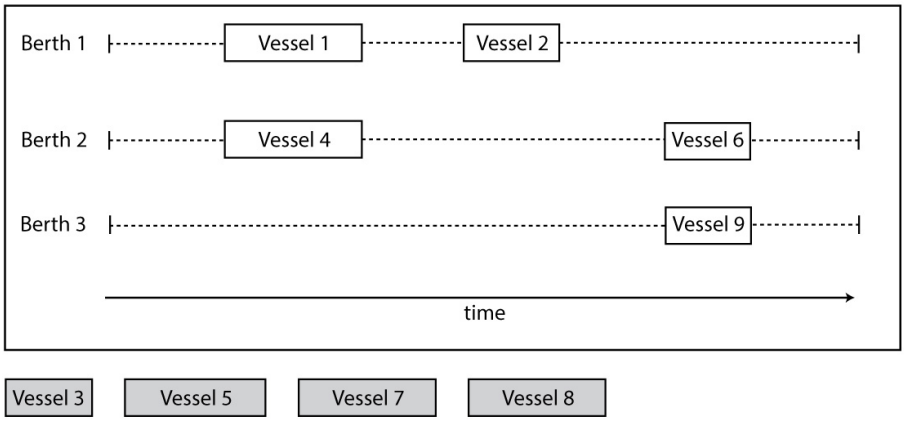
```
V ← removed vessels from the ruin phase
Sort V by increasing arrival time
for i ∈ V do
    S ← ∅                                   ▷ List of feasible insertion positions for vessel i
    Calculate rank k                        ▷ Equation (19)
    for b ∈ B_i do
        S ← S ∪ {first k feasible positions of vessel i in the schedule of berth b}
    end for
    Sort S by increasing vessel assignment cost
    InsertVessel(i, S[k])                   ▷ Insert i at the k-th insertion position
end for
```

5.2.4. Adaptive neighborhood selection

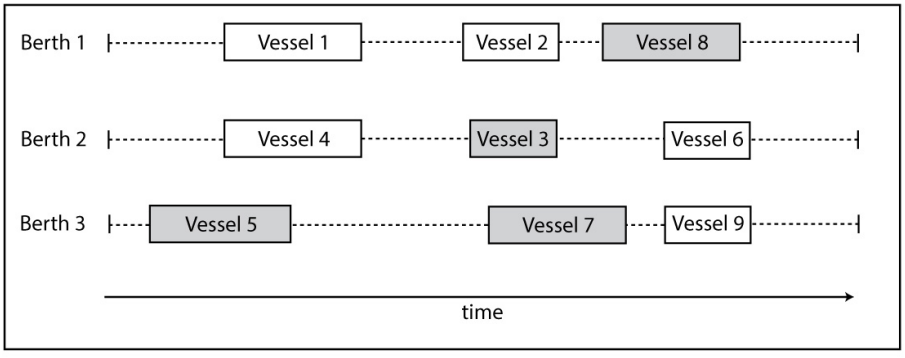
The neighborhood selection mechanism introduced by Røpke and Pisinger (2006) is employed within the proposed heuristic. This adaptive selection mechanism rewards neighborhoods which have contributed towards obtaining better solutions while maintaining the possibility of diversification when



(a) Random selection of vessels to remove, based on the ruin-factor R.



(b) A number of vessels, equal to R, is removed from the berths' schedules.



(c) All removed vessels are randomly reinserted into the set of all berthing schedules.

Figure 7: Ruin & Recreate neighborhood.

historically poorer-performing neighborhoods are selected.

Assuming there are n neighborhoods, each neighborhood $i \in \{1, 2, ..., n\}$ is assigned a score π_i which reflects its recent performance. This score is updated each time the neighborhood is employed by increasing its value by either ρ_1, ρ_2 or ρ_3, which correspond to different improvement situations. ρ_1 is related to the case in which employing the neighborhood results in a new global best solution. In the case of ρ_2, a solution is found which was not accepted before and has a lower cost than the current solution's. Finally, ρ_3 corresponds to the case in which a worse solution is found which was not accepted before.

