Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2015, Article ID 781907, 12 pages
http://dx.doi.org/10.1155/2015/781907

Research Article

Migrating Birds Optimization for the Seaside Problems at Maritime Container Terminals

Eduardo Lalla-Ruiz, Christopher Expósito-Izquierdo, Jesica de Armas, Belén Melián-Batista, and J. Marcos Moreno-Vega

Department of Computer and Systems Engineering, University of La Laguna, 38271 La Laguna, Spain

Correspondence should be addressed to Eduardo Lalla-Ruiz; elalla@ull.es

Received 13 March 2015; Revised 25 May 2015; Accepted 27 May 2015

Academic Editor: Wei Fang

Sea freight transportation involves moving huge amounts of freights among maritime locations widely spaced by means of container vessels. The time required to serve container vessels is the most relevant indicator when assessing the competitiveness of a maritime container terminal. In this paper, two main logistic problems stemming from the transshipment of containers in the seaside of a maritime container terminal are addressed, namely, the Berth Allocation Problem aimed at allocating and scheduling incoming vessels into berthing positions along the quay and the Quay Crane Scheduling Problem, whose objective is to schedule the loading and unloading tasks associated with a container vessel. For solving them, two Migrating Birds Optimization (MBO) approaches are proposed. The MBO is a recently proposed nature-inspired algorithm based on the V-formation flight of migrating birds. In this algorithm, a set of solutions of the problem at hand, called birds, cooperate among themselves during the search process by sharing information within a V-line formation. The computational experiments performed over well-known problem instances reported in the literature show that the performance of our proposed MBO approaches is highly competitive and presents a better performance in terms of running time than the best approximate approach proposed in the literature.

1. Introduction

Maritime container terminals and container vessels are the main components involved in sea freight transportation, where huge amounts of freights are moved among widely spaced locations. Since the international sea freight trade has undergone a relevant growth over the last few decades (United Nations Conference on Trade And Development, http://www.unctad.org/), maritime container terminals have to better use and schedule their resources in order to efficiently face the operational and technical requirements of shipping companies. In this regard, as indicated by Nicoletti et al. [1] and Expósito-Izquierdo et al. [2], the time required to serve container vessels, since their arrival until their departure, is the most representative indicator used by the shipping companies when assessing the competitiveness of a given maritime container terminal. Moreover, the faster these tasks are made, the earlier the containers are available for withdrawal by the corresponding companies, and therefore there will be a better management of the whole terminal (Yeo [3]).

Several logistic problems are relevant in the productivity of the maritime container terminal. One of the most outstanding problems within them is the Berth Allocation Problem (BAP), whose purpose is to allocate and schedule those vessels arriving to port into berthing positions along the quay in order to optimize some objective function. This problem has been extensively studied in the literature over the last years and a multitude of variations has been proposed (Biertwirth and Meisel [4]). One of these variants is the Dynamic Berth Allocation Problem (DBAP), proposed by Cordeau et al. [5]. It considers berth and vessel time windows as well as heterogeneous vessel service times depending on the assigned berth. The other outstanding optimization problem in maritime container terminal is the Quay Crane Scheduling Problem (QCSP), which is aimed at determining the work schedules of the quay cranes allocated to a given container vessel. It is worth mentioning that solving the

DBAP and the QCSP allows the terminal managers to know how to perform the management of the incoming container vessels during a certain planning horizon. This means knowing the berthing position and berthing time of the vessels and how the quay cranes are handled during the transshipment operations. In this regard, their efficient solution prevents traffic bottlenecks and enhances the competitiveness of the whole infrastructure.

Logistic operations, such as those involved in the DBAP and QCSP, require fast and effective solution approaches due to inherent requirements of the context where they appear. In this backdrop, the usage of metaheuristics to find high-quality feasible solutions is advisable. For this reason, in this work we apply and assess a recent nature-inspired metaheuristic based on the V-formation of the migrating birds, called Migrating Birds Optimization (MBO), proposed by Duman et al. [6]. This metaheuristic is a population-based algorithm, where the individuals, called birds, cooperate among themselves during the search process by sharing information about the explored search space. The way they share the information is by considering a V-formation that establishes the relation among birds.

