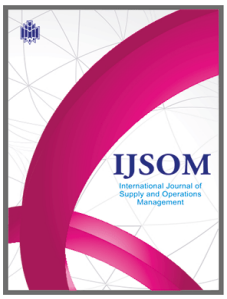
International Journal of Supply and Operations Management

IJSOM

November 2014, Volume 1, Issue 3, pp. 328-346
ISSN-Print: 2383-1359
ISSN-Online: 2383-2525
www.ijsom.com



A new approach for sequencing loading and unloading operations in the seaside area of a container terminal

Azza Lajjam[a*], Mohamed El merouani[a], Yassine Tabaa[a], Abdellatif Medouri[a]

[a]*College of Sciences, Abdelmallek Essaadi University, Tetuan, Morocco*

Abstract
Due to the considerable growth in the worldwide container transportation, optimization of container terminal operations is becoming highly needed to rationalize the use of logistics resources. For this reason, we focus our study on the Quay Crane Scheduling Problem (QCSP), which is a core task of managing maritime container terminals. From this planning problem arise two decisions to be made: The first one concern tasks assignment to quay crane and the second one consists of finding the handling sequence of tasks such that the turnaround time of cargo vessels is minimized. In this paper, we provide a mixed-integer programming (MIP) model that takes into account non-crossing constraints, safety margin constraints and precedence constraints. The QCSP has been shown NP-complete; thus, we used the Ant Colony Optimization (ACO), a probabilistic technique inspired from ants' behavior, to find a feasible solution of such problem. The results obtained from the computational experiments indicate that the proposed method is able to produce good results while reducing the computational time.

Keywords: Quay Crane; Scheduling; Loading and unloading operations; Ant colony optimization; Mathematical formulation; Container terminal.

1. Introduction

Nowadays, container terminals play an essential role as an intermodal interface in a port. Its main function is the effective container transshipment between various transportation modes that are maritime transportation (container vessels) and land transportations (trucks and trains). Exported containers are received from shippers for loading onto vessels and imported containers are discharged from vessels for picking up by consignees. Consequently, container terminals are called

* Corresponding author email address: alajjam@uae.ac.ma

to operate in the most efficient way, that is, as fast as possible, at the least possible cost.

Several major container terminal operations influence the port efficiency, which include the vessel berthing operation, crane unloading/loading operations, container delivery operations using trucks, inspection operations and container storage operations. The productivity of quay cranes is considered as one of the most important determinants of container handling efficiency. in fact, quay cranes are responsible for loading and unloading operations for container vessels. Therefore, our paper deals essentially with the Quay Crane Scheduling Problem (QCSP), which is considered as one of the most important problems that affects the container throughput and handling efficiency. Its main objectives are first to find the assignment of tasks to quay cranes (Lajjam et al., 2014a) and then determine the tasks sequence for each quay crane in order to minimize the time handling while respecting certain constraints, namely spatial constraints and precedence constraints. Spatial constraints consist, first, of respecting the safety margin between adjacent cranes, and secondly avoiding intercrossing among quay cranes that operate on the same tracks and consequently cannot cross each other. In order to respect the non-crossing constraint, Legato et al (Legato et al., 2012) proposed the unidirectional schedules which consider that cranes move along the same direction, either from bow to stern or from stern to bow, while serving the vessel. Precedence relations among tasks can be given to ensure that unloading precedes loading and to represent the stacking of containers as defined by stowage plan. A survey dedicated to the research on QCSP is provided by Bierwirth and Meisel (Bierwirth and Meisel, 2010).

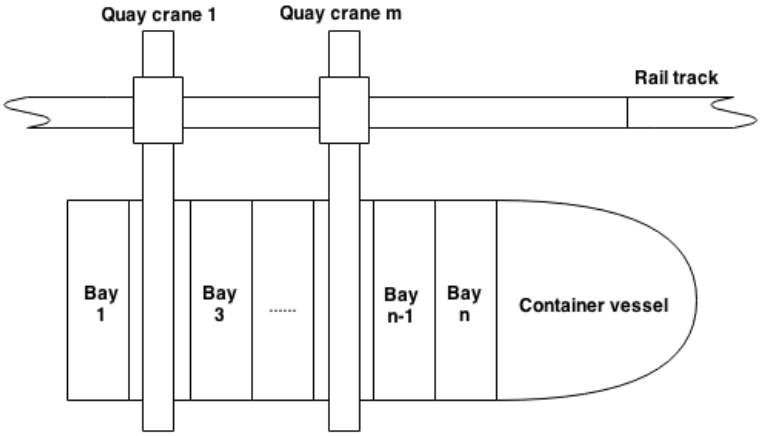


Figure 1. Illustration of QCSP

As shown in Fig.1, a vessel is divided longitudinally into multiple bays in which containers are stored on holds and decks. Holds are about eight containers deep and decks are about six containers high. The vessel could have between 10 to 50 bays. Practically, containers are stored in these bays based on certain criteria (e.g., size, weight, origin and destination). According to Unsal and Oguz (Unsal and Oguz, 2013), there are two classes of the QCSP which are QCSP with complete bays and QCSP with container groups. For the former, a single task is composed of all unloading and loading operations in a single bay. For the latter, i.e. QCSP with container groups, a task represents loading or unloading operations for a group of containers named a cluster, which refer to a collection of adjacent slots holding the containers with the same destination port, the same size, etc.

