

Document downloaded from:

<http://hdl.handle.net/10251/52303>

This paper must be cited as:

Rodríguez Molins, M.; Ingolotti Hetter, LP.; Barber Sanchís, F.; Salido Gregorio, MA.; R. Sierra, M.; Puente, J. (2014). A genetic algorithm for robust berth allocation and quay crane assignment. *Progress in Artificial Intelligence*. 2(4):177-192. doi:10.1007/s13748-014-0056-3.



The final publication is available at

<http://link.springer.com/article/10.1007%2Fs13748-014-0056-3>

Copyright

---

# A Genetic Algorithm for Robust Berth Allocation and Quay Crane Assignment

Mario Rodriguez-Molins · Laura Ingolotti · Federico Barber · Miguel A. Salido ·  
María R. Sierra · Jorge Puente

**Abstract** Scheduling problems usually obtain the optimal solutions assuming that the environment is deterministic. However, actually the environment is dynamic and uncertain. Thus, the initial data could change and the initial schedule obtained might be unfeasible. To overcome this issue, a proactive approach is presented for scheduling problems without any previous knowledge about the incidences that can occur. In this paper, we consider the Berth Allocation Problem and the Quay Crane Assignment Problem as a representative example of scheduling problems where a typical objective is to minimize the service time. The robustness is introduced within this problem by means of buffer times that should be maximized in order to absorb possible incidences or breakdowns. Therefore, this problem becomes a multi-objective optimization problem with two opposite objectives: minimizing the total service time and maximizing the robustness or buffer times.

**Keywords** Scheduling · Planning · Robustness · Genetic Algorithms · Metaheuristics · Multi-Objective · Berthing Allocation · Quay Crane Assignment

---

Instituto de Automática e Informática Industrial, Universitat Politècnica de València (UPV). Camino de vera, s/n, 46022, Valencia, Spain  
Mario Rodriguez-Molins:

E-mail: mrodriguez@dsic.upv.es

Laura Ingolotti:

E-mail: lingolotti@dsic.upv.es

Federico Barber:

E-mail: fbarber@dsic.upv.es

Miguel A. Salido:

E-mail: msalido@dsic.upv.es

· Department of Computer Science, University of Oviedo (Spain)

María R. Sierra:

E-mail: sierramaria@uniovi.es

Jorge Puente:

E-mail: puente@uniovi.es

## 1 Introduction

A container terminal is an open system with three distinguishable areas (berth, container yard and landside areas) where there exist different complex optimization problems. For instance, berthing allocation or stowage planning problems are related to the berth area [17]; remarkshalling problem or transport optimization to the yard area; and, planning and scheduling hinterland operations related to trains and trucks to the landside area [18].

In a container terminal, once a vessel arrives at the port, it waits at the roadstead until it has permission to moor at the quay. The locations where mooring can take place are called berths. These are equipped with giant cranes, known as Quay Cranes (QC), that are used to load and unload containers which are transferred to and from the yard by a fleet of vehicles. These QCs are mounted on the same track (or rail) and, therefore they cannot pass each other. Two scheduling problems are considered in this paper, the Berth Allocation Problem (BAP) and the Quay Crane Assignment Problem (QCAP). The former is a well-known combinatorial optimization problem [14], which consists in assigning berthing positions and mooring times to incoming vessels. The QCAP deals with assigning a certain number of QCs to each berthed vessel such that all required movements of containers can be fulfilled [1].

Nowadays, the point of the view for the scheduling tasks has changed. Due to the fact that the real world is uncertain, imprecise and non-deterministic, there might be unknown information, breakdowns, incidences or changes, which become the initial plans or schedules obtained invalid. Thus, there are new trends to cope these aspects in the optimization techniques. This approach is being studied within the Berth Allocation and the Quay Crane Assignment problems. The uncertainty within these problems is due to the fact, among others, that engines of the Quay Cranes might present a fail-

ure or the movements per time unit are lower than expected. Due to the introduction of this new objective in the scheduling optimization problem, a multi-objective optimization approach needs to be taken into consideration.

The overall collaboration goal of our group at the Universidad Politécnic de Valencia (UPV) with the Valencia Port Foundation and the maritime container terminal MSC (Mediterranean Shipping Company S.A.) is to offer assistance and help in the planning and scheduling of tasks such as the allocation of spaces to outbound containers, to identify bottlenecks, to determine the consequences of changes, to provide support in the resolution of incidents, to provide alternative berthing plans, etc.

A comprehensive survey of BAP and QCAP is given in [1]. These problems have been mostly considered separately, with an interest mainly focused on BAP. An interesting approach for BAP is presented in [12] where a Simulated Annealing metaheuristic is compared with a mathematical model. However, there are some studies on the combined BAP+QCAP considering different characteristics of berths and cranes ([4], [11], [13], [16], [20]).

All the above studies do not take into consideration the uncertainty of the real world to obtain a robust scheduling. However, there are some studies that address the robust scheduling. In [8], a proactive approach for a discrete and dynamic model of the BAP is presented taking into account uncertainties in the arrival and handling times given their probability density functions. They propose a mixed integer programming model and a Genetic Algorithm (GA) for both problems: discrete berth allocation and QC assignment. The objective is to minimize the sum of expected value, the standard deviation of the service time and the tardiness of the incoming vessels. In [9], a robust optimization model for cyclic berthing for a continuous and dynamic BAP is studied by minimizing the maximal crane capacity over different arrival scenarios of a bounded uncertainty given by their arrival agreements.

Robust scheduling based on operational buffers has already been introduced as a proactive approach in the BAP. An approach to robust BAP is presented in [3]. They presented a feedback procedure for the BAP that iteratively improves the robustness of the initial schedule. This feedback procedure determines the time buffers for each vessel by means of adjustment rules.

In [19], another approach to the robust BAP is solved by a scheduling algorithm that integrates simulated annealing and branch-and-bound algorithms. This study introduces the robustness as an objective to maximize and an evaluation is carried out by varying the weights of these functions. The robustness is achieved by a constant buffer time assigned to all vessels.

In [21], the robust BAP problem is studied as a proactive strategy as a multi-objective optimization problem. They solved

this problem with the Squeaky Wheel Optimization (SWO) metaheuristic. The first objective is to minimize the late departures and the deviation from the desired position; and, the second objective is to maximize the robustness of the schedule. They tackle the robustness measure as a diminishing return, specifically the exponential function, to capture the decreasing marginal productivity of slacks in a berthing schedule.

However, some of the above approaches consider discrete berths or previous knowledge about the uncertainty in arrival or handling times to produce robust schedules, but usually this knowledge is not available. Furthermore, other approaches propose how to obtain robust schedules by means of operational buffer times, but these buffers are set independently of the handling time of the vessels.

Nevertheless, in this paper we present a formal mixed integer linear programming (MILP) for the combined dynamic and continuous BAP+QCAP that extends the model presented in [12]. In order to obtain optimized solutions in an efficient way, we develop a metaheuristic GA to obtain near-optimal solutions in competitive computational times (compared with mathematical solvers). Furthermore, we assume that there is no previous knowledge about incidences, so both the MILP and the GA approaches have been adapted to tackle robustness of the BAP+QCAP as a multi-objective optimization problem by using operational buffers within the schedule.

The rest of the paper is organized as follows. In the next section we give a thorough description of the BAP+QCAP. In Section 3, the robustness is formalized for the BAP+QCAP. The multi-objective optimization approach to manage the two objective functions is detailed in Section 4. In Section 5, the mathematical formulation is presented. In Section 6 we give the details of the GA designed for the BAP+QCAP. Section 7 reports the results of the experimental study. Finally, in Section 8 we give the main conclusions of this work.

## 2 Problem description

The objective in BAP+QCAP is to obtain a schedule of the incoming vessels with an optimum order of vessels mooring and a distribution of the docks and QCs for these vessels. Figure 1(b) shows an example of the graphical space-time representation of a berth plan with 6 vessels. Each rectangle represents a vessel, its handling time and length.

Our BAP+QCAP case is classified, according to the classification given by [1], as:

- *Spatial attribute: Continuous layout.* We assume that the quay is a continuous line, so there is no partitioning of the quay and the vessel can berth at arbitrary positions within the boundaries of the quay. It must be taken into

account that for a continuous layout, berth planning is more complicated than for a discrete layout, but it better utilizes the quay space [1].

- *Temporal attribute: Dynamic arrival.* Fixed arrival times are given for the vessels, so that vessels cannot berth before their expected arrival times.
- *Handling time attribute: Unknown in advance.* The handling time of a vessel depends on the number of assigned QCs (*QCAP*) and the moves required.
- *Performance measure: wait and handling times* The objective is to minimize the sum of the waiting ( $w_i$ ) and handling times ( $h_i$ ) of all vessels.