The main goal of this work is to propose and evaluate the use of the MBO technique for solving the main seaside problems at maritime container terminals. With this goal in mind, we have selected two of the most relevant problems in the related literature, DBAP and QCSP. The computational results as well as the comparison with the best approximate algorithms reported in the literature point out a competitive performance of MBO in terms of objective function value and running time. This latter feature makes MBO suitable and competitive to be included in port decision support systems. It is worth highlighting the significance of the running time in these systems, due to the fact that the aforementioned problems may have to be solved frequently (i) because of its direct link with each other and with problems from other parts of the container terminal and (ii) to include possible changes related to terminal resources and (iii) to assist port managers during the negotiations with shipping companies.

The remainder of this paper is structured as follows. Section 2 introduces the DBAP and QCSP. The MBO is presented in Section 3. Afterwards, Section 4 describes the application of MBO to the seaside problems under analysis in this paper: DBAP and QCSP. Section 5 discusses the computational experiments carried out to assess the suitability of MBO. Finally, Section 6 presents the main conclusions extracted from the work and suggests several directions for further research.

2. Seaside Operations

Seaside operations are those related to the transshipment of containers between the container vessels and the maritime container terminal. In this context, three main problems can be identified.

(i) *Berthing of the Vessels.* Each incoming container vessel has to be assigned to a position along the quay of the container terminal according to its particular characteristics (i.e., length, draft, arrival time, etc.).

(ii) *Allocation of Quay Cranes.* A subset of the quay cranes in the container terminal must be allocated to each berthed vessel in order to perform its loading and unloading operations.

(iii) *Scheduling of the Quay Cranes.* The quay cranes allocated to a given container vessel have be scheduled for performing its transshipment operations in such a way that the stay is the shortest as possible.

As indicated by Bierwirth and Meisel [4], the allocation of quay cranes known as Quay Crane Allocation Problem (QCAP) is tightly related to the BAP due to the fact that the handling times of the vessels depend on the number of quay cranes assigned to them. Consequently, the QCAP is usually jointly considered with the BAP or QCSP. Therefore, in the following, these two relevant logistic problems, namely, the management of berths and the schedule of quay cranes at a terminal when serving container vessels, are addressed.

2.1. Dynamic Berth Allocation Problem. The Dynamic Berth Allocation Problem (DBAP) is an $\mathcal{NP}$-hard problem (Cordeau et al. [5]) that seeks to identify the berthing position and berthing time of the container vessels arriving to port over a well-defined time horizon.

In the DBAP, we are given a set of incoming container vessels, V, and a set of berths, B. Each vessel, $i \in V$, must be assigned to a berth, $k \in B$. Each vessel has a known time window, $[tv_i, tv_i']$. Similarly, each berth has its own time window, $[tb_k, tb_k']$. For each vessel $i \in V$, its service time, s_i^k, depends on the berth $k \in B$, where it is assigned to. That is, the service time of a given vessel may differ from one berth to another. Furthermore, each $i \in V$ has a given service priority, denoted as p_i, according to its contractual agreement with the terminal. It should be noted that the higher this value, the higher the priority of the vessel.

In a more detailed way, the assumptions in the DBAP can be enumerated as follows.

(a) Each berth $k \in B$ can only handle one vessel at a time.

(b) The service time of each vessel $i \in V$ is determined by the assigned berth $k \in B$.

(c) Each vessel $i \in V$ can be served only after its arrival time tv_i.

(d) Each vessel $i \in V$ has to be served until its departure time tv_i'.

(e) Each vessel $i \in V$ can only be berthed at berth $k \in B$ after k becomes available at time step tb_k.

(f) Each vessel $i \in V$ can only be berthed at berth $k \in B$ until k becomes unavailable at time step tb_k'.