The main aim of this paper is to apply Ant Colony Optimization (ACO) algorithm to resolve a

part of the QCSP which is the sequencing of loading and unloading operations. The rest of the paper is organized as follows. Section 2 provides a literature review on the QCSP, focusing on the models and the resolution approaches developed to deal with such problems. In section 3, after a formal description of the QCSP, the mathematical formulation of the problem is provided. Section 4 is dedicated to the proposed resolution method. Section 5 presents the numerical results obtained from the experimentation of our proposed method. Finally, we conclude our work before we draw the future research perspective in the last section.

2. Literature review

QCSP is known as one of the most challenging topics that is widely studied in the literature because of its complexity and its practical applicability. Consequently, several studies on QCSP have been conducted in order to improve the performance of container terminal operations. Daganzo(1989) was the first to introduce QCSP in the literature. He provided an exact method which helps to determine the number of cranes to assign to ship-bays of multiple vessels. In a related study, Daganzo and Peterkofsky (Peterkofsky and Daganzo, 1990) resolved the problem with a branch and bound algorithm. They attempted to minimize the delay cost as a criterion. However, both studies considered neither the interference among QCs nor the precedence relationships between tasks. These two kinds of constraints, create the most fundamental difference between the QCSP and the traditional parallel machine scheduling problem(Chen et al., 2014).

In(Kim and Park, 2004), Kim and Park proposed a mixed integer programming that considers precedence constraints as well as non-crossing constraints to avoid congestion between QCs. In order to solve the problem, they first provided a branch and bound method to obtain the optimal solution of the QCSP problem, and then they experimented an heuristic algorithm, called greedy randomized adaptive search procedure (GRASP) in order to obtain high-quality schedules through shorter computational times. Moreover, Moccia et al (Moccia et al., 2006) ameliorated the model proposed by Kim and Park by proposing a branch and cut algorithm to solve large instances.

In another way, Sammarra et al (Sammarra et al., 2007) spited up the QCSP in two sub-problems. The first sub-problem is a routing problem that consists of finding the sequence of tasks that have to be fulfilled by each crane. The second problem concerns the scheduling of these tasks by calculating the starting and finishing times of the tasks. A taboo search is developed in order to solve the routing problem while the scheduling problem is solved by a local search method. Compared with the Greedy randomized adaptive search procedure (GRASP) proposed by Kim and Park, their algorithm's results outperformed GRASP.

on the other hand, Meisel and Bierwirth (Meisel and Bierwirth, 2011) proposed a dedicated platform to evaluate the performance of different model classes and solution procedures. Concretely, they provided a set of benchmark instances with different values for number of quay cranes, number of tasks, task location, and safety margin among the quay cranes. The benchmark instances are available from this link http://prodlog.wiwi.unihalle.de/qcspgen.

Tavakkoli et al (Tavakkoli-Moghaddam et al., 2009) considered the integrated quay crane scheduling and allocation problem(QCSAP) by extending the model proposed by Kim and Park (Kim and Park, 2004) to a model based on a set of vessels in parallel. in fact, they provided a MIP model as a formulation of the proposed problem. Then, they proposed a genetic algorithm

(GA) to obtain a near optimal solution for this complex and large-scale problem. Kaveshgar et al (Kaveshgar et al., 2012) also provided a genetic algorithm (GA) to solve the QCSP. They proposed a new resolution method which reduces the number of decision variables, while it improves the efficiency of the GA using an initial solution based on the S-LOAD rule which was developed by Sammarra et al (Sammarra et al., 2007).

Exposito-Izquierdo et al (Expósito-Izquierdo et al., 2013) studied the QCSP in environments composed of container groups in order to minimize the overall container vessel service time noted as the *makespan*. The contribution of these authors was in form of the development of a hybrid algorithm that combines two methods, namely Estimation of Distribution Algorithm (EDA) and Local Search (LS). Chen et al (Chen et al., 2014) proposed a MIP model which can be solved easily by any standard optimization solver. Their model is based on the unidirectional cluster QCSP. It means that they consider the loading/unloading operations of a group of containers and the fact that cranes move in the same direction.

Diabat and Theodorou (Diabat and Theodorou, 2014) studied the integrated quay crane assignment and scheduling problem (QCASP) which yields better results than solving these problems independently. They proposed a mathematical formulation that considers quay crane positioning conditions, like end-bay positioning and non-crossing constraints. Then, they developed a genetic algorithm (GA) based on a new shorter chromosome representation which always generates feasible solutions. The integrated quay crane assignment and scheduling problem was also studied by Yi-Min et al (Fu et al., 2014). They provided a bidirectional model that takes into account practical constraints such as safety margins and QC ordering using the genetic algorithm (GA) to resolve this problem.

Recently, Kaveshgar and Huynh (Kaveshgar and Huynh, 2014) studied the integrated quay crane and yard truck scheduling problems. In fact, they developed a MIP model for the both problems jointly which takes into consideration a large number of real operational constraints such as precedence, blocking, interference and safety margin. They provided a hybrid method that combines the genetic algorithm (GA) with the greedy algorithm.

