Computers & Operations Research 162 (2024) 106468

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor

The berth allocation and quay crane assignment problem with crane travel and setup times

Juan F. Correcher [a,*], Federico Perea [b], Ramon Alvarez-Valdes [a]

[a] *Universitat de València, Dept. d'Estadística i Investigació Operativa, Doctor Moliner 50, 46100 Burjassot, Spain*
[b] *Universidad de Sevilla, Dept. de Matemática Aplicada II and Instituto de Matemáticas (IMUS), Avda. Reina Mercedes s/n, 41012, Seville, Spain*

ARTICLE INFO

Keywords:
Berth allocation
Quay crane assignment
Setup time
Mixed integer linear programming
Combinatorial optimization
Port logistics
Container terminal

ABSTRACT

In this paper, we propose a new approach for including quay crane travel and setup times in the berth allocation and quay crane assignment problem. We first develop a new mixed integer linear programming model (MILP) for the problem without setups (BACASP), in which berthing positions and times are considered as continuous variables. Several groups of valid inequalities are also set forth. Then, for the BACASP with crane travel and setup times, which we denote as BACASP-S, we propose two MILPs: the first is based on the previous BACASP formulation and the second on routing formulations. Due to the complexity of the BACASP-S, we also propose a genetic algorithm and an exact approach which combines various MILPs with the genetic algorithm. All methods and valid inequalities are computationally tested over two different sets of randomly generated instances. According to the results, the models and algorithms can optimally solve, in less than one hour, BACASP-S instances of up to 40 vessels within a quay one kilometer long and a time horizon of one week. Additionally, extensive experiments were conducted on a new large set of instances to assess the effect of various BACASP-S input parameters on the computational effort required to solve the problem. *Ceteris paribus*, the computational effort required seems to increase with decreasing number of cranes, while vessel processing times and crane setup times seem not to affect it.

1. Introduction

Maritime transportation accounts nowadays for 90% of world trade. Despite current uncertainties, it is expected to continue growing at a rapid pace in the coming years. Among cargo types, containerized transport represents 24% of dry cargo and reached a volume of 165 million TEUs (Twenty-foot Equivalent Unit) in 2021 (UNCTAD, 2022). Most manufactured goods and many other types of products, including food and fresh produce, are increasingly using containers because of their advantages in terms of protection, standardization, and ease of exchange between different transportation modes. Container terminals act as sea-land interfaces between the various modes of transport and therefore play a key role in the efficient operation of the process. In 2022, there were 857 million TEU movements, and ports such as Shanghai moved 47 million TEUs.

Maritime container terminals compete to provide better services to customers and face two main challenges. On the one hand, the shipping industry has undergone a process of mergers and acquisitions that has resulted in three major alliances controlling more than 80% of the traffic on the main inter-oceanic routes, putting them in a strong bargaining position. On the other hand, the size of ships is increasing (note that in this paper we use *vessel* and *ship* interchangeably). Terminals have to cope with larger ships and fewer but longer calls, resulting in peak workloads at some times and idleness at others.

A key indicator of port efficiency and competitiveness is the time that ships spend in the terminal. In 2021, container ships spent a median time of 0.8 days and terminals are under pressure to further reduce it. To minimize the time ships spend in port for a given volume of cargo handled, ships must arrive at their allocated berthing time, as arriving too early entails additional costs and more pollution. Once they reach the quay, their processing must begin immediately and the flow of loading and unloading operations must be fast and reliable.

Although container terminals are interconnected systems and all their subsystems must be synchronized to operate efficiently, the above indicators, taken from UNCTAD's Review of Maritime Transport 2022 (UNCTAD, 2022), directly point to the optimization of processes in the seaside subsystem.

In the seaside area, the most important problem is the Berth Allocation Problem (BAP), where each ship to be served must be assigned a berthing position and time within a planning horizon. When several

* Corresponding author.
E-mail addresses: juan.correcher@uv.es (J.F. Correcher), perea@us.es (F. Perea), ramon.alvarez@uv.es (R. Alvarez-Valdes).

https://doi.org/10.1016/j.cor.2023.106468
Received 28 February 2023; Received in revised form 8 October 2023; Accepted 30 October 2023
Available online 2 November 2023

ships arrive simultaneously at the quay, a Quay Crane Assignment Problem (QCAP) arises to allocate the available cranes. These two problems are usually addressed together, because the number of quay cranes assigned to a ship determines its processing time. In the combined berth and quay crane assignment problem (BACAP), a number of cranes is assigned to each ship, as well as its berthing time and position. A more realistic problem is the BACASP, in which not only a number of cranes, but a specific set of cranes is assigned to each ship, thus ensuring that the cranes can serve the ships without crossing each other (we assume that it is not possible for cranes to cross each other along the quay).

There are two versions of the BACAP and the BACASP. In the time-invariant version, the crane-ship assignment remains constant throughout its operation, while in the variable-in-time version this assignment can change. The variable-in-time version allows for a more efficient use of cranes, since those initially assigned to a ship can be reassigned to newly arriving vessels. However, the solutions may have more complex crane-to-ship assignments, with a greater number of crane movements. In the time-invariant version, the assignment of a crane to a vessel remains constant, resulting in simpler solutions with fewer crane movements, although they often waste crane capacity.

It is very interesting to compare the two versions of the problem, but a key factor is how crane movements are taken into account. If one considers that the time required for cranes to move from one ship to another is negligible, the variable-in-time version may yield more efficient solutions. However, in practice the movement of quay cranes is not an easy task, especially if that movement involves the movement of other adjacent cranes. Here a new question arises. If time is discretized, for example in 1-h periods, as is often considered in the literature, the time allocated to a movement must be a number of periods. Even if we consider only one period, it would be a very rough and rigid estimate that could lead to solutions with very few movements. Consequently, a more realistic and flexible way of assigning times to crane movements has to consider continuous time or a much more granular discretization.

This has been done when designing heuristic algorithms (Hsu et al., 2019), but to the best of our knowledge it has not been attempted in the development of integer linear programming models. In this paper we propose mixed integer linear programming models for the time-invariant version of BACASP in which the two dimensions of the assignment process, time and position, are considered continuous. This allows us to extend the models to include the times for crane setups and movements that consider the initial and final position of a crane when moving from one vessel to another. Although crane travel and setup times are usually much shorter than vessel processing times, they must be taken into account when designing the schedule in order to control the number of moves and to properly reflect their cumulative effect throughout the planning horizon.

This paper presents five main contributions:

1. We propose a mixed integer linear programming model for the BACASP in which no discretization is needed for either for the berthing position of the ships at the quay or their berthing times. The model is enhanced by developing several families of valid inequalities.
2. We include crane travel and setup times in the BACASP, producing a new problem, BACASP-S, for which two alternative MILP models are proposed.
3. We also develop a genetic algorithm for the BACASP-S to provide solutions for large instances.
4. We also propose an exact method for the BACASP-S which combines exact and heuristic procedures for the BACASP-S and the related BACAP and BACASP.
5. We have generated an extensive benchmark of test instances, with controlling factors such as the number of cranes available, vessel processing times and crane speed and setup time.

A computational study shows that the new MILP model for the continuous BACASP is able to optimally solve instances with up to 40 vessels. Although the BACASP-S is more complex, good results are also obtained with one of the proposed models. The flexible exact procedure that allows the inclusion of other simpler related problems is able to reduce the gaps and obtain some optimal solutions in difficult instances.

The rest of this paper is organized as follows. In Section 2, we review the related literature. In Section 3, we describe the problem and the specific assumptions of our approach. In Section 4, we propose a new mixed integer linear programming model for the BACASP. In Section 5, several valid constraints are added to reinforce the MILP. The MILP models are extended to solve the BACASP with crane travel and setup times in Section 6. In Section 7, a genetic algorithm for solving the BACASP-S is introduced. Section 8 presents a MILP-based exact algorithm which aims to solve the proposed problems more efficiently. In Section 9, we describe the experiments conducted and discuss their results. Finally, in Section 10, we draw some conclusions and indicate future research directions.

2. Related literature

In this section we focus on reviewing BACASP studies, especially those that consider crane setup or travel times. Early surveys on BAP and BACAP were conducted by Bierwirth and Meisel (2010, 2015) and Carlo et al. (2015). More recent papers such as those by Agra and Oliveira (2018) and Malekahmadi et al. (2020) contain excellent reviews of the latest proposals for these problems.

2.1. BACASP studies

The study of BACASP can be traced back to Park and Kim (2003) who developed a MILP and a Lagrangean-based heuristic for the BACAP with variable-in-time crane assignments and then a dynamic programming algorithm to solve the corresponding BACASP. Zhang et al. (2010) also proposed an integer linear formulation and a Lagrangean-based procedure to solve the BACASP in a single phase, including ranges in which cranes can move along the quay. Chang et al. (2010) developed a multiobjective approach to the variable-in-time BACASP, including the energy consumption of the cranes, and developed a parallel genetic algorithm. Yang et al. (2012) proposed a Nested Loop Evolutionary Algorithm for time-invariant BACASP, obtaining better results than (Park and Kim, 2003). Le et al. (2012) solved the time-invariant BACASP by a multi-objective MILP model and a multi-objective Particle Swarm Optimization procedure.

Rodriguez-Molins et al. (2014s) developed a GRASP algorithm to solve both time-invariant and variable-in-time versions of the BACASP. Rodriguez-Molins et al. (2014b) also proposed a multi-objective MILP and a multi-objective genetic algorithm to obtain robust solutions for the time-invariant BACASP. Türkoğullari et al. (2014) proposed integer linear formulations for time-invariant BACAP and BACASP and developed a cutting plane algorithm to solve BACASP by repeatedly solving BACAP with additional constraints. Türkoğullari et al. (2016) extended their study to the variable-in-time BACASP, proposing a MILP and a cutting plane procedure based on a decomposition scheme.

Li et al. (2015) studied the time-invariant version with crane ranges and proposed a nonlinear mixed integer formulation. A novel heuristic based on the spatio-temporal analysis of vessel conflicts was applied to obtain good quality solutions. Karam and Eltawil (2016) solved the variable-in-time version by decomposing it into two problems, BAP and SQCAP. The two problems are optimally solved and functionally integrated through a feedback loop. Li et al. (2017) proposed a multi-objective model for the time-invariant version and developed a Chaos Cloud Particle Swarm Optimization algorithm. Agra and Oliveira (2018) also addressed the variable-in-time BACASP, proposing new integer linear programming models and a rolling-horizon heuristic.

Correcher and Alvarez-Valdes (2017) solved the time-invariant BACASP by means of a metaheuristic approach based on a Biased Random-key Genetic Algorithm with memetic characteristics and several Local Search procedures. Correcher et al. (2019a) extended the study by proposing a MILP model, an iterative procedure in which the BACAP model is solved, and if its solution is not feasible for the BACASP, specific constraints are added until an optimal solution for the BACASP is found. Additionally, a branch-and-cut algorithm is proposed based on the cuts used in the iterative procedure. Zheng et al. (2019) studied the variable-in-time BACASP including preventive maintenance activities in which cranes are temporarily unavailable and proposed an integer model, a heuristic and a genetic algorithm. Malekahmadi et al. (2020) addressed the variable-in-time BACASP with tidal constraints and proposed a Random Topology Particle Swarm Optimization algorithm. Ji et al. (2022) tackled the time-invariant BACASP, embedding it into a rolling-horizon procedure to dynamically consider unscheduled incoming vessels and solving it by means of an ALNS algorithm.