In order to present the decision variables, let us define a graph, $G^k = (V^k, A^k) \forall k \in B$, where $V^k = V \cup \{o(k), d(k)\}$ contains a vertex for each vessel as well as the vertices $o(k)$ and $d(k)$, which are the origin and destination nodes for any route in the graph. The set of arcs is defined as $A^k \subseteq V^k \times V^k$, where each one represents the handling time of the vessel. Considering this graph, the decision variables defined in the DBAP are as follows.

(i) $x_{ij}^k \in \{0,1\}, \forall k \in B, \forall (i,j) \in A^k, i \neq j$, set to 1 if vessel j is scheduled after vessel i at berth k and 0 otherwise.

(ii) $T_i^k, \forall k \in B, \forall v \in V$, the berthing time of vessel i at berth k, that is, the time when the vessel berths.

(iii) $T_{o(k)}^k, \forall k \in B$, starting operation time of berth k, that is, the time when the first vessel berths at the berth.

(iv) $T_{d(k)}^k, \forall k \in B$, ending operation time of berth k, that is, the time when the last vessel departs at the berth.

The objective function (1) aims to minimize the total (weighted) service time of all the vessels, defined as the time elapsed between their arrival to the port and the completion of their handling. It should be noted that when vessel $i \in V$ is not assigned to berth $k \in B$, the corresponding term in the objective function is zero because $\sum_{j \in V \cup d(k)} x_{ij}^k = 0$ and $T_i^k = t_i$:

$$\min \sum_{i \in V} \sum_{k \in B} p_i \left[T_i^k - t_i + s_i^k \sum_{j \in V \cup d(k)} x_{ij}^k \right]. \tag{1}$$

A comprehensive description of the DBAP is provided by Cordeau et al. [5], Imai et al. [9], and Lalla-Ruiz et al. [10].

For the sake of clarity, we provide an example of a solution of the DBAP in Figure 1. In this figure, an assignment plan is depicted for 6 container vessels and 3 berths. The rectangles represent the vessels. Within each rectangle the service priority, service time, and the time windows associated with each vessel are provided. The time windows of the berths are delimited by the scratched areas. For instance, berth 1 is opened from time step 0 until time step 13. In the figure, vessel 6 has to wait for berthing in their respective assigned berths. In this regard, since its priority is 2, its waiting time will have less impact on the objective function value than delaying, for example, vessel 5.

The mathematical formulation of this problem provided in [11] allows solving those instances within reasonable computational time. However, as indicated in [10] this mathematical model implemented in CPLEX reaches a memory fault status for problem instances where other characteristics are taken into account. Therefore, approximate approaches are required, in the following, we describe the most recent ones, de Oliveira et al. [12] proposed a clustering search with simulated annealing, and the authors evaluate their approach using only the large-size problem instances proposed in [5]. Their approach allows us to reach high-quality solutions in short computational times. Later, Ting et al. [13] developed a Particle Swarm Optimization (PSO) and solve the small- and large-size instances proposed in [5]. Their approach reports the same quality solutions as [12] in terms of objective function value; nevertheless, it requires less computational time. Finally, Lalla-Ruiz and Voß [14] propose a matheuristic based on POPMUSIC (Partial Optimization Metaheuristic Under Special Intensification Conditions). The authors tested their approach over the largest instances proposed in [5] exhibiting a high robustness in terms of the average objective values reported by their approach.

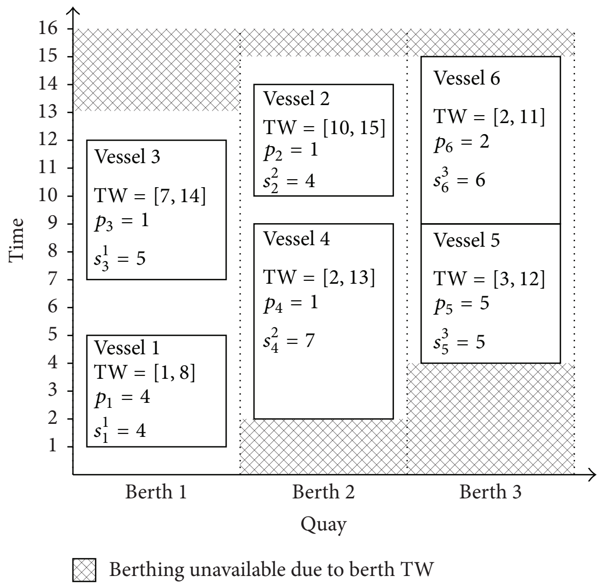


Figure 1: Solution example for $|V| = 6$ vessels and $|B| = 3$ berths.

