An evolutionary approach to solving a new integrated quay crane assignment and quay crane scheduling mathematical model

Ghazwan Alsoufi, Xinan Yang, Abdellah Salhi

Department of Mathematical Sciences, University of Essex, UK

Abstract

This paper puts forward an integrated optimisation model that combines two distinct problems arising in container terminals, namely the Quay Crane Assignment Problem, and the Quay Crane Scheduling Problem. The model is of the mixed-integer programming type with the objective being to minimise the tardiness of vessels. Although exact solutions can be found to the problem using Branch-and-Cut, for instance, they are costly in time when instances are of realistic sizes. To overcome the computational burden of large scale instances, an adapted Genetic Algorithm, is used. Small to medium size instances of the combined model have been solved with both the Genetic Algorithm and the CPLEX implementation of Branch-and-Cut. Larger size instances, however, could only be solved approximately in acceptable times with the Genetic Algorithm. Computational results are included and discussed.

Keywords: Container terminals, Quay crane assignment, Quay crane scheduling, Branch-and-Cut, Genetic algorithm

1. Introduction

Container terminals are essential for the distribution of import/export goods to domestic markets around the world. Quay cranes are important and valuable equipment used in these terminals. They, therefore, must be used optimally whenever possible. However, determining the optimum number of quay cranes to assign to a given vessel moored in a container port, and the optimum sequence in which to perform the tasks of unloading and loading the vessel constitute some of the most important operations at ports. These operations potentially lead to distinct problems as follows.

1. The Berth Allocation Problem (BAP);
2. The Quay Crane Assignment Problem (QCAP);
3. The Quay Crane Scheduling Problem (QCSP).

*Corresponding author
[1]Email address: as@essex.ac.uk (A. Salhi)

Comprehensive reviews of these three problems (seaside problems) individually and when integrated pairwise as in BAP&QCAP, and QCAP&QCSP, can be found in [3, 25]. Each one of these problems considered individually, is complex enough. However, solving them individually and without consideration of the others may lead to suboptimal solutions. For this reason the pairwise integrated problems are investigated to obtain near optimal solutions.

After berth allocation, quay cranes must be assigned to each docked vessel. The number of quay cranes assigned to a vessel impacts on its processing time and therefore the working efficiency of the port. If the number of quay cranes assigned to a vessel is below requirement, the vessel will take unnecessarily longer processing time. This may result in a delay of the vessel's departure, which will, potentially, have a knock-on effect on the allocation of berths to waiting vessels, and other port operations. Equally, if the number of assigned quay cranes is more than what is needed, it may lead to high handling cost for this vessel and potentially delay the processing of other vessels. The other problem faced by the decision maker after assigning quay cranes to vessels is finding the optimum sequence in which to perform tasks (unload and load containers from and onto vessels). This is the problem of scheduling quay cranes to carry out all tasks efficiently and without interference between them.

QCAP and QCSP are closely related as the former feeds the latter with the number of quay cranes as input. Without the solution of the latter one cannot see the accurate processing time of a vessel given the fixed number of quay cranes assigned to it.

The combination of QCAP and QCSP has not been widely attempted so far, possibly because of the complexity of the mathematical formulation and especially its computational demands. Also, to our best knowledge, previous works that combine QCAP and QCSP do not allow quay cranes to move between vessels while still being processed [6, 19, 26].

This paper focuses on the integration of QCAP and QCSP to form QCASP, an extended decision making problem that can be solved in real applications. Contributions are two fold:

1. to formulate a mathematical model that combines QCAP and QCSP and allows quay cranes to move between vessels while they are still being processed. In other words, it allows the number of quay cranes allocated to any vessel to change during the handling of the vessel;
2. to solve realistic instances of the problem using an adapted variant of the Genetic Algorithm (GA).

The paper is organised as follows. Section 2 is a literature review of QCSP, BAP combined with QCAP, and QCAP combined with QCSP. Section 3 presents a mathematical model that combines QCAP and QCSP. In Section 4, a variant of GA is suggested for QCASP. In Section 5, the computational results are reported and discussed. Section 6 concludes the paper and suggests further directions of research on this topic.

2. Container port operations: a review

As said earlier, assigning and scheduling quay cranes is at the heart of container port operations. Here, we review the relevant literature on these problems and related ones.

2.1. The Quay Crane Scheduling Problem (QCSP)

The solution of QCSP is the optimal sequence of moves for quay cranes to perform (unloading and loading containers from and unto the vessel) in order to minimize the handling time of the vessel. Kim and Park [13] studied QCSP. A mixed integer programm was formulated. In their paper, Branch and Bound (B&B) was used in conjunction with the Greedy Randomized Adaptive Search Procedure (GRASP) [7], to overcome the difficulties of B&B on its own. Moccia et al. [17], formulated QCSP as a vehicle routing problem with additional constraints like the precedence relationships between tasks. CPLEX was used to solve small scale instances and the Branch-and-Cut (B&C)algorithm to solve large scale instances.

Sammarra et al. [24] proposed a tabu search heuristic for QCSP in order to minimize the completion time to unload and/or load containers from/unto vessels. They considered precedence and non-simultaneity between tasks. They also observed that QCSP can be decomposed into a routing problem and a scheduling problem.

Lee et al. [14] presented a mixed integer programming model of QCSP and proved that it is NP-complete; they used a genetic algorithm to solve problem instances to near optimality.

Bierwirth and Meisel [2] noticed that the constraints to avoid interference between the quay cranes in their model did not do the job properly. They revised their model accordingly. They also proposed a Unidirectional Schedule (UDS) heuristic for when the quay cranes do not change moving direction from their initial position and have identical directions of movement both from upper and lower bays.

Chung and Choy [5] proposed a modified GA to solve Kim and Park's model of QCSP, [13]. Their results compared well with those obtained by most well established algorithms.

Kaveshgar et al. [12] introduced an efficient GA for QCSP. Their algorithm improved the efficiency of GA search by using an initial solution based on the S-LOAD rule and by reducing the number of genes in the chromosomes to minimize the time of the search.

Nguyen et al. [18] suggested two representations of QCSP one for GA and the other for Genetic Programming (GP). GA uses permutation to decide the priority of tasks, whereas GP relied on a priority function to calculate the priority of tasks.

2.2. The Combined Berth Allocation and Quay Crane Assignment Problem (BACAP)

QCAP is the problem of finding the optimum number of quay cranes that should be assigned to every vessel that docks at a container terminal. This problem can be seen as trivial since knowing the workload of a vessel and the work rate of a quay crane should allow to estimate the number of quay cranes required for the vessel. However, if quay cranes of different work rates are possibly used and once they are allowed to move from one ship to another while work on the ships is ongoing, then their assignment is no longer so simple.