The probability of selecting a neighborhood in the current iteration of the local search depends upon a weight which is adjusted during the search. In particular, the local search process is divided into segments of 100 iterations each and at the end of each segment the weight of each neighborhood is updated by taking into account its score. w_{is} is identified as the weight of neighborhood i during segment s, so the probability of selecting neighborhood j at each iteration in segment s is:

$$\frac{w_{js}}{\sum_{i=1}^{n} w_{is}} \tag{20}$$

In the first segment, the weights of all the neighborhoods are set equal to 1. For segment $s+1$, the weight of each neighborhood i is adjusted as follows:

$$w_{i,s+1} = w_{is}(1 - f) + f\frac{\pi_{is}}{\theta_{is}} \tag{21}$$

In Equation (21), π_{is} is the score obtained by neighborhood i in segment s, whereas θ_{is} corresponds to the number of times i was employed in that segment. Reaction factor f manages the influence of the last score in the new weight. A reaction factor value of 1 means that the weight only depends on the score calculated in the previous segment, while a value of 0 means that the weight does not depend on the score at all and thus is kept fixed to its initial value.

The weight adjustment algorithm is controlled by four parameters (ρ_1, ρ_2, ρ_3, f) with the values proposed by Røpke and Pisinger (2006) ($\rho_1 = 33$, $\rho_2 = 9$, $\rho_3 = 13$, $f = 0.1$) being employed in present work.

5.2.5. Perturbation

To prevent the local search from converging to a local optimum, a perturbation is performed on its current best solution s^*. The perturbation employs an adjusted Ruin & Recreate neighborhood in which the ruin factor is set to twice the value of R.

5.2.6. Acceptance and stopping criteria

A solution obtained during local search replaces the current solution only if it meets the acceptance criterion. The present approach applies Late Acceptance Hill Climbing (Burke and Bykov, 2017), which only accepts a solution if it is not worse than the solution evaluated L iterations ago.

The local search terminates after $stop_{LS}$ iterations without improvement. Afterwards, the current solution is perturbed and local search continues from the resulting solution. The complete ILS procedure is stopped when a computation time limit of $stop_{ILS}$ seconds is reached.

The parameter values applied for selection, acceptance and stopping criteria are discussed in Section 6.4.

6. Computational experiments

The quality of both the MILP model and the heuristic approach was assessed by conducting experiments on a variety of instances. Three different instance sets were generated in order to cover a wide range of different circumstances and terminal layouts.

First, historical data which provides berthing schedules as they were conducted over the course of a full year was collected from a tank terminal located in the port of Antwerp. Processing this data results in two datasets: *Realistic* and *Realistic-week*. The second dataset limits the scheduling horizon to one week, making it possible to evaluate the performance of both methods under increasing vessel congestion conditions.

A third set was generated to assess the time required to optimally solve the problem with respect to various instance factors. All benchmark instances have been made publicly available at `http://gent.cs.kuleuven.be/bap_irregular_layouts`.

6.1. Realistic instance set

The characteristics of this instance set correspond with the situation observed at the tank terminal participating in this study. The realistic set consists of 100 instances, 10 for each number of vessels considered: $N_V \in \{10, 20, \ldots, 100\}$. Each instance consists of 11 berths. Every berth $k \in B$ is available from the beginning of the planning period. There are ten pairs of oppositional berths and five pairs of adjacent berths. In all cases, the clearances required are $c_k^a = 10$ and $c_k^o = 30$ metres.

The generation of vessels in each instance occurs as follows. First, a single day is randomly selected from the historical information for the entire year. The first N_V records from that day are selected, representing berthing information of N_V vessels in increasing order of arrival at the terminal. As this information only contains the specific berth at which vessels were moored at, compatible berths for

each vessel must be pre-calculated according to the characteristics of both the vessel and each berth at the terminal, such as length, beam (width), draft, deadweight tonnage and type of vessel.

The data provided by the tank terminal only contains detailed information regarding vessel dimensions for a limited number of vessels admitted. Therefore, actual ship dimensions were obtained from a vessel tracking website[1]. All generated vessels for the realistic set are randomly selected from vessels tracked, after which a random deviation of $U[-20, 20]$ units is applied to its relevant characteristics, namely length (metres), beam (metres), draft (metres) and deadweight tonnage (tonnes).

