Research Article

Robust Scheduling for Berth Allocation and Quay Crane Assignment Problem

M. Rodriguez-Molins, M. A. Salido, and F. Barber

Instituto de Automática e Informática Industrial, Universitat Politècnica de València, Camino de Vera, s/n, 46022 València, Spain

Correspondence should be addressed to M. A. Salido; msalido@dsic.upv.es

Received 22 July 2014; Revised 24 November 2014; Accepted 1 December 2014; Published 31 December 2014

Academic Editor: Andrzej Swiernia

Copyright © 2014 M. Rodriguez-Molins et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Decision makers must face the dynamism and uncertainty of real-world environments when they need to solve the scheduling problems. Different incidences or breakdowns, for example, initial data could change or some resources could become unavailable, may eventually cause the infeasibility of the obtained schedule. To overcome this issue, a robust model and a proactive approach are presented for scheduling problems without any previous knowledge about incidences. This paper is based on proportionally distributing operational buffers among the tasks. In this paper, we consider the berth allocation problem and the quay crane assignment problem as a representative example of scheduling problems. The dynamism and uncertainty are managed by assessing the robustness of the schedules. The robustness is introduced by means of operational buffer times to absorb those unknown incidences or breakdowns. Therefore, this problem becomes a multiobjective combinatorial optimization problem that aims to minimize the total service time, to maximize the buffer times, and to minimize the standard deviation of the buffer times. To this end, a mathematical model and a new hybrid multiobjective metaheuristic is presented and compared with two well-known multiobjective genetic algorithms: NSGAII and SPEA2+.

1. Introduction

Within a container terminal, operations related to move containers can be divided into four different subsystems (ship-to-shore, transfer, storage, and delivery/receipt) [1]. In each subsystem, terminal operators must deal with different complex optimization problems that can be overcome by using artificial intelligence techniques. For instance, berthing allocation or stowage planning problems are related to the ship-to-shore area [2–5], remarshalling problem and transport optimization [6] to the storage and transfer subsystems, respectively, and planning and scheduling hinterland operations related to trains and trucks to the delivery/receipt subsystem [7].

In this paper, we focus on two problems related to the ship-to-shore area, the berth allocation problem (BAP) and the quay crane assignment problem (QCAP). The former is a well-known combinatorial optimization problem [8], which consists in assigning berthing positions and mooring times to incoming vessels. The QCAP deals with assigning a certain number of quay cranes (QCs) to each moored vessel such that all required movements of containers can be fulfilled [9].

A comprehensive survey of BAP and QCAP is given in [9]. These problems have been mostly considered separately, with an interest mainly focused on BAP. An interesting approach for BAP is presented by Kim and Moon [10] 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 [11–15].

Most of the research in scheduling has been focused on deterministic and complete information, but they are usually not satisfied in real-world environments. Due to the fact that the real world is uncertain, imprecise, and nondeterministic, there might be unknown information, breakdowns, incidences, or changes, which make the initial plans or the obtained schedules become invalid. Thus, there are new trends to cope these aspects in the optimization techniques: proactive and reactive approaches [16]. In this paper, a proactive approach is studied within the berth allocation and
the quay crane assignment problems. The uncertainty within these problems is due to lower movements per time unit than expected or engine failures in quay cranes, among others. Due to the introduction of this new objective in the scheduling optimization problem, a multiobjective optimization approach needs to be taken into consideration.

All the above studies do not take into consideration the uncertainty of the real world to obtain a robust scheduling. Robustness is a measure of the performance characterization of an algorithm in the presence of uncertainties [17]. However, there are some studies that address the robust scheduling. In [18], 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. In [19], 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.

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 [20]. 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 [21], 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 be maximized 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 [22], the robust BAP problem is studied as a proactive strategy as a multiobjective 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; 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, most 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 (or processing) time of the vessels.

Overcoming the above approaches, hybrid metaheuristics for both single and multiobjective combinatorial optimization problems have received a significant interest from the research community [23, 24], and also they have been used in a wide range of real-world applications [25].

In this paper, we introduce a robust model to deal with limited incidences with no previous knowledge about them (Section 3) as well as a multiobjective approach to face this problem (Section 4). A formal mixed integer programming (MIP) is presented for the dynamic and continuous robust BAP + QCAP that extends the model presented in [10] (Section 5). Section 6 presents our proposed hybrid multiobjective genetic algorithm based on the scheme NSGAII [26] in order to obtain near-optimal solutions in an efficient way. This hybrid algorithm is used to solve the BAP + QCAP with a continuous quay and dynamic arrivals as well as to provide robust solutions by using operational buffers. As there is no previous knowledge about the incidences, these operational buffers are proportionally distributed among the tasks to be able to absorb as many incidences as possible. Thereby, a new objective function (standard deviation of robustness measures) was introduced to pursue this goal. This algorithm is compared with the mathematical model presented and two well-known multiobjective genetic algorithms: NSGAII and SPEA2+ [27] (Section 7). The development of the technique presented in this paper will provide the terminal operators with different robust berthing plans which are able to absorb limited incidences.

The overall collaboration goal of our group at the Universitat Politècnica de València (UPV) with the Valencia Port Foundation and the maritime container terminal 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, and to provide alternative berthing plans. Thus, the development of the technique presented in this paper will provide the terminal operators with different robust berthing plans which are able to absorb limited incidences.

2. Berthing Allocation and Quay Crane Assignment: BAP + QCAP

Let $V$ be a set of incoming vessels; BAP + QCAP consists in obtaining an optimal (or near-optimal) schedule of the vessels $V$ by assigning mooring times, berthing positions, and QCs to each vessel. Our BAP + QCAP model is classified, according to the classification given by Bierwirth and Meisel [9] as follows.

(i) **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 [9].

(ii) **Temporal Attribute: Dynamic Arrival.** Fixed arrival times are given for the vessels, so that vessels cannot berth before their expected arrival times.
The variables derived from the previous ones are

(i) $H_{ik}$: loading and unloading time at quay (handling time) of vessel $i$ using $k$ QCs ($1 \leq k \leq K$). This handling time depends on $c_i$ and is defined by

$$H_{ik} = \left[ \frac{c_i}{k + \text{movsQC}} \right] \quad \forall i \in V, \forall k = 1, \ldots, K,$$

(ii) $h_i$: required handling time of vessel $i$ when $q_i$ QCs are assigned to it. This value is set by means of $H_{ik}$,

(iii) $t_{ik}$: working time of the $k$th QC ($1 \leq k \leq K$) that is assigned to vessel $i$,

(iv) $d_i$: departure time of vessel $i$ ($d_i = m_i + h_i$),

(v) $s_i$ and $c_i$: indexes of the first and last QC assigned to vessel $i$, respectively.

In this study, the following assumptions are considered.

(i) All the information related to the waiting vessels is known in advance (arrival, priority, moves, and length).

(ii) Every vessel has a draft that is lower than or equal to the draft of the quay.

(iii) 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.

(iv) Simultaneous berthing is allowed, subject to the length of the berth.