BACAP is the problem of allocating berthing times and berthing positions to vessels and, at the same time, determining the optimum number of quay cranes to service them.

Legato et al. [15], addressed QCAP with a predetermined berth position and time following the solution of BAP. They assumed that quay cranes could not move between vessels before all tasks are performed and the vessels processing is completely finished. A mathematical model was presented to determine the number of quay cranes for each vessel that is ready for processing.

Meisel and Bierwirth [16] integrated BAP, and QCAP into BACAP. The proposed problem is formulated taking into account some of the real issues faced by the decision maker at the port. In addition to the mathematical model, they also suggested two meta-heuristic approaches for the problem: the Squeaky Wheel Optimization (SWO), and Tabu Search (TS).

Cheong et al. [4] considered the multi-objective optimization aspect of BACAP; indeed, it involves simultaneous optimization of two highly-coupled container terminal operations. Optimization results show that the multi-objective approach offers the port manager flexibility in selecting a desirable solution for implementation.

Yang et al. [27] suggested to solve BACAP by solving simultaneously solution BAP and QCAP. They formulated a mathematical model which integrates the BAP constraints of Guan et al. [10] and the QCAP constraints of Legato et al. [15]. The objective function for this model is the combination of the objective functions of BAP and QCAP previous models. An evolutionary algorithm was developed to find the solution to this coarse combined problem.

2.3. The Combined Quay Crane Assignment and Quay Crane Scheduling Problem (QCASP)

The decision maker in QCASP focuses on determining the number of quay cranes for each vessel and finding the best sequence in which tasks will be performed by these quay cranes. There is obviously a strong relationship between the two goals.

Daganzo [6] addressed quay crane scheduling for multiple vessel arrivals. They proposed both an exact and an approximate solution approach with the objective being to minimise the tardiness of all vessels. Peterkofsky and Daganzo [19] developed a Branch-and-Bound (B&B) algorithm to solve QCSP. Both of these papers did not consider the interference between quay cranes.

Tavakkoli et al. [26] studied QCASP. They formulated a mixed integer program to determine the optimal number of quay cranes for every vessel that will arrive at the terminal and at the same time the optimal sequence in which the tasks should be carried out on the vessel. An evolutionary approach is suggested to solve large scale instances of this type of problem.

3. Mathematical formulation

This section describes a mixed integer programming model of a container terminal with a continuous berth, the solution of which is the optimum number of quay cranes to be assigned to each docked vessel and their scheduling to carry out all necessary moves in an optimum way. Note that the number of quay cranes assigned to a vessel at the beginning of the operation may not be the same as at the end because our model allows quay cranes to move between vessels. Comparing with the traditional way of handling these two problems, i.e. individually, the combined model as proposed here, does not require us to estimate the processing time when allocating quay cranes to vessels. Therefore, it allows in general to find more accurate solutions.

The improvement on the two step planning models is that we do not force the number of quay cranes allocated to a vessel to be fixed during the whole processing period of the vessel. If necessary, a quay crane may move from one vessel to another before the processing of this vessel has finished. This is more flexible than using a fixed number of quay cranes to handle a vessel; it has the potential to give better working plans [1], and also, to some extent, to mitigate uncertainty linked to the performance of quay cranes. To illustrate the case of better plans, consider the situation in which two quay cranes are available to handle two vessels with data as given in Table 1, and each vessel has two tasks to process.

Table 1: Input data of Example 1

Ready (crane)	0	0
Initial location (crane)	11	14
Variable cost(crane)	1	1
Arrival time	0	0
Processing time of tasks ,vessel 1	85	27
Processing time of tasks ,vessel 2	18	33
Location task, vessel 1	1	3
Location task, vessel 2	1	4
Expected departure time of vessel	85	33
Berthing position	10	15
Tardiness cost (per unit time)	5	1
Earliness income (per unit time)	1	1

In the previous models where a fixed number of quay cranes is allocated to every vessel during the whole processing period, the optimal working plan is described in Figure 2 with the objective value being equal to 328. Even though quay crane 2 finished its work on vessel 2 at 54(2+18+1+33), it is not allowed to move away from vessel 2 according to the mathematical model. This wastes the effective working time of this quay crane which will result into a sub-optimal solution. In contrast, in our model, a variable number of quay cranes is used during the processing period, which allows quay crane 2 to move to vessel 1 to perform other tasks as showing in Figure 1. As a result, vessel 1 processing has been completed 27 time units earlier than in the previous plan with an objective value of 197, due to making the best use of both quay cranes. Note that we only allow the quay crane to move after it has finished its work on vessel 2.

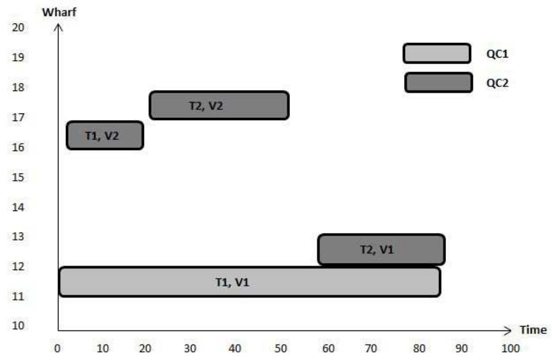


Figure 1: Suggested model solution

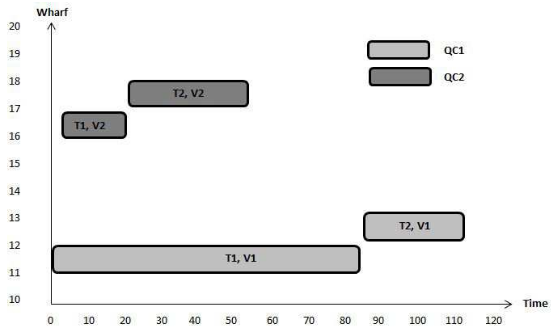


Figure 2: Previous model solution

Our model also allows quay cranes to share tasks on the same vessels to which they are allocated. Consider the input data for two vessels arriving at a container terminal as given in Table 2.

Table 2: Input data of Example 2

Ready (crane)	0	0
Initial location (crane)	22	24
Variable cost(crane)	0	0
Arrival time	0	0
Processing time of tasks,vessel 1	28	22
Processing time of tasks,vessel 2	39	34
Location task, vessel 1	1	2
Location task, vessel 2	1	2
Expected departure time of vessel	28	39
Berthing position	20	25
Tardiness cost (per unit time)	1	1
Earliness income (per unit time)	1	1

In Example 2, the objective value returned by previous models is 70 time units, (see Figure 4). Since we allow quay cranes to move between vessels if no interference constraints are violated, the solution of our model returns an objective value of only 35 time units, as can be seen in Figure 3.