All the above studies considered the discretization of time or position on the quay, so some of the related variables in the models were always integer or binary. In fact, for some time, this seemed to be the only way to address the problem. However, in the recent work by Cheimanoff et al. (2022) both time and position variables are continuous. Their results show that this approach compares favorably with discrete-time approaches. The model that they developed is similar to the one proposed here for the BACASP, although with some differences in the way they address the crane assignment.

2.2. Considering crane travel and setup times

Many of the above-mentioned papers consider crane travel times to be negligible, so they are not included in the solutions. A crane may be in one position on the quay at the end of one period and in another position at the beginning of the next period (Park and Kim, 2003; Chang et al., 2010; Zhang et al., 2010; Le et al., 2012; Rodriguez-Molins et al., 2014b; Türkoğullari et al., 2014; Li et al., 2015, 2017; Correcher and Alvarez-Valdes, 2017; Song et al., 2019; Zheng et al., 2019; Abou Kasm et al., 2020; Malekahmadi et al., 2020; Grubisic et al., 2020).

However, some authors do consider these crane movements and there is a growing trend in recent years to take them into account in different ways. Zampelli et al. (2013), in their Constraint Programming approach to the BACAP, consider that when a crane moves it is unavailable for 30 min, regardless of the movement. The model proposed by Shang et al. (2016) takes crane setup times into account. As time is discrete in their model, these setups take one period, irrespective of the positions of cranes and vessels. The authors extend their study by including uncertainties and proposing a robust optimization model as well as a genetic and a heuristic insertion algorithm. Karam and Eltawil (2016) approximate the time needed to move a crane to a constant average value and multiply it by the number of moves when solving the SQCAP to minimize the vessel processing time. Liu et al. (2016) consider the variable-in-time BACASP with crane disruptions and develop a two-phase procedure in which Phase I solves the BACAP and Phase II solves the crane assignment. In Phase II, the objective is to minimize the number of crane changes between vessels, although crane movements do not affect the BACAP solution. Their strategy is similar to that of Yang et al. (2012) in their Nested Loop Evolutionary Algorithm.

Türkoğullari et al. (2016) include crane setup costs in the objective function of their variable-in-time BACASP integer model. These setup costs are general and may include travel times between vessel positions. In particular, they are calculated in an iterative decomposition procedure, but they do not affect vessel processing times. Agra and Oliveira (2018) use an auxiliary model for finding the solution with the minimum number of movements among the optimal solutions of the BACASP, in which they do not consider crane travel times. Hsu et al. (2019) compare the time-invariant and variable-in-time versions of a discrete BACAP, considering that crane reassignments are only made when a vessel leaves the quay and the cranes servicing it are released. To make the comparison fair, the objective function takes into account the cost of cranes setups and movements. They propose a genetic algorithm that provides solutions for the time-invariant case and an event heuristic that modifies them by allowing crane reassignments. They find that with their parameters and costs the variable-in-time version obtains better solutions. Salhi et al. (2019) study an integrated problem that includes BAP, QCAP, and the crane tasks scheduling problem (QCSP). In their model they include the travel times of the cranes between consecutive task locations performed by the same crane, considering berthing positions and times as continuous variables. They propose a MILP model able to solve only small instances, and a genetic algorithm for large instances.

In summary, the issue of considering quay crane travel and setup times as part of the planning process cannot be properly addressed with discrete-time models. These times can be considered neither constant nor integer multiples of the discrete periods. In order to handle jointly the loading/unloading processing times and the travel and setup times of cranes, a continuous time scale is needed. In this context, only Salhi et al. (2019) use continuous variables for berthing times and count the crane travel times as part of the solution. However, they address a more complex problem, including also the scheduling of the cranes, and their mixed integer linear programming model for the combined problem is only able to solve instances with a few vessels. In this paper, we focus on the BACASP, develop models with continuous berthing positions and times, with and without crane setups and travel times, and solve instances with up to 40 vessels within a time horizon of one week and a quay one kilometer long.

3. Description of the problem

In this section we describe the problem addressed in this paper. We begin with a brief outline of its main characteristics, followed by a formal description.

3.1. Outline

We consider a set of ships arriving at a port terminal with a single quay that will be scheduled over a planning horizon. The quay is represented as a continuous line along which vessels can berth. The vessels must be processed at the quay to be loaded and unloaded. These tasks are carried out by a number of cranes, available on the quay. No crane can process more than one vessel at a time. The number of cranes assigned to a vessel remains constant throughout the vessel's processing (time-invariant). The objective of the problem is to determine a schedule for all the vessels that minimizes the sum of waiting times, delays, and deviations from their desired positions on the quay.

More specifically, we assume that: (1) Each vessel has a known arrival time and a maximum desired departure time. Waiting times before berthing and delays with respect to departure times are penalized. (2) Each vessel requires a number of quay cranes to be served, between a minimum and a maximum value. The processing time of a vessel depends on the number of cranes assigned. (3) Each vessel has a desired position on the quay, due to its proximity to the area of the yard from which containers are to be loaded into the vessel or to which containers are to be unloaded from the vessel. Deviations from this position are penalized because they increase the distance to be covered by the vehicles moving containers between the yard and the quay (Hendriks et al., 2013; Bierwirth and Meisel, 2015; Türkoğullari et al., 2016; Salhi et al., 2019).

The schedule will explicitly consider travel and setup times required for the cranes between the processing of two consecutive vessels. The crane travel time is variable and depends on the crane travel speed and the distance between the two vessels at the quay, while the setup time has a fixed value and comprehends the activities necessary to decouple the crane after servicing a vessel and the activities needed to setup the crane to serve another vessel. We call this problem the Berth Allocation and Crane Assignment Problem with Setup (BACASP-S).

3.2. Input data

The following sets are defined:

- $V = \{1, \dots, N\}$, $N \in \mathbb{Z}^+$ is the set of vessels. For modeling purposes, let $V^0 = V \cup \{0\}$, where vessel 0 is a dummy vessel. The vessels are indexed by i and j.
- $K = \{1, \dots, Q\}$, $Q \in \mathbb{Z}^+$ is the set of available cranes, indexed by k, k' (which refer to specific cranes) and q (which refers to the number of cranes assigned to a vessel). Cranes are numbered from 1 to Q, increasing with their distance from the start of the quay. As the cranes can move along the quay but cannot cross each other, their relative ordering remains constant.
- $L > 0$ is the length of the quay.

The following input data are assumed to be known and deterministic. For each vessel $i \in V$ we know:

- Length: $l_i \in [0, L]$, which also includes the safety distance between vessels, so it actually is the required space for the vessel to be properly moored.
- Arrival time: $a_i \geq 0$
- Cost per unit of waiting time for berthing after the expected arrival time: $C_i^w \geq 0$
- Maximum desired departure time: $s_i > 0$
- Cost per unit of delay time after the maximum desired departure time: $C_i^d \geq 0$
- Desired position at the quay: $b_i \in [0, L - l_i]$
- Cost per unit of length away from the desired position at the quay: $C_i^p \geq 0$
- Minimum and maximum number of cranes that can be assigned to the vessel: $q_i^{\min}, q_i^{\max} \in \mathbb{Z}^+$, with $q_i^{\min} \leq q_i^{\max} \leq Q$.
- We define $K_i = \{1, \dots, Q - q_i^{\min} + 1\} \subseteq K, \forall i \in V$, as the set of cranes that can be the first crane assigned to vessel i. Cranes assigned to a vessel are consecutive, so the first crane assigned to vessel i cannot be greater than $Q - q_i^{\min} + 1$.
- Let $Q_i = \{q_i^{\min}, \dots, q_i^{\max}\}$ be the set of all numbers of cranes admitted for vessel i.
- Processing time if q cranes are assigned to the vessel: $u_{iq} > 0$, $q \in Q_i$. The processing time of a vessel can vary linearly with the number of cranes or can be related to the number of cranes by a more complex expression, considering crane interference. It may include the time estimated for berthing and unberthing operations.

When considering crane travel and setup time, for each crane $k \in K$ the following are also known:

- $\alpha_k > 0$: the speed of crane k, in order to measure how long it will take to move from one position to another on the quay. Let $\alpha = \min_{k \in K}(\alpha_k)$.
- $\beta_k > 0$: the setup time that crane k requires between serving two consecutive vessels. Let $\beta = \max_{k \in K}(\beta_k)$.

The notation is summarized in Table 10 (Appendix A). Fig. 1 shows an example of a berth plan depicting the terminology used.

4. A model for the time-invariant BACASP with continuous space and time

In this section we present a new mixed integer linear programming model for the BACASP problem, with the novelty that it considers both time and space as continuous, as opposed to the more usual discretization of these dimensions. This model does not consider crane movements and setup times.

4.1. Variables

- $t_i \geq a_i$: berthing time of vessel i. Note that we impose the condition that vessel i cannot be moored before its arrival time.
- $p_i \in [0, L - l_i]$: the berthing position for vessel i at the quay, measured as the shortest distance from the vessel to the start of the quay.
- $d_i \geq 0$: delay incurred in processing vessel i with respect to its maximum desired departure time.
- $e_i \geq 0$: deviation of the berthing position of vessel i with respect to its desired position at the quay.
- $\sigma_{ij} = 1$ if the departure time of vessel i is prior to the berthing time of vessel j, 0 otherwise.
- $\delta_{ij} = 1$ if vessel i is completely below vessel j, 0 otherwise.
- $r_{iq} = 1$ if vessel i is served exactly by q cranes, 0 otherwise.
- $w_{ik} = 1$ if k is the first crane in the sequence of consecutive cranes assigned to i, 0 otherwise.

4.2. Objective function and constraints

Using the input data and the variables defined above, we propose the following MILP model for the time-invariant BACASP with continuous space and time:

$$\min \sum_{i \in V} \left(C_i^w (t_i - a_i) + C_i^d d_i + C_i^p e_i \right) \quad (1)$$

subject to

$$e_i \geq p_i - b_i, \quad \forall i \in V \quad (2)$$

$$e_i \geq -(p_i - b_i), \quad \forall i \in V \quad (3)$$

$$d_i \geq t_i + \sum_{q \in Q_i} u_{iq} r_{iq} - s_i, \quad \forall i \in V \quad (4)$$

$$p_j \geq p_i + l_i - L(1 - \delta_{ij}), \quad \forall i, j \in V, i \neq j \quad (5)$$

$$t_j \geq t_i + \sum_{q \in Q_i} u_{iq} r_{iq} - T(1 - \sigma_{ij}), \quad \forall i, j \in V, i \neq j \quad (6)$$

$$\sigma_{ij} + \sigma_{ji} + \delta_{ij} + \delta_{ji} \geq 1, \quad \forall i, j \in V, i \neq j \quad (7)$$

$$\sum_{q \in Q_i} r_{iq} = 1, \quad \forall i \in V \quad (8)$$

$$\sum_{k \in K_i} w_{ik} = 1, \quad \forall i \in V \quad (9)$$

$$\sum_{k \in K_i} k w_{ik} + \sum_{q \in Q_i} q r_{iq} \leq Q + 1, \quad \forall i \in V \quad (10)$$

$$\sum_{k \in K_j} k w_{jk} \geq \sum_{k' \in K_i} k' w_{ik'} + \sum_{q \in Q_i} q r_{iq} - Q(\delta_{ji} + \sigma_{ij} + \sigma_{ji}), \quad \forall i, j \in V, i \neq j \quad (11)$$