In this work, our main goal is to focus our study on the sequencing of loading and unloading operations which is considered as a part of the Quay Crane Scheduling Problem (QCSP). Concretely, we will provide a MIP model which differs from Kim's (Kim and Park, 2004) model by enhancing the objective function in a way that minimizes the total cost. Then, we will propose a innovative approach based on ACO algorithm to find the optimized sequence in a reasonable time compared to an exact method usually used in the resolution of such problems, namely, the branch and bound method.

3. Problem description and formulation

3.1. Problem description

Considering the unidirectional cluster-based QCSP, and given a set of tasks $\Omega=\{1,\ldots n\}$ which are located at a set of bays $B=\{1,\ldots,b\}$ and a set of identical quay cranes $Q=\{1,\ldots,q\}$. Each task $i \in \Omega$ represents a loading or unloading operation of a container group. Each task has a

processing time p_i which represents the time required to complete task *i* and a bay position $l_i, l_i \in B$. From an operational point of view, a single QC without preemption must process each individual task.

The set Φ contains pairs of tasks that have a precedence relationship, i.e. for each $(i,j) \in \Phi$, task *i* must be completed before task *j* starts. It ensures that the discharging tasks precede the loading tasks; when a discharging operation is performed in a bay, the tasks on the deck must be accomplished before the tasks in the hold of the same bay; conversely, the loading tasks in the hold must precede the loading tasks on the deck of the same bay. The set Ψ contains pairs of tasks that cannot be processed simultaneously. Moreover, QCs are operated on the same rail track along the quay; consequently, several QCs cannot operate at the same bay simultaneously and they cannot cross each other. Thus, we suppose that QCs move in the same direction to ensure non-crossing constraint. Furthermore, a safety distance (measured in bay units) has to be kept continuously in order to prevent collisions between QCs.

3.2. Problem formulation

We use the following notations for the mathematical formulation

Indices:

i,j : Tasks indices which are ordered in an increasing order according to their locations on the ship-bay

q: QCs where *k*=1,…,*K*. QCs are also ordered in an increasing order of their relative locations in the direction of increasing ship-bay numbers

Sets:

Ω :Set of tasks $\Omega = \{1, \ldots, n\}$

Φ: Set of task pairs (*i,j)* (task *j* cannot start before the completion of task *i*)

Ψ: Set of task pairs that cannot be processed simultaneously

Problem data:

p_j: Processing time of task *j*

β_k : Fixed cost of using QC *k*

α_k : Variable cost of using QC *k*

w_j : Tardiness cost of task *j*

d_j : Due date of task *j*

r_k: Release time of QC k

l_i: Bay position of task i

t_{ij}^k: The travel time of a QC from bay position l_i of task *i* to bay position l_j of task j

M : A sufficiently large constant

Decision variables:

X_{ij}^k=1, if task *j* immediately follows task *i* on QC *k*; 0, otherwise. Tasks 0 and T will be considered to be the initial and final states of each QC, respectively. Thus when task *j* is the first task of QC *k*, $X_{0j}^k = 1$ Also, when task *j* is the last of QC *k*, $X_{jT}^k = 1$.

$Y_{ij} = 1$, if task *j* starts later than the completion time of task *i*; 0, otherwise.

$Z_k = 1$, if QC *k* is selected; 0, otherwise.

Q_k : Completion time of QC *k*.

C_i : Completion time of task *i*.

C_{max} : *Makespan*

As in Lajjam et al. (Lajjam et al., 2013), the QCSP problem can be formulated as follows :

$$\sum_{k=1}^{K}(\alpha_k Q_k + \beta_k Z_k) + \sum_{i=1}^{n} w_i \max\{0, c_i - d_i\} \quad (1)$$

Subject to

$$Q_k \leq C_{max} \qquad \forall k = 1, \dots, K \tag{2}$$

$$\sum_{j\in\Omega} X_{0j}^k = 1 \qquad \forall k = 1, \dots, K \tag{3}$$

$$\sum_{i\in\Omega} X_{iT}^k = 1 \qquad \forall k = 1, \dots K \tag{4}$$

$$\sum_{k=1}^{K} \sum_{i\in\Omega} X_{ij}^k = 1 \qquad \forall j \in \Omega \tag{5}$$

$$\sum_{j\in\Omega} X_{ij}^k - \sum_{j\in\Omega} X_{ji}^k = 0 \qquad \forall i \in \Omega, \forall k = 1, \dots, K \tag{6}$$

$$C_i + t_{ij}^k + p_j - C_j \leq M\left(1 - X_{ij}^k\right) \qquad \forall i, j \in \Omega, \forall k = 1, \dots, K \tag{7}$$

$$C_i + p_j \leq C_j \qquad \forall (i,j) \in \Phi \tag{8}$$

$$C_i - C_j + p_j \leq M(1 - Y_{ij}) \tag{9}$$

$$Y_{ij} + Y_{ji} = 1 \qquad \forall (i,j) \in \tag{10}$$

$$\sum_{v=1}^{k} \sum_{u\in\Omega} X_{uj}^v - \sum_{v=1}^{k} \sum_{u\in\Omega} X_{ui}^v \leq M\left(Y_{ij} + Y_{ji}\right) \qquad \forall i, j \in \Omega, l_i < l_j \ , \forall k = 1, \dots, K \tag{11}$$