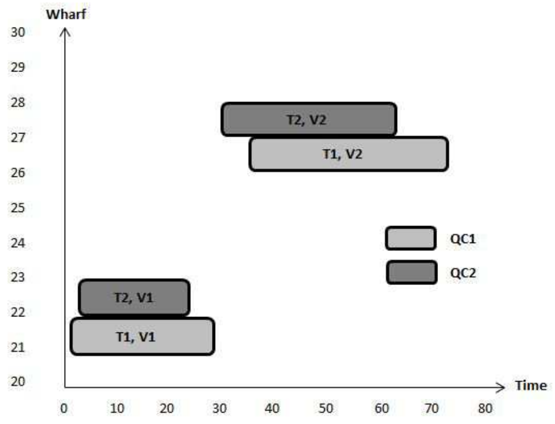


Figure 3: Suggested model solution

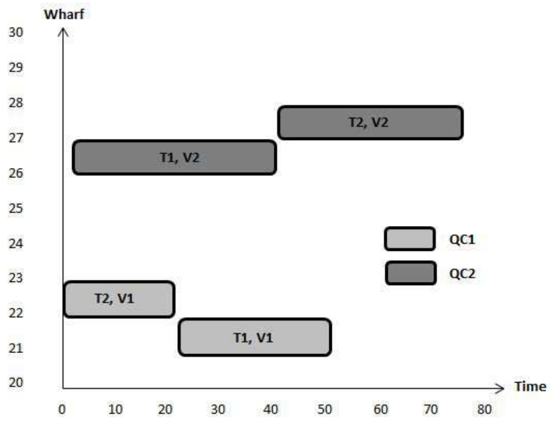


Figure 4: Previous model solution

3.1. Assumptions

Consider a continuous berth container terminal with fixed length and berth allocation already decided. Now assume that

1. the berthing position and berthing time of vessels are given as inputs;
2. each vessel is divided longitudinally into bays; all bays have the same length. Thus, the length of a vessel is given as the number of bays;
3. the safety distance between each pair of adjacent quay cranes depends on the width of a bay;
4. once a quay crane starts processing a task, it can leave only after it has finished the workload of this task;
5. quay cranes are on the same rail and thus they cannot cross over each other;
6. there are precedence relationships between some tasks, i.e. some tasks cannot be performed simultaneously.

3.2. Indices

Q	Number of quay cranes $(q, q_i, q_j = 1, 2, ...Q)$.
V	Number of vessels $(v, v_i, v_j = 1, 2, ..., V)$.
B_v	Number of tasks on vessel v $(b, b_i, b_j = 1, 2, ..., B_v)$.

3.3. Parameters

p_b^v	Time required to perform task b on vessel v.
l_b^v	Location of task b on vessel v expressed by the ship bay number on vessel v.
r_q	Earliest available time of the q^{th} quay crane.
I_0^q	Initial location of quay crane q which is relative to the ship-bay number.
$t_{b_i b_j}^{qv}$	Travel time of the q^{th} quay crane from the location $l_{b_i}^v$ of task b_i to the location $l_{b_j}^v$ of task b_j. $t_{b_0 b_j}^{qv}$ Represents the travel time from the initial position I_0^q of the q^{th} quay crane to the location $l_{b_j}^v$ of the task b_j on vessel v.
T_v	Berthing time of vessel v.
d_v	Requested departure time for vessel v.
H_q	Variable cost (per-hour) of using the q^{th} quay crane.
W_v	Tardiness cost (per-hour) of vessel v per time unit.
R_v	Earliness incoming of vessel v per time unit .
Ψ	Set of pairs of tasks that cannot be performed simultaneously. When tasks b_i and b_j cannot be performed simultaneously, then $(b_i, b_j) \in \Psi$.
Φ	Set of ordered pairs of tasks for which there is a precedence relationship. When task b_i must precede task b_j, then we have $(b_i, b_j) \in \Phi$.
M	Arbitrary large positive number.

3.4. Binary decision variables

$$X_{b_i b_j}^{qv} = \begin{cases} 1 & \text{if the } q^{th} \text{ quay crane performs task } b_j \text{ immediately after performing task } b_i \text{ on vessel } v. \\ 0 & \text{otherwise} \end{cases}$$

Tasks b_0 and b_{B_v+1} are considered as the dummy initial and final states of each quay crane, respectively. Thus, when task b_j is the first task of the q^{th} quay crane on vessel v then $X_{b_0 b_j}^{qv} = 1$. Similarly, when task b_j is the last task of the q^{th} quay crane on vessel v then $X_{b_j b_{B_v+1}}^{qv} = 1$.

$$Z^{v}_{b_i b_j} = \begin{cases} 1 & \text{if task } b_j \text{ starts later than the finish time of task } b_i \text{ on vessel } v. \\ 0 & \text{otherwise.} \end{cases}$$

$$Y^{q}_{v_i v_j} = \begin{cases} 1 & \text{if the } q^{th} \text{ quay crane is assigned to vessel } v_j \text{ immediately after finishing its task on vessel } v_i. \\ 0 & \text{otherwise.} \end{cases}$$

$$\alpha^{v_i v_j}_{b_i b_j} = \begin{cases} 1 & \text{if the task } b_j \text{ on vessel } v_j \text{ is located below the task } b_i \text{ on vessel } v_i. \\ 0 & \text{otherwise.} \end{cases}$$

$$\beta^{v_i v_j}_{b_i b_j} = \begin{cases} 1 & \text{if the task } b_j \text{ on vessel } v_j \text{ starts later than the finish time of task } b_i \text{ on vessel } v_i \\ 0 & \text{otherwise.} \end{cases}$$

3.5. Continuous decision variables

E_v	Earliness of vessel v.
A_v	Tardiness of vessel v.
S_{qv}	Starting time of q^{th} quay crane on vessel v.
$D^{v}_{b_i}$	Completion time of task b_i on vessel v.
C_{qv}	Completion time of q^{th} quay crane on vessel v.
F_v	Finishing (departure) time of the vessel v.