The objective function (1) sums up the total cost due to vessels waiting for berthing after their arrival times, the total cost of delay after their maximum desired departure times, and the total cost of deviations from their desired positions at the quay. Constraints (2) and (3) define the deviation of vessels from their desired position at the quay and constraints (4) the delay with respect to their maximum desired departure times. Constraints (5), (6), and (7) ensure that vessel assignments do not overlap. If δ_{ij} or δ_{ji} take value 1, vessels i and j are separated in space by (5). If σ_{ij} or σ_{ji} take value 1, vessels are separated in time by (6). Note that T is an upper bound on the total time required to berth all the vessels. It can be given as input data (i.e. the time horizon) or determined by using algorithms such as Algorithm 2 in Correcher et al. (2019b). By (7), at least one of these variables takes value one. Constraints (8) force the number of cranes assigned to vessel i to be within its allowed range and constraints (9)

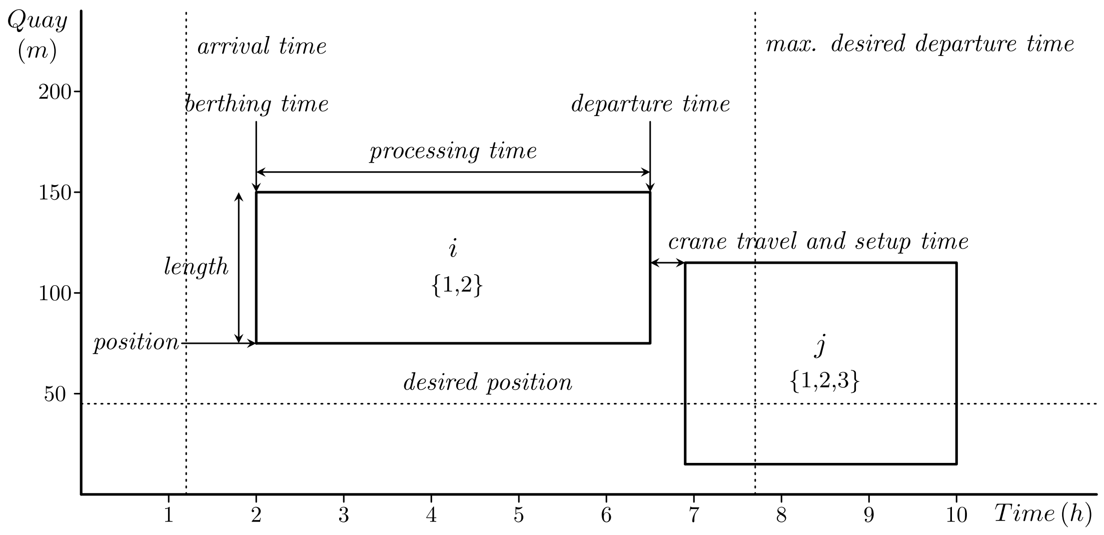

Fig. 1. Graphic representation of a berth plan with two vessels i and j. The cranes assigned to the vessels appear in brackets. The data indicated correspond to vessel i. Both vessels use cranes 1 and 2, so the processing of vessel j cannot start until those cranes are in position and ready after serving vessel i.

set the number of the crane with the lowest number assigned to each vessel. Together, (8) and (9) fix the number of cranes assigned to each vessel. By constraints (10), the numbers of the cranes assigned to a vessel cannot exceed the number of available cranes. Constraints (11) ensure that the cranes assigned to each pair of vessels i and j that are served simultaneously at the quay do not cross each other. The number of the first crane assigned to vessel j is greater than the number of the last crane assigned to vessel i if vessel i is below vessel j ($\delta_{ij} = 1$) and both vessels are concurrent ($\sigma_{ij} = \sigma_{ji} = 0$).

5. Valid inequalities

With the aim of improving the previous formulation, in this section we propose several families of valid inequalities.

5.1. Transitivity

For any three vessels i, j, k, if i and j, and j and k are separated, in space or in time, then i and k must also be separated:

$$\delta_{ij} + \delta_{jk} \leq 1 + \delta_{ik}, \qquad \forall i,j,k \in V, \tag{12}$$

$$\sigma_{ij} + \sigma_{jk} \leq 1 + \sigma_{ik}, \qquad \forall i,j,k \in V. \tag{13}$$

Many inequalities could be generated in this way, so we introduce only those that we consider interesting, namely those in which the vessel assignments are sufficiently separated in space/time. In particular, in the case of the delta inequalities, we include only those for which $\forall i,j,k \in V \ : \ a_i + h_\delta \cdot u_{iq_i^{\min}} < a_j$ and $a_j + h_\delta \cdot u_{jq_j^{\min}} < a_k$, being h_δ a constant greater than 0. Regarding sigma inequalities, we include only those for which $\forall i,j,k \in V \ : \ b_i + h_\sigma \cdot l_i < b_j$ and $b_j + h_\sigma \cdot l_j < b_k$, h_σ being a constant greater than 0.

5.2. Use of cranes

Constraints (10) of the model apply to each individual vessel. A valid constraint on the use of the cranes by a subset S of vessels would be:

$$\sum_{i\in S}\sum_{q\in Q_i} q r_{iq} \leq Q + M \sum_{i,j\in S, i\neq j} \sigma_{ij}, \quad \forall S \subseteq V, \tag{14}$$

where $M = \sum_{i\in S} q_i^{\max} - Q$. To limit the number of such inequalities, we introduce only those related to relevant sets of vessels S such that:

- $|S| \leq \bar{c}$, where $\bar{c}$ is the maximum number of vessels allowed to constitute a set S;
- all vessels in S would be concurrent if they were assigned at their time of arrival and their departure time would be $a_i + h_c \cdot u_{iq_i^{\min}}, \forall i \in S$, h_c being a constant greater than 0;
- the sum of their minimum number of cranes is less than or equal to Q: $\sum_{i\in S} q_i^{\min} \leq Q$;
- the sum of their maximum number of cranes is greater than or equal to Q: $\sum_{i\in S} q_i^{\max} > Q$;
- the sum of their lengths is less than or equal to L: $\sum_{i\in S} l_i \leq L$.

5.3. Separation in time of vessels using the same cranes

If vessels i and j use the same cranes they must be separated in time:

$$w_{ik} + w_{jk'} \leq 1 + \sigma_{ij} + \sigma_{ji}, \quad \forall i,j \in V, i \neq j, \forall k \in \{1,\dots,Q - q_i^{\min} + 1\},$$
$$k' \geq k,\ k' < k + q_i^{\min},\ k' \leq Q + q_j^{\min} + 1. \tag{15}$$

For each vessel i, we introduce only the inequalities related to the first η vessels j that would concur with i if they were assigned to their time of arrival with their maximum number of cranes allowed, ordered by non-increasing time of concurrence with i, η being an integer greater than 0.

5.4. Non-concurrence by the sum of the lengths or the sum of the minimum number of cranes

For all subsets of vessels $F \subseteq V \ : \ \sum_{i\in F} l_i > L$ and for all $G \subseteq V$: $\sum_{i\in G} q_i^{\min} > Q$, at least one vessel in the subset must be separated in time with respect to another vessel in the subset. In other words, not all vessels in the subset can be concurrent. Let $D = F \cup G$:

$$\sum_{i\in G}\sum_{j\in G, j\neq i} \sigma_{ij} \geq 1, \quad \forall S \in D. \tag{16}$$

To limit the number of inequalities included, we only consider the minimal subsets S in which the vessels would be concurrent if they were assigned to their arrival time with their minimum number of cranes allowed and $|S| \leq \bar{\sigma}$, $\bar{\sigma}$ being an integer strictly greater than 1. To illustrate the concept of a minimal set, let $S_1, S_2 \in D$: if $S_1 \subset S_2$, then S_2 would not be minimal.

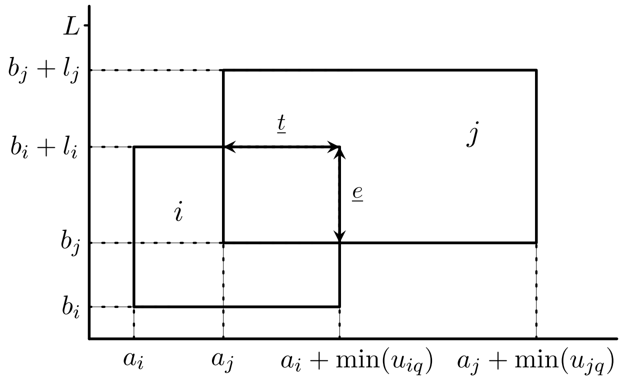

Fig. 2. Minimal separation in time and space.

5.5. *Coherence between variables w_{ik} and r_{iq}*

Constraints (10) can be disaggregated as follows:

$$kw_{ik} + qr_{iq} \leq Q+1, \qquad \forall i \in V, \forall k \in K_i, \forall q \in Q_i,\ q+k > Q+1. \quad (17)$$

If k is the first crane assigned to vessel i, the number of cranes assigned to it must not be greater than $Q-k+1$. Similarly, if vessel i has q cranes assigned, the first crane cannot be greater than $Q-q+1$. Note that if $q+K \leq Q+1$, the previous constraint would be trivial.

Moreover, we can replace this set of constraints with a stronger version of them:

$$w_{ik} + r_{iq} \leq 1, \qquad \forall i \in V, \forall k \in K_i, \forall q \in Q_i,\ q+k > Q+1. \quad (18)$$

The latter are stronger because both k and q are less than $Q+1$, and therefore $\frac{k}{Q+1}w_{ik} + \frac{q}{Q+1}r_{iq} \leq w_{ik} + r_{ik}$. Moreover, those constraints can be lifted, so their final version would be:

$$\sum_{k'=k}^{Q-q_i^{\min}+1} w_{ik'} + \sum_{q'=q}^{q_i^{\max}} r_{iq'} \leq 1, \quad (19)$$

$$\forall i \in V, \forall k \in K_i \,|\, \exists g \in Q_i, g+k > Q+1, q = \arg\min_{g \in Q_i}(g+k > Q+1).$$

5.6. *Minimum separation in time and space (Correcher et al., 2019a)*

A vessel has a least-cost assignment if it is berthed at its arrival time and in its desired position at the quay. If a pair of vessels i and j overlap in their least-cost assignments, as shown in Fig. 2, then in any feasible solution they must be separated in space and/or in time. If they are separated in time ($\sigma_{ij}+\sigma_{ji} = 1$), we can compute the minimum separation $\underline{t}$ (in Fig. 2: $\underline{t} = a_i + \min(u_{iq}) - a_j$) and then:

$$t_i + t_j \geq a_i + a_j + \underline{t} \cdot (\sigma_{ij} + \sigma_{ji}), \quad \forall i, j \in V, i \neq j. \quad (20)$$

If they are separated in space ($\delta_{ij} + \delta_{ji} = 1$), we can compute the minimum separation $\underline{e}$ (in Fig. 2: $\underline{e} = b_i + l_i - b_j$) and hence:

$$e_i + e_j \geq \underline{e} \cdot (\delta_{ij} + \delta_{ji}), \quad \forall i, j \in V, i \neq j. \quad (21)$$

In general, $\underline{t} = \min(\max(0, a_i + u_{iq^{\max}} - a_j), \max(0, a_j + u_{jq^{\max}} - a_i))$ and $\underline{e} = \min(\max(0, b_j + l_j - b_i), \max(0, b_i + l_i - b_j))$.

5.7. *Cover constraints on ordered sets of vessels that together exceed the length of the quay (Correcher et al., 2019a)*

If there is a set of vessels S such that $\sum_{i \in S} l_i > L$, then not all of them can be served simultaneously and at least one of them must be separated in time. Fig. 3 shows an example with $S = \{i, j, k\}$. In this case, the inequality $\delta_{ij} + \delta_{jk} + \delta_{ki} \leq 1$ must be satisfied, and we can add similar constraints for other orderings, for instance, $\delta_{ji} + \delta_{ik} + \delta_{kj} \leq 1$.