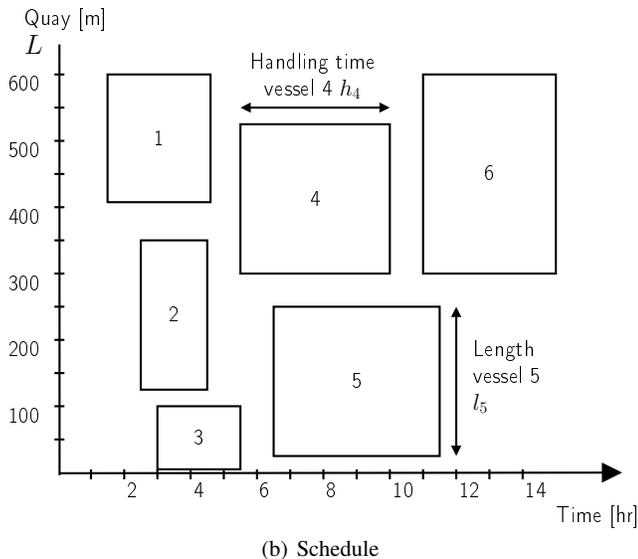
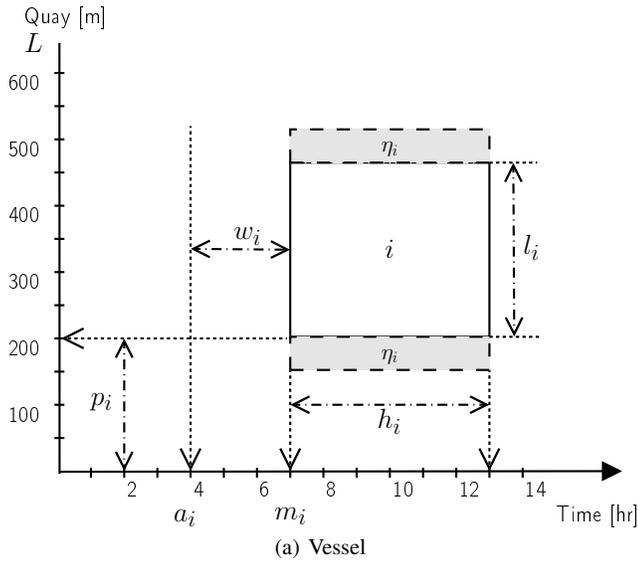


Fig. 1 Representation of the BAP+QCAP problem

Let  $V$  be the set of incoming vessels. Following, we introduce the notation used for each vessel  $i \in V$  (Figure 1(a)). The data variables are:

- *QC* : Available QCs in the container terminal. All QCs carry out the same number of movements per time unit (*movsQC*), given by the container terminal.
- *L* : Total length of the berth in the container terminal.
- *H* : Planning horizon for this schedule. It is calculated as the last departure when the FCFS policy is applied to the incoming vessels.
- $a_i$  : Arrival time of the vessel  $i$  at port.
- $c_i$  : Number of required movements to load and unload containers of  $i$ .
- $l_i$  : Vessel length.

The decision variables are:

- $m_i$  : Mooring time of  $i$ . Thus, waiting time ( $w_i$ ) of  $i$  is calculated as ( $w_i = m_i - a_i$ ).
- $p_i$  : Berthing position where  $i$  moors.
- $q_i$  : Number of assigned QCs to  $i$ .
- $u_{ik}$  : Indicates whether the QC  $k$  works (1) or not (0) on the vessel  $i$ .

The variables derived from the previous ones are:

- $h_i$  : Loading and unloading time at quay (handling time) of vessel  $i$ . This time depends on  $q_i$  and  $c_i$ , that is :  $\left(\frac{c_i}{q_i \text{ movsQC}}\right)$ .
- $t_{ik}$  : Working time of the QC  $k$  that is assigned to vessel  $i$ .
- $d_i$  : Departure time of vessel  $i$  ( $d_i = m_i + h_i$ ).
- $s_i, e_i$  : indexes for the first and last QC used in vessel  $i$ , respectively.

In this study, the following assumptions are considered:

- The number of QCs assigned to a vessel do not vary along the moored time. Once a QC starts a task in a vessel, it must complete it without any pause or shift (non-preemptive tasks). Thus, all QCs assigned to the same vessel have the same working time ( $t_{ik} = h_i, \forall k \in \text{QC}, u_{ik} = 1$ ).
- All the information related to the waiting vessels is known in advance (arrival, priority, moves and length).
- Every vessel has a draft that is lower than or equal to the draft of the quay.
- Movements of QCs along the quay as well as berthing and departure times of vessels are not considered since it supposes a constant penalty time for all vessels.
- Simultaneous berthing is allowed, subject to the length of the berth.

And the following constraints must be accomplished:

- Moored time must be at least the same that its arrival time ( $m_i \geq a_i$ ).

- It must be enough contiguous space at berth to moor a vessel of length ( $l_i$ ).
- There is a safety distance between two moored ships. We assume that each vessel has a 2.5% of this length at each side as a safety distance ( $\eta_i$ ) (Figure 1(a)). This safety distance is added to the length of each vessel  $i$ :  $l_i := l_i + 2\eta_i$ .
- There must be at least one QC to assign to each vessel. The maximum number of assigned QCs by vessel  $i$  ( $QC_i^+$ ) depends on its length, since a safety distance is required between two contiguous QCs (safeQC), and the maximum number of QCs that the container terminal allows per vessel (maxQC). Both parameters are given by the container terminal.

Our objective is to allocate all vessels according to several constraints minimizing the total weighted waiting and handling or processing time, known as the service time, for all vessels:

$$T_s = \sum_{i \in V} (w_i + h_i) \quad (1)$$

### 3 Robustness in BAP+QCAP

Container terminals are uncertain and non-deterministic systems in the same way than many other real-world systems. In the BAP+QCAP problem, the initial obtained schedules might become invalid due to different reasons: breakdowns in QCs, late arrivals of the vessels, bad weather, a lower ratio of movements per QC than expected, etc.

The robustness concept means that, given a schedule, this initial schedule remains feasible when incidences occur in its actual scenario.

The usual disruptions to be considered in BAP+QCAP are the followings:

- The arrival of a vessel is delayed from its expected arrival time ( $a_i$ ) assuming that the order remain unchanged.
- The handling time of a vessel is larger than its expected handling time ( $h_i$ ).

In case of a delay in the arrival of a vessel, it can be modeled as the departure time of this vessel is delayed by the same amount. Therefore, an incidence related to the arrival of a vessel is represented as a larger handling time for this vessel.

Given these two disruptions, we consider that a schedule is robust if a disruption in one vessel does not affect or alter the mooring times of the other vessels.

In BAP+QCAP, the robustness of a schedule might be guaranteed through two periods of time:

- The waiting time of vessels ( $w_i$ ). One vessel might be late  $w_i$  time units without disrupting the schedule of the others vessels.

- The buffer times of vessels ( $b_i$ ). The handling time of a vessel might be delayed  $b_i$  time units which correspond to the time between vessel  $i$  and each vessel  $j$  that shares the berth and moors just after vessel  $i$  (Figure 2).

The schedule could absorb delays or breakdowns that do not exceed the sum of those two periods ( $w_i + b_i$ ). Therefore, both times should be maximized in order to achieve the maximum robustness and ensure that there is no need to re-schedule the vessels. However, it should be kept in mind that the first objective of the BAP+QCAP is to minimize the total service time of the incoming vessels ( $w_i + h_i$ ). Therefore, we focus on maximizing only the second period of time, buffer times ( $b_i$ ), to obtain robust schedules.

The buffer time ( $b_i$ ) for each vessel  $i$  is obtained by Equation 4. We define  $\varphi_i$  as the set of vessels that succeed vessel  $i$  and occupy some berth space of vessel  $i$ ; and,  $\tau_{ij}$  is the difference between the departure time of vessel  $i$  ( $d_i$ ) and the mooring time of vessel  $j$  ( $m_j$ ). As example, Figure 2 shows the buffer times ( $b_i$ ) for each scheduled vessel as an empty rectangle.

$$\varphi_i = \{j \in V, m_j \geq d_i \wedge [p_i, p_i + l_i] \cap [p_j, p_j + l_j] \neq \emptyset\} \quad (2)$$

$$\tau_{ij} = m_j - d_i \quad \forall i \in V, \forall j \in \varphi_i \quad (3)$$

$$b_i = \begin{cases} +\infty & , |\varphi_i| = 0 \\ \min_{j \in \varphi_i} (\tau_{ij}) & , \text{otherwise} \end{cases} \quad \forall i \in V \quad (4)$$

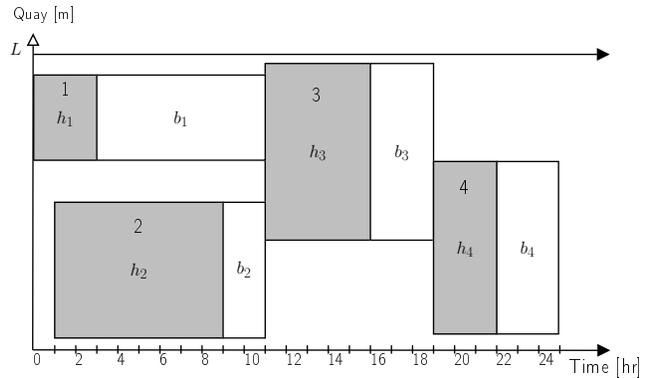


Fig. 2 Buffer times  $b_i$  given an example schedule.