3.6. The mathematical model

$$\min(Z) = \sum_{q=1}^{Q} H_q C_{qv} + \sum_{v=1}^{V} W_v A_v - \sum_{v=1}^{V} R_v E_v \tag{1}$$

s.t

$$d_v - F_v = E_v - A_v, \qquad \forall v \tag{2}$$

$$\sum_{v_j=1}^{V} Y^{q}_{v_0 v_j} = 1, \qquad \forall q \tag{3}$$

$$\sum_{v_i=1}^{V} Y^{q}_{v_i (V+1)} = 1, \qquad \forall q \tag{4}$$

$$\sum_{v_j=1}^{V+1} Y^{q}_{v v_j} - \sum_{v_j=0}^{V} Y^{q}_{v_j v} = 0, \qquad \forall v, q \tag{5}$$

$$\sum_{v_i=0}^{V}\sum_{q=1}^{Q} Y^q_{v_iv_j} \geq 1, \qquad \forall v_j \tag{6}$$

$$S_{qv} \geq r_q - M(1 - Y^q_{v_0v}), \qquad \forall v, q \tag{7}$$

$$S_{qv} \geq T_v - M(1 - \sum_{v_j=1}^{V+1} Y^q_{vv_j}), \qquad \forall v, q \tag{8}$$

$$S_{qv_j} \geq C_{qv_i} - M(1 - Y^q_{v_iv_j}), \qquad \forall v_i, v_j; v_i \neq v_j; q \tag{9}$$

$$\sum_{b_j=1}^{B_v} X^{qv}_{b_0b_j} = \sum_{v_i=0}^{V} Y^q_{v_iv}, \qquad \forall v, q \tag{10}$$

$$\sum_{b_j=1}^{B_v} X^{qv}_{b_jb_{B_v+1}} = \sum_{v_i=0}^{V} Y^q_{v_iv}, \qquad \forall v, q \tag{11}$$

$$\sum_{b_j=1}^{B_v+1} X^{qv}_{bb_j} - \sum_{b_j=0}^{B_v} X^{qv}_{b_jb} = 0, \qquad \forall b, v, q \tag{12}$$

$$\sum_{q=1}^{Q}\sum_{b_i=0}^{B_v} X^{qv}_{b_ib_j} = 1, \qquad \forall b_j, v \tag{13}$$

$$\sum_{b_i=0}^{B_v}\sum_{b_j=1}^{B_v+1} X^{qv}_{b_ib_j} \leq M \sum_{v_i=0}^{V} Y^q_{v_iv}, \qquad \forall v, q \tag{14}$$

$$D^v_{b_i} + p^v_{b_i} + t^{qv}_{b_ib_j} - D^v_{b_j} \leq M(1 - X^{qv}_{b_ib_j}), \qquad \forall b_i, b_j, v, q \tag{15}$$

$$S_{qv} + p^v_{b_j} + t^{qv}_{b_0b_j} - D^v_{b_j} \leq M(1 - X^{qv}_{b_0b_j}), \qquad \forall b_j, v, q \tag{16}$$

$$D^v_{b_j} - C_{qv} \leq M(1 - X^{qv}_{b_jb_{B_v+1}}), \qquad \forall b_j, v, q \tag{17}$$

$$C_{qv} - F_v \leq M(1 - \sum_{v_j=1}^{V+1} Y^q_{vv_j}), \qquad \forall v, q \tag{18}$$

$$D^v_{b_i} + p^v_{b_j} \leq D^v_{b_j}, \qquad \forall (b_i, b_j) \in \Phi_v; b_j \neq b_i; \forall v \tag{19}$$

$$D^v_{b_i} - D^v_{b_j} + p^v_{b_j} \leq M(1 - Z^v_{b_ib_j}), \qquad \forall b_i, b_j; b_i \neq b_j; \forall v \tag{20}$$

$$Z^v_{b_ib_j} + Z^v_{b_jb_i} = 1, \qquad \forall (b_i, b_j) \in \Psi_v; b_j \neq b_i; \forall v \tag{21}$$

$$\sum_{\theta=0}^{Q}\sum_{\kappa=0}^{B_v} X^{\theta v}_{\kappa b_j} - \sum_{\theta=0}^{Q}\sum_{\kappa=0}^{B_v} X^{\theta v}_{\kappa b_i} \leq M(Z^v_{b_ib_j} + Z^v_{b_jb_i}), \qquad \forall b_i, b_j; j \neq i; l_{b_i} < l_{b_j}; \forall v, q \tag{22}$$

$$P_{v_i} + l^v_{b_i} \leq P_{v_j} + l^v_{b_j} + M(1 - \alpha^{v_iv_j}_{b_ib_j}), \qquad \forall b_i, b_j, v_i, v_j; v_j \neq v_i \tag{23}$$

$$D^{v_i}_{b_i} - D^{v_j}_{b_j} + p^{v_j}_{b_j} \leq M(1 - \beta^{v_iv_j}_{b_ib_j}), \qquad \forall b_i, b_j, v_i, v_j; v_j \neq v_i \tag{24}$$

$$\beta^{v_iv_j}_{b_ib_j} + \beta^{v_jv_i}_{b_jb_i} + \alpha^{v_jv_i}_{b_jb_i} \geq \sum_{\kappa=0}^{Bv} X^{q_iv_i}_{\kappa b_j} + \sum_{\kappa=0}^{Bv} X^{q_jv_j}_{\kappa b_i} - 1, \qquad \forall b_i, b_j, v_i, v_j, q_i, q_j; v_j \neq v_i; q_i < q_j \tag{25}$$

$$X^{qv}_{b_ib_j}, Z^v_{b_ib_j}, Y^q_{v_iv_j}, \alpha^{v_iv_j}_{b_ib_j}, \beta^{v_iv_j}_{b_ib_j} \in \{0,1\}, \tag{26}$$

$$C_{qv}, F_v, D^v_{b_j}, P_v, T_v \geq 0. \tag{27}$$

In the objective function (1), the first term $\sum_{q=1}^{Q} H_q C_{qv}$ represents the cost of using quay cranes. The second term $\sum_{v=1}^{V} W_v A_v$ represents the tardiness cost if the departure time of a vessel is greater than its due time. The third term $\sum_{v=1}^{V} R_v E_v$ represents the earliness income if the finishing time of a vessel is less than its due time. Note that in practice this reward for earliness may be zero. Constraints (2) calculate the earliness or tardiness of a vessel depending on the difference between its due time and its finishing time.

The constraints (3-6) represent the main conditions for QCAP. However, constraints (3) and (4) respectively select the first and the last ships for each quay crane. Constraints (5) guarantee that ships are processed in a well-defined sequence. Constraints (6) guarantee that each vessel be handled by at least one quay crane.

The constraints (7-9) determine the starting time of quay cranes. Constraints (7) force the starting time of the earliest vessel that is to be done by the q^{th} quay crane to be after the ready time of the q^{th} quay crane. Note that vessel v_0 is a dummy vessel from which the working sequence starts. Constraints (8) say that the starting time of the q^{th} quay crane on the vessel v is no earlier than the berthing time of vessel v if the q^{th} quay crane is assigned to serve this vessel. Constraints (9) ensure that the starting time of the q^{th} quay crane on vessel v_j is no earlier than the finishing time of its predecessor vessel v_i.