Usually in container terminals, the number of QCs could vary during execution at the quay. This issue has been studied in Rodriguez-Molins et al. [5]. However, in this paper and without loss of generality, we study the robustness of the schedules assuming that the number of QCs assigned to one vessel does not vary along the moored time. Once a QC starts a task in a vessel, it must complete it without any pause or shift (nonpreemptive tasks). Thus, all QCs assigned to the same vessel $i$ have the same working time on the vessel ($t_{ik} = h_i, \forall k = 1, \ldots, K, u_{ik} = 1$).

The following constraints must be accomplished.

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

(ii) There is a safe distance between two moored ships. We assume that each vessel $i$ has a 2.5% of this length at each side as a safe distance ($\eta_i$) (Figure 1). This safe distance is added to the length of each vessel $i$: $l_i := \ell_i + 2\eta_i$.

(iii) There must be enough contiguous space at berth to moor a vessel $i$ of length ($l_i$).

(iv) There must be at least one QC assigned to each vessel. Furthermore, there is a maximum number of QCs that can be assigned to vessel $i$ ($QC_{ij}^+$). This value, ($QC_{ij}^+$), depends on the length of each vessel ($\ell_i$), since a safe distance is required between two contiguous
QCs (safeQC) and the maximum number of QCs that the container terminal allows per vessel (maxQC) (equation (2)). Both safeQC and maxQC parameters are given by the container terminal:

\[
QC^*_i = \min \left( \maxQC, \max \left( 1, \frac{\ell_i}{\text{safeQC}} \right) \right) \quad \forall i \in V. \quad (2)
\]

Our objective is to allocate all vessels according to several constraints minimizing the total waiting \((T_w)\) and handling or processing time \((T_h)\), known as the service time \((T_s)\), for all vessels:

\[
T_w = \sum_{i \in V} w_i, \quad (3)
\]

\[
T_h = \sum_{i \in V} h_i, \quad (4)
\]

\[
T_s = T_w + T_h, \quad (5)
\]

3. Robust BAP + QCAP Model

Uncertainty and nondeterminism of real-world environments may cause difficulties in the initial plans made by the decision makers. In container terminals, the initial obtained schedules for the BAP + QCAP problem might become invalid due to different reasons: breakdowns in QCs, late arrivals of the vessels, extreme weather events, a lower ratio of movements per QC than expected, and so forth.

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

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

(i) early or late arrival of a vessel \(i\) from its expected arrival time \(a_i\);

(ii) the handling time of a vessel \(i\) is larger than its expected handling time \(h_i\).

In this paper, we focus just on the disruptions affecting the handling time which eventually delay the departure time. In case of incidences related to late arrivals, they could also be modeled as delays in the handling time of the vessels which eventually also delay their departure time.

Definition 1. Given the possible 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.

The robustness of a schedule of BAP + QCAP might be guaranteed through two periods of time related to each vessel: waiting time of a vessel \(w_i\) and buffer time after the departure of each vessel \(b_i\) [28]. Without loss of generality, early arrivals are not taken into account since they only increase waiting times but they do not alter mooring times.

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 reschedule the involved 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, following the proposal given by Davenport et al. [29], we focus on maximizing only the second period of time, buffer times \((\sum b_i)\), to obtain robust schedules.

Let \(\varphi_i\) be the vessels that succeed vessel \(i\) and occupy some berth space of vessel \(i\) \((\varphi_i^p)\) or use any of QCs assigned to vessel \(i\) \((\varphi_i^q)\). The buffer time of vessel \(i\) \((b_i)\) is the minimum difference \((\tau_{ij})\) between the departure time of vessel \(i\) \((d_i)\) and the mooring time of vessel \(j\) \((m_j, j \in \varphi_i)\). In case there is no vessel in \(\varphi_i\), the maximum buffer time is assigned to \(b_i\) (an infinite value). Figure 2 shows an example of the buffer times \((b_i)\) assigned for each scheduled vessel as an empty rectangle:

\[
\varphi_i^p = \left\{ \forall j \in V, m_j \geq d_i \right\}
\]

\[
\varphi_i^q = \left\{ \forall j \in V, m_j \geq d_i \land \exists k, 1 \leq k \leq K \right\}
\]

\[
\tau_{ij} = m_j - d_i \quad \forall i \in V, j \in \varphi_i
\]

\[
b_i = \begin{cases} +\infty, & |\varphi_i| = 0 \\ \min \{\tau_{ij}\}, & \text{otherwise} \end{cases} \quad \forall i \in V.
\]

In this paper, we assume that the more handling time is, the more likely it is to suffer incidences. Therefore, in general, the larger the buffers are, the more robust the schedules are. Nevertheless, regarding the concept of decreasing productivity (or diminishing returns) presented in [22], there is a certain buffer size beyond which no more robustness is added to the schedule. Thereby, there is no need to assign large buffer times 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 highly likely that this vessel...
suffers some breakdown or delay and so it becomes invalid this schedule.

Furthermore, we consider that the magnitude of the incidence is related to the handling time of the vessel. Thus, the robustness measure of each vessel $i$ ($r_i \in [0,1]$) is related to the buffer time $b_i$ and the average handling time $h_i^*$ (equation (7)). It should be mentioned that other functions, for example, exponential function, could be adopted to define the robustness of each vessel:

\[
h_i^* = \frac{c_i}{\left(1 + QC_i^r\right)/2} \text{movesQC.} \tag{7}
\]

Given the robustness of each vessel, the robustness of a schedule $R \in [0,|V|]$ is defined by (9), where $\omega_i$ is a weighting factor ($\omega_i \geq 1$) which depends on historical data, if available. A $\omega_i = 1$ value represents that vessel $i$ used to finish its tasks as expected, and $\omega_i > 1$ value denotes that vessel $i$ used to be delayed:

\[
r_i = \min \left(1, \frac{b_i}{\omega_i h_i^*} \right), \quad \forall i \in V, \tag{8}
\]

\[
R = \sum_{i \in V} r_i \tag{9}
\]

In this paper, we address the BAP + QCAP problem without knowledge of the incidences; thus, the weighting factor is the same for all the vessels ($\omega_i = 1, \forall i \in V$).

**Example 2.** Figure 3 shows two different schedules given the same set of 9 incoming vessels. Each vessel is labeled with its vessel’s ID and the assigned QC number in brackets. Furthermore, the buffer time between vessels is also showed. On the one hand, Figure 3(a) represents a robust schedule since limited incidences 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 highly likely unfeasible if any incidence occurs.

Figures 3(a) and 3(b) are an example of the well-known trade-off between optimality and robustness. However, a robust schedule is not only achieved by extending an optimal schedule over the time. A robust schedule must also consider an optimized allocation of vessels to achieve the maximum sum of buffer sizes with a proper distribution among all vessels. Note that the optimality is not directly the makespan of the schedule but the total service time (waiting and handling times).