Following the concept of decreasing productivity (diminishing returns) of the buffers presented in [21], in this study it is considered that the more handling time, the more likely to suffer incidences. Nevertheless, there is no need to assign a large buffer time to each vessel. For instance, in Figure 2, Vessel 1 would not need 8 time units of buffer time ( $b_1$ ) since its handling time is only 3 time units. It is not likely that this vessel would suffer a delay of that magnitude. However, Vessel 2, with a handling time of 8 time units, has only 2 time units of buffer time ( $b_2$ ). In this case, it is high

likely that this vessel would suffer some breakdown or delay and so it becomes invalid this schedule.

It is known that the late arrivals may be due to other factors such as the travel time from its immediately origin. To simplify this study, we only consider that the magnitude of the incidence is related to the handling time.

Thus, the robustness measure is related to the buffer time  $b_i$  and the average handling time  $h_i^*$  of each vessel  $i$ . Therefore, it is proposed the following objective function (Equation 6) as a measure of the robustness of a schedule.

$$r_i = \min\left(1, \frac{b_i}{h_i^*}\right), \quad \forall i \in V \quad (5)$$

$$R = \sum_{i \in V} r_i \quad (6)$$

where  $h_i^*$  indicates the handling time when its possible average number of QCs  $\left(\frac{1+QC_i^+}{2}\right)$  are assigned.

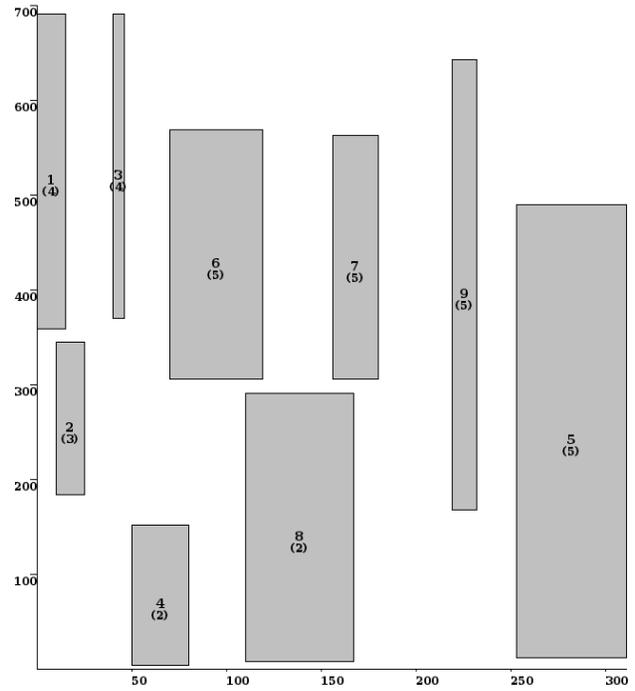
Let's see a simple example, Figure 3 shows two different schedules given the same set of incoming vessels. Each vessel is labeled with its vessel's ID and the assigned QC number in brackets. On the one hand, Figure 3(a) represents a robust schedule since almost any incidence over any vessel could be absorbed. On the other hand, Figure 3(b) shows a schedule with the optimal solution according to the objective function  $T_s$ . The latter schedule will be high likely unfeasible if any incidence occurs.

From Figure 3(a) and Figure 3(b), it is clear that there is a trade-off between optimality and robustness. However, getting a robust schedule is not achieved by extending an optimal schedule over the time, but getting an optimized allocation of vessels to achieve the maximum buffer size with proper distribution among all vessels. Note that the optimality is not the makespan of the schedule, but the total service time (waiting and handling times).

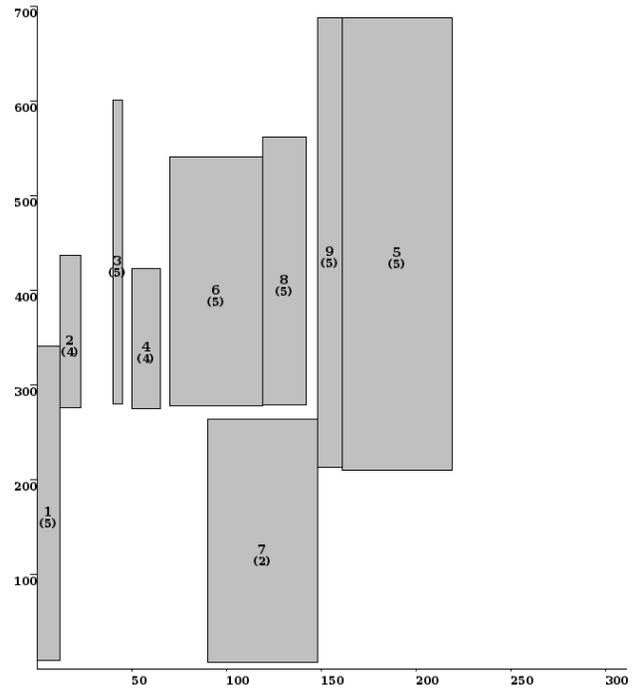
#### 4 Multi-objective approach for the BAP+QCAP

Solving the robust BAP+QCAP involves two optimization objectives: the service time ( $T_s$ ) and the robustness ( $R$ ). These objective functions must be normalized in order to apply the search process correctly. Equation 9 and Equation 10 define the normalized service time and the normalized robustness function, respectively.

Normalizing the service time function into the interval  $(0,1)$  implies to normalize both the waiting time (Equation 7) and the handling time (Equation 8). The handling time is just a linear normalization since the maximum ( $h_i^+$ ) and minimum ( $h_i^-$ ) times are known by assigning the minimum and the maximum number of QCs to vessel  $i$  ( $QC_i^+$ ). Normalizing the waiting time requires to determine a maximum total waiting time ( $W_F$ ). In this case, ( $W_F$ ) value is the total



(a) Robust schedule.



(b) Optimal schedule according to  $T_s$ .

**Fig. 3** Two possible schedules given the same incoming vessels.

waiting time of the incoming vessels when a FCFS policy is applied.

$$T_w = \frac{1}{W_F} \sum_{i \in V} (m_i - a_i) \quad (7)$$

$$T_h = \frac{1}{|V|} \sum_{i \in V} \left( \frac{h_i - h_i^-}{h_i^+ - h_i^-} \right) \quad (8)$$

$$\hat{T}_s = \frac{T_w + T_h}{2} \quad (9)$$

$$\hat{R} = \frac{R}{|V|} \quad (10)$$

Thereby, the objective function for the robust BAP+QCAP is to minimize the function  $F$  (Equation 11). The coefficient  $\lambda$  ( $0 \leq \lambda \leq 1$ ) assigns different weights to each component in order to establish an aggregate function.

$$F = \lambda \hat{T}_s - (1 - \lambda) \hat{R} \quad (11)$$

## 5 Mathematical formulation

In this section, the mathematical formulations for BAP+QCAP and the robust BAP+QCAP are presented. The first MILP model (Figure 4) solves the BAP+QCAP by minimizing the function given by the Equation 1, where  $M$  denotes a sufficiently large number, subject to the given constraints.

The given formulation expands the model presented in [12] by adding the needed constraints to take into consideration QCs. Thereby, the handling time of vessels depends on the number of QCs and these QCs cannot pass each other when are relocated.

In the proposed model, there are two auxiliary variables:  $z_{ij}^x$  is a decision variable that indicates if vessel  $i$  is located to the left of vessel  $j$  on the berth ( $z_{ij}^x = 1$ ); and,  $z_{ij}^y = 1$  indicates that vessel  $i$  is moored before vessel  $j$  in time (see constraint 32). Moreover, constraint 12 ensures that vessels must moor once they arrive at the terminal. Constraint 15 guarantees that a moored vessel does not exceed the length quay. Constraints 13 and 14 establish the waiting and departure times according to  $m_i$ . Constraints 16, 17, 18, 19 and 20 assign the number of QCs to the vessel  $i$ . Constraint 21 establishes the minimum handling time needed to load and unload their containers according to the number of assigned QCs. Constraint 22 assigns the handling time for vessel  $i$ . Constraint 23 ensures that QCs that are not assigned to vessel  $i$  have  $t_{ik} = 0$ . Constraint 24 forces all assigned QCs to vessel  $i$  working the same number of hours. Constraint 25 avoids that one QC is assigned to two different vessels at the same time. Constraints 26 and 27 force the QCs to be contiguously assigned (from  $s_i$  up to  $e_i$ ). Constraint 28 takes into account the safety distance between vessels. Constraint 29 avoids that one vessel uses a QC which should cross through the others QCs. Constraint 30 avoids that vessel  $j$