Constraints (10) ensure that if a quay crane is assigned to a vessel, then it will start its processing from one of the tasks on that vessel. Constraints (11) ensure that if a quay crane is assigned to a vessel, then it will finish its processing with one of tasks on that vessel. Constraints (12) show a flow balance ensuring that tasks are performed in a well-defined sequence on every vessel. Constraints (13) ensure that every task on each vessel must be handled by exactly one quay crane. Constraints (14) ensure that tasks on a vessel are handled by a quay crane only if this quay crane is allocated to that vessel. Constraints (15) simultaneously determine the completion time for each task and eliminate sub-tours; sub-tours here are the looping on tasks which have already been done. To illustrate, let Task 1, Task 2, and Task 3, be carried out in this order. A sub-tour would be to do Task 1, Task 2, Task 3, and Task 2 again, for instance. Constraints (15) remove this possibility. Constraints (16) determine the quay crane starting time on vessel v and the completion time of the same quay crane is computed by constraints (17). Constraints (18) determine the finishing time of each vessel.

When required, constraints (19) force task b_i to be completed before the starting of task b_j for all the task pairs $(b_i, b_j) \in \Phi$. Constraints (20) define $Z^v_{b_i b_j}$ such that $Z^v_{b_i b_j} = 1$ when the operation of task b_j on vessel v starts after the completion of task b_i on the same vessel. Constraints (21) ensure that the pair of tasks that are members of the set Ψ will not be handled simultaneously.

By constraints (22), interference between quay cranes is avoided. Suppose that tasks b_i and b_j are performed simultaneously and $l_i < l_j$, then $Z^v_{b_i b_j} + Z^v_{b_j b_i} = 0$. Note that both quay cranes and tasks are ordered in an increasing order of their relative location in the direction of increasing ship-bay number.

Suppose that, for $q_1 < q_2$, quay crane q_1 performs tasks b_j and quay crane q_2 performs task b_i. Then, interference between quay cranes q_1 and q_2 results as in such case, $\sum_{\theta=1}^{q_1}\sum_{\kappa=0}^{Nv} X_{\kappa j}^{\theta v} - \sum_{\theta=1}^{q_1}\sum_{\kappa=0}^{Nv} X_{\kappa i}^{\theta v} = 1$. This violates constraints (22), since we have $Z_{b_i b_j}^{v} + Z_{b_j b_i}^{v} = 0$ as mentioned earlier.

Constraints (23) define $\alpha_{b_i b_j}^{v_i v_j}$ such that $\alpha_{b_i b_j}^{v_i v_j} = 0$ if the berthing position of vessel v_i plus the location of task b_i on that vessel is greater than the berthing position of vessel v_j plus the location of task b_j on that vessel. Figure 5 illustrates how the value of $\alpha_{b_i b_j}^{v_i v_j}$ is computed.

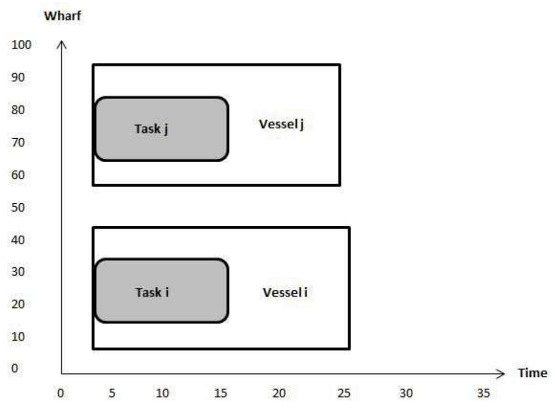


Figure 5: No location overlap between two vessels

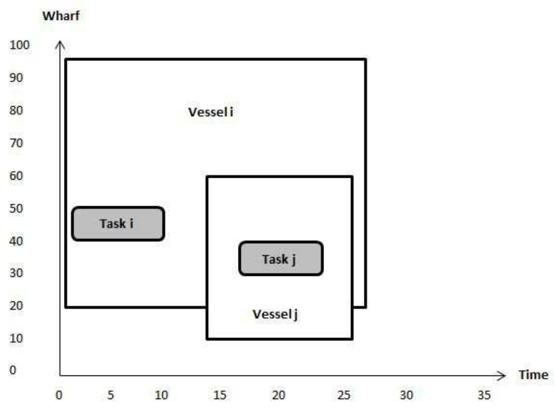


Figure 6: Location overlap between two vessels

The value of $\alpha_{b_i b_j}^{v_i v_j}$ in Figure 5 can be 1 because $P_{v_i} + l_{b_i}^{v} \leq P_{v_j} + l_{b_j}^{v}$. The value of $\alpha_{b_i b_j}^{v_i v_j}$ in Figure 6 equals 0, because $P_{v_i} + l_{b_i}^{v} > P_{v_j} + l_{b_j}^{v}$. This means there is overlap in the position between these two tasks on these two vessels.

Constraint (24) defines $\beta_{b_i b_j}^{v_i v_j}$ such that $\beta_{b_i b_j}^{v_i v_j} = 0$ if the finishing time of task b_i on vessel v_i plus the processing time of task b_j on vessel v_j is greater than the finishing time of task b_j on vessel v_j. Figure 7 illustrates how the value of $\beta_{b_i b_j}^{v_i v_j}$ is computed.