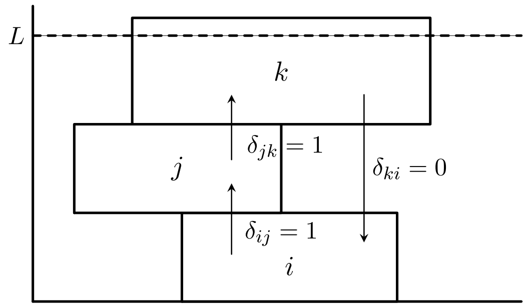

Fig. 3. Vessels that do not fit together at the quay.

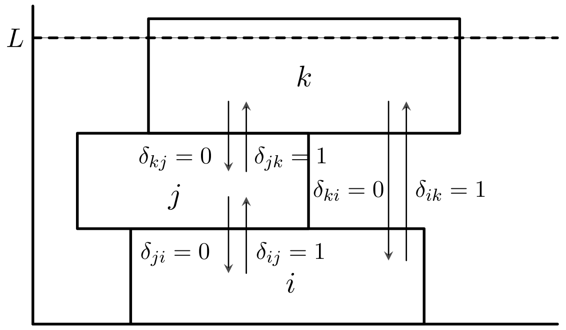

Fig. 4. Cover constraint on a non-ordered set of vessels violated.

In general, if we identify a subset of vessels S such that the sum of their lengths exceeds the length of the quay, for each permutation of vessels in S, $i_1, i_2, \dots, i_{|S|}$, we have a constraint:

$$\delta_{i_1 i_2} + \delta_{i_2 i_3} + \cdots + \delta_{i_{|S|-1} i_{|S|}} + \delta_{i_{|S|} i_1} \leq |S| - 2, \quad i_1, i_2, \dots, i_{|S|} \in S. \quad (22)$$

We only consider minimal subsets S of vessels that do not fit together at the quay and coincide in time if they were moored at their arrival times, considering their maximum processing times. We also limit the cardinality of the subsets, using a parameter $\bar{\delta}$, so $|S| \leq \bar{\delta}$.

5.8. *Cover constraints on non-ordered sets of vessels that together exceed the length of the quay (Correcher et al., 2019a)*

Other valid constraints can be developed if all possible variables separating vessels in time are considered together, as can be seen in Fig. 4:

$$\sum_{i \in S} \sum_{j \in S, j \neq i} \delta_{ij} \leq \frac{|S|^2 - |S|}{2} - 1, \ \forall S : \sum_{i \in S} l_i > L. \quad (23)$$

As in the previous case, we only consider minimal subsets of vessels S that do not fit together at the quay and coincide in time if they are moored at their arrival times, considering their maximum processing times. Here we do not limit the cardinality of S, as there are not so many such inequalities.

6. A model for the time-invariant BACASP-S

We now propose a MILP for the BACASP with crane travel and setup times (BACASP-S), which is based on the previous BACASP model presented in Section 4. A second model, based on traditional routing models, has also been designed. However, since its results were clearly outperformed by the first model, we only show it in the Appendix II.

6.1. *Model based on Section 4*

This new model needs, in addition to the variables in the BACASP model, the following:

- $v_{ij} = 1$ if vessel i is below vessel j and the crane with the highest number among those assigned to vessel i (its last crane) has a number lower than the number of the crane with the lowest number among those assigned to j (its first crane); 0 otherwise.
- $f_{ij} \geq 0$ is the distance between vessels i and j, measured as the distance between their middle positions.

A MILP model for the BACASP-S consists of minimizing (1) subject to constraints (2)–(11) and:

$$f_{ij} \geq \left(p_i + \frac{l_i}{2}\right) - \left(p_j + \frac{l_j}{2}\right), \qquad \forall i,j \in V, i \neq j \quad (24)$$

$$f_{ij} \geq \left(p_j + \frac{l_j}{2}\right) - \left(p_i + \frac{l_i}{2}\right), \qquad \forall i,j \in V, i \neq j \quad (25)$$

$$t_j \geq t_i + \sum_{q\in Q_i} u_{iq} r_{iq} + \frac{f_{ij}}{\alpha} + \beta - T(1 - \sigma_{ij} + v_{ij} + v_{ji}), \quad \forall i,j \in V, i \neq j \quad (26)$$

$$v_{ij} \geq \delta_{ij} - \sigma_{ji} - \sigma_{ij}, \qquad \forall i,j \in V, i \neq j \quad (27)$$

$$\sum_{k\in K} k w_{jk} \geq \sum_{k'\in K} k' w_{ik} + \sum_{q\in Q_i} q r_{iq} - Q(1 - v_{ij}), \qquad \forall i,j \in V, i \neq j \quad (28)$$

Constraints (24) and (25) define the distance between vessels i and j. Constraints (26) extend constraints (6) by including between the starting times of vessels i and j not only the processing time of vessel i, but also the setup times of the cranes that are assigned to both consecutively. Constraints (27) link the new variables v_{ij} with variables δ_{ij} and σ_{ij} defined to avoid overlaps. Constraints (28) ensure that the cranes assigned to each pair of vessels i and j do not cross each other. Now constraints (11) become redundant with (28) and can be removed, although they may be kept as additional cuts.

The BACASP-S model can be simplified under certain conditions. Let F be the set of pairs of vessels $(i,j), i \neq j$, satisfying $q_i^{\min} + q_j^{\min} > Q$. For these pairs, as vessels i and j cannot be handled concurrently and will use at least one of the same cranes, constraints (6) and (26) for those pairs can be replaced by:

$$t_j \geq t_i + \sum_{q\in Q_i} u_{iq} r_{iq} + \frac{f_{ij}}{\alpha} + \beta - T(1 - \sigma_{ij}), \quad \forall (i,j) \in F, \quad (29)$$

and variables and constraints related to v_{ij}, δ_{ij} $\forall (i,j) \in F$ need not be included. Moreover, their corresponding constraints (7), can be replaced by:

$$\sigma_{ij} + \sigma_{ji} = 1, \quad \forall (i,j) \in F, \quad (30)$$

and constraints (5) and (11) can be eliminated likewise.

This changes the interpretation of variables $\sigma_{ij}, \forall (i,j) \in F$. Thus, for those pairs $\sigma_{ij} = 1$ now means that vessel j starts after the processing of vessel i ends *plus* the time needed for the cranes to move to the position of j. Note that the valid inequalities proposed in Section 5 can also be applied to this model.

7. A genetic algorithm for the BACASP-S

In this section we propose a Biased Random-key Genetic Algorithm (BRKGA) for the BACASP-S based on our BRKGA developed for the continuous BACAP and discrete-time BACASP (Correcher and Alvarez-Valdes, 2017). The main differences from these previous heuristics are: a new constructive algorithm considering continuous time and crane setup time, with a crane utilization list developed to reduce computational time; and a new local search matheuristic which uses the proposed BACASP-S model to solve the problem with some variables fixed according to an input solution.

The BRKGA works as a genetic algorithm with N_{pop} populations with N_{ind} individuals each. The best individual in each population is added to the other populations each certain number of generations. During each generation, a percentage of the best individuals in the population is declared elite and passes directly to the next generation. A percentage of the worst individuals are replaced by new random ones. The rest of the individuals are generated by crossover, which is performed in pairs, always with a random individual from the elite set and a random individual from the entire population. The elite individual is favored during crossover with a bias, so the probability that a gene from the elite individual is passed on to the offspring is a value in the interval $[0.5, 1]$. The genetic algorithm is run for some time, and afterwards the local search procedure is applied over the individuals of the resulting populations, ordered by non-decreasing cost.

An individual codifies a solution in the form of a chromosome. A chromosome consists of two lists: a list of vessels and a list with the number of cranes assigned to each vessel. Each element of these lists is considered a gene for crossover purposes and the constructive algorithm is able to decode them and construct a feasible solution to the problem.

To avoid redundancies, in the following we will focus on the new contributions, since the detailed explanation of the whole approach can be found in Correcher and Alvarez-Valdes (2017).

7.1. Constructive algorithm

The constructive algorithm is based on the procedure proposed in the cited paper for the BACASP with discretized times, adapting it to the case considered here: continuous time and including crane travel and setup times.

The algorithm creates a solution from an ordered list of pairs (i,q), i being a vessel and $q \in Q_i$ a valid number of cranes for i. For each $i \in V$, only one pair is admitted in the list. Starting from an empty berth schedule, the algorithm adds the vessels to the schedule in the order of the list, one by one, with their corresponding number of cranes q. At each step, the algorithm finds the berthing time and position that minimize the objective function for the vessel, given the partial solution with the previous vessels in the list already included.

To do this, once a vessel i is taken to be included in the schedule, the algorithm starts from the candidate point in which the vessel would be at its zero-cost position on the quay and with berthing time equal to its time of arrival. If the vessel cannot be assigned there, a list of alternative candidate assignments is generated and iteratively explored until the minimum-cost feasible assignment is found. Feasibility requires not only that there is no overlap with previous vessel assignments, but also compatibility with the number of cranes and the specific cranes used. Thus, we must ensure that (1) there will be at least q consecutive cranes available and ready on the quay for the entire vessel processing time, u_{iq}; and (2) that the vessel assignment satisfies travel and setup time constraints for all the specific cranes allocated to serve the vessel.

As time is continuous, we cannot keep track of the number of cranes used per time period. Instead, we generate a list of tuples $(t, v_{in}, v_{out}, q_{used})$, where t is a time instant, v_{in} and v_{out} are the number of vessels berthing and departing at that time, respectively, and q_{used} is the number of cranes in use from that time until the next time in the list. The list is ordered by increasing time and updated as the vessels are included in the schedule.

Once a feasible BACASP solution is obtained, the crane travel and setup time constraints are checked, as it is necessary to know which specific cranes are assigned to each vessel. If any of these constraints are not met, the solution is partially destroyed and recreated. Thus, if a crane k is assigned to two vessels i and j such that vessel j berths after i departs and the travel and setup times of the crane moving from i to j are not respected, vessel j and all the vessels with berthing time equal to or greater than the departure time of j are removed from the solution. The algorithm is then applied over the resulting partial solution to include j, but now its berthing time is forced to be the berthing time it had in the previous solution plus the time needed to satisfy the travel and setup constraints. The remaining vessels are included using the algorithm in the order established in the original input list. The process is repeated until a feasible solution is found for the BACASP-S with crane travel and setup times.

7.2. Matheuristic local search

The local search procedure (LS) is a final attempt to refine the best solutions obtained by the BRKGA. This particular algorithm is a matheuristic, as it solves a subproblem using a solver running the proposed BACASP-S MILP.

Given an input solution, this procedure first constructs a list of vessels ordered by non-increasing score, which is calculated, for each vessel, as the sum of its cost and the cost of its adjacent vessels in the space–time diagram (i.e., the vessels whose rectangle is in contact with the rectangle representing the schedule of the vessel in question). Then, a vessel is selected from the list with a predefined probability, starting from the first element. If the size of the cluster is less than a predefined value, the vessel selected and the adjacent vessels are removed and reintroduced in the solution by solving the model for the BACASP-S with the variables corresponding to the remaining vessels fixed at the values they have in the solution, while the rest are not fixed. Otherwise, another vessel is selected. By limiting the number of adjacent vessels admitted we ensure that the model will be solved very fast. If the solution obtained is not the best solution found so far, another vessel is selected. Otherwise, the process is repeated over the new best solution and the procedure continues until the time limit is reached.