$$m_i \geq a_i \quad \forall i \in V \quad (12)$$

$$w_i = m_i - a_i \quad \forall i \in V \quad (13)$$

$$d_i = m_i - h_i \quad \forall i \in V \quad (14)$$

$$p_i + l_i \leq L \quad \forall i \in V \quad (15)$$

$$q_i = \sum_{k \in QC} u_{ik} \quad \forall i \in V \quad (16)$$

$$1 \leq q_i \leq QC_i^+ \quad \forall i \in V \quad (17)$$

$$1 \leq s_i, e_i \leq |QC| \quad \forall i \in V \quad (18)$$

$$s_i \geq e_i \quad \forall i \in V \quad (19)$$

$$q_i = e_i - s_i + 1 \quad \forall i \in V \quad (20)$$

$$\sum_{k \in QC} t_{ik} \text{ movsQC} \geq c_i \quad \forall i \in V \quad (21)$$

$$h_i = \max_{k \in QC} t_{ik} \quad \forall i \in V \quad (22)$$

$$t_{ik} - u_{ik}M \leq 0 \quad \forall i \in V, \forall k \in QC \quad (23)$$

$$h_i - M(1 - u_{ik}) - t_{ik} \leq 0 \quad \forall i \in V, \forall k \in QC \quad (24)$$

$$u_{ik} + u_{jk} + z_{ij}^x \leq 2 \quad \forall i, j \in V, \forall k \in QC \quad (25)$$

$$M(1 - u_{ik}) + (e_i - k) \geq 0 \quad \forall i \in V, \forall k \in QC \quad (26)$$

$$M(1 - u_{ik}) + (k - s_i) \geq 0 \quad \forall i \in V, \forall k \in QC \quad (27)$$

$$p_i + l_i \leq p_j + M(1 - z_{ij}^x) \quad \forall i, j \in V, i \neq j \quad (28)$$

$$e_i + 1 \leq s_j + M(1 - z_{ij}^x) \quad \forall i, j \in V, i \neq j \quad (29)$$

$$d_i \leq m_j + M(1 - z_{ij}^y) \quad \forall i, j \in V, i \neq j \quad (30)$$

$$z_{ij}^x + z_{ji}^x + z_{ij}^y + z_{ji}^y \geq 1 \quad \forall i, j \in V, i \neq j \quad (31)$$

$$z_{ij}^x, z_{ij}^y, u_{ik} \quad 0/1 \text{ integer} \quad \forall i, j \in V, i \neq j, \forall k \in QC \quad (32)$$

**Fig. 4** Mathematical model for the BAP+QCAP

moors while the previous vessel  $i$  is still at the quay. Finally, constraint 31 establishes the relationship between each pair of vessels.

$$\sum_{i \in V} (m_i - a_i) \leq W_F \quad (33)$$

$$z_{ij}^x = z_{ij}^x + z_{ji}^x + z_{ij}^y \quad \forall i, j \in V, i \neq j \quad (34)$$

$$\tau_{ij} = M \quad \forall i, j \in V, z_{ij}^x = 0 \wedge i \neq j \quad (35)$$

$$d_i + \tau_{ij} = m_j + M(1 - z_{ij}^y) \quad \forall i, j \in V, i \neq j \wedge z_{ij}^x = 1 \quad (36)$$

$$\tau_{ij} = M \quad \forall i, j \in V, z_{ij}^x = 2 \wedge i \neq j \quad (37)$$

$$b_i = \min \left( \min_{\substack{j \in V \\ i \neq j}} (\tau_{ij}), h_i^* \right) \quad \forall i \in V \quad (38)$$

$$r_i h_i^* = b_i \quad \forall i \in V \quad (39)$$

$$0 \leq z_{ij}^x \leq 2 \quad \forall i, j \in V, i \neq j \quad (40)$$

**Fig. 5** Mathematical model for the robust BAP+QCAP

The robust BAP+QCAP model minimizes the objective function presented in Equation 11. This model requires the 12-32 constraints presented above and the ones presented in Figure 5. The decision variable  $z_{ij}^x$  (see constraint 40) indi-

cates if a vessel  $j$  moors later than  $i$  and, at the same time, the vessel  $j$  intersects with the berth length occupied by vessel  $i$  ( $z'_{ij}$ ). Constraint 33 ensures that the total waiting time of the schedule does not exceed the maximum total waiting time ( $W_F$ ). Constraints 34-36 assign the time between the departure time of vessel  $i$  and the mooring time of vessel  $j$ . For those vessels  $j$  that  $z'_{ij} \neq 1$ , they are assigned  $M$  as a value. Constraints 38 and 39 set the value of the available buffer time after vessel  $i$  and its robustness value, respectively.

These mathematical models have been coded in IBM ILOG CPLEX Optimization Studio 12.5 as detailed in the Evaluation Section.

## 6 Genetic Algorithm

In this section, we describe two implementations of genetic algorithms to solve the problems: BAP+QCAP (Figure 4) and robust BAP+QCAP (Figure 5), respectively.

In the first problem, the goal is to minimize the service time  $T_s$  (Equation 1). Thus, the fitness  $F$  associated to each chromosome is the service time ( $T_s$ ) of that solution.

In the second one, the goal is to find a set of non-dominated solutions (henceforth efficient set  $M$ ) that is as close as to the optimal Pareto front. In this case, two objective functions are associated to each chromosome:  $\hat{T}_s$  and  $\hat{R}$  (see Equations 9 and 10, respectively), and the fitness  $F$  is computed according to the Equation 11. Below, we describe in detail both approaches.

### 6.1 A Genetic Algorithm to solve the BAP+QCAP

Algorithm 1 shows the structure of the genetic algorithm (GA) we have considered herein. The core of this algorithm is taken from [7, 6] and is quite similar to others generational GAs described in the literature ([10], [5] or [15]). In the first step, the initial population is generated and evaluated. Then, the GA iterates over a number of steps or generations. In each iteration, a new generation is built from the previous one by applying the genetic operators of selection, reproduction and replacement. These operators can be implemented in a variety of ways and, in principle, are independent from each other. However, in practice all of them should be chosen considering their effect on the remaining ones in order to get a successful overall algorithm.

The approach taken in this work is the following:

- In the selection phase all chromosomes are grouped into pairs, and then each one of these pairs is mated or not in accordance with a crossover probability ( $P_c$ ) to obtain two offspring.
- Each offspring, or parent if the parents were not mated, undergoes mutation in accordance with the mutation probability ( $P_m$ ).

- Finally, the replacement is carried out as a tournament selection (4:2) among each pair of parents and their offspring.

We consider that all chromosomes have the same number of genes  $N$  (number of vessels  $|V|$ ), and all populations have the same number of chromosomes,  $popsize$ .

---

#### Algorithm 1 The genetic algorithm

---

**Require:** A BAP+QCAP instance  $P$

**Ensure:** A mooring schedule for instance  $P$

1. Generate the initial population;
2. Evaluate the population;
- while** No termination criterion is satisfied **do**
  3. Select chromosomes from the current population;
  4. Apply the reproduction operators to the chromosomes selected at step 3 to generate new ones;
  5. Evaluate the chromosomes generated at step 4;
  6. Apply the replacement criterion to the set of chromosomes selected at step 3 together with the chromosomes generated at step 4;

**end while**

**return** The schedule from the best chromosome evaluated so far;

---

The coding schema is based on permutations of vessels, each one with a given number of QCs. So a gene is a pair  $(i, q_i)$ ,  $1 \leq q_i \leq QC_i^+$ , and a chromosome includes a gene like this for each one of the vessels. For example, for an instance with 5 vessels where the maximum number of QCs is 2, 3, 4, 3 and 2, respectively, two feasible chromosomes are the following ones:

$$chr_1: ((1\ 1)\ (2\ 1)\ (3\ 1)\ (4\ 2)\ (5\ 1))$$

$$chr_2: ((3\ 2)\ (1\ 2)\ (2\ 2)\ (5\ 2)\ (4\ 3))$$

Note that, the same vessel may have different number of QCs in each chromosome. In accordance with this encoding, a chromosome expresses the number of QCs that each vessel is assigned in the solution and an order for building the schedule.

The order of vessels in chromosomes is used as a dispatching rule. Hence, we use the following decoding algorithm: the genes are visited from left to right in the chromosome sequence. For each gene  $(i, q_i)$  the vessel  $i$  is scheduled at the earliest mooring time with  $q_i$  consecutive QCs available, so that none of the constraints is violated. If there are several positions available at the earliest time, that closest to one of the berth extremes is selected. Also, the QCs are chosen starting from the same extreme of the berth.