$$C_j + t_{jT}^k - Q_k \leq M\left(1 - X_{jT}^k\right) \qquad \forall j \in \Omega, \forall k = 1, \dots, K \tag{12}$$

$$r_k - C_j + t_{0j}^k - p_j \leq M\left(1 - X_{0j}^k\right) \qquad \forall j \in \Omega, \forall k = 1, \dots, K \tag{13}$$

$$X_{ij}^k, Y_{ij}, Z_k = 0 \text{ or } 1 \quad \forall i, j \in \Omega, \forall k = 1, \dots, K \tag{14}$$

$$Q_k, C_i \geq 0 \forall i \in \Omega, \forall k = 1, \dots, K \tag{15}$$

In this model, the objective function is composed of two terms. The first term is the QC holding cost and the second term is the total cost of tardiness penalties. This objective function reflects the balance between the system cost and the cost related to the job tardiness penalties. Constraint (2) determines the *makespan* that corresponds to the completion time of all tasks. Constraints (3) and

(4) define the first and the last tasks for each QC, respectively. Constraint (5) ensures that every task must be completed by exactly one crane. Constraint (6) guarantees that, on each QC, tasks are performed in well-defined sequences. Constraint (7) determines the completion time for each task. Constraint (8) ensures that task *i* should be completed before task *j* if there is a precedence relationship between them. Constraint (9) defines Y_{ij} such that $Y_{ij} = 1$ when task *j* starts after task *i* is completed. Constraint (10) guarantees that tasks *i* and *j* cannot be processed simultaneously if (*i*,*j*) belongs to the non-simultaneity set. By constraint (11), interference among QCs can be avoided. Constraint (12) defines the completion time of each QC. Constraint (13) ensures that the first task of a crane is not started before the crane is ready. Finally, (14) and (15) define the domains of the decision variables.

4. Resolution approach

In practice, the number of containers to be loaded/unloaded on or from the vessel is often very large. For practical size, the model presented in the last section cannot be solved using standard software within reasonable time. Otherwise, as QCSP is NP-hard problem, we proposed a meta-heuristic widely used for solving the scheduling problem which is the Ant Colony Optimization (ACO) (Lajjam et al., 2014b).

4.1. Ant Colony Optimization

This study proposes an ant colony optimization (ACO) algorithm to solve the quay crane scheduling problem. The ACO algorithm is a meta-heuristic introduced by Dorigo et al(Dorigo et al., 1996) and it is widely used for solving combinatorial optimization problems such as the traveling salesman problem, quadratic assignment problem, production scheduling, time-tabling, vehicle routing, telecommunications routing and investment planning. Chandra Mohan and Baskaran (Chandra Mohan and Baskaran, 2012) provided an exhaustive survey about ACO and its implementations on several engineering domain.

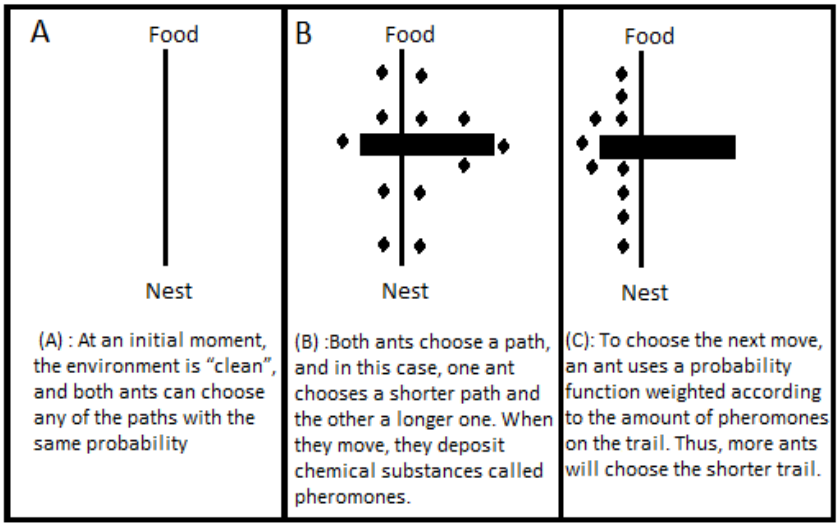


Figure 2. Basic Ant Colony Optimization Behavior

As shown in Fig.2, ants leave a chemical pheromone trail on the ground that can be smelled by ants while moving. When the cycle repeats itself, the shorter path will have a stronger pheromone trail more quickly as a result of a higher number of trips occurring on shorter routes in a unit of time. To choose the next move, an ant uses a probability function weighted according to the amount of pheromones on the trail. Thus, more ants will choose the shorter trail. After a certain time, the first pheromones that were dropped in the long path evaporate, and the pheromone trail on the shorter path becomes dominant. In this case, all the ants will choose the shorter trail. As a conclusion, we can say that the pheromone trails will guide other ants to the food source.
Practically, ACO algorithm is inspired from that behavior of real ants to find the shortest route between their nests and target food locations. The artificial ants seek the solutions according to a constructive procedure. Solutions are built in a probabilistic way by taking into account the pheromone trails, denoted by τ_{ij},which change in a dynamic way to reflect the experience acquired by the agents. Furthermore, the construction of solutions is influenced by the heuristic information, denoted by η_{ij}, which depends on the problem type.