8. A multi-model and multi-heuristic exact approach

To solve the BACASP and BACASP-S, we can use a model for the problem that does not consider specific crane assignment, namely, the problem in which only a number of cranes is assigned to each vessel (BACAP). In past experiments we observed that the BACAP model (Correcher et al., 2019a) can be solved significantly faster than the BACASP model, and that it can provide good lower bounds for the latter in a very short time. Furthermore, the genetic algorithms proposed to solve all the three problems (BACAP, BACASP and BACASP-S) can provide the MILP solver with good feasible starting solutions. Taking all these remarks into consideration, we propose a solution scheme in which the problems are solved consecutively using the models and the metaheuristics to take full advantage of the lower and upper bounds obtained with them. The proposed algorithm is the following:

(1) Run the BRKGA+LS for the BACAP of Correcher and Alvarez-Valdes (2017) for τ_1 seconds.
(2) Solve the MILP for the BACAP of Correcher et al. (2019a) for τ_2 seconds, using the best solution obtained in step 1 as the starting solution.
(3) Run the BRKGA+LS for the BACASP of Correcher et al. (2019a) for τ_3 seconds.
(4) Solve the MILP for the BACASP proposed in Section 4 for τ_4 seconds, using the best solution obtained in step 3 as the starting solution.
(5) Run the BRKGA+LS for the BACASP-S proposed in Section 7 for τ_5 seconds.
(6) Solve the MILP for the BACASP-S proposed in Section 6 for τ_6 seconds, using the best solution obtained in step 5 as the starting solution.

Since the best lower bound achieved in a step is also a lower bound for the next step, it is possible to calculate a final optimality gap with the best lower and upper bounds obtained through the chain.

Note that it is not necessary to run all the steps of the algorithm, but just those we find useful empirically to solve the final problem, either the BACASP or the BACASP-S. In Section 9 we will evaluate different combinations of this chain of steps. This algorithm is summarized in Fig. 5.

9. Computational study

In this section, the models and the genetic algorithm are tested and evaluated in several computational experiments. In Section 9.1 we present and describe the sets of instances used throughout all the experiments. Then, in Section 9.2, we compare the computational results of the proposed continuous-time model with those of our previous discrete-time model. The effect of the valid inequalities is analyzed in Section 9.3, and the BACASP-S model is evaluated and discussed in Section 9.4. Next, we assess the multimodel multiheuristic approach in Section 9.5, and finally we study the effect of some relevant problem parameters on the computation effort required by the solver in Section 9.6.

The solution methods were implemented in C++ using GCC–G++ 9.4 and CPLEX 12.10, limiting the size of the search tree to 30 GiB. The compilation was performed specifying the parameters –O2 and -fopenmp in order to generate an executable file that uses the OpenMP parallelization library. The experiments were run on a cluster with 16 AMD Opteron 6344 at 2.6 GHz with 31.4 gibibytes of RAM, running the Ubuntu 20.04 operating system. The instances and results can be found at https://github.com/juanfdata/bacasp_setup.

9.1. Problem instances

We use a number of sets of instances generated by Correcher and Alvarez-Valdes (2017), following the schemes of Park and Kim (2003) and Meisel and Bierwirth (2009), extended to include the special features of the new BACASP-S. We now detail each set.

- *GenPK*, generated according to the criteria of Park and Kim (2003). All instances consider a time horizon of $T = 300$ h and a quay length of $L = 1200$ meters, discretized in units of 1 h and 10 m, respectively, with $Q = 11$ cranes. The set consists of 50 instances, 10 for each number of vessels $N \in \{20, 25, 30, 35, 40\}$. The vessel characteristics are generated from uniform distributions, the arrival time from $U[1, 170]$, the length from $U[15, 35]$, the number of crane-hours required from $U[10, 48]$, and the desired position on the quay from $U[1, 120]$. The maximum desired departure time is determined by applying the expression: $s_i = a_i + 1.5 \min_q\{u_{iq}\}$ (Meisel and Bierwirth, 2009). The processing time for each number of cranes is calculated by dividing the number of crane-hours required for the vessel by the number of cranes and rounding up to the next integer. The cost coefficients are $C_i^w = 1000$, $C_i^d = 2000$, $C_i^p = 200$, $\forall i \in V$. Additionally, crane setup time $\beta = 6$ minutes and crane speed $\alpha = 40$ meters per minute.
- *GenMB*, generated according to the criteria of Meisel and Bierwirth (2009). The time horizon is $T = 210$ h, the quay length is $L = 1000$ meters, discretized as before, with $Q = 10$ cranes. There are 50 instances, 10 for each number of vessels $N \in \{20, 30, 40, 50, 60\}$. The vessels are divided into three types: Feeder, Medium, and Jumbo, and each instance has 60%, 30%, and 10% of vessels of each type. Arrival times are taken from $U[1, 168]$, corresponding to one week, although the time horizon is longer to avoid unfeasible instances. Lengths, crane-hours, and minimum and maximum number of cranes are generated according to Table 1, taken from Meisel and Bierwirth (2009). The desired position of each vessel i on the quay is generated from $U[1, L+1-l_i]$. The processing time for each number of cranes, the maximum desired departure time, the cost coefficients and the crane setup time and speed are the same as in the *GenPK* set.
- *LargeMB*, consisting of 2430 instances of 30 vessels, generated in the same manner as *GenMB*. Specifically, the generation process was as follows. First, a total of 90 base instances were constructed with $T = 300$ h, $L = 500$ meters and the corresponding vessel data. Then, from each of these base instances, a total of 27

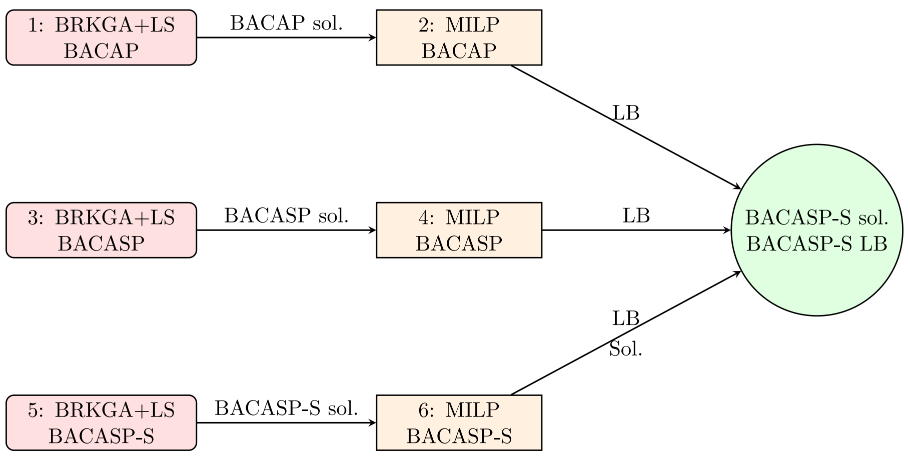

Fig. 5. Summary of the multi-model and multi-heuristic exact approach. LB stands for the best lower bound found.

Table 1
Specifications for the different classes of vessels in set *GenMB*.

Class	l_i	crane-hours	$q_i^{\min}$	$q_i^{\max}$
Feeder	$U[8,21]$	$U[5,15]$	1	2
Medium	$U[21,30]$	$U[15,50]$	2	4
Jumbo	$U[30,40]$	$U[50,65]$	4	6

instances were generated and included in the set, resulting in all the combinations of the following parameters: number of cranes $Q \in \{5, 10, 15\}$, crane speed and setup times (α in meters/minute, β in minutes) $\in \{(26.7, 12), (33.3, 9), (40, 6)\}$, and vessel processing times multiplied by 1, 1.5 or 2.

9.2. Continuous-time BACASP model vs. previous discrete-time BACASP model

The continuous-time BACASP model proposed here and our previous discrete-time BACASP model (Correcher et al., 2019a) were run over *GenMB* and *GenPK* instance sets using CPLEX with a time limit of 3600 s for each instance. The results are shown in Table 2. Instances are grouped by number of vessels. For each number of vessels, the table shows: *Avg.cost*, the average cost obtained on the instances for which at least one feasible solution was obtained; *Feasible*, the number of instances in each group for which a feasible solution was found; *Optimum*, the number of instances in each group for which an optimal solution was found (and proven optimal); *Avg. time*, the average time (in seconds) needed by the solver; *Av. gap*, the average MILP gap, in percentage; and *Max gap*, the maximum MILP gap found in the corresponding group of instances.

It can be observed that the new model reduces the computational times in both sets and optimally solves more instances with 50 vessels in *GenMB* and 35 and 40 vessels in *GenPK*. The average and maximum gaps are also reduced and, in contrast to the discrete-time model, feasible solutions are found in all the instances with 60 and 40 vessels, in *GenMB* and *GenPK*, respectively. The better performance of the continuous model may be due to its lower number of variables and constraints. The number of variables in the discrete model is $O(N^2 + NQT)$, in particular: $O(N^2)$ variables σ_{ij} and δ_{ij}, $i, j \in V$; $O(NQT)$ variables r_{ijt}, $i, j \in V, t \in \{1, \dots, T\}$; and $O(N)$ variables t_i, p_i and e_i, $i \in V$. However, the continuous model has only $O(N^2 + NQ)$ variables, as its r variables do not have the time as a subindex. Moreover, the number of constraints in the discrete model is $O(N^2 + Q^2T)$, while in the continuous model it is $O(N^2)$.

9.3. Evaluating the effect of the valid inequalities

Preliminary experiments were conducted to determine the best values for the parameters of the various families of valid inequalities, from which we finally obtained $h_\sigma = 5$, $h_\delta = 2$, $h_c = 2$, $\eta = 1$, $\bar{\delta} = 4$, $\bar{c} = \bar{\sigma} = 100000$. This parameter configuration was used in the remaining experiments.

Table 3 shows the effect of the valid inequalities described in Section 5 on the performance of the model in Section 4, for the two sets of *GenMB* and *GenPK* instances. Each inequality class has been tested separately, to assess its individual effect, while the last row shows the effect of all inequalities applied together. The solver was run for fifteen minutes on each instance. The gap reduction, defined as the difference between the gap without the valid inequalities and the gap with the valid inequalities, has been calculated only for the non-optimally solved instances. Conversely, the time reduction has been calculated for the optimally solved instances. In both cases, the table shows the mean absolute and percentage differences.

It can be seen that the effect of the valid inequalities is different in the two sets of instances. Concerning gap reductions, the effect is more uniform in *GenMB*, in which all inequalities have a positive (small) effect, whereas it is more irregular in *GenPK*, although the joint effect is larger. Running times are slightly reduced in *GenMB* and increased in *GenPK*, but it is worth noting that the inclusion of a large number of new inequalities does not significantly slow down the solution process for instances that are solved optimally. Therefore, all valid inequalities were included in the other computational tests in this section.

9.4. BACASP model vs. BACASP-S model

Table 4 compares the results obtained by the model without travel and setup times with the same model including them. Instances and results are grouped in the same way as in Table 2.

It can be seen how, when travel and setup times are added, the costs increase, the number of solved instances in one of the groups (50 vessels) decreases, the number of optimal solutions decreases in most of the groups, the average gaps increase and so do the maximum gaps. These increments get larger with problem size. For small problems, both models are rapidly solved to optimality, but as the number of vessels increases, the model with travel and setup times becomes much harder to solve. The example in Fig. 6 illustrates how crane travel and setup times impact the whole solution. While in subfigure (a) the BACASP is solved without considering crane travel and setup times, in subfigure (b) they are taken into account. As can be seen, cranes 1 to 3 must move from their positions for vessel 1 to their positions on vessel 3 and prepare for new operations. The time they take to do so affects

Table 2
Comparison between our previous MILP model for the BACASP and the new one.