For chromosome mating we have considered a classical crossover operator such as Generalized Position Crossover (GPX) which is commonly used in permutation based encodings. This is a two points crossover operator which work as follows. Let us consider two parents like:

$$pt_1: ((1\ 1) | (2\ 1) (3\ 1) | (4\ 2) (5\ 1))$$

$$pt_2: ((3\ 2) | (1\ 2) (2\ 2) | (5\ 2) (4\ 3))$$

Symbols "|" represent crossover positions, 1 and 3 respectively in this example, which are selected at random for each mating. Then two offsprings are built taking the substrings between positions 1 and 3 in each parent and then filling the remaining positions with the genes representing the remaining vessels taken from the other parent keeping their relative order. So in this case the two offsprings are:

$$off_1: ((1\ 2) | (2\ 1) (3\ 1) | (5\ 2) (4\ 3))$$

$$off_2: ((3\ 1) | (1\ 2) (2\ 2) | (4\ 2) (5\ 1))$$

For mutation we have implemented an operator that shuffles a random substring of the chromosome and at the same time changes the number of QCs assigned to each one of the shuffled genes at random, provided that the number of QCs is kept in between the proper limits for the vessel.

The initial population is generated at random, i.e. a random order for the vessels is chosen and each vessel  $i$  is assigned a number of QCs chosen uniformly in  $[1, QC_i^+]$ . The termination condition is given in one of these three forms: (1) a number of generations, (2) a time limit or (3) a number of evaluations.

## 6.2 A Genetic Algorithm to solve the robust BAP+QCAP

In this section, we describe a GA that proposes a set of potentially efficient solutions  $M$  considering two optimization criteria: the service time and the robustness. The objective of this approach is to find a set  $M$  that is as closest as to the optimal Pareto front of the same instance. Thus, we have modified the Algorithm 1 in the following issues:

- The initial population is constructed in such a way a percentage of this population is at least, as good as the solution that the FCFS policy would propose.
- For each generation and for each chromosome, a pair of random weights that sum 1, are assigned to each objective function.
- Gene representation changes. The initial berthing position is added to the pair  $(i, q_i)$ . Thus, each gene is represented by  $(i, q_i, p_i)$ , where  $q_i$  and  $p_i$  are feasible values for vessel  $i$ .
- Before performing crossover, mutation and replacement, daughter and son inherits mom and dad weights, respectively. Therefore, the replacement operation compares daughter and mom, and son with dad. The two best chromosomes pass to the following generation.
- In the decoding process to evaluate each chromosome, the initial berthing position is given by each gene. Therefore, this value is not computed by the decoding process. A mooring time should be assigned to vessel  $i$  so that is

feasible, taking into account that vessel  $i$  must be berthed at quay from position  $p_i$  with  $q_i$  QCs without break any of the problem constraints.

- Once the population of the next generation is obtained, the efficient set  $M$  is updated.

Following, we describe in detail the issues that have been modified with respect to the single-objective GA.

**Representation** Structure of each chromosome is similar to the structure that is used in single-objective GA (Section 6.1). However, the gene has been modified to consider the initial berthing position as part of its data. Thus, this value is assigned by the GA instead of being computed by the decoding function. Structure of each gene and each chromosome are shown by Figure 6.

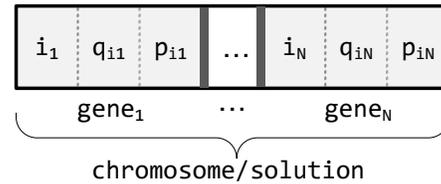


Fig. 6 Chromosome representation

It should be noted that each gene must be composed by feasible values with respect to vessel  $i$ . That is, according to the problem constraints, each vessel  $i$  can be assigned at most  $QC_i^+$  cranes. Therefore,  $1 \leq q_i \leq QC_i^+$ . Likewise, if the berth length is  $L$ , then  $\eta_i \leq p_i \leq L - l_i - \eta_i$ .

Algorithm 2 shows an outline of the multi-objective GA. Variable  $O$  indicates the number of objective functions that are considered by the algorithm.

---

### Algorithm 2 multi-objective genetic algorithm

---

**Require:** A BAP+QCAP instance  $P$ ,  $popsize$ ,  $N$ ,  $O$

**Ensure:**  $M$ : set of non-dominated solutions

$pop_0 \leftarrow$  generate\_the\_initial\_population( $popsize, N, O$ )  
 $t \leftarrow 0$

**while** No termination criterion is satisfied **do**

    assign\_weights( $pop_t, N, O$ )

$pop_{t+1} \leftarrow$  construct\_next\_generation( $popsize, N, O$ )

$t \leftarrow t + 1$

**end while**

**return** schedule for each element of efficient set  $M$

---

Below, we describe the main steps of this algorithm.

**Initialization** Construction of initial population is performed so that the service time of a percentage of the initial population (GA parameter) is at least as good as the solution provided by the policy FCFS. The other chromosomes are constructed by instantiating each gene in the following way:

- Vessel identifier ( $i$ ): an integer, between 1 and  $N$  is chosen randomly. Two genes of the same chromosome cannot have the same vessel identifier.
- Number of QCs: an integer, between 1 and  $QC_i^+$ , is chosen randomly.
- Initial position: an integer  $p_i$ , between  $\eta_i$  and  $L - l_i - \eta_i$ , is chosen randomly.

Each chromosome  $x$  in the initial population is considered to update the efficient set  $M$ . Following, we describe how is performed this operation [2].

**Update of efficient set  $M$**  Updating set  $M$  with a new solution  $x$  consists of:

- adding  $x$  to  $M$  if there is no other solution  $y \in M$  such that  $y$  dominates  $x$ ,
- removing from set  $M$  all solutions dominated by  $x$ .

**Generations** After the initial population has been generated, the algorithm constructs a new generation of population for each iteration, until some stop criterion is fulfilled.

Let  $pop_t$  be the population obtained in the generation  $t$ , Algorithm 3 shows how each generation is constructed.

**Crossover** Crossover operation is performed by using the same operator that is used by the single-objective GA (GPX). However, some differences there exist and these are explained in the following.

The crossover receives one pair of chromosomes ( $mom$  and  $dad$ ), which are in the current population  $pop_t$  and have been selected randomly. The objective of this operator is to construct two offspring chromosomes ( $daughter$  and  $son$ ). For that, each time the crossover operation is performed, the following steps are made:

1. Two cross points are chosen randomly,  $k_1$  and  $k_2$  ( $1 \leq k_1 < k_2 \leq N$ ). Figures 7(a) and 7(b) show an example, where  $N = 8$ ,  $k_1 = 3$  and  $k_2 = 6$ .

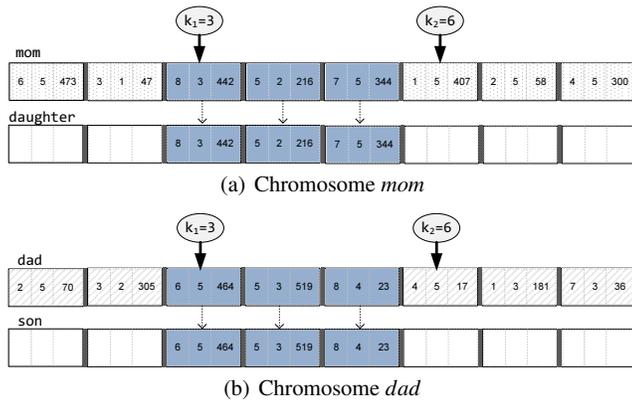


Fig. 7 Crossover points applied to the parents.

---

### Algorithm 3 construct\_next\_generation

---

**Require:**  $pop_t, popsize, N, O$

**Ensure:**  $pop_{t+1}$

$i \leftarrow 0$

assign\_weights( $pop_t, N, O$ )

$i \leftarrow 0$

**while**  $i < popsize$  **do**

$m \leftarrow$  random integer between 1 and  $popsize$ , which has not been selected yet in the current loop.

$d \leftarrow$  random integer between 1 and  $popsize$ , which has not been selected yet, in the current loop.

$p \leftarrow$  random real between 0 and 1

$daughter.weights \leftarrow chromosome_m.weights$

$son.weights \leftarrow chromosome_d.weights$

**if**  $p \leq$  crossover probability  $P_c$  **then**

crossover( $chromosome_m, chromosome_d, daughter, son$ )

**else**

$daughter \leftarrow chromosome_m$

$son \leftarrow chromosome_d$

**end if**

$p \leftarrow$  random real between 0 and 1

**if**  $p \leq$  mutation probability  $P_m$  **then**

mutation( $daughter$ )

**end if**

$p \leftarrow$  random real between 0 and 1

**if**  $p \leq$  mutation probability  $P_m$  **then**

mutation( $son$ )

**end if**

**if**  $daughter \neq chromosome_m$  **then**

evaluate  $daughter$

$M \leftarrow$  update\_efficient\_set( $daughter$ )

**end if**

**if**  $son \neq chromosome_d$  **then**

evaluate  $son$

$M \leftarrow$  update\_efficient\_set( $son$ )

**end if**

**if**  $daughter.fitness \leq chromosome_m.fitness$  **then**

$pop_{t+1} \leftarrow pop_{t+1} \cup \{daughter\}$

**else**

$pop_{t+1} \leftarrow pop_{t+1} \cup \{chromosome_m\}$

**end if**

**if**  $son.fitness \leq chromosome_d.fitness$  **then**

$pop_{t+1} \leftarrow pop_{t+1} \cup \{son\}$

**else**

$pop_{t+1} \leftarrow pop_{t+1} \cup \{chromosome_d\}$

$i \leftarrow i + 2$

**end if**

**end while**

---

2. Each gene in chromosome  $mom$  and  $dad$ , which is in position  $p$ ,  $k_1 \leq p < k_2$  is copied to the same position in chromosomes  $daughter$  and  $son$ , respectively.
3. Each gene in chromosome  $mom/dad$ , which is in position  $p$ ,  $1 \leq p < k_1$  is copied to the same position in chromosome  $son/daughter$ .
4. Each gene in chromosome  $mom/dad$ , which is in position  $p$ ,  $k_2 \leq p \leq N$  is copied to the same position in chromosome  $son/daughter$ .