An important issue in this paper is that there is no available information about how likely the incidences or breakdowns occur. Therefore, it is interesting that these buffers are proportionally distributed among all the vessels. Thereby, a third objective is introduced into the model in order to improve the robustness of a schedule: minimizing the standard deviation (\(\sigma\)) of the robustness measures of all vessels ($r_i \forall i \in V$):

\[
\sigma = \sqrt{\frac{1}{|V|} \sum_{i \in V} \left( r_i - \bar{r} \right)^2}, \tag{10}
\]

where $\bar{r}$ is the average of the buffers of the schedule and $|V|$ is the number of incoming vessels.

Both measures presented above, robustness of a schedule ($R$) and standard deviation of these values (\(\sigma\)), represent...
the actual robustness of a schedule \( R \) to be maximized (see (11)). This measure guarantees the absorption of incidences that imply at most a delay of a \( \mathcal{R} \% \) of the weighted average handling time \( (\omega_i h_i^r) \):

\[
\mathcal{R} = R - \sigma.
\]  

(11)

**Example 3.** Figure 4 shows two different schedules with a similar robustness value \((R = 0.7)\) but different standard deviations, \( \sigma_1 = 0.17 \) and \( \sigma_2 = 0.45 \). With these values, the first schedule has an actual robustness value of \( \mathcal{R}_1 = 0.7 - 0.17 = 0.53 \). Thus, in average, this schedule guarantees that it could absorb incidences that imply at most a delay of the 53\% of the average handling time of the vessels. In contrast, the second schedule has an actual robustness value of \( \mathcal{R}_2 = 0.7 - 0.45 = 0.25 \); thus, it is able to absorb only incidences that imply at most a 25\% of the average handling time of the vessels.

Surico et al. [30] presented a close function to measure the robustness of a schedule \( \text{avg}(\omega_i) - \alpha \sigma(\omega_i), \forall i \in V \). \text{avg}(\omega_i) and \( \sigma(\omega_i) \) denote the average and the standard deviation of the waiting times, respectively; \( \alpha \) is a constant weighting factor that must be set. However, this measure does not reflect the relationship between the handling or processing time of the task and the buffer times. Thus, to our best knowledge, there is no other study which, considering the BAP + QCAP with a continuous quay and dynamic arrivals, tackles the robustness without any previous knowledge about the incidences.

Figure 4 shows two different schedules of 10 vessels with the same high value of the robustness measure. However, schedule of Figure 4(a) has a greater value for the standard deviation \( (\sigma) \) than schedule of Figure 4(b). Thereby, it is important to note that buffer times from schedule in Figure 4(a) are not equally distributed and this schedule will fail if an incidence which delays the departure time just 1 time unit over vessels 4 or 6 occurs or more than 3 time units over vessel 1. However, in the schedule in Figure 4(b), it is highly unlikely to be invalid since all vessels have enough buffer time after its schedule departure time.

### 4. Multiobjective Approach for the Robust BAP + QCAP

Three different objectives must be optimized to solve the robust BAP + QCAP: the service time \( (\bar{h}_i) \) (equation (5)), the robustness \( (R) \) (equation (9)), and the standard deviation of the robustness measures \( \sigma(R) \) (equation (10)). These objective functions must be normalized in order to apply the search process correctly.

Equation (14) shows how to normalize the service time objective into the interval \([0, 1]\) \((\bar{T}_i)\) and it implies to normalize both the waiting time \( \bar{T}_w \) (equation (12)) and the handling time \( \bar{T}_h \) (equation (13)). On the one hand, the handling time is just a linear normalization since the maximum \( (h_i^r) \) and minimum \( (\bar{h}_i^r) \) times are known by assigning the minimum \((1)\) and the maximum number of QCs to vessel \( i \) \((\text{QC}_i^r)\). On the other hand, normalizing the waiting time requires to determine a maximum total waiting time \((W_F)\). In this case, \( W_F \) value is the total waiting time of the incoming vessels when a first-come, first-served (FCFS) policy is applied, assigning 2 QCs to each vessel, and just one vessel is allowed in the berth at the same time (see, for example, Figure 5). The maximum total waiting time \((W_F)\) could also be obtained by assigning just one QC to each incoming vessel, but in that case, \( W_F \) value would be too large and all the normalized waiting times would be close to zero:

\[
\bar{T}_w = \frac{1}{W_{F,\text{rev}}} \sum_{i \in V} (m_i - a_i) \quad \bar{T}_w \in [0, 1],
\]  

(12)
Mathematical Problems in Engineering 7

1

(2)

(2) (2)

(2) (2) 2 3

4 5

50 100 150 200 250 300 350 400 450 500 ...

assigned by means of constraints (23)–(28) as follows:

∀𝑖∈𝑉 𝑞𝑖 =

|𝑉|∑

𝑘=1

𝑢𝑖𝑘, (23)

∀𝑖∈𝑉

|𝑉|∑

𝑘=1

𝑛𝑖𝑘 =1, (24)

Figure 5: Schedule generated to obtain the maximum value of wait-

(2)

ing time 𝑊 .

[36x772]Mathematical Problems in Engineering 7

[36x191]better off without making at least one criterion worse off as a solution in which it is impossible to make any one criterion optimal solutions with different trade-offs between their objective functions. Pareto efficiency, or Pareto optimality, is no single solution wherein all its objectives are simulta-

(2)

nously optimized. However, there may exist a set of Pareto optimal solutions with different trade-offs between their objective functions. Pareto efficiency, or Pareto optimality, is a solution in which it is impossible to make any one criterion better off without making at least one criterion worse off [31]. Pareto optimal solutions are defined by means of the dominance concept. Considering the robust BAP + QCAP, let 𝑥 and 𝑦 be two different solutions; 𝑥 dominates 𝑦 if at least one of the following conditions is satisfied:

|𝑉|新冠

(13)

(14)

Robustness objective function must also be normalized into the interval [0, 1] (̂𝑅) as defined by (15). The third objective, standard deviation of robustness measures, is already normalized due to the fact that 𝑟𝑖 values are already in the interval [0, 1] (equation (10)):

|𝑉|新冠

(15)

Thereby, the objective function for the robust BAP + QCAP is to minimize the function 𝐹 (equation (16)). Each coefficient 𝜆𝑖 (0 ≤ 𝜆𝑖 ≤ 1) assigns different weights to each component or objective function in order to establish an aggregate function:

意义

|𝑉|新冠

(16)

These coefficients 𝜆𝑖 are subject to ∑ 𝜆𝑖 = 1.

In a multiobjective optimization problem, usually there is no single solution wherein all its objectives are simulta-

naneously optimized. However, there may exist a set of Pareto optimal solutions with different trade-offs between their objective functions. Pareto efficiency, or Pareto optimality, is a solution in which it is impossible to make any one criterion better off without making at least one criterion worse off [31]. Pareto optimal solutions are defined by means of the dominance concept. Considering the robust BAP + QCAP, let 𝑥 and 𝑦 be two different solutions; 𝑥 dominates 𝑦 if at least one of the following conditions is satisfied:

意义

|𝑉|新冠

(17)

Given a set of feasible solutions 𝐷, a solution 𝑥 ∈ 𝐷 is Pareto optimal solution if it is nondominated by any other

solution 𝑥′ ∈ 𝐷. The Pareto optimal set is the set of all the solutions that are Pareto optimal solutions [31].