2.2. Quay Crane Scheduling Problem. The Quay Crane Scheduling Problem (QCSP) is stated as determining the finishing times of the tasks performed by the available quay cranes allocated to a container vessel berthed at the terminal. In this environment, a task represents the loading/unloading of a group of containers onto/from a given deck or hold of the container vessel at hand. Alternative definitions of tasks are proposed by Meisel and Bierwirth [15].

Input data for the QCSP consist of a set of tasks $\Omega = \{1, 2, \ldots, |\Omega|\}$ (loading or unloading operations associated with a container group) and a set of quay cranes $Q = \{1, 2, \ldots, |Q|\}$ with similar technical characteristics. Each $t \in \Omega$ is located in a certain position along the container vessel, l_t, and has a positive handling time, p_t.

The objective of the QCSP is to minimize the service time of the container vessel at hand. That is, its makespan (Kim and Park [16]):

$$\min c_T, \tag{2}$$

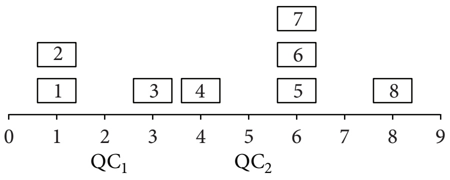
where c_i is the finishing time of the task $t \in \Omega$ and T is a dummy task that represents the end of the service.

The QCSP has a set of particular constraints which differentiates it from other well-known scheduling problems found in the scientific literature.

(a) Each quay crane performs a task without any interruption. This means that once a quay crane starts to (un)load the containers related to a given task, this goes on until all the containers included into the relevant group are (un)loaded.

(b) Each quay crane $q \in Q$ is only available after its earliest ready time, $r^q \geq 0$.

(c) Each quay crane $q \in Q$ is initially located on a known position, l_0^q.



Task, t	1	2	3	4	5	6	7	8
Position, l_t	1	1	3	4	6	6	6	8
Processing time, p_t	10	8	10	15	7	6	5	10

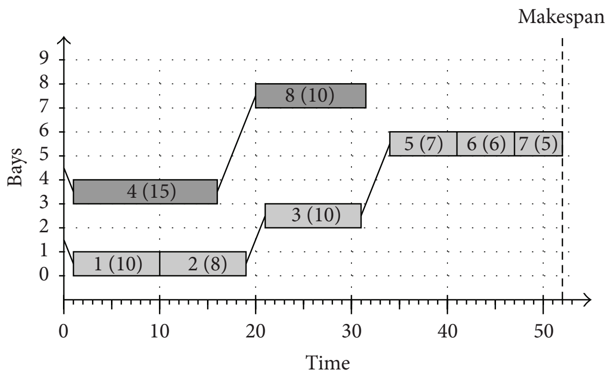


Figure 2: Example of a QCSP instance and schedule with 8 tasks and 2 quay cranes.

(d) Each quay crane $q \in Q$ can travel between two adjacent positions of the container vessel with a travel time, $\hat{t} > 0$.

(e) Within each position of the container vessel, the relevant tasks are sorted according to their precedence relationships. For instance, unloading tasks must be performed before loading tasks.

(f) The quay cranes are rail mounted. This means they can only move from left to right along the container vessel and vice versa.

(g) The quay cranes cannot work in the same position of the vessel simultaneously and they cannot cross each other.

(h) The quay cranes have to keep a safety distance, $\delta > 0$, between them in order to prevent collisions. This gives rise to that certain tasks cannot be performed simultaneously.