Figure 8 shows the position of genes that will be copied from the chromosome  $mom$  to the offspring  $son$ . The same

procedure is made for the chromosome *dad* and the offspring *daughter*.

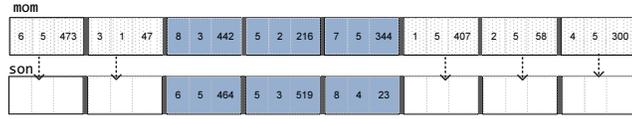


Fig. 8 Copy of genes to complete the offspring

In one chromosome cannot be two genes with the same vessel identifier. Therefore, if the vessel identifier in the gene that will be copied already exists in the offspring (*daughter/son*), the vessel identifier in the new chromosome must be modified.

The new value ( $i'$ ) is obtained from the first gene in the chromosome parent (*mom/dad*) that has a vessel identifier that does not exist in the offspring. Due to the original vessel identifier has been modified, it is necessary to ensure that the number of QCs and the initial position in the gene are feasible with respect to the new vessel identifier ( $i'$ ). If the number of QCs or the initial position in the gene is higher than the maximum value that is allowed for vessel ( $i'$ ), these values are also modified, by the maximum values corresponding to vessel ( $i'$ ).

Figure 9 shows an example where the identifier 6 must be changed by the identifier 2, because the vessel 6 already exists in the offspring. Once the identifier is modified by 2, the procedure must verify the number of QCs and initial position in the same gene are feasible, with respect to the vessel 2. In this case, the maximum initial position of vessel 2 is 400. Therefore, the value 473 in the same gene must be modified to 400.

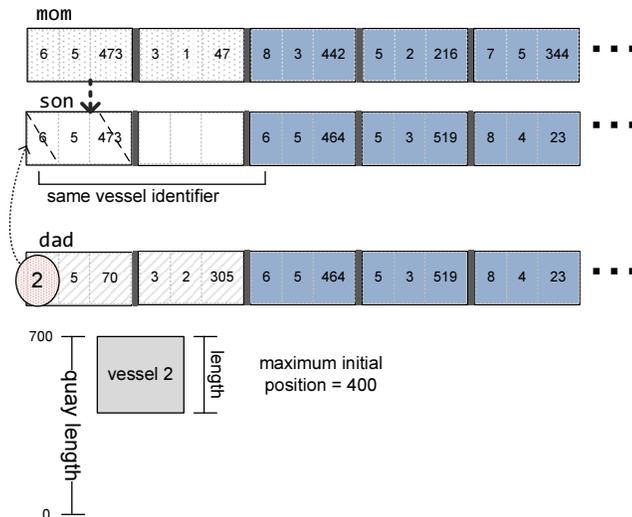


Fig. 9 Ensure offspring feasibility

**Mutation** Mutation operation is performed on one chromosome, following these steps:

1. Two positions ( $k_1$  and  $k_2$ ) of the chromosome are chosen randomly ( $1 \leq k_1 < k_2 \leq N$ ).
2. Positions of genes that are between  $k_1$  and  $k_2$ , both included, are modified randomly.
3. The number of QCs in each gene that is between  $k_1$  and  $k_2$ , both included, is modified by a feasible random value with respect to the vessel in the same gene.
4. The initial position in each gene that is between  $k_1$  and  $k_2$ , both included, is modified by a random value that is feasible with respect to the vessel that is in the same gene.

Figure 10 shows how the offspring *son*, which has been obtained after the crossover operation, is mutated. First, two values  $k_1 = 2$  and  $k_2 = 4$  are selected randomly. Then, all genes between both positions are shuffled. The gene 2 is moved to position 4, the gene 3 to position 1 and gene 4 to position 3. Finally, the number of QCs and the initial position of each gene in position  $p$ ,  $2 \leq p \leq 4$  are modified by selecting feasible random values for each one.

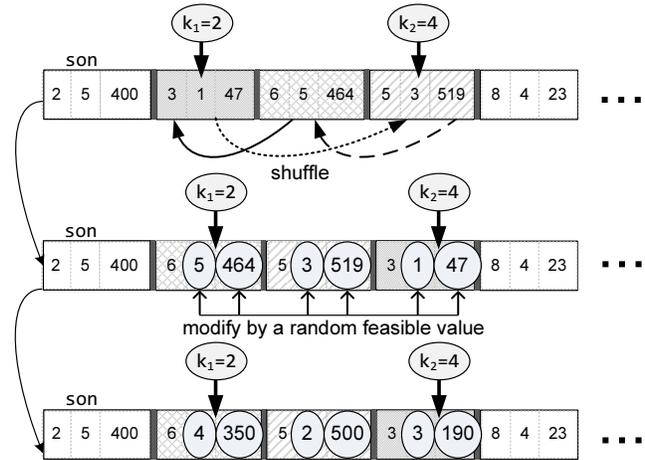


Fig. 10 An example of mutation operation

**Evaluation of one chromosome/solution** Function that evaluates each objective function of one chromosome and its fitness (evaluate in Algorithm 3), is the responsible of assigning a valid mooring time to each vessel. For that, it processes each vessel in the same order that it appears in the chromosome. For each gene  $g$  in the chromosome, this procedure assigns the mooring time most near to the arrival time of the vessel  $g.i$ , such that all the problem constraints are fulfilled, considering the initial position of the vessel  $i$  must be  $g.p_i$  and the number of QCs that must be available at this time is at least  $g.q_i$ .

Once a valid mooring time has been assigned to each gene  $g$ , the two normalized objective functions: service time

(Equation 9) and robustness (Equation 10) are computed; and, the fitness assigned to the chromosome is computed according to the Equation 11.

## 7 Evaluation

The experiments were performed in a corpus of 100 instances randomly generated following the suggestions of container terminal operator. Each one is composed of a queue from 5 to 20 vessels. These instances follow an exponential distribution for the inter-arrival times of the vessels ( $\beta = 20$ ). The number of required movements and length of vessels are uniformly generated in  $[100, 1000]$  and  $[100, 500]$ , respectively. In all cases, the berth length ( $L$ ) was fixed to 700 meters; the number of QCs was 7 (corresponding to a determined MSC berth line) and the maximum number of QCs per vessel was 5 ( $\text{maxQC}$ ); the safety distance between QCs ( $\text{safeQC}$ ) was 35 meters and the number of movements that QCs carry out was 2.5 ( $\text{movsQC}$ ) per time unit.

The two approaches developed in this paper, the GA and the MILP model, were coded using C++ and the IBM ILOG CPLEX Optimization Studio 12.5, respectively. They were solved on a Core 2 Quad 2.83Ghz with 4Gb RAM.

In the GA, the population size was 500. Mutation and crossover probabilities were  $P_m = 0.1$  and  $P_c = 0.8$ , respectively. Due to the stochastic nature of the GA process, each instance was solved 5 times and the results show the average obtained values.

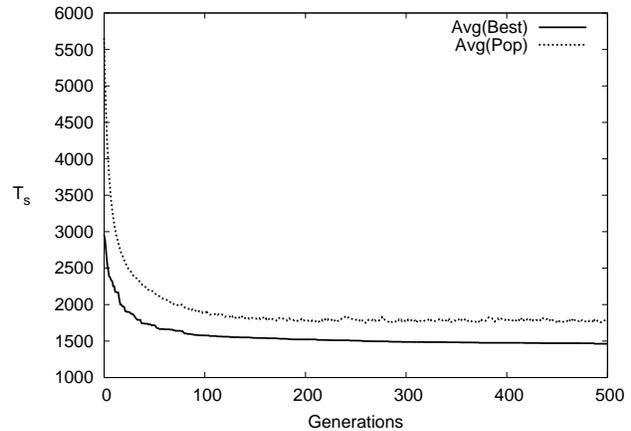
### 7.1 BAP+QCAP

Table 1 shows the average results for CPLEX and GA for each group of 100 instances with the same number of vessels (5 to 20). The timeout was 10 seconds. For CPLEX, the reported values are the average value of  $T_s$  for the solutions reached ( $\text{Avg}T_s$ ), the number of instances solved to optimality ( $\#\text{Opt}$ ) and the number of instances solved without certify optimality ( $\#\text{NOpt}$ ). The last two columns show the best and the average values of the solutions obtained by the GA in 5 runs, respectively. Obviously, in all cases, the objective function ( $T_s$ ) increases as the number of incoming vessels increases from 5 up to 20.