In general, generating the Pareto optimal set is expensive computationally and it is often impracticable. Therefore, algorithms try to find a good approximation of the Pareto optimal set. In this work, we refer that each approximation as Pareto front, which contains solutions that, although are nondominated among them, could be dominated by other solutions not found by our algorithms.

5. Mathematical Formulation

A mixed integer programming (MIP) model is presented to solve the robust BAP + QCAP. The objective function of this model is to minimize (16). This mathematical model is based on the model presented in [10, 28].

In the proposed model, 𝐷 denotes a sufficiently large number (as it is used in MIP). Furthermore, there are four auxiliary binary variables. 𝑧𝑖𝑗 is a decision variable that indicates if vessel 𝑖 is located to the left of vessel 𝑗 on the berth (𝑧𝑖𝑗 = 1); 𝑧𝑖𝑗 = 1 indicates that vessel 𝑖 is moored before vessel 𝑗 in time. The auxiliary variable 𝑢𝑖𝑘 indicates whether the QC 𝑘 works (1) or not (0) on vessel 𝑖; 𝑛𝑖𝑘 = 1 denotes that the number of QCs assigned to vessel 𝑖 is 𝑘:

|𝑉|新冠

(18)

In the proposed model, there are four auxiliary binary variables. 𝑧𝑖𝑗 is a decision variable that indicates if vessel 𝑖 is located to the left of vessel 𝑗 on the berth (𝑧𝑖𝑗 = 1); 𝑧𝑖𝑗 = 1 indicates that vessel 𝑖 is moored before vessel 𝑗 in time. The auxiliary variable 𝑢𝑖𝑘 indicates whether the QC 𝑘 works (1) or not (0) on vessel 𝑖; 𝑛𝑖𝑘 = 1 denotes that the number of QCs assigned to vessel 𝑖 is 𝑘.

The constraints of this mathematical model are detailed below. Constraint (19) ensures that vessels must moor after they arrive at the terminal:

意义

|𝑉|新冠

(19)

Constraints (20) and (21) establish the waiting and departure times according to 𝑚𝑖 and 𝑡𝑖:

意义

|𝑉|新冠

(20)

(21)

Constraint (22) guarantees that a moored vessel does not exceed the length quay:

意义

|𝑉|新冠

(22)

The number of QCs to the vessel 𝑖 are assigned by means of constraints (23)–(28) as follows:

意义

|𝑉|新冠

(23)

(24)
∀_{i∈V} \sum_{k=1}^{K} n_{ik} k = q_i, \quad (25)

∀_{i∈V} 1 \leq q_i \leq QC_i^+, \quad (26)

∀_{i∈V} 1 \leq s_i \leq e_i \leq K, \quad (27)

∀_{i∈V} q_i = e_i - s_i + 1. \quad (28)

Constraints (29)–(31) establish the minimum handling time needed to load and unload their containers according to the number of assigned QCs:

∀_{i∈V} \sum_{k=1}^{K} t_{ik} \text{ movs QC} \geq c_i \quad (29)

∀_{i∈V} \sum_{k=1}^{K} n_{ik} H_{ik} = h_i \quad (30)

∀_{i∈V} h_i = \max_{\forall k=1,...,K} t_{ik}. \quad (31)

Constraint (32) ensures that QCs that are not assigned to vessel \( i \) have \( t_{ik} = 0 \):

∀_{i∈V} \forall_{k=1,...,K} t_{ik} - Mu_{ik} \leq 0. \quad (32)

Constraint (33) forces all assigned QCs to vessel \( i \) working the same number of hours:

∀_{i∈V} \forall_{k=1,...,K} h_i - M (1 - u_{ik}) - t_{ik} \leq 0. \quad (33)

Constraint (34) avoids that one QC is assigned to two different vessels at the same time:

∀_{i,j∈V} \forall_{k=1,...,K} u_{ik} + u_{jk} + z_{ij}^x \leq 2. \quad (34)

Constraints (35) and (36) force the QCs to be contiguously assigned (from \( s_i \) up to \( e_i \)):

∀_{i∈V} \forall_{k=1,...,K} M (1 - u_{ik}) + (e_i - k) \geq 0, \quad (35)

∀_{i∈V} \forall_{k=1,...,K} M (1 - u_{ik}) + (k - s_i) \geq 0. \quad (36)

The safety distance between vessels is taken into account by constraint (37) as follows:

∀_{i,j∈V} p_i + l_i \leq p_j + M (1 - z_{ij}^x). \quad (37)

Constraint (38) avoids that one vessel uses a QC which should cross through the others QCs:

∀_{i,j∈V} e_j + 1 \leq s_j + M (1 - z_{ij}^x). \quad (38)

Constraint (39) avoids that vessel \( j \) moors while the previous vessel \( i \) is still at the quay:

∀_{i,j∈V} d_i \leq m_j + M (1 - z_{ij}^x). \quad (39)

Constraint (40) establishes the relationship between each pair of vessels avoiding overlaps:

∀_{i,j∈V} z_{ij}^x + z_{ji}^x + z_{ij}^y + z_{ji}^y \geq 1. \quad (40)

Constraint (41) ensures that the total waiting time of the schedule does not exceed the maximum total waiting time \( W_F \):

\sum_{i∈V} u_i \leq W_F. \quad (41)

Constraints (42)–(44) assign the time between the departure time of vessel \( i \) and the mooring time of vessel \( j \). For those vessels \( j \) so that \( z_{ij}^x \neq 1 \), they are assigned \( M \) as a value representing an unbounded time for the robustness:

∀_{i,j∈V} z_{ij}^x = z_{ij}^y + z_{ji}^x + z_{ji}^y, \quad (42)

∀_{i,j∈V} \forall_{x,y,z} \tau_{ij} = M, \quad (43)

∀_{i,j∈V} \forall_{x,y,z} d_i + \tau_{ij} = m_j + M \left(1 - z_{ij}^y\right). \quad (44)

Constraints (45) and (46) set the value of the available buffer time after vessel \( i \) and its robustness value, respectively:

∀_{i∈V} h_i = \min_{\forall j} \left(\min_{\forall x,y,z} \left(\min_{\forall x,y,z} \tau_{ij}, h_i^* \right)\right), \quad (45)

∀_{i∈V} \forall_{x,y,z} r_i h_i^* = h_i. \quad (46)

The decision variable \( z_{ij}^x \) (see constraint (47)) indicates if a vessel \( j \) moors later than \( i \) and, at the same time, the vessel \( j \) intersects with the berth length occupied by vessel \( i \) (\( z_{ij}^x \)):

∀_{i,j∈V} 0 \leq z_{ij}^x \leq 2 \quad (47)

6. Multiobjective Genetic Algorithms: MOGA + SA

Commonly approximations of the Pareto optimal sets of a multiobjective optimization problem are obtained by means of multiobjective evolutionary algorithms [31]. Furthermore, nowadays, metaheuristics are usually hybridized with other techniques or algorithms in order to enhance their effectiveness and performance [23, 24]. One of the most common forms of hybrid genetic algorithm involves incorporating local search to a canonical genetic algorithm. Genetic algorithm is used to perform global exploration among the population, and local search is used to perform local exploitation around the chromosomes. Because of the complementary properties of genetic algorithms and local search methods, the hybrid approach often outperforms either methods operating alone [32].