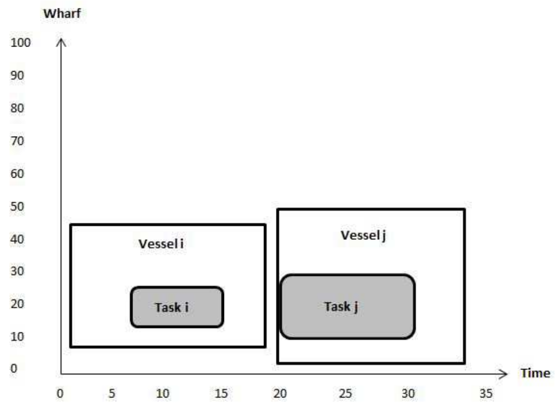


Figure 7: No location overlap between two vessels

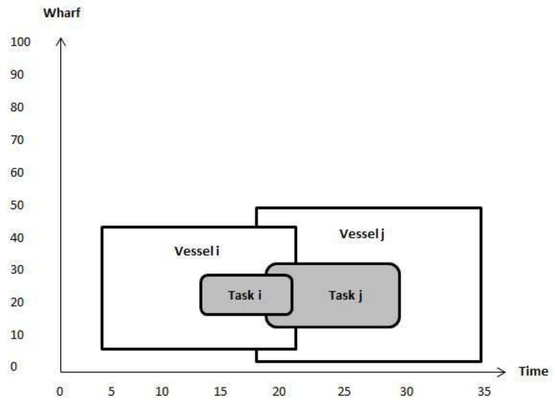


Figure 8: Location overlap between two vessels

The value of $\beta_{b_i b_j}^{v_i v_j}$ in Figure 7 can be 1 because $D_{b_i}^{v_i} + p_{b_j}^{v_j} \leq D_{b_j}^{v_j}$, whereas the value of $\beta_{b_j b_i}^{v_j v_i}$ in the same figure equals 0 because $D_{b_j}^{v_j} + p_{b_i}^{v_i} > D_{b_i}^{v_i}$. The value of $\beta_{b_i b_j}^{v_i v_j}$ in Figure 8 equals 0 because $D_{b_i}^{v_i} + p_{b_j}^{v_j} > D_{b_j}^{v_j}$ and the value of $\beta_{b_j b_i}^{v_j v_i}$ in the same figure equals 0 because $D_{b_j}^{v_j} + p_{b_i}^{v_i} > D_{b_i}^{v_i}$. This means there is overlap in the time between these two tasks on these two vessels. Constraints (25) prevent the interference between quay cranes depending on the values of $\beta_{b_i b_j}^{v_i v_j}$ and $\alpha_{b_i b_j}^{v_i v_j}$, respectively.

4. Application of the Genetic Algorithm to QCASP

Once an instance of the model is defined and the data is available, submitting it to CPLEX is straightforward. However, to apply GA requires that we define a representation for the problem solutions as well as other algorithmic components necessary for its implementation as can be seen later. GA is an adaptive heuristic method based on evolution through natural selection ideas due to Darwin and others, [11]. It is a population based approach, i.e. it searches for solutions by maintaining a population of solutions that are then updated from generation to generation using a number of possible genetic operators such as Crossover, Mutation, and Reproduction. Over successive generations the population evolves towards an optimal solution. GA is particularly effective on difficult problems referred to as being NP-Hard. QCASP has been shown to be NP-hard, [14, 26, 8].

4.1. Solution representation: chromosome

GA starts with a randomly generated population of solutions. Each solution is called a chromosome and consists of a sequence of genes, here representing a sequence of holds (tasks) for all docked vessels. The value of a gene is randomly picked from the index set of all holds; it cannot, therefore, be duplicated, i.e. each gene is unique. Each chromosome consists of $v \times b$ genes, where v represents the number of vessels and b the number of tasks on each vessel. A simple chromosome for the case of two vessels, each with three tasks, is illustrated in Figure 9. Here, genes (1,2,3) represent the tasks on the first vessel and genes (4,5,6) represent those on the second vessel. Based on the sequence of tasks for all vessels

1	4	3	6	5	2

Figure 9: Chromosome representation

represented by the chromosome, a quay crane schedule can be constructed using the following steps that are the extension of the procedure proposed by Lee et al. [14] used for each vessel separately. In this paper, however, we assume that the berth allocation plan (berthing time and berthing position of each vessel arriving at the container terminal) and the initial position of each quay crane at the beginning of scheduling are known.

Quay crane scheduling procedure, [14]:

Begin

Step 1: Based on the current position of each quay crane, determine which quay cranes can handle the first unassigned task in the chromosome without interference with other quay cranes. If only one quay crane is available, this task is assigned to this quay crane and it is deleted from the chromosome; the position and the completion time of the assigned quay crane are updated. The completion time of task i is also computed. If two quay cranes are available, go to Step 2.

Step 2: Compare the completion time of the two available quay cranes, and assign this task to the quay crane with earlier completion time. This task is then deleted from the chromosome, and both the position and the completion time of the assigned quay crane are updated. Also the completion time of task i is computed. If their completion times are equal, go to Step 3.

Step 3: Compare the distance between this task and these two available quay cranes, and assign the task to the quay crane with the shorter distance. Then, this task is deleted from the chromosome, and both the position and the completion time of the assigned quay crane are updated. Also the completion time of task i is computed. If their distances are equal, go to Step 4.

Step 4: Assign this task to the quay crane with the smaller number. Then, this task is deleted from the chromosome, and both the position and the completion time of the assigned quay crane are updated. Also the completion time of task i is computed.

Step 5: Steps 1–4 are repeated until all the tasks in the chromosome are assigned.

Stop

4.2. Solution validation

To validate chromosomes/solutions, two important situations must be considered. The first one is the precedence relationship between tasks. For instance, some of the bays of a given vessel need to be unloaded and loaded. The discharging of containers from a bay must precede the loading of the bay. For this reason the generated chromosome should be checked to see if it satisfies this condition, i.e. constraints (19). The second situation is the non-simultaneity of some of tasks, i.e. constraints (21) must also be satisfied. If either of constraints (19) or (21) are violated or both are violated, the generated chromosomes are discarded by putting a high penalty to their fitness values.

4.3. Evaluation of fitness

The objective of the QCASP is to minimise the cost of using quay cranes and at the same time, to minimise the tardiness of vessels. The completion time of each quay crane can be computed by summing up the processing time of all the tasks that have been performed by this quay crane plus the travel time which it takes to move from one hold to another. The tardiness of each vessel can be computed by subtracting the finishing time from the expected departure time. The finishing time represents the maximum processing time of the vessel required by the quay cranes assigned to it. The objective function used by the GA in MATLAB is the same objective function as that of the mathematical model. Thus, the fitness value of a chromosome is calculated by Equation (28).

$$Fitness(chromosome) = \frac{1}{\sum_{q=1}^{Q} H_q C_{qv} + \sum_{v=1}^{V} W_v A_v - \sum_{v=1}^{V} R_v E_v} \tag{28}$$

4.4. Generating new populations

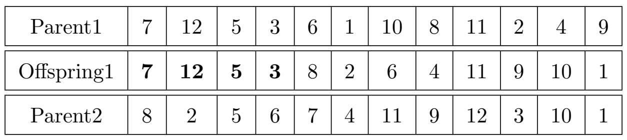
From the initial randomly generated population, subsequent generations of children, i.e. new populations, must be created. This is achieved by using genetic operators such as crossover, mutation and reproduction (copying of individuals unmodified into subsequent populations), [21, 20, 23, 22].