Several versions of ant-based algorithms were developed to solve scheduling problems such as parallel machine scheduling, flow shop and job shop scheduling. Tavares Neto and Godinho Filho (Tavares Neto and Godinho Filho, 2013) provided a rich literature review about the application of ACO algorithm for the resolution of scheduling problem. Recently, this meta-heuristic is extended to deal with multi-objective problem.

4.2. Solution construction

Let Ω={1,…,n} be the set of jobs, B={1,…,b} the set of bays and $J_b = \{1, \dots, J\}$ the set of jobs in bay *b*. During the construction of the solution, the ants will select a bay $b \in B$ first, and then select a job $i \in J_b$ to be scheduled, until all jobs are scheduled, while respecting the precedence constraint. As shown in fig.3, the solutions generated by the ACO algorithm are represented in a graph composed of two types of nodes (Keskinturk et al., 2012). Super-nodes represent ship bays and nodes represent the handling jobs in this bay, insofar that each super-node encloses a number of nodes. Every node in a super-node can be connected to all the other nodes in other super-nodes. We also consider a dummy node as the start and end point of an ant's tour which is connected to every other node on the graph. As a result, the graph has a dummy node, $|B|$ supernodes and $|J_b||B|$ nodes. In order to construct a solution, the artificial ant travels to each super-node, where it visits a sequence of nodes and then returns back to the dummy node when it completes the tour on the graph. We consider tab.1 as the input of our algorithm. It contains the sufficient

information that allowed the generation of a schedule representing a sequence of tasks.

Table 1. An example of the input data of QCSP

Quay Crane ID	Bay ID	Job ID	Number of containers	Operation type	Location
109	1	1001	14	D*	Deck
109	1	1010	6	D	Deck
109	1	1017	9	L*	Deck
109	1	1026	9	L	Deck
109	2	2005	21	D	Hold
109	2	2007	12	L	Hold
109	2	2009	7	D	Deck
109	2	2020	3	L	Deck
109	2	2023	6	L	Deck
109	5	5001	9	D	Deck
109	5	5012	9	D	Hold
109	5	5015	18	L	Deck

* D (Discharging), L (Loading)

To illustrate this route construction, we consider Fig.2 that represents the sequencing of tasks gathered from tab.1 where tree bays are represented. In the first bay, we have four jobs to be scheduled; five jobs in the second bay and tree jobs in the third bay. The path of the artificial ant, represented by arcs, is (0), (1,1), (1,10), (1,26), (1,17), (2,5), (2,9), (2,7), (2,20), (2,23), (3,1), (3,12), (3,15), (0), where (*l*,*i*) is (bay, job). This route results in the following solution: we start with the first bay and we schedule job 1, job 10, job 26 and then job 17; next, we pass to the second bay and schedule job 9, job 7, job 20 and job 23; finally, we pass to the bay 3 and we schedule job 1, job 12 and job 15.