Thereby, a hybrid multiobjective genetic algorithm (MOGA) has been implemented in this paper. The NSGAII
schema has been extended with a multiobjective local search based on the multiobjective simulated annealing proposed by Bandyopadhyay et al. [33] (AMOSA), hereinafter named as MOGA + SA (see Algorithm 1). Moreover, two different schemes from the literature have been assessed NSGAII [26] and SPEA2+ [27].

The same chromosome structure is used in these three MOGAs. This chromosome has as many genes as incoming vessels $|V|$. Each gene consists of three values (see Figure 6): (1) the ID of the next vessel to dispatch ($i$); (2) the number of QCs assigned ($q_i$); (3) the buffer size after this vessel ($b_i$).

It should be noted that each gene must be composed of feasible values with respect to vessel $i$. That is, according to the problem constraints, each vessel $i$ can be assigned at most $QC_i^C$ QCs. Therefore, $1 \leq q_i \leq QC_i^C$. Likewise, if the berth length is $L$, then $b_i \leq L - d_i - \eta$. In the following subsections, genetic operators that are used by the implementations of NSGAII and SPEA2+ are described.

6.1. Decoding and Evaluation of One Chromosome/Solution. The structure of the chromosome, specifically the order of the vessels, 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 $(i, q_i, b_i)$, the vessel $i$ is scheduled at the earliest mooring time with $q_i$ consecutive QCs available, so that none of the constraints are violated. In case there are several positions available at the earliest mooring time, the one closest to the berth extremes is selected. After the departure of the vessel $i$ ($d_i$), it is ensured that there are $b_i$ unit times where no other vessel $j$ ($\forall j \in V, j \neq i$) uses the QCs assigned to vessel $i$ nor moors where vessel $i$ does $[p_i, p_i + l_i]$.

Once a valid mooring time ($m_i$) and initial position ($p_i$) have been assigned to each vessel $i$, the fitness of the chromosome (equation (16)) is obtained by computing each one of the objective functions: total service time ($\hat{T}_i$), robustness ($\hat{R}$), and standard deviation of the robustness ($\psi$).

6.2. Generation of Initial Population. Construction of initial population (generateInitialPopulation procedure) 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 FCFS policy. The other chromosomes (or solutions) are constructed by instantiating each gene in the following way.

(i) Vessel identifier ($i$): an integer, between 1 and $N$, is randomly chosen. Two genes of the same chromosome cannot have the same vessel identifier.

(ii) Number of QCs ($q_i$): an integer, between 1 and $QC_i^C$, is randomly chosen.

(iii) Buffer size ($b_i$): the initial buffer size is 0 for all genes of the initial population.

Once all chromosomes in the initial population have been instantiated, their fitness values are obtained as described in Section 6.1. Furthermore, the Pareto front $\mathcal{X}$ is updated considering all these chromosomes. Let $x$ be a chromosome (or solution); $x$ is added to the Pareto front $\mathcal{X}$ if there is no other solution $y \in \mathcal{X}$ such that $y$ dominates $x$. If $x$ is added to $\mathcal{X}$, then all solutions dominated by $x$ are removed from $\mathcal{X}$.

6.3. Evolution of One Population. In each iteration of the MOGA, a new population is built from the previous one (or the initial) by applying the genetic operators of selection, reproduction, and replacement. The proposed approach follows the scheme:

(i) selection: all chromosomes in the actual population are randomly grouped into pairs;

(ii) reproduction: (1) each one of these pairs is mated or not according to the crossover probability $P_c$ generating two offspring; (2) each offspring, or parent if the parents were not mated, undergoes mutation in accordance with the mutation probability $P_m$;

(iii) replacement: after evaluating the chromosomes previously generated, a tournament selection (4:2) is carried out among each pair of parents and their offspring as a replacement.

6.4. Crossover. The crossover operator receives one pair of chromosomes ($P_1$ and $P_2$), which are in the current population $pop$ and have been randomly selected. The objective of this operator is to construct two offspring chromosomes ($O_1$ and $O_2$). For that, each time the crossover operation is performed, the following steps are made.

(1) Two cross points are randomly chosen, $k_1$ and $k_2$ ($1 \leq k_1 < k_2 \leq N$).

(2) Each gene in chromosomes $P_1$ and $P_2$ which is in position $p$, $k_1 \leq p < k_2$, is copied to the same position in chromosomes $O_1$ and $O_2$, respectively.

(3) Each gene in chromosomes $P_1$ and $P_2$ which is in position $p$, $1 \leq p < k_1$, is copied to the same position in chromosomes $O_1$ and $O_2$, respectively.

(4) Each gene in chromosomes $P_1$ and $P_2$ which is in position $p$, $k_2 \leq p \leq N$, is copied to the same position in chromosomes $O_1$ and $O_2$, respectively.

Figure 7 is a graphical representation of the procedure that is used to perform the crossover operation, which is based on the technique generalized position crossover [34] that is commonly used in permutation based encodings.

In one chromosome there 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 ($O_1/O_2$)
Input: \( I \): instance of robust BAP + QCAP; 
\( \text{size} \): number of chromosomes; 
\( k \): number generations for local search 

Output: \( X \): set of nondominated schedules 

{Generate the parent population \( P_0 \)}
\[
P_0 \leftarrow \text{generateInitialPopulation(\text{size}, I)}
\]
{Generate the offspring population \( Q_0 \)}
\[
Q_0 \leftarrow \text{evolvePopulation}(P, \text{size})
\]
\( t \leftarrow 0 \)

while no termination criterion is satisfied do

\[
\text{unionSet} \leftarrow \text{makeUnionSet}(P_t, Q_t)
\]
\[
F \leftarrow \text{fastNondominatedSort(unionSet)}
\]
{Create the next parent population \( P_{t+1} \)}
\[
P_{t+1} \leftarrow 0
\]
\( i \leftarrow 0 \)

while \( i < |F| \land |P_{t+1}| + |F_i| < \text{size} \) do

\( P_{t+1} \leftarrow P_{t+1} \cup F_i \)
\( i \leftarrow i + 1 \)
end while

if \( |P_{t+1}| < \text{size} \) then

\( F_i \leftarrow \text{crowdingDistance}(F_i) \)
\( \text{sort}(F_i) \)
\( P_{t+1} \leftarrow P_{t+1} \cup \left\{ F_i[j] \right\} \)
\( j \leftarrow j + 1 \)
end if

{Perform the local search each \( k \) generations.}
\( t \leftarrow t + 1 \)
end while

return Schedule of each element of the Pareto front \( X' \)

\[
\text{Algorithm 1: Implementation of MOGA + SA.}
\]

during steps 2 or 3, a new gene must be selected from the chromosome parent \( (P_1/P_2) \).