From these results, it can be observed that CPLEX was not able to reach any optimal solution by the given timeout in at least 25% of the instances with 8 vessels or more. In addition, it cannot get any optimal solution from 18 up to 20 vessels with this timeout. Regarding GA, all instances were solved and it can be observed that the average values were better than those from CPLEX, the differences being in direct ratio with the number of vessels. Here, it is important to remark that GA reached 2053 generations in 10 seconds. However, the GA was able to converge in lower times.

**Table 1** Comparison CPLEX with GA (timeout 10 secs)

V	CPLEX			GA	
	Avg $T_s$	#Opt	#NOpt	Best $T_s$	Avg $T_s$
5	267.08	100	0	267.08	267.08
6	339.25	97	3	339.21	339.21
7	417.52	88	12	416.70	416.80
8	501.41	74	26	497.70	498.05
9	585.94	58	42	575.57	576.42
10	690.91	38	62	667.50	669.57
11	797.15	24	76	759.07	763.13
12	927.85	18	82	854.45	861.35
13	1065.32	12	88	949.55	959.35
14	1212.86	6	94	1055.05	1066.19
15	1406.21	3	97	1158.88	1173.59
16	1610.21	2	98	1276.12	1296.09
17	1796.58	1	99	1379.01	1407.02
18	2101.27	0	100	1497.33	1526.94
19	2333.46	0	100	1621.19	1658.57
20	2603.36	0	100	1798.63	2090.61



**Fig. 11** Convergence of the Genetic Algorithm.

Figure 11 shows the GA convergence for one representative instance of 20 vessels indicating for each generation the average of the best value ( $\text{Avg}(\text{Best})$ ) and the average of the entire population ( $\text{Avg}(\text{Pop})$ ), so that near-optimal values were obtained after 150 generations, taking 0.73 seconds. Furthermore, Table 2 shows how the average  $T_s$  for 20 vessels decreases as more computation time was allowed. In this experiment, the timeout was set from 5 to 60 seconds. It can be observed, the GA approach does not require a large timeout (the improvement is lower than 1% beyond 5 seconds). Moreover, CPLEX was given a timeout up to 300 seconds and it obtained an average  $T_s$  (1876.27) greater than the one obtained by the GA after 5 seconds (1807.90).

### 7.2 Robust BAP+QCAP

Table 3 shows the results of the CPLEX and the GA for the robust BAP+QCAP for each group of 100 instances with

**Table 2** Average  $T_s$  for 20 vessels setting different timeouts.

Timeout (Secs.)	CPLEX	Best GA	Avg. GA
5	3187.01	1807.90	2097.67
10	2603.36	1798.63	2090.61
20	2406.16	1791.38	2088.08
40	2251.88	1783.68	2079.25
60	2126.56	1780.74	2073.32

the same number of vessels (5 to 20) and the same weights  $\lambda \in \{0, 0.2, 0.5, 0.8, 1\}$ . The timeout was set to 10 seconds. For CPLEX, the reported values were the average value of  $F$  for the solutions reached ( $AvgF$ ), the number of instances solved to optimality ( $\#Opt$ ), the number of instances solved without certify optimality ( $\#NOpt$ ) and the number of instances for which no solution was reached by the timeout ( $\#NSol$ ). Note that CPLEX was not able to reach the optimal solution even for instances with 5 vessels (e.g.  $\lambda = 0.2$ ). Furthermore, for instances with more than 8 vessels, CPLEX was not able to reach a feasible solution for all instances while robust GA always obtains feasible solutions for all instances. Note that robust GA was able to get better solutions (lower  $F$  values) for instances with more than 10 vessels.

The evaluation of the robust BAP+QCAP model presented in Section 3, as a multi-objective optimization problem, requires the study of the dominance of solutions as well as the Pareto front. Thus, for each instance, a set of non-dominated solutions after a timeout of 10 seconds was obtained. Figure 12 shows the Pareto front for two representative instances of 10 and 20 vessels, respectively. This Pareto front is an important tool for the decision maker (the container terminal operators), to assess the trade-off between  $\hat{T}_s$  and  $\hat{R}$  of the different schedules obtained according to their preferences. Note that the GA obtained a large set of solutions over the Pareto front. Moreover, in the case of 10 vessels, the solutions obtained by the GA dominates the ones obtained by CPLEX (Figure 12(a)). Figure 12(b) shows the Pareto front for a representative schedule of 20 vessels. In this case, CPLEX was not able to reach feasible solutions.

The performance (robustness) of the schedules obtained by the GA with a timeout of 10 seconds was evaluated by generating actual scenarios with some incidences in the actual handling time of the vessels. An incidence over a vessel  $i$  is modeled as a delay  $d$  in the handling time of this vessel  $i$ . An incidence is absorbed if there is enough buffer time behind the vessel  $i$  as to not alter the mooring time of the subsequent vessels. For each instance, the vessels that vary their handling times were uniformly chosen among all the vessels.

In this experiment, 100 instances of 20 vessels were evaluated. For each instance, four different schedules were chosen from the set of efficient solutions of the multi-objective GA according to their robustness: the one with the minimum

**Table 3** Average  $F$  values obtained by CPLEX with robust BAP+QCAP model.

V	$\lambda$	Robust CPLEX				Robust GA	
		Avg $F$	#Opt	#NOpt	#NSol	Best $F$	Avg. $F$
5	0	-0.723	100	0	0	-0.672	-0.674
	0.2	-0.481	99	1	0	-0.468	-0.475
	0.5	-0.141	97	3	0	-0.219	-0.223
	0.8	0.150	99	1	0	-0.301	-0.306
	1	0.324	100	0	0	0.481	0.357
6	0	-0.698	82	18	0	-0.670	-0.685
	0.2	-0.450	83	17	0	-0.415	-0.430
	0.5	-0.106	86	14	0	-0.069	-0.078
	0.8	0.180	90	10	0	-0.091	-0.093
	1	0.346	92	8	0	0.522	0.439
7	0	-0.684	53	47	0	-0.640	-0.653
	0.2	-0.438	48	52	0	-0.416	-0.435
	0.5	-0.091	57	43	0	-0.021	-0.030
	0.8	0.184	70	30	0	0.234	0.234
	1	0.342	83	17	0	0.478	0.426
8	0	-0.637	27	73	0	-0.632	-0.664
	0.2	-0.391	29	70	1	-0.379	-0.402
	0.5	-0.059	34	65	1	-0.020	-0.029
	0.8	0.217	42	57	1	0.284	0.273
	1	0.368	60	40	0	0.471	0.432
9	0	-0.612	15	80	5	-0.597	-0.631
	0.2	-0.353	14	81	5	-0.370	-0.399
	0.5	-0.019	16	81	3	-0.006	-0.021
	0.8	0.254	27	70	3	0.287	0.287
	1	0.404	40	58	2	0.447	0.415
10	0	-0.549	9	80	11	-0.585	-0.627
	0.2	-0.294	6	80	14	-0.357	-0.390
	0.5	0.019	12	74	14	-0.003	-0.018
	0.8	0.310	14	79	7	0.301	0.299
	1	0.454	24	68	8	0.443	0.416
15	0	-0.461	0	17	83	-0.508	-0.562
	0.2	-0.167	0	15	85	-0.297	-0.339
	0.5	0.103	0	17	83	0.020	0.008
	0.8	0.372	0	16	84	0.326	0.324
	1	0.546	1	18	82	0.388	0.367
20	0	-	0	0	100	-0.422	-0.462
	0.2	-0.035	0	1	99	-0.229	-0.263
	0.5	0.243	0	1	99	0.060	-0.047
	0.8	0.520	0	1	99	0.350	0.346
	1	-	0	0	100	0.381	0.365

robustness ( $R_m$ ), the one with the maximum robustness ( $R_M$ ) and two intermediate robust schedules ( $R_1$  and  $R_2$ ).

The incidences (delays,  $d$ ) introduced were randomly chosen from a range. This range varies from a minimum value (1) to a maximum value, which is related to the handling time ( $h_i$ ) of the vessel affected by the incidence (first column of Table 4). For each range, 100 incidences were uniformly created and applied to the four schedules ( $R_m$ ,  $R_1$ ,  $R_2$  and  $R_M$ ) of each instance.