Handling times in compatible berths are also subjected to similar deviations in the interval $U[-20, 20]$. As the objective of the terminal operators is to minimize the stay of vessels at the terminal, the cost coefficients are set to $C_i^w = 0$ and $C_i^d = 1$ for all vessels, while their desired departure times are equal to their arrival times. The set of tuples representing mooring restrictions (Set D) is generated according to a set of rules provided by the terminal, thereby complying with their safety regulations. Rules are generated with respect to all vessel and berth characteristics known in advance, such as length or deadweight tonnage. An example of such a rule is:

> *Any pair of vessels i and j cannot simultaneously be moored at Berth 6 and Berth 7, respectively, if $l_i \geq 135\,\mathrm{m}$ and $l_j \geq 150\,\mathrm{m}$.*

Similarly, the set of tuples representing blocking restrictions (F) was generated by means of rules such as:

> *Any vessel i cannot berth at or depart from Berth 2 if $l_i \geq 110\,\mathrm{m}$ and when Berth 3 or Berth 4 are occupied.*

Sixteen real-world rules are included in the instances, of which 12 relate to mooring restrictions and the remaining 4 concern blocking constraints.

6.2. Realistic-week instance set

An additional set similar to the realistic instance set, namely *Realistic-week*, is generated such that the vessels' arrival times lie within a time horizon of one week. This set represents various vessel congestion conditions. All the instances from *Realistic* set were copied and those in which the arrival time of a vessel exceeded the planning horizon of one week (168 hours) were transformed. In those instances, the arrival time of each vessel i was adjusted, such that $new\_a_i = prev\_a_i \cdot prop$, where

[1] https://www.vesselfinder.com

$prop = 168/\max_{j \in V}(prev\_a_j)$ is rounded to the nearest integer. The desired departure time was also adjusted such that: $new\_s_i = prev\_s_i \cdot prop$, again rounded to the nearest integer. A total of 33 instances were modified this way: two instances with 60 vessels, four with 70 vessels, eight with 80 vessels, nine with 90 vessels and all ten instances containing 100 vessels.

6.3. Diverse instance set

This set was generated to assess both the computational time requirements when solving the BAP optimally and the performance of the MILP and heuristic approaches in a more general sense by way of different terminal layouts and number of vessels. Let $Perc_{moor}$ be the total number of adjacent and oppositional restrictions divided by the number of berths, $Perc_{block}$ the number of blocking restrictions divided by the total number of berths and $Perc_{comp}$ the maximum number of compatible berths for each vessel divided by the number of berths. Ten random instances were generated for each combination of the following parameters:

- $N_V \in \{10, 20, 30, 40, 50, 60, 70\}$,
- $N_B \in \{8, 16, 24\}$,
- $Perc_{comp} \in \{0.25, 0.50, 0.75\}$,
- $Perc_{moor} \in \{0, 0.5, 1\}$,
- $Perc_{block} \in \{0, 0.25, 0.50\}$.

The release times of all berths in all instances are equal to 0 in all instances. $N_B \cdot 0.5 Perc_{moor}$ random pairs of adjacent berths, $N_B \cdot 0.5 Perc_{moor}$ random pairs of oppositional berths and $N_B \cdot Perc_{block}$ random blocking restrictions were generated which prevent opposite berths in each pair from being adjacent.

The distances between adjacent berths were randomly generated in metres in the interval $U[50, 400]$. This selection was performed while preventing the formation of cyclical relations and considering chains of adjacency relations being at most three berths in length. Likewise, the distances between opposite berths were obtained from $U[50, 200]$. All adjacent berths require a clearance of 10 metres, while oppositional berths require a clearance of 30 metres.

The set of blocking berths corresponding to each blocking restriction was also generated randomly, obtaining a number of berths in the interval $U[1, 4]$. The berth blockable by those berths was also selected randomly while preventing mutual blocking restrictions, namely those in which a blockable berth is also simultaneously a blocking berth. All possible combinations of vessels were considered when deriving blocking restrictions.

Each vessel i was generated randomly, with its length in metres obtained from $U[30, 430]$ and its beam rounded to the nearest integer resulting from: $l_i^{2/3} + rand$, with $rand$ obtained from $U[0, 5]$. Each

vessel has a number of compatible berths obtained from $U[2, N_B \cdot Perc_{comp}]$. Finally, the arrival time in hours was obtained from $[0, 168]$; the processing time in each compatible berth from $U[5, 20]$ hours; and the desired departure time is equal to $a_i + 1.25 \cdot \min_{k \in B_i}(h_i^k)$. The cost coefficients are equal for all the vessels: $C_i^w = 1, C_i^d = 2$, meaning no priorities are enforced. A total of 5670 instances were generated in this way.