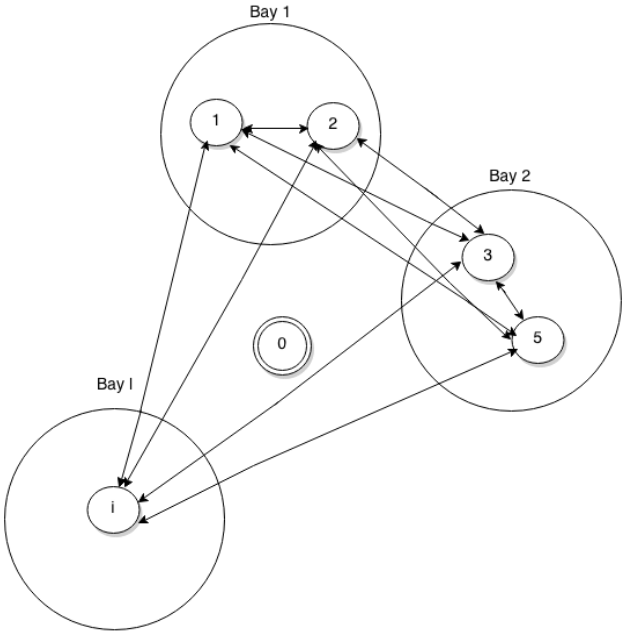


Figure 3. Network of sequencing loading and unloading operations on ACO

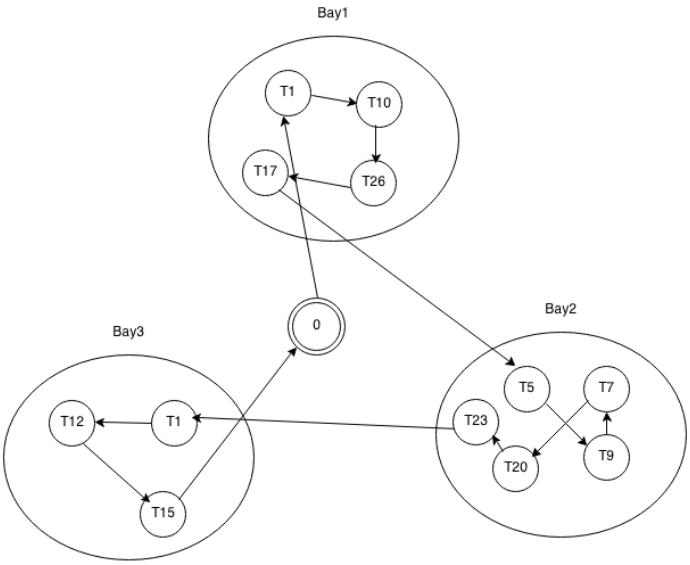


Figure 4. Example of ant tour using ACO

4.2.1. Bay selection

In bay selection, an ant selects a bay by using heuristic information that favors the lowest-indexed bay. Let q_i be a random number in [0,1] and q_{i0} a parameter $(0 \leq q_{i0} \leq 1)$. The transition rule to select bay l for an ant is defined by:

$$l = \begin{cases} argmax_{l \in Tabu_b}\{\varphi_{bk}\} & if\ q_l \leq q_{l0} \\ L & Otherwise \end{cases} \quad (16)$$

The parameter q_{i0} weighs the relative importance of exploitation versus exploration: every time an ant has to choose bay *l*, it samples a random number q_l. If $q_l \leq q_{l0}$, then exploitation is favored and the ant selects bay *l* with maximum desirability φ_{bl}. Otherwise, an ant randomly selects bay *L* by favoring exploration. L is a random variable having the probability P_{bl} defined by

$$P_{bl} = \frac{\varphi_{bl}}{\sum_{k \in Tabu_b} \varphi_{bk}} \quad ,\forall l \in B \quad (17)$$

$\varphi_{bl}$is the desirability to select a bay *l*, used as heuristic information. It can be represented as the reciprocal of the bay index given by

$$\varphi_{bl} = \frac{1}{l} \quad (18)$$

4.2.2. Job selection

Once the bay *l* is selected, an ant has to select job *i* to be fulfilled in that bay. The job *j* is selected according to the transition rule defined by:

$$i = \begin{cases} argmax_{j \in Tabu_T}(\tau_{ij})^{\alpha}.(\eta_{ij})^{\beta} & if\ q_i \leq q_{i0} \\ I & otherwise \end{cases} \quad (19)$$

Where q_{i0} is a user-specified parameter to control the relative influence between exploitation and exploration as for the bay selection. In fact, when an ant has to choose job *i*, it samples a random number q_i in [0,1]. If $q_i \leq q_{i0}$, then exploitation is favored and the ant selects job *i* with maximum desirability, otherwise, an ant randomly selects job *i* by favoring exploration. *i* is a random variable having the probability distribution formed by the probabilities defined by

$$P_{ij} = \frac{(\tau_{ij})^{\alpha}.(\eta_{ij})^{\beta}}{\sum_{l \in Tabu_T}(\tau_{il})^{\alpha}.(\eta_{il})^{\beta}} \quad (20)$$

Where *Tabu* is the set of unscheduled jobs. $\tau_{ij}$is the pheromone quantity deposited by ants on the arc *(i,j)* which measures the acquired desirability between the selected bay *l* and the unscheduled

job *i*. The equation 21 gives the heuristic information η_{ij} which favored the shortest processing time.

$$\eta_{ij} = \frac{1}{p_i} \quad (21)$$

p_i is the processing time of job *i*. α and β are two parameters that manage the relative importance of pheromone trails and heuristic information, respectively.

4.2.3. Local pheromone update

Initially, all the components (*i*,*j*) are initialized to weak amounts of pheromone ($\tau_{ij} = \tau_0$). During the solution construction, the pheromone quantity of component (*i*,*j*) visited by ant is locally updated using the rule given by

$$\tau_{ij} = (1-\rho)\tau_{ij} + \rho\tau_0 \quad (22)$$

With ρ the evaporation rate that represents the evaporation of trail between iteration.

4.2.4. Global pheromone update

The global pheromone update can be presented by

$$\tau_{ij} = (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij} \quad (23)$$

Let T be the sum of QC holding cost and tardiness penalty cost of all tasks visited by the ant *k* at iteration *t*. Then, after iteration *t* is completed, the pheromone level on ant γ's path is updated using the following formula:

$$\Delta\tau_{ij}^{\gamma}(t) = \begin{cases} \frac{1}{T} & if\ ant\ \gamma\ travels\ on\ edge\ (i,j) \\ 0 & Otherwise \end{cases} \quad (24)$$

$$T = \sum_{k=1}^{K}(\alpha_k Q_k + \beta_k Z_k) + \sum_{i=1}^{N} w_i \max\{0, c_i - d_i\} \quad (25)$$

The overall change in the pheromone level as a result of the ants is calculated as:

$$\Delta\tau_{ij}(t) = \sum_{\gamma=1}^{\Gamma} \Delta\tau_{ij}^{\gamma}(t) \quad (26)$$

4.2.5. The proposed algorithm SACO

In this section, we give the pseudo code of the algorithm SACO to optimize the sequencing of jobs for each quay crane using the rules and concepts described in the previous sections.