Once the vessel identifier of the selected gene does not exist in the offspring, then the gene is copied to the offspring in the corresponding position.

6.5. Mutation. Mutation operation is performed on one chromosome, following these steps.

(1) Two positions \( (k_1 \text{ and } k_2) \) of the chromosome are randomly chosen \((1 \leq k_1 < k_2 \leq N)\).

(2) Genes that are in positions between \( k_1 \) and \( k_2 \) (both included) are shuffled.

(3) The number of QCs in each gene located 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 buffer size in each gene located between \( k_1 \) and \( k_2 \), both included, is modified by a random value that
is between 0 and the average handling time \( h_i^* \), of the vessel \( i \) in the same gene.

Figure 8 shows how the offspring \( o_i \), 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. Gene 2 is moved to position 4, gene 3 to position 1, and gene 4 to position 3. Finally, the number of QCs and the buffer size of each gene in position \( p \), \( 2 \leq p \leq 4 \), are modified by selecting feasible random values for each one.

6.6. Local Search. The multiobjective simulated annealing algorithm presented by Bandyopadhyay et al. [33] has been modified and included into the MOGA as a local search to solve the robust BAP + QCAP. The neighborhood structure of a solution takes advantage of the chromosome structure and their neighbors are obtained by changing the values of the variables presented in its genes \( (i, q_i, \text{ and } b_i) \). Algorithm 2 describes how to modify a given solution \( cur \) in order to create a neighbor \( new \). In this process, two operations are applied to solution \( cur \):

(i) interchanging the position of two vessels \( (v_{i1} \text{ and } v_{i2}) \) in the chromosome, randomly chosen,

(ii) changing the values of number of QCs assigned \( (q_i) \) and buffer size \( (b_i) \) of a vessel \( i \) randomly chosen.

This multiobjective simulated annealing algorithm \( (mosa) \) is computed every \( k \) iterations. It receives, as parameter, the Pareto front \( \mathcal{X} \) of the actual iteration of the MOGA + SA. As a result, it returns two different sets of solutions or schedules:

(i) a Pareto front \( \mathcal{X}' \) where the solutions from \( \mathcal{X} \) have been improved to obtain a local optimal following the AMOSA scheme,

(ii) a new set \( S_n \) consisting of the new nondominated solutions found in the search which are part of \( \mathcal{X}' \).

Unlike AMOSA [33], the simulated annealing algorithm implemented in this paper makes use of a different clustering method (see Algorithm 3) based on the crowding distance used in the NSGAII algorithm. This clustering method selects the representative solutions of the population according to the density of solutions surrounding a particular solution. After this local search process is performed, the solutions in \( S_n \) set replace the solutions in population \( \text{impPop} \) whose crowding distances are the lowest ones. The solution to be replaced is chosen by means of the same clustering method used in the simulated annealing. The purpose is to improve the quality of the population by keeping the solutions that are most spread around the search space.
Input: $P$: population; $N_m$: maximum number of chromosomes; 
Output: $P'$: population with exactly $N_m$ chromosomes; 
- [calculate the crowding distance for each element/chromosome in $P$] 
- [sort the elements in ascending order] 
- [choose the biggest $N_m$ elements according to the crowding distance] 
return $P'$

Algorithm 3: Clustering.

Table 1: Setting of the algorithms.

(a) NSGAII and SPEA2+ settings

| Parameter                  | MOGA scheme |
|----------------------------|-------------|
| Number of generations      | 500         |
| Number of chromosomes      | 100         |
| Mutation probability ($P_m$) | 0.1         |
| Crossover probability ($P_c$) | 0.9         |

(b) Multiobjective local search settings from MOGA + SA

| Parameter                  | MOGA + SA |
|----------------------------|-----------|
| Initial temperature ($T_{max}$) | 20        |
| Minimum temperature ($T_{min}$) | 0.001     |
| Annealing factor ($\alpha$)    | 0.9       |
| Termination criterion        | $T < T_{min}$ |
| Cycle length                | 2         |

7. Evaluation

As no benchmark is available in the literature, the experiments were performed in a corpus of 100 instances randomly generated, where parameters ($\text{maxQC, safeQC, etc.}$) follow the suggestions of container terminal operators. All these benchmarks are freely available at [http://gps.webs.upv.es/bap-qcap/](http://gps.webs.upv.es/bap-qcap/). Each one is composed of a queue from 100 vessels. These instances follow an exponential distribution for the number of required movements and length of vessels are uniformly generated in $[100, 1000]$ and $[70, 400]$, 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 (maxQC); the safe distance between QCs (safeQC) was 35 meters and the number of movements that QCs carry out was 2.5 (movsQC) per unit time.

The approaches developed in this paper, NSGAII, SPEA2+, and MOGA + SA, were coded using C++; their settings are showed in Tables I(a) and I(b). Due to the stochastic nature of the GA process, each instance was solved 30 times and the results show the average obtained values. The mathematical model was coded and solved by using IBM ILOG CPLEX Optimization Studio 12.5. Due to the fact that the square root function defines concave region, standard deviation function could not be introduced into the objective function in the mathematical solver. They were solved on an Intel i7-2600 3.4 Ghz with 8 Gb RAM.

CPLEX is able to obtain a schedule for an instance for a given $\lambda$ value. Algorithm 4 describes how to obtain a Pareto front using CPLEX solver for a given instance in order to be compared with the MOGA.

Figure 9 shows the Pareto fronts obtained of a representative instance by both the MOGA + SA and CPLEX. In this experiment, the timeout for the CPLEX solver was set to 1000 seconds for each $\lambda$ value. It is important to note that the greater the incoming vessels are, the fewer the solutions obtained by CPLEX solver are. Given this timeout, CPLEX was only able to get optimal solutions when $\lambda = 0.0$ and the incoming vessels were set to 10 and 20. Considering the Pareto fronts obtained by MOGA + SA and CPLEX, they were very similar with a queue of 10 vessels (see Figure 9(a)). However, for instance, with a queue 20 vessels, the solutions obtained by CPLEX were not able to reach the quality of the Pareto front of MOGA + SA (see Figure 9(b)). Furthermore, it turned out that for 40 incoming vessels only one nonoptimal solution was obtained (see Figure 9(d)) and even more there was no solution with 50 vessels.

Multiobjective optimization algorithms are not comparable directly since there is no a unique optimal solution. Zitzler et al. [35] propose different measures to compare Pareto front approximations. Among these measures, the size of the dominated space or the hypervolume is one of the most used measures to differentiate two algorithms [36]. This measure is related to a reference point $p$ and it is set according to the suggestion of While et al. [36]. To this end, for each objective, the worst value from any of the sets being compared is chosen and increased by an $\epsilon$ value.