Figure 2 illustrates an example of an instance for the QCSP with 8 tasks and 2 quay cranes. The quay cranes are initially located in the positions $l_0^1 = 2$ and $l_0^2 = 5$ of the container vessel. The right figure depicts a schedule for the instance at hand where the quay cranes are available from the beginning of the service time, $r^q = 0, \forall q \in Q$. Additionally, the quay cranes have to keep a safety distance $\delta = 1$ and they can move between two consecutive positions of the vessel with $\hat{t} = 1$. In this example, tasks 1, 2, 3, 5, 6, and 7 are performed by quay crane 1, whereas tasks 4 and 8 are performed by quay crane 2. As can be seen, the makespan of this schedule is 52 time units.

The QCSP is already known to be $\mathcal{NP}$-hard (Sammarra et al. [17]). A mathematical formulation for the QCSP is proposed by Legato et al. [18]. Moreover, the QCSP has been suitably dealt in the literature through several papers. It was introduced in the early work by Daganzo [19] and later approximately addressed by Kim and Park [16] and Sammarra et al. [17]. The computational results of these works brought to light the necessity of developing high efficient optimization techniques to tackle the QCSP by means of reasonable computational times. In this regard, an interesting approach to solve the QCSP was put forward by Bierwirth and Meisel [7]. In their proposal, the authors suggest exploring the search space of unidirectional schedules. A given schedule is termed *unidirectional* if all the quay cranes move with similar direction of movement along their service time. This approach was afterwards deeply exploited by Legato et al. [18] and Expósito-Izquierdo et al. [8]. Lastly, the interested reader is referred to the work by Meisel and Bierwirth [15] to obtain an exhaustive review or the related literature. Unlike most of the previous proposals found in the related literature, the MBO proposed in this work (see Section 3) allows us to reach a high diversification level of the search space during exploration whereas it properly exploits the promising regions in order to find a large number of local optima solutions. This suitable balance between diversification and intensification is mainly due to its population-based structure, which avoids stagnation in low-quality regions of the search space.

3. Migrating Birds Optimization

The Migrating Birds Optimization (MBO) algorithm was initially proposed by Duman et al. [6]. In that work, the authors propose a nature-inspired metaheuristic based on the V-formation flight of migrating birds. This algorithm consists of a set of individuals, where each is associated to a solution and termed as *birds* in MBO. Moreover, the individuals are aligned in a V-formation. Figure 3 shows an illustrative scheme of the V-formation, in which the first individual corresponds to the leader bird in the flock and it is represented by the doubled circle at the top. The remaining circles represent the rest of the flock. The arrows in the figure represent how the information is shared among the individuals.

In MBO, the leader individual attempts to improve itself by generating a number of neighbour solutions. Then, the following individual in the V-formation evaluates a number of its own neighbours and a number of the best discarded neighbour solutions received from the previous individual. In case one of those solutions leads to an improvement with respect to the solution associated with the individual then it is replaced by that solution yielding the maximum improvement. Once all the individuals have been considered, the process is repeated for iter$_{\max}$ iterations. Once those

```
(1)  Generate $n_{\text{birds}}$ initial solutions in a random manner and place them on a hypothetical $V$ formation arbitrarily
(2)  $g = 0$
(3)  **while** $g < K$ **do**
(4)      **for** ($l = 0; l < \text{iter}_{\max}; l{+}{+}$) **do**
(5)          Try to improve the leading solution by generating and evaluating $\lambda$ neighbours of it
(6)          $g = g + \lambda$
(7)          **for all** (solutions $s$ in the flock (except leader)) **do**
(8)              Try to improve the leading solution by generating and evaluating $\lambda - \delta$
                 neighbours of it and the $\delta$ unused best neighbours from the solution in the front.
(9)              $g = g + (\lambda - \delta)$
(10)         **end for**
(11)     **end for**
(12)     Move the leader solution to the end and forward one of the solutions following it to the leader position
(13) **end while**
(14) Return best solution in the flock
```

Algorithm 1: Migrating Birds Optimization algorithm (Duman et al. [6]).