6.4. Parameter settings

The ILS procedure requires various parameters to be set for the acceptance and stopping criteria as well as for the Ruin & Recreate neighborhood.

The automatic configuration package iRace was employed to determine the most appropriate combination of parameter settings. Table 2 presents these values as indicated by iRace on a budget of 10 000 runs for the ILS procedure. The value range considered for each parameter is also reported.

Parameter	Search interval	Step size	Value obtained
R	$[0.001 \cdot N_V, N_V]$	0.001	$0.544 \cdot N_V$
$stop_{LS}$	[500 000, 1 000 000]	1	964 621
L	$[1, 30]$	1	20
β	$[0.001, 1]$	0.001	0.047

Table 2: iRace parameter settings and resulting values.

The parameter $stop_{ILS}$ was not tuned, given that good solutions were reported when $stop_{ILS}$ was set to $3 \cdot N_V$ seconds, which relates computation time to the problem's size. This runtime limit respects the brief time windows within which terminal operators must allocate berths.

6.5. Computational results

The MILP model was implemented in CPLEX 12.6 using Java 8, while the heuristic was coded in Java 8. Experiments were conducted on an Intel(R) Core(TM) i7-2600 CPU, 3.40 gigahertz computer with 31.4 gibibytes of RAM running Ubuntu Linux 12.04 LTS.

Experiments regarding the MILP model were run with a time limit of one hour for each instance, considering maximum departure time M and initial upper bound obtained by means of Algorithm 2. CPLEX was configured with emphasis on optimality (parameter *MIP emphasis* = 2). Five runs of the heuristic were performed for each instance with different random seeds in order to assess its average performance. Tables 3–8 show the results per instance set.

6.5.1. Realistic set

Table 3 presents the MILP results aggregated over all instances with the same number of vessels: number of instances for which a feasible solution was obtained, number of instances optimally solved, average and maximum optimality gaps reported by CPLEX, average cost and average computation time in seconds (including the time required to construct the MILP). Gaps are calculated as follows: $gap = (UB - LB)/UB$, where UB and LB are the best upper bound and lower bound achieved, respectively. Regarding the heuristic approach, the average cost over the five runs (*cost*) and the average time elapsed until the reported solution was found (*time*) were calculated for each instance. The optimality gap for the heuristic was calculated considering the best lower bound achieved by CPLEX as follows: $gap_H = (cost - LB)/cost$. The relative difference (*dev*) between the average cost and the MILP cost ($bestMILP$) was calculated as: $dev = (cost - bestMILP)/bestMILP$. The table details the number of instances solved optimally in at least one of the five runs, the average gap_H, the average and maximum *dev* in percentage, the average *cost* and finally the average *time* in seconds.

	MILP						Heuristic					
Vessels	Feasible	Optim.	Avg. gap (%)	Max. gap (%)	Avg. cost	Avg. time (s)	Optim.	Avg. gap_H (%)	Avg. dev (%)	Max. dev (%)	Avg. cost	Avg. time (s)
10	10	10	0.0	0.0	113.1	0.16	10	0.0	0.0	0.0	113.1	0.37
20	10	10	0.0	0.0	211.1	0.48	10	0.0	0.0	0.3	211.2	2.93
30	10	9	1.2	11.6	356.5	468.64	8	1.9	0.8	5.1	360.9	9.05
40	10	9	3.2	32.0	515.3	948.92	7	4.0	0.7	8.7	517.9	14.04
50	10	4	4.8	14.9	629.4	2179.36	4	5.9	1.3	6.7	639.7	48.22
60	10	3	7.3	19.6	796.6	2700.34	2	8.3	1.3	9.9	808.3	56.50
70	10	4	13.5	49.2	1062.5	2657.75	2	12.9	-0.9	5.7	1044.8	94.21
80	9	1	17.2	44.2	1234.3	3343.22	1	15.0	-3.0	9.5	1170.7	105.20
90	10	1	17.2	63.5	1474.2	3280.85	1	14.9	-3.2	3.6	1403.7	163.76
100	10	0	23.1	58.9	1688.5	3600.59	0	14.7	-11.3	0.3	1423.8	166.66

Table 3: Results from instance set *Realistic*, aggregated per 10 instances considering equal numbers of vessels.