Comparison among these different schemes has been performed by using the Kruskal-Wallis’ nonparametric statistical test, according to Zitzler et al. [35]. This test assesses whether there are significant differences among different sets of values: in this case, sets of hypervolume measures. Table 2 shows the values obtained for this test given the results after solving five instances of 50 vessels with the three different algorithms. Kruskal-Wallis test revealed a significance effect of the algorithms on the hypervolumes ($P$ value $< 0.01$).
As there was a significance difference among them, a post-hoc test using a pairwise comparison test (Wilcoxon) with Bonferroni correction was carried out and showed the significant differences between the different algorithms. As an example, Table 3 shows the results of the Wilcoxon test for the fifth instance. Note that, MOGA + SA algorithm produces Pareto fronts which are statistically different with respect to the other algorithms. Examining the average values in Table 2, it can be determined that MOGA + SA obtained better Pareto front approximations.

Figure 10 shows the Pareto fronts obtained by NSGAII and MOGA + SA algorithms. The schedules with the minimum and maximum values for each objective are highlighted by circles. It is important to note that MOGA + SA algorithm...
Table 3: Wilcoxon test over the hypervolumes obtained for a given instance.

|          | NSGAII | SPEA2+ |
|----------|--------|--------|
| SPEA2+   | 0.00001| —      |
| MOGA + SA| <2e−16 | <2e−16 |

Table 4(a) shows the percentage of incidences absorbed by each type of schedule. It can be observed that the more robust the schedule is, the more incidences absorbed. For instance, with delays \( d \in [1,0.5h_1] \), the \( R_{\text{min}} \) schedule only absorbed 18.73% of incidences in average, but the \( R_{\text{Max}} \) schedule absorbed up to 99.85%. Note that as the delay became larger, fewer schedules can absorb the incidences. With delays in the range of \([1,0.2h_1]\), the \( R_{\text{Max}} \) schedule can absorb 99.95% of incidences in average. However, with larger ranges, the incidences absorbed decreased down to 97.15% in average. This pattern was also repeated in Table 4(b). The lower \( T_o \), the lower incidences absorbed due to the fact that either there would not be buffers among vessels or there would be small buffers.

The performance of the schedules obtained by our approach (MOGA + SA) 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 vessel \( i \). This incidence is absorbed if there is enough buffer time behind 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 scheduled vessels.

In this experiment, 100 instances of 100 vessels were evaluated. For each instance, three different schedules were chosen from the Pareto front according to their robustness (see Table 4(a)): the one with the minimum robustness (\( R_{\text{min}} \)), the one with the maximum robustness (\( R_{\text{Max}} \)), and one intermediate robust schedule (\( R_i \) where \( R_{\text{min}} < R_i < R_{\text{Max}} \)). Likewise, three schedules were chosen according to their service time (see Table 4(b)) and other three schedules according to their standard deviation of the robustness (see Table 4(c)).

The incidences (or delays, \( d \)) introduced were randomly chosen from different ranges. These ranges vary 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 (see first column in Table 4). For each range, 100 incidences were uniformly created and applied to the three schedules of each instance.

8. Conclusions

The competitiveness among container terminals causes the need to improve the efficiency of each one of the subprocesses or scheduling problems that are performed within them. However, this efficiency is affected by the uncertainty of the environment. This uncertainty might provoke delays in the arrivals of the vessels or handling times greater than expected due to extreme weather events, breakdowns in engines, delays, and so forth. Furthermore, these scheduling problems are even harder since they are interrelated and sometimes there is no previous knowledge about these incidences. To this end, we introduce the robustness into these scheduling problems.
problems. In this paper, we introduce the robustness into one of the main scheduling problems in container terminals: berth allocation and quay crane assignment problems. Its objective function is to minimize the total service time of the incoming vessels. The robustness, as second objective function, has been modeled as a measure related to the likelihood of a schedule to absorb incidences. This robustness has been related to the operational buffers found after each assigned vessel. The greater the operational buffers are, the higher the robustness of the schedule is. However, due to the lack of the knowledge about incidences, operational buffers should be distributed among vessels proportionally,
Table 4: Percentages of incidences absorbed in schedules of 100 vessels.

(a) Delays applied to schedules with different levels of $\hat{R}$

| Range     | $R_{\min}$ | $R_{\hat{R}}$ | $R_{\max}$ |
|-----------|-------------|----------------|-------------|
| $d \in [1, 0.2h_i]$ | 21.78 | 95.51 | 99.95 |
| $d \in [1, 0.5h_i]$ | 18.73 | 93.58 | 99.85 |
| $d \in [1, 0.8h_i]$ | 16.64 | 90.49 | 98.96 |
| $d \in [1, 1.0h_i]$ | 13.92 | 87.10 | 98.01 |
| $d \in [1, 1.2h_i]$ | 12.93 | 85.31 | 97.15 |

(b) Delays applied to schedules with different levels of $T_i$

| Range     | $T_{i_{\min}}$ | $T_{i_{\hat{R}}}$ | $T_{i_{\max}}$ |
|-----------|----------------|-------------------|----------------|
| $d \in [1, 0.2h_i]$ | 20.17 | 79.91 | 99.94 |
| $d \in [1, 0.5h_i]$ | 16.16 | 73.88 | 99.75 |
| $d \in [1, 0.8h_i]$ | 13.89 | 65.17 | 98.94 |
| $d \in [1, 1.0h_i]$ | 11.85 | 60.25 | 98.08 |
| $d \in [1, 1.2h_i]$ | 10.32 | 56.76 | 96.74 |

(c) Delays applied to schedules with different levels of $\sigma(\hat{R})$

| Range     | $\sigma(\hat{R})_{\min}$ | $\sigma(\hat{R})_{\hat{R}}$ | $\sigma(\hat{R})_{\max}$ |
|-----------|---------------------------|-----------------------------|--------------------------|
| $d \in [1, 0.2h_i]$ | 99.96 | 75.48 | 61.85 |
| $d \in [1, 0.5h_i]$ | 99.95 | 72.23 | 54.42 |
| $d \in [1, 0.8h_i]$ | 99.94 | 67.88 | 48.72 |
| $d \in [1, 1.0h_i]$ | 98.87 | 65.34 | 47.77 |
| $d \in [1, 1.2h_i]$ | 97.39 | 62.38 | 45.75 |

Table 5: Percentages of incidences absorbed in schedules of 100 vessels obtained using or not LS (timeout 30 secs).

| Range     | $R_{\max}$ no LS | $R_{\max}$ with LS |
|-----------|-------------------|---------------------|
| $d \in [1, 0.2h_i]$ | 99.53 | 99.88 |
| $d \in [1, 0.5h_i]$ | 99.40 | 99.70 |
| $d \in [1, 0.8h_i]$ | 97.62 | 98.51 |
| $d \in [1, 1.0h_i]$ | 95.33 | 97.34 |
| $d \in [1, 1.2h_i]$ | 94.04 | 96.00 |