Algorithm SACO

Set parameters N_c(iteration number),N_a(ants number),$\alpha, \beta, q_{i0}, q_{l0}, \tau_0, \rho$

Initialize c=1 // c is the iterations counter;

Initialize$\tau_{ij} = \tau_0, \forall(i,j), i = 1$ to *m*and *j=1* to n

Repeat

For each ant

Do

Repeat

Step 1// Bay selection

Generate$q_l, q_l \sim U[0,1]$

Select bay *l* with the following transition rule:

$$l = \begin{cases} argmax_{l \in Tabu_b}\{\varphi_{bl}\} & if\ q_l \leq q_{l0} \\ L & Otherwise \end{cases}$$

Where $\varphi_{bl} = \frac{1}{l}$,$\forall l \in B$ and L $\sim$ probability distribution with

$$P_{bl} = \frac{\varphi_{bl}}{\sum_{k \in Tabu_b} \varphi_{bk}} \quad , \forall l \in B$$

Step 2 // job selection

Generate$q_i \sim$U[0,1]

Select the job *i* for the selected bay *l* by the following transition rule

$$i = \begin{cases} argmax_{j \in Tabu_T}(\tau_{ij})^{\alpha}.(\eta_{ij})^{\beta} & if\ q_i \leq q_{i0} \\ I & otherwise \end{cases}$$

Where $\eta_{ij} = \frac{1}{p_i}$ and $I \sim$ probability distribution with:

$$P_{ij} = \frac{(\tau_{ij})^{\alpha}.(\eta_{ij})^{\beta}}{\sum_{l \in Tabu_T}(\tau_{il})^{\alpha}.(\eta_{il})^{\beta}}$$

Until *j* > *n* // all the *n* jobs are scheduled;

Pheromone local update using the equation;

$$\tau_{ij} = (1-\rho)\tau_{ij} + \rho\tau_0$$

End do

Step 3 Pheromone global update using the equation;

$$\tau_{ij} = (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij}$$

Until ($c > N_c$) // Maximum number of iterations reached;

4.3. Tuning the algorithm parameters

The ACO algorithm contains a number of parameters that must be adjusted in order to obtain the best results in terms of solution cost and convergence time of the algorithm. Thus, we identified 7 parameters of the algorithm that are presented in tab.2. These parameters are all continuous, therefore lower and higher bounds of values ranges that can be taken during the design of experiments study are associated to them.

Table2. ACO parameters

Parameter	Higher bound	Lower bound	Default value
α	0.5	3	1
β	0.5	3	0.5
φ_0	0.1	2	0.2
τ_0	0.1	2	1.5
ρ	0.05	0.3	0.25
N_c	80	120	100
N_a	60	120	100

5. Experimental environment and results

In this section, we experiment the performance of our SACO approach with the exact method called branch and bound. Thus, we consider 10 numerical examples for two problem sizes: small

and medium size. Small and medium size examples were solved optimally by a branch and bound (B&B) method under the LINGO 14.0 software while our SACO algorithm was implemented in MATLAB. We used a personal computer to carry out all the experiences; it is a Dell laptop with an Intel Core i3-330M processor and 3 GB of RAM.

Table 3. Computational results of random instances in small sizes

Experiment No	Problem information		B&B		SACO		GAP (%)
	Bay no	Job no	CPU time (s)	OFV	CPU time (s)	OFV	
1	2	3	4	480	3	480	0
2	3	3	9	514	4	517	0,6
3	2	5	18	579	7	581	0,3
4	3	5	37	588	9	590	0,3
5	2	7	172	556	11	559	0,5
6	3	7	860	864	13	879	1,7
7	4	7	2580	882	18	895	1,5
8	2	9	3787	884	20	896	1,3
9	3	9	4080	902	22	919	1,8
10	4	9	4220	912	24	935	2,5

Table 4. Computational results of random instances in medium sizes

Experiment No	Problem information		B&B		SACO		GAP (%)
	Bay no	Job no	CPU time (s)	OFV	CPU time (s)	OFV	
1	5	12	7130	1404	74	1430	1,8
2	6	12	8025	1456	96	1493	2,5
3	7	12	9120	1513	114	1558	2,9
4	5	16	10650	1570	143	1597	1,7
5	6	16	10800	1728	165	1778	2,8
6	7	16	10980	1804	182	1874	3,7
7	5	21	TO*	___	195	5446	___
8	6	21	TO*	___	225	5475	___
9	7	21	TO*	___	244	5514	___
10	8	21	TO*	___	263	5561	___

***** Timeout before reaching a solution

We proceed to the resolution of all the examples according to the two approaches, namely branch and bound (B&B) method and the proposed SACO method. Then we proceed to the comparison of both obtained results in terms of two indicators: the CPU time and the objective function value

mentioned here as OFV readings from the average results of ten experiments for each problem. We define a timeout execution of 12000 seconds to finish processing.