The results from the *Realistic* set (Table 3) indicate that the MILP optimally solves most instances with up to 40 vessels and some additional instances containing up to 70 vessels. Larger instances are occasionally solved optimally, however in cases where optimality is not proven the average and maximum gaps are rather significant. Nevertheless, the MILP obtained feasible solutions for all instances except one. Instances with fewer than 30 vessels are solved in under a second, whereas larger instances require increasing computational effort, as expected.

Despite the heuristic's computation time limitation of three times the number of vessels in seconds, optimal or near-optimal solutions were obtained for all instances. The short computation times required by the heuristic indicate it converges fast and sufficiently explores the solution space. For large instances (70 vessels and above) a negative average deviation is reported, indicating that on average the heuristic

outperformed the MILP. This demonstrates the relevance of employing a heuristic approach in real-world scenarios, where good quality solutions are required within short computation times.

6.5.2. Realistic-week set

Table 4 provides the results obtained for the *Realistic-week* instances. The congestion resulting from the arrival of more than 60 vessels within a time horizon of one week heavily impacts the performance of the MILP approach. The average gap reported by CPLEX increases with the number of vessels. Indeed, for six instances with 90 and 100 vessels no feasible solution was obtained. In contrast to the MILP, the heuristic achieved optimal or near-optimal solutions for medium sized instances in far shorter run times and obtained better solutions for instances where the number of vessels ranges from 70 to 100. The heuristic's gap, measuring the distance from the heuristic solution to the CPLEX lower bound, increases with instance size, so in the case of large instances, although the heuristic solutions are better than those achieved by CPLEX, it is difficult to assess their quality.

	MILP						Heuristic					
Vessels	Feasible	Optim.	Avg. gap (%)	Max. gap (%)	Avg. cost	Avg. time (s)	Optim.	Avg. gap_H (%)	Avg. dev (%)	Max. dev (%)	Avg. cost	Avg. time (s)
10	10	10	0.0	0.0	113.1	0.16	10	0.0	0.0	0.0	113.1	0.4
20	10	10	0.0	0.0	211.1	0.48	10	0.0	0.0	0.3	211.2	2.9
30	10	9	1.2	11.6	356.5	468.64	8	1.9	0.8	5.1	360.9	9.1
40	10	9	3.2	32.0	515.3	948.92	7	4.0	0.7	8.7	517.9	14.0
50	10	4	4.8	14.9	629.4	2179.36	4	5.9	1.3	6.7	639.7	48.2
60	10	2	7.3	19.6	797.6	2939.91	1	8.3	1.2	9.9	809.0	56.7
70	10	1	14.6	49.2	1078.5	3390.83	1	14.2	-0.7	5.7	1062.1	108.4
80	10	0	25.8	59.6	1431.1	3600.33	0	23.1	-3.4	15.6	1347.4	157.6
90	8	0	37.0	64.0	1952.0	3600.53	0	32.7	-7.0	1.2	1791.2	186.0
100	6	0	59.4	76.7	3289.5	3600.44	0	47.3	-23.6	2.0	2438.7	217.4

Table 4: Results from instance set *Realistic-week*, aggregated per 10 instances considering equal numbers of vessels.

6.5.3. Diverse set

An exhaustive experiment was conducted on the *Diverse* set to investigate the required computation time in relation to the number of vessels, the number of berths, the maximum percentage of compatible berths for each vessel, the percentage of mooring restrictions and the percentage of blocking restrictions. The MILP model was run with a time limit of 600 seconds per instance.

Table 5 details the number of instances optimally solved, grouped by number of vessels and number of berths on one hand and number of vessels and maximum percentage of compatible berths for each vessel ($Perc_{comp}$) on the other. Table 6 groups the results by percentage of mooring restrictions ($Perc_{moor}$) and percentage of blocking restrictions ($Perc_{block}$).