The results showed that the MIP model was able to obtain robust and efficient schedules up to 10 incoming vessels. However, MOGA + SA achieved better Pareto fronts than the MIP model for queues of incoming vessels greater than or equal to 20 vessels. Thereby, the schedules obtained by MOGA + SA were more efficient and robust than the schedules obtained by the MIP model. Furthermore, the MIP model was unable to find any solution with a given timeout for a queue of 50 incoming vessels. Additionally, differences between the MOGAs have been assessed by means of nonparametric statistical tests. It turned out to be that MOGA + SA obtained better Pareto fronts according to the hypervolume measures. Furthermore, different sets of incidences were simulated into the schedules obtained by the NSGAI and the MOGA + SA. The results returned that the schedules obtained by MOGA+SA were more robust due to the fact that they could absorb more incidences.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work has been partially supported by the Spanish Government under research project MINECO TIN2013-46511-C2-1-P, the project PIRES-GA-2011-294931 (FP7-PEOPLE-2011-IRSES), and the predoctoral FPU fellowship (AP2010-4405).

References

[1] L. Henesey, Multi-Agent Systems for Container Terminal Management, Citeseer, 2006.
[2] A. Imai, H. C. Chen, E. Nishimura, and S. Papadimitriou, “The simultaneous berth and quay crane allocation problem,” Transportation Research E: Logistics and Transportation Review, vol. 44, no. 5, pp. 900–920, 2008.
[3] Q.-M. Hu, Z.-H. Hu, and Y. Du, “Berth and quay-crane allocation problem considering fuel consumption and emissions from vessels,” Computers & Industrial Engineering, vol. 70, pp. 1–10, 2014.
[4] M. A. Salido, M. Rodriguez-Molins, and F. Barber, “Integrated intelligent techniques for remarshaling and berthing in maritime terminals,” Advanced Engineering Informatics, vol. 25, no. 3, pp. 435–451, 2011.
[5] M. Rodriguez-Molins, M. A. Salido, and F. Barber, “A GRASP-based metaheuristic for the berth allocation problem and the quay crane assignment problem by managing vessel cargo holds,” Applied Intelligence, vol. 40, no. 2, pp. 273–290, 2014.
[6] K. Park, T. Park, and K. R. Ryu, “Planning for remarshaling in an automated container terminal using cooperative coevolutionary algorithms,” in Proceedings of the ACM Symposium on Applied Computing (SAC ’09), pp. 1098–1105, March 2009.
[7] R. Stahilbock and S. Voß, “Operations research at container terminals: a literature update,” OR Spectrum, vol. 30, no. 1, pp. 1–52, 2008.
[8] A. Lim, “The berth planning problem,” Operations Research Letters, vol. 22, no. 2–3, pp. 105–110, 1998.
[9] C. Bierwirth and F. Meisel, “A survey of berth allocation and quay crane scheduling problems in container terminals,” European Journal of Operational Research, vol. 202, no. 3, pp. 615–627, 2010.
[10] K. H. Kim and K. C. Moon, “Berth scheduling by simulated annealing,” Transportation Research Part B: Methodological, vol. 37, no. 6, pp. 541–560, 2003.

[11] G. Giallombardo, L. Moccia, M. Salani, and I. Vacca, “Modeling and solving the tactical berth allocation problem,” Transportation Research Part B: Methodological, vol. 44, no. 2, pp. 232–245, 2010.

[12] C. Liang, J. Guo, and Y. Yang, “Multi-objective hybrid genetic algorithm for quay crane dynamic assignment in berth allocation scheduling,” Journal of Intelligent Manufacturing, vol. 22, no. 3, pp. 471–479, 2011.

[13] A. Diabat and E. Theodorou, “An integrated quay crane assignment and scheduling problem,” Computers and Industrial Engineering, vol. 73, pp. 115–123, 2014.

[14] Y.-M. Park and K. H. Kim, “A scheduling method for berth and quay cranes,” OR Spectrum, vol. 25, no. 1, pp. 1–23, 2003.

[15] C. Zhang, L. Zheng, Z. Zhang, L. Shi, and A. J. Armstrong, “The allocation of berths and quay cranes by using a sub-gradient optimization technique,” Computers & Industrial Engineering, vol. 58, no. 1, pp. 40–50, 2010.

[16] O. Lambrechts, E. Demeulemeester, and W. Herroelen, “Proactive and reactive strategies for resource-constrained project scheduling with uncertain resource availabilities,” Journal of Scheduling, vol. 11, no. 2, pp. 121–136, 2008.

[17] J.-C. Billaut, A. Moukrim, and E. Sanlaville, Flexibility and Robustness in Scheduling, vol. 56, Wiley, 2010.

[18] M. Hendriks, M. Laumanns, E. Lefeber, and J. T. Udding, “Robust cyclic berth planning of container vessels,” OR Spectrum, vol. 32, no. 3, pp. 501–517, 2010.

[19] X.-L. Han, Z.-Q. Lu, and L.-F. Xi, “A proactive approach for simultaneous berth and quay crane scheduling problem with stochastic arrival and handling time,” European Journal of Operational Research, vol. 207, no. 3, pp. 1327–1340, 2010.

[20] Y. Du, Y. Xu, and Q. Chen, “A feedback procedure for robust berth allocation with stochastic vessel delays,” in Proceedings of the 8th World Congress on Intelligent Control and Automation (WCICA ’01), pp. 2210–2215, July 2010.

[21] Y. Xu, Q. Chen, and X. Quan, “Robust berth scheduling with uncertain vessel delay and handling time,” Annals of Operations Research, vol. 192, no. 1, pp. 123–140, 2012.

[22] L. Zhen and D.-F. Chang, “A bi-objective model for robust berth allocation scheduling,” Computers and Industrial Engineering, vol. 63, no. 1, pp. 262–273, 2012.

[23] C. Blum, J. Puchinger, G. R. Raidl, and A. Roli, “Hybrid metaheuristics in combinatorial optimization: a survey,” Applied Soft Computing, vol. 11, no. 6, pp. 4135–4151, 2011.

[24] M. Ehrigott and X. Gandibleux, “Hybrid metaheuristics for multi-objective combinatorial optimization,” in Hybrid Metaheuristics, C. Blum, M. Aguiler, A. Roli, and M. Sampels, Eds., vol. 114 of Studies in Computational Intelligence, pp. 221–259, Springer, Berlin, Germany, 2008.

[25] R. Hanafi and E. Kozan, “A hybrid constructive heuristic and simulated annealing for railway crew scheduling,” Computers & Industrial Engineering, vol. 70, no. 1, pp. 11–19, 2014.

[26] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182–197, 2002.

[27] M. Kim, T. Hiroyasu, M. Miki, and S. Watanabe, “SPEA2+: improving the performance of the strength pareto evolutionary algorithm 2,” in Parallel Problem Solving from Nature—PPSN VIII, vol. 3242 of Lecture Notes in Computer Science, pp. 742–751, Springer, Berlin, Germany, 2004.