Table 4 shows the percentage of incidences absorbed by each type of schedule. It can be observed that the more robust schedule, the more incidences absorbed. For instance, with delays  $d \in [1, 0.5h_i]$ , the  $R_m$  schedule only absorbed 25.75% of incidences in average, but the  $R_M$  schedule ab-

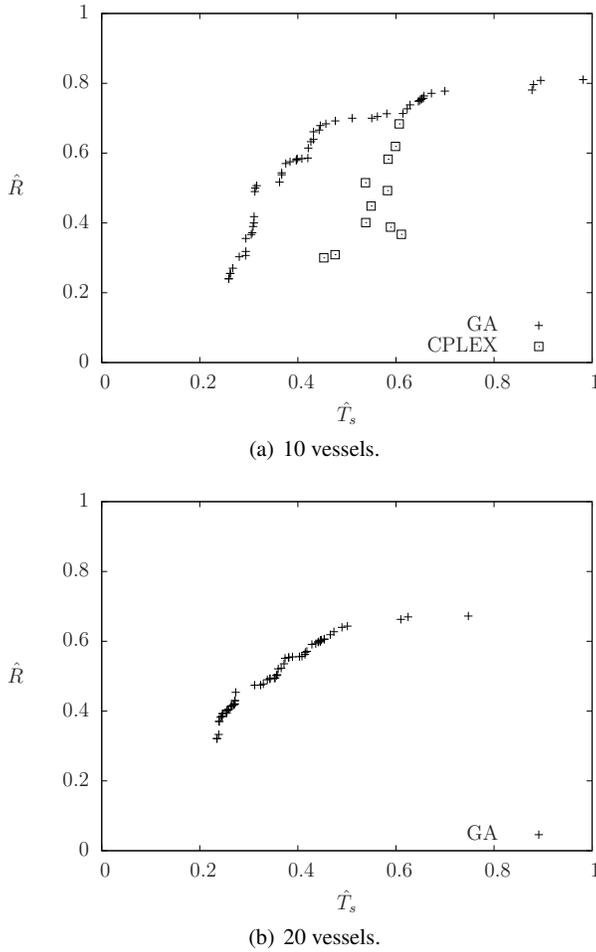


Fig. 12 Pareto Front for representative instances.

Table 4 Average of incidences absorbed in schedules of 20 vessels.

Range	$R_m$	$R_1$	$R_2$	$R_M$
$d \in [1, 0.2h_i]$	30.18	36.68	56.14	66.82
$d \in [1, 0.5h_i]$	25.75	32.82	54.65	65.63
$d \in [1, 0.8h_i]$	23.40	30.28	51.80	64.09
$d \in [1, 1.0h_i]$	22.65	29.95	50.94	62.41
$d \in [1, 1.2h_i]$	20.90	26.99	47.89	60.44

sorbed up to 65.63%. Note that as the delay became larger, fewer schedules can absorb the incidences. With delays between ranges of  $[1, 0.2h_i]$ , the  $R_M$  schedule can absorb 66.82% of incidences in average. However, with larger ranges, the incidences absorbed decreased down to 60.44% in average.

## 8 Conclusions

The competitiveness among container terminals causes the need to improve the efficiency of each one of the subprocesses that are performed within them. However, this efficiency is affected by the uncertainty of the environment

such as bad weather, breakdowns of the engines, delays, etc. This paper focuses on two of the main related problems, the Berth Allocation and Quay Crane Assignment Problems, in an integrated way. To this end, a mixed integer linear programming model and a Genetic Algorithm were developed in a dynamic and continuous BAP+QCAP. The MILP model was unable to get optimal solutions when a reasonable timeout is set or when the problem becomes harder (more than 10 vessels). Moreover, many of the instances were solved but without any guarantees of being the optimal ones since the timeout was reached. However, the GA approach was able to obtain near-optimal solutions in lower computational times and it also maintained a rapid convergence of the results even with large vessel queues. From these results, it is concluded the adequacy of a metaheuristic approach based on GA for solving the BAP+QCAP problem.

These two approaches, the MILP model and the GA, were extended as a multi-objective optimization problem to cope the uncertainty by obtaining robust schedules. The robustness of a schedule is related to the operational buffer times between the vessels. This new problem (robust BAP+QCAP) becomes harder and even instances of 5 vessels were not optimally solved by the MILP model in a reasonable time. Since the two objectives considered in this study, 'service time' and 'robustness', are opposite, there is no single optimal solution for these scheduling problems such that it is necessary to evaluate the trade-off between these objectives functions. Thereby, the multi-objective GA was able to obtain a set of efficient solutions and a study of the Pareto front was presented. Visualizing this Pareto front helps to the container terminal operators to decide which schedule is better depending on the actual state.

The robustness that is tackled in this study is related to the resource of the berth. In further works, since there is no previous knowledge about incidences, there should be analyzed how to distribute the operational buffer times among all the vessels equally. Furthermore, robustness related to shared QCs is going to be studied. When QCs are considered, a delay over a vessel may not prevent another vessel to moor. However, it might prevent all QCs allocated in subsequent vessels were employed. Thus, an incidence is propagated in these subsequent vessels. In this case, a further analysis is needed to re-allocate the QC correctly, which gives rise to a new re-scheduling problem.

**Acknowledgements** This research has been supported by the Spanish Government under research projects TIN2010-20976-C02-01 and TIN2010-20976-C02-02 (Min. de Ciencia e Innovación, Spain), the project PIRSES-GA-2011-294931 (FP7-PEOPLE-2011-IRSES) and the predoctoral FPU fellowship (AP2010-4405).

## References

1. Bierwirth, C., Meisel, F.: A survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research* **202**, 615–627 (2010)
2. Czyzak, P., Jaskiewicz, A.: Pareto simulated annealing—a metaheuristic technique for multiple-objective combinatorial optimization. *Journal of multi-criteria decision analysis* **7**, 34–47 (1998)
3. Du, Y., Xu, Y., Chen, Q.: A feedback procedure for robust berth allocation with stochastic vessel delays. In: *Intelligent Control and Automation (WCICA)*, 2010 8th World Congress on, pp. 2210–2215. IEEE (2010)
4. Giallombardo, G., Moccia, L., Salani, M., Vacca, I.: Modeling and solving the tactical berth allocation problem. *Transportation Research Part B: Methodological* **44**(2), 232 – 245 (2010)
5. Goldberg, D.: *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley (1985)
6. Gonzalez-Rodríguez, I., Vela, C., Puente, J.: A memetic approach to fuzzy job shop based on expectation model. In: *Proceedings of IEEE International Conference on Fuzzy Systems, FUZZ-IEEE2007*, pp. 692–697. IEEE (2007)
7. González-Rodríguez, I., Vela, C., Puente, J.: A genetic solution based on lexicographical goal programming for a multiobjective job shop with uncertainty. *Journal of Intelligent Manufacturing* **21**(1), 65–73 (2010)
8. Han, X.l., Lu, Z.q., Xi, L.f.: A proactive approach for simultaneous berth and quay crane scheduling problem with stochastic arrival and handling time. *European Journal of Operational Research* **207**(3), 1327–1340 (2010)
9. Hendriks, M., Laumanns, M., Lefeber, E., Udding, J.T.: Robust cyclic berth planning of container vessels. *OR spectrum* **32**(3), 501–517 (2010)
10. Holland, J.: *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control and artificial intelligence* (1975)
11. Imai, A., Chen, H., Nishimura, E., Papadimitriou, S.: The simultaneous berth and quay crane allocation problem. *Transportation Research Part E: Logistics and Transportation Review* **44**(5), 900–920 (2008)
12. Kim, K., Moon, K.: Berth scheduling by simulated annealing. *Transportation Research Part B: Methodological* **37**(6), 541–560 (2003)
13. Liang, C., Guo, J., Yang, Y.: Multi-objective hybrid genetic algorithm for quay crane dynamic assignment in berth allocation planning. *Journal of Intelligent Manufacturing* **22**, 471–479 (2011)
14. Lim, A.: The berth planning problem. *Operations Research Letters* **22**(2-3), 105–110 (1998). DOI 10.1016/S0167-6377(98)00010-8
15. Michalewicz, Z.: *Genetic algorithms+ data structures = Evolution Programs*, third, revised and extended. Springer (1996)
16. Park, Y., Kim, K.: A scheduling method for berth and quay cranes. *OR Spectrum* **25**(1), 1–23 (2003)
17. Salido, M.A., Rodríguez-Molins, M., Barber, F.: Integrated intelligent techniques for remarkshaling and berthing in maritime terminals. *Advanced Engineering Informatics* **25**(3), 435–451 (2011)
18. Stahlbock, R., Voß, S.: Operations research at container terminals: a literature update. *OR Spectrum* **30**(1), 1–52 (2008)
19. Xu, Y., Chen, Q., Quan, X.: Robust berth scheduling with uncertain vessel delay and handling time. *Annals of Operations Research* **192**(1), 123–140 (2012)
20. Zhang, C., Zheng, L., Zhang, Z., Shi, L., Armstrong, A.: The allocation of berths and quay cranes by using a sub-gradient optimization technique. *Computers & Industrial Engineering* **58**(1), 40–50 (2010)
21. Zhen, L., Chang, D.F.: A bi-objective model for robust berth allocation scheduling. *Computers & Industrial Engineering* **63**(1), 262–273 (2012)