The MILP solved 4267 instances optimally (75.3%) and found a non-optimal feasible solution for 258 instances (4.5%). However, no feasible solution could be found for 1145 instances (20.2%). The

number of instances solved optimally decreases with increasing number of vessels, maximum percentage of compatible berths, percentage of blocking restrictions or percentage of mooring restrictions. Instances become more difficult to solve as these factors increase, whereas difficulty appears to decrease when the number of berths at the terminal increases, given that the number of instances solved optimally increases accordingly.

Vessels	Berths			$Perc_{comp}$			Total
	8	16	24	0.25	0.50	0.75	
10	269	270	270	269	270	270	**809**
20	270	270	270	270	270	270	**810**
30	245	265	265	270	267	238	**775**
40	189	224	233	256	234	156	**646**
50	134	195	187	225	165	126	**516**
60	80	145	161	184	102	100	**386**
70	51	130	144	145	92	88	**325**
Total	**1238**	**1499**	**1530**	**1619**	**1400**	**1248**	**4267**

Table 5: Number of *Diverse* instances solved optimally by the MILP, grouped by number of vessels and number of berths (left) and by number of vessels and maximum percentage of compatible berths (right).

$Perc_{moor}$	$Perc_{block}$			Total
	0	0.25	0.50	
0	629	468	393	**1490**
0.5	611	431	373	**1415**
1	583	428	351	**1362**
Total	**1823**	**1327**	**1117**	**4267**

Table 6: Number of *Diverse* instances solved optimally by the MILP, grouped by percentage of mooring restrictions and percentage of blocking restrictions.

The heuristic was also run on the set *Diverse* to not only evaluate its performance on random and diverse instances but also to verify the insights provided by the MILP concerning the problem's difficulty. Five random runs were performed per instance considering, for each run, a time limit in seconds equal to three times the number of vessels in the instance.

The heuristic's solutions were as good as or better than the MILP solution for 5356 instances (94.5%). From the 4267 instances solved optimally by the MILP, the heuristic obtained the optimum for 4182 instances in at least one of the five runs.

Tables 7 and 8 show the average time elapsed until the heuristic found the solution reported for each instance (in seconds). The average time increases with increasing number of vessels, percentage of

mooring restrictions, or percentage of blocking restrictions, while it decreases with increasing number of berths or maximum percentage of compatible berths. Therefore, the computational effort required by the heuristic and the MILP are similarly dependent upon instance characteristics. The number of instances solved optimally when grouped by $Perc_{comp}$ equal to 0.25, 0.50 and 0.75 is 1576, 1375 and 1231, respectively; while the mean deviations of the heuristic's solutions relative to those obtained by the MILP are 0.19%, -0.56% and -0.2%. These results indicate that the heuristic better utilizes the available computation time for instances with a limited number of compatible berths per vessel. In the remaining groups, the heuristic converges faster but is slightly less successful at achieving optimal or near-optimal solutions.

Vessels	Berths			$Perc_{comp}$			Mean
	8	16	24	0.25	0.50	0.75	
10	0.03	0.02	0.02	0.02	0.02	0.02	0.02
20	0.09	0.05	0.03	0.07	0.05	0.05	0.06
30	0.92	0.13	0.06	0.32	0.60	0.19	0.37
40	5.70	0.79	0.23	3.34	2.32	1.06	2.24
50	20.28	0.98	0.34	10.38	6.04	5.18	7.20
60	50.40	5.42	1.39	25.21	16.20	15.80	19.07
70	101.84	12.30	2.61	48.33	34.43	34.00	38.92
Mean	**25.61**	**2.81**	**0.67**	**12.52**	**8.52**	**8.04**	**9.70**

Table 7: Heuristic's computation times in seconds on set *Diverse*, averaged by number of vessels and number of berths (left) and by number of vessels and maximum percentage of compatible berths (right).

$Perc_{moor}$	$Perc_{block}$			Mean
	0	0.25	0.50	
0	3.64	6.10	7.32	**5.69**
0.5	7.66	9.99	12.38	**10.01**
1	11.39	13.63	15.15	**13.39**
Mean	**7.57**	**9.90**	**11.62**	**9.70**

Table 8: Heuristic's computation times in seconds on set *Diverse*, averaged by percentage of mooring restrictions and blocking restrictions.