Tab3. presents the obtained results from the execution of random instances in small sizes, as shown, we can observe that the time needed to find a solution using an implementation of the B&B version of the QCSP in the LINGO grows randomly as the instance size increases. However, when we use the proposed SACO, we obtain the near-optimal solution in a practical time (few seconds in comparison with B&B results). We also note that the average GAP between the two approaches in terms of the OFV is approximately 1% and a standard deviation of 0.78 something that is very promising. In the second experimentation, we chose to execute random medium size instances.Tab.4 illustrates the obtained results. Thus, the associated processing times to optimize the fitness function using branch and bound (B&B) implementation in the LINGO software grow exponentially until exceeding the threshold defined as a timeout execution, while the proposed SACO implementation continues to demonstrate its efficiency even for medium size instances. Therefore, the average gap between our proposed SACO and the best solutions obtained using branch and bound method (for problems which the processing doesn't exceed timeout) is about 2.6%.

6. Conclusions and perspectives

In this paper, we have studied the QCSP, which is considered as one of the most important problems treated in a container terminal. In fact, a number of studies have proposed different meta-heuristics to solve this problem such as Genetic Algorithm (GA). In our case, we have proposed a new approach based on the ACO paradigm, a meta-heuristic widely used for solving scheduling problem. Experimental results using random instances showed that the developed ACO approach provides solutions in faster time for medium problems compared to branch and bound method. The obtained results encourage us to study more complex systems. Moreover, it seems to be very interesting to study the integrated three problems encountered in the seaside area of a container terminal, namely, Berth Allocation Problem (BAP), Quay Crane Assignment Problem (QCAP) and Quay Crane Scheduling Problem (QCSP). In fact, the next studies will focus on the resolution of the integrated Quay Crane Assignment and Scheduling Problem (QCASP) which seems to yield better solution.

References

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

Chandra Mohan,B. & Baskaran,R. (2012) A survey: Ant Colony Optimization based recent research and implementation on several engineering domain. *Expert Systems with Applications, 39, 4618-4627.*

Chen, J .H. Lee, D.-H. & Goh, M. (2014) An effective mathematical formulation for the unidirectional cluster-based quay crane scheduling problem. *European Journal of Operational Research, 232, 198-208.*

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

Diabat, A. & Theodorou, E. (2014) An Integrated Quay Crane Assignment and Scheduling Problem. *Computers & Industrial Engineering. 73, 115-123.*

Dorigo, M.,Maniezzo, V. & Colorni, A. (1996) Ant system: optimization by a colony of cooperating agent. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 26, 29-41*

Exposito- Izuierdo, C., Gonzalez-Velarde, J.L., Melian-Batista, *B. & MARCOS MORENO-VEGA, J. (2013)* Hybrid Estimation of Distribution Algorithm for the Quay Crane Scheduling problem. *Applied Soft Computing. 13, 4063-4076.*

Fu, Y.-M., Diabat, A. & Tsai, I. T. (2014) Amulti-vessel quay crane assignment and schedualing problem: Formulation and heuristic solution approach. *Expert Systems with Applications, 41, 6959-6965.*

Kaveshgar, N. & Huynh, N. (2014) Integrated quay crane and yard truck scheduling for unloading inbound containers. *International Journal of Production Economics*.

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

Keskinturk, T., Yildirim, M. B. & Barut, M. (2012) An ant colony optimization algorithm for load balancing in parallel machines with sequence-dependent setup times. *Computers & Operations Research,39, 1225-1235.*

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

Lajjam, A., El Merouani, M., Medouri, A. & Tabaa, Y. (2013) Mathematical model for Quay Crane Scheduling Problem with spatial constraints. *International Journal of Innovation and Applied Studies,*4, 547-551.

Lajjam, A., El Merouani, M., Tabaa, Y. & Medouri, A. (2014a) An Efficient Algorithm for Solving Quay Crane Assignment Problem. 13-18.

Lajjam, A., El Merouani, M. & Medouri, A. (2014b) Ant colony system for solving quay crane scheduling problem in container terminal. In: *2nd IEEE Conf. on Logistics Operations Management* (Ed. IEEE). Proc IEEE, Rabat, Morocco.

Legato, P., Trunfio, R. & Meisel, F. (2012) Modeling and solving rich quay crane scheduling problems. *Computers & Operations Research,*39**,** 2063-2078.

Meisel, F. & Bierwirth, C. (2011) A unified approach for the evaluation of quay crane scheduling models and algorithms. *Computers & Operations Research,*38**,** 683-693.

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**,** 45-59.

Peterkofsky, R. I. & Daganzo, C. F. (1990) A branch and bound solution method for the crane scheduling problem. *Transportation Research Part B: Methodological,*24**,** 159-172.

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**,** 327-336.

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**,** 241-248.

Tavares Neto, R. F. & Godinho Filho, M. (2013) Literature review regarding Ant Colony Optimization applied to scheduling problems: Guidelines for implementation and directions for future research. *Engineering Applications of Artificial Intelligence,*26**,** 150-161.

Unsal, O. & Oguz, C. (2013) Constraint programming approach to quay crane scheduling problem. *Transportation Research Part E: Logistics and Transportation Review,*59**,** 108-122.