4.4.1. Selection process

The selection process picks chromosomes from the current population to be parents to new individuals (children/solutions) in the new population created using one of many genetic operators as listed above. To give priority to the best chromosomes to pass their genes into the next generation, the fitness proportionate selection approach is implemented using a roulette wheel. High fitness individuals/solutions have high probability to be selected to contribute to the next population. In other words there is a bias toward their selection which means their genes are likely to be passsed into the next generation.

4.4.2. Crossover operator

To produce a new chromosome (offspring) the ‘Order Crossover’ of Gen and Cheng [9] is used. Order crossover is a permutation-based crossover. It works as follows. A subsequence of consecutive alleles from parent 1 is selected and used to partially make the offspring; the remaining alleles to complete the creation of the offspring are chosen from parent 2 avoiding any repetitions. The same procedure is then applied starting from parent 2 to make the second offspring. The crossover operator always creates two offspring. Figures 10 and 11 illustrate the order crossover.



Parent1	7	12	5	3	6	1	10	8	11	2	4	9
Offspring1	**7**	**12**	**5**	**3**	8	2	6	4	11	9	10	1
Parent2	8	2	5	6	7	4	11	9	12	3	10	1

Figure 10: Offspring 1

Parent1	7	12	5	3	6	1	10	8	11	2	4	9
Offspring2	**8**	**2**	**5**	**6**	7	12	3	1	10	11	4	9
Parent2	8	2	5	6	7	4	11	9	12	3	10	1

Figure 11: Offspring 2

4.4.3. Mutation operator

To prevent the population getting trapped in a local solution, the mutation operator is used to enable the genetic algorithm to escape and explore the search space globally by changing one or more genes. In this algorithm, two genes from the chromosome are randomly selected and then swapped with each other. Figure 12 illustrates the mutation operator.

8	4	**3**	6	5	2	7	**9**	1
8	4	**9**	6	5	2	7	**3**	1

Figure 12: Mutation operator

4.5. Stopping

Two stopping criteria are used in the genetic algorithm: the maximum number of generations, and the the number of generations without any improvement in the best solution found so far; this number is pre-set by the user.

5. Computational experiments

Twenty four instances of the above mathematical model of QCASP with different numbers of vessels, tasks, and quay cranes have been solved using CPLEX and GA. They are recorded in Tables 3 and 4. All instances have randomly generated processing time of tasks for each hold from the uniform distribution U(30,180).

CPLEX solved problems 1 to 17. These are relatively small in size. They were solved in acceptable times, although 16, 23 and 18 hours were required for problems 13, 15 and 17, respectively. These times are hardly acceptable in the context of container terminal operations. The rest of the problems, 18 to 24, which are the realistic instances, could not be solved with CPLEX in acceptable times (¿ 100 hours of CPU time). In some cases they could not be solved at all due to the limitations of the computing platform used.

GA, coded in Matlab, managed to solve all 24 instances. For the small size problems of Table 3, the GA parameters of population size, rate of crossover, rate of mutation, and the maximum number of generations are set as 150, 0.8, 0.1, and 100, respectively. In the case of the large size instances of Table

4, population size, crossover and mutation rates, and the maximum number of generations are set as 500, 0.8, 0.2, and 1000, respectively.

All experiments have been performed on a PC with Intel Core 2 and 2.40 GHz CPU with 4 GB RAM running Windows 7 Operating System. The 24 problems and their corresponding results are presented in Tables 3 and 4, containing small problems 1 through 12 and problems 13 through to 24, respectively.

5.1. Results

On the small size problems of Table 3, the number of constraints, the number of decision variables, and the CPLEX computational time grow exponentially with the instance number of vessels, tasks and quay cranes. Note, however, that in column 10 of Table 3 showing the CPU time required by GA to solve these problems, this time does not change much with the increase in the problem size. It is also important to note that in most cases, GA obtains the optimal or near optimal solution within 100 generations. The average gap between CPLEX and GA returned objective values is about 1.4%.

Table 3: Computational results for small scale instances

No.	Problem Information			Problem Size			CPLEX		GA		Gap(%)
	Vessels	Tasks	Q.Cranes	Constraints	Dec.Vars	Int.Vars	CPU(s)	Obj.Val	CPU(s)	Obj.Val	
1	2	3	5	528	270	243	2	180	6	180	0.0
2	3	2	5	601	294	254	10	62	6	64	3.2
3	2	4	4	628	342	316	77	450	8	450	0.0
4	4	2	4	736	380	332	881	618	8	668	8.0
5	3	3	4	821	408	370	1788	367	8	367	0.0
6	2	4	5	840	402	373	115	165	8	165	0.0
7	4	2	5	1010	444	389	856	216	8	222	2.7
8	2	4	6	1084	462	430	171	41	9	41	0.0
9	3	3	5	1126	474	431	2146	76	8	76	0.0
10	4	2	6	1332	508	446	1086	565	7	583	3.1
11	2	5	4	918	484	456	7749	231	9	231	0.0
12	3	3	6	1485	540	492	2315	696	6	696	0.0

On the larger size instances, shown in Table 4, CPLEX did not solve some of these instances, marked with a dash in the CPLEX colum. GA, however, finds the optimal or near solutions for all the instances in a reasonable CPU times (see column 10 of Table 4). The average gap between the objective values of the solutions found by CPLEX and those found by GA is 1.12%. Note that the gap is missing for the problems that CPLEX could not solve.

Table 4: Computational results for large scale instances

No.	Problem Information			Problem Size			CPLEX		GA		Gap(%)
	Vessels	Tasks	Q.Cranes	Constraints	Dec.Vars	Int.Vars	CPU(h:m:s)	Obj.Val	CPU(h:m:s)	Obj.Val	
13	5	2	4	1098	535	474	16:44:23	445	00:00:08	451	1.3
14	2	5	5	1234	566	535	01:21:19	321	00:00:08	321	0.0
15	5	2	5	1525	620	550	23:25:56	1216	00:00:10	1216	0.0
16	5	2	6	2032	705	626	01:37:35	1726	00:00:09	1759	1.9
17	2	6	6	2224	870	834	18:30:35	457	00:00:10	468	2.4
18	5	8	8	43826	6455	6328	-	-	00:14:08	1479	-
19	5	10	8	67686	9675	9538	-	-	00:17:45	237	-
20	5	12	8	97146	13575	13428	-	-	00:21:09	2483	-
21	5	12	10	148610	15345	15180	-	-	00:21:29	50	-
22	5	16	10	263350	26385	26200	-	-	00:29:55	3691	-
23	5	12	12	203659	17115	16932	-	-	00:13:02	696	-
24	5	16	12	375154	29355	29152	-	-	00:17:20	1801	-

6. Conclusion

This paper describes QCASP, a mathematical formulation of the combined problem of Quay Crane Assignment and Quay Crane Scheduling. It is a mixed integer programming model which allows quay cranes to move between two holds of the same ship and between two holds on different vessels. The time it takes for these cranes to move between holds is taken into account in the optimisation process. Interference between quay cranes is avoided by introducing non-interference constraints which consider both the potential interference of tasks on the same vessel and those on different vessels. The Branch-and-Cut algorithm as implemented in CPLEX 12.6 has been used to find the optimal solutions of relatively small instances of QCSAP. It cannot cope with larger instances of the problem which are of practical size. GA, however, coped well with all problems. It required almost the same CPU time for all problems of small size and CPU times of the same magnitude for the larger instances. Moreover, on most of the 17 instances that CPLEX managed to solve, GA also found the optimum. Overall the average discrepancy between the objective function values of the solutions found by both algorithms is 1.3. This shows that GA, while substantially more efficient than B&C, it is also quite robust on most instances considered as this average discrepancy shows.