In summary, the MILP approach provides a good means of optimally solving most instances with up to 40 vessels arriving within a time horizon of one week. Although the heuristic does not guarantee optimality, it is able to obtain optimal or near-optimal solutions in small instances and good solutions in larger ones within very short computation times. Indeed, the solutions achieved by the heuristic in instances with 70 or more vessels are better, in average, than the solutions obtained by the MILP

approach. The heuristic is therefore the preferred method in practice given its high quality performance within short computation times.

7. Conclusions and future research

The Berth Allocation Problem in terminals with irregular layouts is a variant of the BAP where additional constraints reflect berthing operations in real-world terminals. Whereas existing literature work addresses simple mooring rules in case-specific terminals, present work reflects complicated berth operations in a general way. Terminal operators can easily construct rules which forbid vessels with certain characteristics from being moored at one or more berths as per given circumstances.

Both an MILP model and a heuristic were proposed to address this problem. An MILP solver is capable of generating optimal solutions for small to medium sized instances. The heuristic was employed to obtain solutions for larger instances which correspond to situations faced at real terminals. According to the experiments' results, the heuristic provides optimal or near-optimal solutions within reasonable computation time. A terminal located in the port of Antwerp employs the heuristic proposed. Terminal's safety rules regarding mooring and blocking restrictions were easily included in both the heuristic and the MILP model.

This study considered fixed estimates for vessel handling times. Future research may broaden the problem's scope such that resources determining handling times are also allocated and thus included in the optimization problem.

Finally, berthing schedules in real terminals may be disrupted due to maintenance, defective resources and delays occurring during quay operations. Although anticipating on schedule disruptions was not within the scope of present work, it may be an interesting thought for future research.

Acknowledgments

This work was supported by Agidens, Oiltanking Stolthaven Antwerp NV (OTSA) and Leuven Mobility Research Center, and funded by research project 140876 of Agentschap Innoveren & Ondernemen (VLAIO). This study was also partially supported by the Spanish Ministry of Economy and Competitiveness (project DPI2014-53665-P, cofinanced by FEDER funds) and by Generalitat Valenciana (project PROMETEO/2013/049 and grant BEFPI 2016). Editorial consultation provided by Luke Connolly (KU Leuven).

References

Agra, A. and Oliveira, M. (2018). MIP approaches for the integrated berth allocation and quay crane assignment and scheduling problem. *European Journal of Operational Research*, 264(1):138–148.

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.

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

Bridi, I., Rodrigues, G., Rosa, R. D. A., Gomes, T. C., and Ribeiro, G. M. (2016). Mathematical model for the Berth Allocation Problem in ports with cargo operation limitations along the pier. *Gestão & Produção*, 23(4):771–786.

Burke, E. K. and Bykov, Y. (2017). The late acceptance hill-climbing heuristic. *European Journal of Operational Research*, 258(1):70 – 78.

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(1):224–262.

Christiaens, J. and Vanden Berghe, G. (2016). A fresh ruin & recreate implementation for the capacitated vehicle routing problem. KU Leuven working paper.

Correcher, J. F. and Alvarez-Valdes, R. (2017). A biased random-Key genetic algorithm for the time-invariant berth allocation and quay crane assignment problem. *Expert Systems with Applications*, 89:112–128.

de León, A. D., Lalla-Ruiz, E., Melián-Batista, B., and Marcos Moreno-Vega, J. (2017). A Machine Learning-based system for berth scheduling at bulk terminals. *Expert Systems with Applications*, 87:170–182.

Ernst, A. T., Oğuz, C., Singh, G., and Taherkhani, G. (2017). Mathematical models for the berth allocation problem in dry bulk terminals. *Journal of Scheduling*, 20(5):459–473.

Frojan, P., Correcher, J. F., Alvarez-Valdes, R., Koulouris, G., and Tamarit, J. M. (2015). The continuous Berth Allocation Problem in a container terminal with multiple quays. *Expert Systems with Applications*, 42(21):7356–7366.

Hansen, P. and Oğuz, C. (2003). A Note on Formulations of Static and Dynamic Berth Allocation Problems. *Les Cahiers du GERAD*, 30:1–17.

He, J. (2016). Berth allocation and quay crane assignment in a container terminal for the trade-off between time-saving and energy-saving. *Advanced Engineering Informatics*, 30(3):390–405.

Hsu, H. P. (2016). A HPSO for solving dynamic and discrete berth allocation problem and dynamic quay crane assignment problem simultaneously. *Swarm and Evolutionary Computation*, 27:156–168.