GenMB				*GenPK*			
Vessels		Previous	New MILP	Vessels		Previous	New MILP
20	Avg. cost	13 300	13 300	20	Avg. cost	22 620	22 620
	Feasible	10	10		Feasible	10	10
	Optimum	10	10		Optimum	10	10
	Avg. time (s)	55.6	0.8		Avg. time (s)	284.2	1.2
	Avg. gap (%)	0	0		Avg. gap (%)	0	0
	Max. gap (%)	0	0		Max. gap (%)	0	0
30	Avg. cost	42 160	42 160	25	Avg. cost	31 920	31 920
	Feasible	10	10		Feasible	10	10
	Optimum	10	10		Optimum	10	10
	Avg. time (s)	526.7	40.4		Avg. time (s)	752.3	2.4
	Avg. gap (%)	0	0		Avg. gap (%)	0	0
	Max. gap (%)	0	0		Max. gap (%)	0	0
40	Avg. cost	56 960	56 620	30	Avg. cost	52 520	52 520
	Feasible	10	10		Feasible	10	10
	Optimum	9	10		Optimum	10	10
	Avg. time (s)	1625.6	123.3		Avg. time (s)	1556	25.9
	Avg. gap (%)	2.02	0		Avg. gap (%)	0	0
	Max. gap (%)	20.14	0		Max. gap (%)	0	0
50	Avg. cost	663 233	162 480	35	Avg. cost	1 556 622	86 260
	Feasible	6	10		Feasible	10	10
	Optimum	0	0		Optimum	5	7
	Avg. time (s)	3600	3600		Avg. time (s)	2806.7	1282.2
	Avg. gap (%)	67.49	38.84		Avg. gap (%)	25.26	6.39
	Max. gap (%)	95.03	62.52		Max. gap (%)	99.25	28.42
60	Avg. cost	1 669 000	337 880	40	Avg. cost	7 060 800	144 060
	Feasible	1	10		Feasible	1	10
	Optimum	0	0		Optimum	0	5
	Avg. time (s)	3600	3600		Avg. time (s)	3600	1957.2
	Avg. gap (%)	95.27	64.63		Avg. gap (%)	99.1	24.74
	Max. gap (%)	95.27	83.71		Max. gap (%)	99.1	73.24

Table 3
Effect of the valid inequalities on the performance of the model.

Inequality class (Section)	*GenMB*				*GenPK*			
	Gap red. (%)		Time red. (s)		Gap red. (%)		Time red. (s)	
Transitivity (5.1)	1.23	(3.97%)	7.03	(5.5%)	1.64	(9.0%)	7.28	(4.8%)
Use of cranes (5.2)	2.03	(4.12%)	25.80	(12.8%)	2.66	(23.8%)	0.75	(3.8%)
Sigma w (5.3)	0.74	(2.38%)	22.47	(4.5%)	−0.14	(7.0%)	1.95	(−8.1%)
Sum. sigma cranes (5.4)	1.48	(4.73%)	17.23	(4.9%)	0.10	(−8.5%)	1.15	(−0.6%)
Sum. sigma length (5.4)	2.40	(5.55%)	10.67	(−1.1%)	−0.28	(−2.6%)	8.63	(2.7%)
Coherence wr (5.5)	0.99	(1.69%)	8.63	(6.4%)	−0.70	(2.9%)	−0.75	(−3.9%)
Minimum separation (5.6)	0.02	(0.45%)	9.33	(14.2%)	−0.46	(11.1%)	3.80	(1.3%)
Ordered delta covers (5.7)	1.99	(1.35%)	2.87	(2.4%)	−0.60	(−5.3%)	1.50	(−2.4%)
Non-ordered delta covers (5.8)	1.10	(3.70%)	11.03	(12.3%)	−0.83	(−1.7%)	9.23	(8.3%)
All inequalities	3.61	(8.50%)	21.87	(10.9%)	8.13	(38.5%)	−5.33	(−14.1%)

the start and departure times of vessel 3 and thereby the operations of vessels 4 and 5. Now vessel 5 starts and departs later than in solution (a), incurring an unavoidable waiting cost; while vessel 4 keeps its berthing time, cannot use crane 3 and incurs a cost of deviating from its desired position to prevent the increase in the penalty for waiting before berthing, which in this case is greater than the deviation penalty. In a week-long planning horizon such as the ones used in the literature and in our experiments, this domino effect propagates to many other vessels and thus leads to quite different optimal solutions.

9.5. The multi-model multi-heuristic exact algorithm

We evaluated the solution scheme proposed in Section 8 over the set of instances of 50 vessels in *GenMB*, as this problem size is the first for which our MILP cannot solve the instances to optimality in 1 h, according to previous experiments. By using the exact solution approach, we expected to reduce optimality gaps and, hopefully, optimally solve some of the instances.

The total time limit of the process was set to 5 h (18000 s) and the time devoted to each genetic algorithm together with its local search was 200 s, of which the local search used 20 s. The remaining time was distributed equally to set the time limits for each of the models selected in each combination. For example, the full execution of the 6 steps comprised the following time limits: 200 s for the BACAP BRKGA+LS, 5800 s for the BACAP model, 200 s for the BACASP BRKGA+LS, 5800 s for the BACASP model, 200 s for the BACASP-S BRKGA+LS and the remaining time for the BACASP-S model. Note that when a model is solved before reaching its time limit, the final model in the chain has more time available.

The genetic algorithm and the local search were run with the best configuration of their parameters according to our previous paper (Correcher and Alvarez-Valdes, 2017). In the case of the genetic algorithm for the BACASP-S, the same configuration of the BACASP genetic algorithm was used. As for the models, they included all the valid inequalities selected in their final version.

Table 5 shows the results for the process applied to solve the BACASP. In particular, the best lower bound (LB) and upper bound (UB)

Table 4
Comparison between model without setup and model with setup.

GenMB				*GenPK*			
Vessels		No Setup	Setup	Vessels		No Setup	Setup
20	Avg. cost	13 300	13 379	20	Avg. cost	22 620	23 027
	Feasible	10	10		Feasible	10	10
	Optimum	10	10		Optimum	10	10
	Avg. time (s)	0.8	1.7		Avg. time (s)	1.2	3.2
	Avg. gap (%)	0	0		Avg. gap (%)	0	0
	Max. gap (%)	0	0		Max. gap (%)	0	0
30	Avg. cost	42 160	43 680	25	Avg. cost	31 920	32 761
	Feasible	10	10		Feasible	10	10
	Optimum	10	10		Optimum	10	10
	Avg. time (s)	40.4	328.5		Avg. time (s)	2.4	9.7
	Avg. gap (%)	0	0		Avg. gap (%)	0	0
	Max. gap (%)	0	0		Max. gap (%)	0	0
40	Avg. cost	56 620	59 450	30	Avg. cost	52 520	55 299
	Feasible	10	10		Feasible	10	10
	Optimum	10	9		Optimum	10	9
	Avg. time (s)	123.3	984.7		Avg. time (s)	25.9	529.6
	Avg. gap (%)	0	0.92		Avg. gap (%)	0	0.16
	Max. gap (%)	0	9.13		Max. gap (%)	0	1.59
50	Avg. cost	162 480	185 985	35	Avg. cost	86 260	91 635
	Feasible	10	10		Feasible	10	10
	Optimum	0	0		Optimum	7	6
	Avg. time (s)	3600	3600		Avg. time (s)	1282.2	1704.4
	Avg. gap (%)	38.83	54.58		Avg. gap (%)	6.39	12.24
	Max. gap (%)	62.52	72.05		Max. gap (%)	28.42	39.66
60	Avg. cost	337 880	967 270	40	Avg. cost	144 060	166 908
	Feasible	10	4		Feasible	10	10
	Optimum	0	0		Optimum	5	1
	Avg. time (s)	3600	3600		Avg. time (s)	1957.2	3594.4
	Avg. gap (%)	64.63	77.88		Avg. gap (%)	24.74	36.67
	Max. gap (%)	83.71	95.64		Max. gap (%)	73.24	76.58

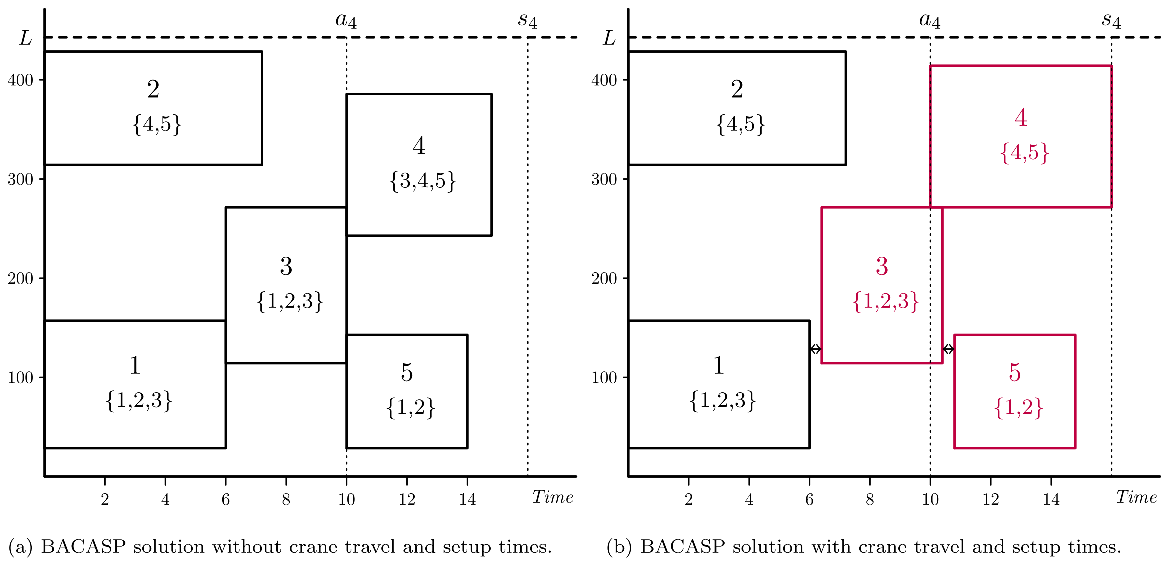

Fig. 6. BACASP solutions for the same vessels in a terminal with 5 quay cranes, with and without considering crane travel and setup times. The cranes assigned to the vessels are shown in brackets.

obtained during the process are reported for each of the configurations tested: step 4 (BACASP MILP), steps 3 and 4 (BACASP GA+MILP) and steps 1 to 4 (BACAP GA+MILP & BACASP GA+MILP). The gap is calculated as $100(UB - LB)/UB$, as is the gap provided by CPLEX. The results are rounded to units and those marked in bold indicate that they are equal to or better than the results obtained by the BACASP MILP alone (step 4).

First, we can see that the use of the BACASP GA to provide a starting solution to the solver (steps 3 and 4) improves the UB in instances 2, 3, 4 and 5, and the LB in instances 3, 4, 6 and 10. The overall outcome is a lower gap in instances 3, 4, 5, 6 and 9, but at the expense of increasing the gap in instances 7 and 8. In general, it seems that running steps 3 and 4 slightly improves on the results of the MILP alone.

As for the use of both BACAP and BACASP GA+MILP (steps 1 to 4), we can see that both lower bounds and gaps are drastically improved; indeed, the average gap reduction is 71%. However, this is attained at the expense of worsening the upper bound in some instances. This configuration of the algorithm appears to be useful to reduce the

Table 5
Solving the BACASP. Comparison between the execution of different configurations of the multimodel algorithm.