The combined model presented here is obviously the way forward as it is more likely to provide better solutions than those found by solving seaside operations problems individually. It is also a substantial improvement on combined variants which do not allow for quay cranes movement between vessels. However, there is scope for doing even better by combining Berth Allocation, Quay Crane Assignment and Quay Crane Scheduling into a single model. We are currently working on this problem and will report our findings in a future paper.

7. Acknowledgment

We would like to thank Tom Corkhill and Stephen Peck of the Port of Felixstowe for providing real instances of the problems considered, as well as practical insights into port operations. We also like to thank the Iraqi Ministry of Higher Education and Scientific Research for sponsoring this work.

8. References

[1] Alsoufi, G., Yang, X., Salhi, A., 2015. A combinatorial Benders' cuts approach to the seaside operations problem in container ports. Presented in the 11th metahueristics international conference, Agadir, Morocco, june 7-10.

[2] Bierwirth, C., Meisel, F., 2009. A fast heuristic for quay crane scheduling with interference constraints. Journal of Scheduling 12 (4), 345-360.

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

[4] Cheong, C.Y., Habibullah, M.S., Goh, R.S.M., Fu, X., 2010. Multi-objective optimization of large scale berth allocation and quay crane assignment problems. In: 2010 IEEE International Conference, Systems, Man and Cybernetics (SMC).

[5] Chung, S., Choy, K.L., 2012. A modified genetic algorithm for quay crane scheduling operations. Expert Systems with Applications 39 (4), 4213-4221.

[6] Daganzo, C.F., 1989. The crane scheduling problem. Transportation Research Part B: Methodological 23 (3), 159-175.

[7] Feo, T.A., Resende, M.G., 1995. Greedy randomized adaptive search procedures. Journal of Global Optimization 6 (2), 109-133.

[8] Garey, M.R., Johnson, D.S., 1979. Computers and intractability: a guide to NP-completeness. WH Freeman New York.

[9] Gen, M., Cheng, R., 1997. Genetic algorithms and engineering design. John Wily and Sons, New York.

[10] Guan, Y., Cheung, R.K., 2004. The berth allocation problem: models and solution methods. OR Spectrum 26 (1), 75-92.

[11] Holland, J.H., 1975. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. U Michigan Press.

[12] Kaveshgar, N., Huynh, N., Rahimian, S.K., 2012. An efficient genetic algorithm for solving the quay crane scheduling problem. Expert Systems with Applications 39 (18), 13108-13117.

[13] Kim, K.H., Park, Y.M., 2004. A crane scheduling method for port container terminals. European Journal of Operational Research 156 (3), 752-768.

[14] Lee, D.-H., Wang, H.Q., Miao, L., 2008. Quay crane scheduling with non-interference constraints in port container terminals. Transportation Research Part E: Logistics and Transportation Review 44 (1), 124-135.

[15] Legato, P., Gullì, D., Trunfio, R., 2008. The quay crane deployment problem at a maritime container terminal. In: proceedings of the 22th European Conference on Modelling and Simulation.

[16] Meisel, F., Bierwirth, C., 2009. Heuristics for the integration of crane productivity in the berth allocation problem. Transportation Research Part E: Logistics and Transportation Review 45 (1), 196-209.

[17] Moccia, L., Cordeau, J.-F., Gaudioso, M., Laporte, G., 2006. A Branch-and-Cut algorithm for the quay crane scheduling problem in a container terminal. Naval Research Logistics (NRL) 53 (1), 45-59.

[18] Nguyen, S., Zhang, M., Johnston, M., Chen Tan, K., 2013. Hybrid evolutionary computation methods for quay crane scheduling problems. Computers & Operations Research 40 (8), 2083-2093.

[19] Peterkofsky, R.I., Daganzo, C.F., 1990. A Branch-and-Bound solution method for the crane scheduling problem. Transportation Research Part B: Methodological 24 (3), 159-172.

[20] Rodríguez, J.A.V., Salhi, A., 2006. A synergy exploiting evolutionary approach to complex scheduling problems. In: Computer Aided Methods in Optimal Design and Operations, Series on Computers and Operations Research, World Scientific Publishing Co. Pvt. Ltd 59-68.

[21] Rodríguez, J.A.V., Salhi, A., 2006. Hybrid Evolutionary Methods for the Solution of Complex Scheduling Problems. In: Research in Computing Science: Advances in Artificial Intelligence 20, 17-28.

[22] Salhi, A., Fraga, E.S., 2011. Nature-inspired optimisation approaches and the new plant propagation algorithm. In: proceedings of the ICeMATH2011, K2-1–K2-8.

[23] Salhi, A., Rodríguez, J.A.V., 2014. Tailoring hyper-heuristics to specific instances of a scheduling problem using affinity and competence functions. Memetic Computing 6 (2), 77-84.

[24] Sammarra, M., Cordeau, J.-F., Laporte, G., Monaco, M.F., 2007. A tabu search heuristic for the quay crane scheduling problem. Journal of Scheduling 10 (4-5), 327-336.

[25] Steenken, D., and Voß, S., Stahlbock, R., 2004. Container terminal operation and operations research-a classification and literature review. OR spectrum 26 (1), 3-49.

[26] Tavakkoli-Moghaddam, R., Makui, A., Salahi, S., Bazzazi, M., Taheri, F., 2009. An efficient algorithm for solving a new mathematical model for a quay crane scheduling problem in container ports. Computers & Industrial Engineering 56 (1), 241-248.

[27] Yang, C., Wang, X., Li, Z., 2012. An optimization approach for coupling problem of berth allocation and quay crane assignment in container terminal. Computers & Industrial Engineering 63 (1), 243-253.