Hu, Q.-M., Hu, Z.-H., and Du, Y. (2014). Berth and quay-crane allocation problem considering fuel consumption and emissions from vessels. *Computers & Industrial Engineering*, 70(1):1–10.

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

Imai, A., Nishimura, E., Hattori, M., and Papadimitriou, S. (2007). Berth allocation at indented berths for mega-containerships. *European Journal of Operational Research*, 179(2):579–593.

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

Imai, A., Nishimura, E., and Papadimitriou, S. (2013). Marine container terminal configurations for efficient handling of mega-containerships. *Transportation Research Part E: Logistics and Transportation Review*, 49(1):141–158.

Iris, Ç., Pacino, D., and Ropke, S. (2017). Improved formulations and an Adaptive Large Neighborhood Search heuristic for the integrated berth allocation and quay crane assignment problem. *Transportation Research Part E: Logistics and Transportation Review*, 105:123–147.

Karam, A. and Eltawil, A. B. (2016). Functional integration approach for the berth allocation, quay crane assignment and specific quay crane assignment problems. *Computers and Industrial Engineering*, 102:458–466.

Lalla-Ruiz, E., Expósito-Izquierdo, C., Melián-Batista, B., and Moreno-Vega, J. M. (2016). A Set-Partitioning-based model for the Berth Allocation Problem under Time-Dependent Limitations. *European Journal of Operational Research*, 250(3):1001–1012.

Lalla-Ruiz, E. and Voß, S. (2016). POPMUSIC as a matheuristic for the berth allocation problem. *Annals of Mathematics and Artificial Intelligence*, 76(1-2):173–189.

Li, C.-l., Cai, X., and Lee, C.-y. (1998). Scheduling with multiple-job-on-one-processor pattern. *IIE Transactions*, 30(5):433–445.

Lim, A. (1998). The berth planning problem. *Operations Research Letters*, 22:105–110.

Liu, C., Xiang, X., and Zheng, L. (2017). Two decision models for berth allocation problem under uncertainty considering service level. *Flexible Services and Manufacturing Journal*, 29(3-4):312–344.

Ma, H. L., Chung, S. H., Chan, H. K., and Cui, L. (2017). An integrated model for berth and yard planning in container terminals with multi-continuous berth layout. *Annals of Operations Research*, pages 1–23.

Pratap, S., Nayak, A., Kumar, A., Cheikhrouhou, N., and Tiwari, M. K. (2017). An integrated decision support system for berth and ship unloader allocation in bulk material handling port. *Computers and Industrial Engineering*, 106:386–399.

Qin, T., Du, Y., and Sha, M. (2016). Evaluating the solution performance of ip and cp for berth allocation with time-varying water depth. *Transportation Research Part E: Logistics and Transportation Review*, 87:167–185.

Røpke, S. and Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation science*, 40(4):455–472.

Shang, X. T., Cao, J. X., and Ren, J. (2016). A robust optimization approach to the integrated berth allocation and quay crane assignment problem. *Transportation Research Part E: Logistics and Transportation Review*, 94:44–65.

Türkoğullari, Y. B., Taşkin, Z. C., Aras, N., and Altinel, I. K. (2016). Optimal berth allocation, time-variant quay crane assignment and scheduling with crane setups in container terminals. *European Journal of Operational Research*, 254(3):985–1001.

Umang, N., Bierlaire, M., and Erera, A. L. (2017). Real-time management of berth allocation with stochastic arrival and handling times. *Journal of Scheduling*, 20(1):67–83.

Ursavas, E. and Zhu, S. X. (2016). Optimal policies for the berth allocation problem under stochastic nature. *European Journal of Operational Research*, 255(2):380–387.

Venturini, G., Iris, Ç., Kontovas, C. A., and Larsen, A. (2017). The multi-port berth allocation problem with speed optimization and emission considerations. *Transportation Research Part D: Transport and Environment*, 54:142–159.

Xiang, X., Liu, C., and Miao, L. (2017). A bi-objective robust model for berth allocation scheduling under uncertainty. *Transportation Research Part E: Logistics and Transportation Review*, 106:294–319.

Zhen, L. (2015). Tactical berth allocation under uncertainty. *European Journal of Operational Research*, 247(3):928–944.

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.