Instance	Step 4			Steps 3 and 4			Steps 1 to 4		
	UB	LB	Gap (%)	UB	LB	Gap (%)	UB	LB	Gap (%)
1	115 000	115 000	0	**115 000**	**115 000**	**0**	**115 000**	**115 000**	**0**
2	226 600	101 361	55	**224 000**	100 086	**55**	237 400	**193 400**	**19**
3	99 600	89 732	10	**99 600**	**92 009**	**8**	**99 600**	**92 247**	**7**
4	108 400	95 253	12	**108 000**	**98 865**	**8**	**108 000**	**104 027**	**4**
5	192 600	95 246	51	**183 800**	93 667	**49**	**183 800**	**164 200**	**11**
6	178 400	124 746	30	181 400	**129 612**	**29**	181 400	**160 200**	**12**
7	224 600	135 944	39	238 600	127 664	46	238 600	**209 400**	**12**
8	162 200	122 640	24	**162 200**	116 917	28	163 200	**154 400**	**5**
9	119 200	119 200	0	**119 200**	**119 200**	**0**	**119 200**	**119200**	**0**
10	181 600	103 766	43	**168 200**	**118 930**	**29**	**166 600**	**156800**	**6**
Avg.	160 820	110 289	26	160 000	111 195	25	161 280	146 887	8

Table 6
Solving the BACASP-S. Comparison between the execution of different configurations of the multimodel algorithm.

Instance	Step 6			Steps 5 and 6			Steps 1, 2, 5 and 6			Steps 1 to 6		
	UB	LB	Gap (%)	UB	LB	Gap (%)	UB	LB	Gap (%)	UB	LB	Gap (%)
1	120 501	95 979	20	**120 501**	**97 269**	**19**	**120 501**	**111 000**	**8**	128 304	**112 127**	**13**
2	407 419	84 486	76	**407 419**	**89 575**	78	**407 419**	**193 400**	**53**	436 735	**193 400**	**56**
3	106 024	77 092	32	**106 024**	76 640	**28**	134 302	**88 600**	34	114 232	**88 600**	**22**
4	115 409	79 059	31	**115 409**	**81 018**	**30**	139 233	**103 000**	**26**	**115 410**	**103 000**	**11**
5	361 233	82 247	62	**361 233**	79 380	78	385 604	**164 200**	**57**	**271 210**	**164 200**	**39**
6	290 137	86 731	65	**290 137**	**97 446**	66	544 902	**160 200**	71	579765	**160 200**	72
7	232 828	104 553	82	**232 828**	**127 063**	**45**	235 883	**209 400**	**11**	235 718	**209 400**	**11**
8	180 810	100 501	46	**180 810**	96 265	47	**168 650**	**154 400**	**8**	201 227	**154 400**	**23**
9	128 887	92 062	29	**128 887**	**101 674**	**21**	**128772**	**115 200**	**11**	128 997	**115 957**	**10**
10	178 489	99 205	58	**178 489**	96 443	**46**	185 044	**156 800**	**15**	186 707	**156 800**	**16**
Avg.	212 174	90 192	50	212 174	94 277	46	245 031	145 620	29	239 831	145 808	27

uncertainty on the interval where the optimal value is found (MILP gap) and, furthermore, leads us to conclude that the best feasible solutions obtained by the solver running the MILP alone are quite close to the optimal ones.

Similarly, Table 6 shows the results for the process applied to solve the BACASP-S considering different configurations: step 6 (BACASP-S MILP), steps 5 and 6 (BACASP-S GA+MILP), steps 1, 2, 5 and 6 (BACAP GA+MILP & BACASP-S GA+MILP), and steps 1 to 6 (BACAP GA+MILP & BACASP GA+MILP & BACASP-S GA+MILP). The results are analogous to the previous ones. The use of the BACASP-S GA+MILP slightly improves on the LBs and the gaps in some instances with respect to the MILP alone, while none of the UBs worsen. The configuration with steps 1, 2, 5 and 6 achieves considerably better LBs and an average gap reduction of 40%, although the UBs are worse in 6 instances. Finally, in the configuration running steps 1 to 6 the UBs are worse in almost all instances, but the LB obtained is much better in all of them, leading to an average gap reduction of 47%. In short, the conclusions that can be drawn in this case are similar to those on the BACASP configurations. Running the MILP with a GA start-up solution is slightly better than running it alone. The use of our previous BACAP and BACASP GAs and MILPs allows us to find much better LBs and leads us to conclude that the MILP alone achieves good-quality solutions.

9.6. *The effect of problem parameters on computational effort*

An important question regarding the new BACASP-S problem is how some key parameters affect the computational effort required to solve it. In previous works (Frojan et al., 2015) we concluded that in the BAP the computation effort required increases, *ceteris paribus*, with increasing number of vessels, vessel length and range of vessel estimated time of arrival. Now we aim at assessing the effect of other parameters that we consider relevant in the BACASP-S. To this end, we conducted an extensive experiment in which the MILP model was applied to the *LargeMB* set of instances, constructed with instances of 30 vessels, varying the number of cranes (5, 10 and 15), the vessel processing times (multiplied by 1, 1.5 and 2), and the crane speed (α in meters/minute) and setup time (β in minutes) in three different configurations: *Slow* ($\alpha = 26.7, \beta = 12$), *Medium* (33.3, 9) and *Fast* (40, 6).

The solver was run for 120 s on each instance and, from a total of 2430 instances, a feasible solution was found in all of them and an optimum in 951. The average gap in non-optimal instances was 43.9%. Tables 7, 8 and 9 show the mean gap of the instances grouped by pairs of problem parameters (number of cranes, processing time multiplier and crane speed and setup time) to assess how the gap changes as they vary alone and together. In Table 7 we can see that the mean gap decreases when the number of cranes in the instance increases. The differences observed are statistically significant (Welch's test $p < 0.0001$), especially in the instances with 5 cranes versus the group with 10 and 15 cranes, as the latter may be considered homogeneous (post-hoc t-test $p = 0.053$). As for the other parameters, the mean gap is not statistically different between the groups defined by the processing time multiplier (Welch's test $p = 0.669$) nor between the groups defined by crane speed (ANOVA test $p = 0.522$). Besides, the mean gap does not change due to the interaction of the pairs of parameters studied. In summary, we can conclude that the computational effort required to solve the BACASP-S increases with decreasing number of cranes, whereas the processing times and the crane travel and setup times do not affect it.

10. Conclusions

In this paper we have introduced a realistic variation of the berth allocation and quay crane assignment problem (BACASP) in which cranes need travel and setup times between serving two consecutive vessels. Including these times in berthing plans makes it possible to control

Table 7
Mean gap (%) in instances grouped by number of cranes and processing time multiplier.

#Cranes	Proc. time mult.			Total
	1.0	1.5	2.0	
5	54.2	58.4	59.9	57.5
10	13.0	11.8	12.0	12.3
15	11.3	9.6	10.6	10.5
Total	26.2	26.6	27.5	26.8

Table 8
Mean gap (%) in instances grouped by number of cranes and crane speed and setup time.

#Cranes	Crane setup speed			Total
	Slow	Medium	Fast	
5	58.7	57.1	56.7	57.5
10	13.2	12.5	11.0	12.3
15	10.7	10.9	9.9	10.5
Total	27.6	26.8	25.9	26.8

Table 9
Mean gap (%) in instances grouped by processing time multiplier and crane speed and setup time.

Proc. time mult.	Crane setup speed			Total
	Slow	Medium	Fast	
1.0	26.8	26.1	25.6	26.2
1.5	27.6	26.5	25.6	26.6
2.0	28.2	27.9	26.3	27.5
Total	27.6	26.8	25.9	26.8

the number of moves and to take into account their cumulative effect over time. This new problem is denoted as BACASP-S and adds more reliability to the resulting berth plan. To address it, we first propose a BACASP MILP model that, unlike traditional formulations, does not rely on the integrality of space and time variables. We strengthen this model with new families of valid inequalities and finally we extend it to tackle the BACASP-S. Both models prove to be efficient in solving instances of moderate size (40 vessels or less within a one-week time horizon), but alternative methods are needed to confront larger ones. This is why we also propose a genetic algorithm and an exact method, which can have up to 6 different phases and takes advantage of solutions to the BACAP problem, which is much easier to solve than the BACASP and BACASP-S.

Our experiments showed that the proposed BACASP model, with continuous variables for time and position, is more efficient than previous BACASP models (Türkoğullari et al., 2014; Correcher et al., 2019a) that rely on discretization of time and position. For the largest instances in the existing benchmarks, with 60 vessels in *GenMB* and 40 vessels in *GenPK*, the new model obtains feasible solutions for all of them and several more optimal solutions. The BACASP-S is a new problem, so the results obtained cannot be compared with previous approaches, but it can be observed that it is harder to solve than the BACASP, as expected, and the difficulty increases with problem size. The proposed valid inequalities do help the solver, although the improvement in computational efficiency thus attained is not enough for the MILPs to deal with large instances. The exact method combined with the genetic algorithm can find good quality solutions to instances of 50 vessels within a one-week time horizon and drastically reduces the optimality gaps. Finally, an extensive experiment allowed us to empirically prove that BACASP-S instances become more difficult to solve as the number of cranes decreases, whereas vessel processing time and crane speed and setup time do not affect the difficulty.

This topic allows for various directions in future research. We are planning to extend these models to accommodate more realistic assumptions, such as the variable-in-time assignment of cranes to vessels and the possibility of planning off-quay maintenance of specific cranes. Berthing and unberthing operations should also be tackled in future models together with crane setup times, as those operations depend on the availability of tugboats, the shape of the terminal layout and the presence of vessels in the quay, and so affect the berth plan. An additional open problem that may be studied in the future is the integration of the BACASP-S and the quay crane task scheduling problem (QCSP).

CRediT authorship contribution statement

Juan F. Correcher: Conceptualization, Methodology, Software, Validation, Data curation, Writing – original draft, Writing – review & editing. **Federico Perea:** Conceptualization, Methodology, Validation, Writing – original draft, Writing – review & editing. **Ramon Alvarez-Valdes:** Conceptualization, Methodology, Validation, Writing – original draft, Writing – review & editing.

Data availability

I have shared the link to my data inside the manuscript.

Acknowledgments

This study has been partially supported by the Ministerio de Ciencia, Innovación y Universidades del Gobierno de España, as part of the projects "OPTEP-Optimización de operaciones en terminales portuarias" (RTI2018-094940-B-I00), cofinanced with the European Regional Development's fund (ERDF), and "Nuevas tendencias en optimización combinatoria para la gestión de recursos y datos" (PID2021-122344NB-I00), cofinanced by the European Union. The first author also acknowledges financial support by the Generalitat Valenciana under the project "Advanced Optimization in Logistics" (CIGE/2022/057). Also, the second author acknowledges financial support from the Agencia Estatal de Investigación (AEI) and the European Regional Development's fund (ERDF): PID2020-114594GB-C21; Regional Government of Andalusia: projects FEDERUS-1256951, AT21_00032, and P18-FR-1422; and Fundación BBVA: project NetmeetData (2019). Special thanks are due to the three anonymous referees for their valuable comments.

Appendix A. Summary of the input data

See Table 10.

Appendix B. Routing-based model

Here we present another way of modeling the BACASP-S, using ideas from vehicle routing models. For this purpose, we need the following new variables:

- $y_{kij} = 1$ if crane k handles vessel i immediately before processing vessel j (we say that i and j are consecutive for crane k)
- $x_{ik} = 1$ if vessel i is handled by crane k

With these variables, and the input data defined in Section 3.2, a VRP-based MILP model for the BACASP-S, denoted as BACASP-S-1, consists of:

Table 10
Notation used for BACASP and BACASP-S input.

V	Set of vessels, $V = \{1, \dots, n\}$. For modeling purposes, let $V^0 = V \cup \{0\}$, vessel 0 being a dummy vessel
K	Set of available cranes, $K = \{1, \dots, Q\}$, indexed by k, k' (referring to specific cranes) and q (referring to the number of cranes assigned to a vessel). Cranes are numbered from 1 to Q, increasing with their distance from the start of the quay. As the cranes can move along the quay but cannot cross each other, their relative position remains constant
L	Length of the quay.
l_i	Length of vessel i, $l_i \in [0, L]$. Includes safety clearance.
a_i	Arrival time of vessel i
C_i^w	Cost per unit of waiting time for berthing after the expected time of arrival of vessel i
s_i	Maximum desired departure time of vessel i
C_i^d	Cost per unit of time delay after the maximum desired departure time of vessel i
b_i	Desired position at the quay of vessel i, $b_i \in [0, L - l_i]$
C_i^p	Cost per unit of length away from the desired position at the quay of vessel i
$q_i^{\min}$	Minimum number of cranes to be assigned to vessel i
$q_i^{\max}$	Maximum number of cranes that can be assigned to vessel i, $q_i^{\min} \le q_i^{\max} \le Q$
K_i	Set of cranes that can be the first crane assigned to vessel i. As the cranes assigned to a vessel are consecutive, $K_i = \{1, \dots, Q - q_i^{\min} + 1\}$
Q_i	Set of all the numbers of cranes admitted for vessel i, $Q_i = \{q_i^{\min}, \dots, q_i^{\max}\}$
u_{iq}	Processing time of vessel i if q cranes are assigned to it, $q \in Q_i$. The processing time of a vessel can vary linearly with the number of cranes or can be related to the number of cranes by a more complex expression, considering crane interference. It may have a non-integer value
α_k	Speed of crane k. If cranes are homogeneous, $\alpha = \min_{k \in K}(\alpha_k)$
β_k	The setup time that crane k requires between serving two consecutive vessels. If cranes are homogeneous, $\beta = \max_{k \in K}(\beta_k), \forall k \in K$

min (1)

s.t.: (2), (3), (4), (5), (7), (8), (24), (25)

$$\sum_k x_{ik} = \sum_{q=q_i^{\min}}^{q_i^{\max}} q r_{iq}, \qquad \forall i \in V \quad (31)$$

$$\sum_{j \in V^0, j \ne i} y_{kij} = x_{ik}, \qquad \forall i \in V, k \in K \quad (32)$$

$$\sum_{i \in V^0, j \ne i} y_{kij} = x_{jk}, \qquad \forall j \in V, k \in K \quad (33)$$

$$\sum_{j \in V} y_{k0j} \le 1, \qquad \forall k \in K \quad (34)$$

$$t_j \ge t_i + \sum_{q=q_i^{\min}}^{q=q_i^{\max}} u_{iq} r_{iq} + \frac{1}{\alpha_k} f_{ij} + \beta_k + M(y_{kij} - 1), \quad \forall i, j \in V, i \ne j, \forall k \in K \quad (35)$$

$$t_j \ge t_i + \sum_{q=q_i^{\min}}^{q=q_i^{\max}} u_{iq} r_{iq} + M(\sigma_{ij} - 1), \qquad \forall i, j \in V, i \ne j \quad (36)$$

$$k x_{jk} - k' x_{ik'} \ge 1 - Q(\delta_{ji} + \sigma_{ij} + \sigma_{ji}) - Q(2 - x_{ik'} - x_{jk}),$$
$$\forall i, j \in V, i \ne j, \forall k, k' \in K, k \ne k' \quad (37)$$

Constraints (31) set exactly q variables x_{ik} to 1. Constraints (32) and (33) define the unique successor and predecessor (which could be the dummy vessel) of each vessel and crane, provided that the crane handles the vessel. Constraints (34) ensure that each crane has at most one first vessel to handle. As for constraints (35), they ensure that the processing of vessel j starts when all cranes that were processing previous vessels are available, and help to properly define variables y. In these constraints we take into account the time it takes for a crane to go from one position at the quay to another, and the time needed to get the crane ready between vessels. Similar constraints are (36), which help to properly define σ variables, establishing that, if $\sigma_{ij} = 1$, then vessel i leaves before vessel j arrives. Finally, constraints (37) avoid crane crossing. The constraint is activated when two conditions are met:

(1) $\delta_{ji} = \sigma_{ij} = \sigma_{ji} = 0$ which, by (7), implies that $\delta_{ij} = 1$ and vessel i is below vessel j,
(2) $x_{ik'} = x_{jk} = 1$, that is, when crane k' handles vessel i (the vessel that is below) and crane k handles vessel j (the crane that is above).

In this case the constraint ensures that $k > k'$ and therefore they do not have to cross at any time.

References

Abou Kasm, O., Diabat, A., Cheng, T., 2020. The integrated berth allocation, quay crane assignment and scheduling problem: mathematical formulations and a case study. Ann. Oper. Res. 291, 435–461.

Agra, A., Oliveira, M., 2018. MIP approaches for the integrated berth allocation and quay crane assignment and scheduling problem. European J. Oper. Res. 265 (1), 138–148.

Bierwirth, C., Meisel, F., 2010. A survey of berth allocation and quay crane scheduling problems in container terminals. European J. Oper. Res. 202 (3), 615–627.

Bierwirth, C., Meisel, F., 2015. A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. European J. Oper. Res. 244 (3), 675–689.

Carlo, H.J., Vis, I.F.A., Roodbergen, K.J., 2015. Seaside operations in container terminals: literature overview, trends, and research directions. Flex. Serv. Manuf. J. 27 (1), 224–262.

Chang, D., Jiang, Z., Yan, W., He, J., 2010. Integrating berth allocation and quay crane assignments. Transp. Res. Part E: Logist. Transp. Rev. 46 (6), 975–990.

Cheimanoff, N., Fontane, F., Kitri, M.N., Tchernev, N., 2022. Exact and heuristic methods for the integrated berth allocation and specific time-invariant quay crane assignment problems. Comput. Oper. Res. 141, 105695.

Correcher, J., Alvarez-Valdes, R., 2017. A biased random-key genetic algorithm for the time-invariant berth allocation and quay crane assignment problem. Expert Syst. Appl. 89, 112–128.

Correcher, J., Alvarez-Valdes, R., Tamarit, J., 2019a. New exact methods for the time-invariant berth allocation and quay crane assignment problem. European J. Oper. Res. 275, 80–92.

Correcher, J.F., Van den Bossche, T., Alvarez-Valdes, R., Vanden Berghe, G., 2019b. The berth allocation problem in terminals with irregular layouts. European J. Oper. Res. 272 (3), 1096–1108.

Frojan, P., Correcher, J.F., Alvarez-Valdes, R., Koulouris, G., Tamarit, J.M., 2015. The continuous berth allocation problem in a container terminal with multiple quays. Expert Syst. Appl. 42 (21), 7356–7366.

Grubisic, N., Krljan, T., Maglic, L., 2020. The optimization process for seaside operations at medium-sized container terminals with a multi-quay layout. J. Mar. Sci. Eng. 8, 891.

Hendriks, M.P.M., Lefeber, E., Udding, J.T., 2013. Simultaneous berth allocation and yard planning at tactical level. OR Spectrum 35, 441–456.

Hsu, H.-P., Chiang, T., Wang, C.-N., Fu, H.-P., Chou, C.-C., 2019. A hybrid GA with variable quay crane assignment for solving berth allocation problem and quay crane assignment problem simultaneously. Sustainability 11, 2018.

Ji, B., Tang, M., Wu, Z., Yu, S.S., Zhou, S., Fang, X., 2022. Hybrid rolling-horizon optimization for berth allocation and quay crane assignment with unscheduled vessels. Adv. Eng. Inform. 54, 101733.

Karam, A., Eltawil, A., 2016. Functional integration approach for the berth allocation, quay crane assignment and specific quay crane assignment problems. Comput. Ind. Eng. 102, 458–466.

Le, M., Wu, C., Zhang, H., 2012. An integrated optimization method to solve the berth-QC allocation problem. In: 2012 8th International Conference on Natural Computation. IEEE, pp. 753–757.

Li, M.W., Hong, W.C., Geng, J., Wang, J., 2017. Berth and quay crane coordinated scheduling using multiobjective chaos cloud particle swarm optimization algorithm. Neural Comput. Appl. 28, 3163–3182.

Li, F., Sheu, J.B., Gao, Z.Y., 2015. Solving the continuous berth allocation and specific quay crane assignment problems with quay crane coverage range. Transp. Sci. 49 (4), 968–989.

Liu, C., Zheng, L., Zhang, C., 2016. Behavior perception-based disruption models for berth allocation and quay crane assignment problems. Comput. Ind. Eng. 97, 258–275.

Malekahmadi, A., Alinaghian, M., Hejazi, S., Saidipour, M., 2020. Integrated continuous berth allocation and quay crane assignment and scheduling problem with time-dependent physical constraints in container terminals. Comput. Ind. Eng. 106672.

Meisel, F., Bierwirth, C., 2009. Heuristics for the integration of crane productivity in the berth allocation problem. Transp. Res. 45, 196–209.

Park, Y., Kim, K., 2003. A scheduling method for berth and quay cranes. OR Spectrum (25), 1–23.

Rodriguez-Molins, M., Salido, M.A., Barber, F., 2014b. Robust scheduling for berth allocation and quay crane assignment problem. Math. Probl. Eng. 2014, 834927.

Rodriguez-Molins, M., Salido, M.A., Barber, F., 2014s. A GRASP-based metaheuristic for the berth allocation problem and the quay crane assignment problem by managing vessel cargo holds. Appl. Intell. 40 (2), 273–290.

Salhi, A., Alsoufi, G., Yang, X., 2019. An evolutionary approach to a combined mixed integer programming model of seaside operations as arise in container ports. Ann. Oper. Res. 272, 69–98.

Shang, X.T., Cao, J.X., Ren, J., 2016. A robust optimization approach to the integrated berth allocation and quay crane assignment problem. Transp. Res. 94, 44–65.

Song, Y., Zhang, J., Liu, M., Chu, C., 2019. The berth allocation optimisation with the consideration of time-varying water depths. Int. J. Prod. Res. 57, 488–516.

Türkoğullari, Y.B., Taşkin, Z.C., Aras, N., Altinel, I.K., 2014. Optimal berth allocation and time-invariant quay crane assignment in container terminals. European J. Oper. Res. 235 (1), 88–101.

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

UNCTAD, 2022. Review of maritime transport 2022. In: United Nations Conference on Trade and Development. URL https://unctad.org/system/files/official-document/rmt2022_en.pdf.

Yang, C., Wang, X., Li, Z., 2012. An optimization approach for coupling problem of berth allocation and quay crane assignment in container terminal. Comput. Ind. Eng. 63 (1), 243–253.

Zampelli, S., Vergados, Y., Schaeren, R.V., Dullaert, W., Raa, B., 2013. The berth allocation and quay crane assignment problem using a CP approach. Lecture Notes in Comput. Sci. 8124, 880–896.

Zhang, C., Zheng, L., Zhang, Z., Shi, L., Armstrong, A.J., 2010. The allocation of berths and quay cranes by using a sub-gradient optimization technique. Comput. Ind. Eng. 58 (1), 40–50.

Zheng, F., Li, Y., Chu, F., Liu, M., Xu, Y., 2019. Integrated berth allocation and quay crane assignment with maintenance activities. Int. J. Prod. Res. 57, 3478–3503.