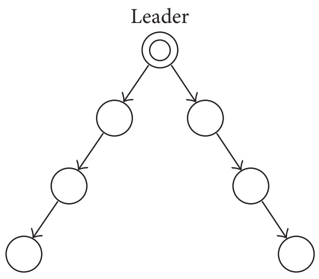


Figure 3: Example of the V-formation of the MBO for 7 individuals (birds).

iterations have been reached, the leader individual is moved to the end of one of the lines of the V-formation and one of its direct follower individuals becomes the new leader of the flock. For this new formation, the process restarts for another $\text{iter}_{\max}$ iterations. The complete MBO process is carried out until a given a stopping criterion is met.

The initial position of the individuals along the V-formation depends on the generation order. That is, the first individual generated will be the leader individual, and therefore the leader bird of the flock, the second and third individuals generated will be its direct followers, and so forth. After generating the initial population, the individuals are organized into a V-formation, as shown in Figure 3.

The input parameters of the MBO algorithm defined by the user are the following:

n_{birds}: number of individuals termed as birds;

K: maximum number of neighbour solutions generated by the individuals;

$\text{iter}_{\max}$: number of iterations before changing the leader individuals;

λ: number of random neighbours generated by each individual;

δ: number of best discarded solutions to share among individuals.

Algorithm 1 depicts the pseudocode of MBO, as reported by Duman et al. [6]. The first step is to generate n_{birds} individuals (line 1). The number of generated solutions by the population, g, is set to 0 (line 2). Once the population is generated, the MBO search process starts (lines 3–13). During the search process, firstly, the leader individual generates λ random neighbour solutions by means of a neighbourhood structure. In case the best solution generated leads to an improvement in terms of objective function value, the solution associated to the leader is replaced by that neighbour solution (line 5). Secondly, each direct follower individual generates a $\lambda - \delta$ neighbour solutions selected at random (lines 7–10) by means of a neighbourhood structure. Also, each individual receives the best δ discarded solutions from the individual in front of it. If one of the solutions, generated or received, leads to an improvement of the solution associated to the individual, then the improved solution replaces it (line 8). This V-formation is maintained until a prefixed number of iterations, $\text{iter}_{\max}$, is reached. Once that, the leader individual becomes the last solution and one of its direct follower individuals becomes the new leader (line 12). Right after, the search process is restarted for other $\text{iter}_{\max}$ iterations. The MBO search process is executed until a number of neighbour solutions, K, have been generated through the search process (line 3). For a more detailed description of the MBO algorithm, the reader is referred to the work by Duman et al. [6].

4. Migrating Birds Optimization for the Seaside Problems at Maritime Container Terminals

In the following, we apply the Migrating Birds Optimization (MBO) introduced in Section 3 to the DBAP and the QCSP. In both cases, we also evaluate the use of improvement procedures applied to the best solution provided by MBO. The rationale behind this is to (i) assess the capability of MBO for pointing out promising regions of the search space that can be exploited by using a improvement procedure and

(ii) measure the contribution of an improvement method based to the quality of the solutions provided by MBO.

4.1. Migrating Birds Optimization for the Dynamic Berth Allocation Problem

4.1.1. Solution Representation. In the context of the DBAP, the MBO implementation considers a solution S_{DBAP} as a sequence composed of the vessel identifiers, where each berth is delimited by a 0. The service order of each vessel is determined by its position in the sequence. The solution structure for the example of Figure 1 for 3 berths and 6 vessels is as follows: $S_{\text{DBAP}} = \{1, 3, 0, 4, 2, 0, 5, 6\}$.

4.1.2. Neighbourhood Structures. The neighbourhood structures considered in this approach are generated by using the following movements.

(a) *Reinsertion Movement.* A vessel i is removed from a berth k and reinserted into another berth k' ($\forall k, k' \in B$, $i \neq i'$) at any of the possible positions. For example, in $S_{\text{DBAP}} = \{1, 3, 0, 4, 2, 0, 5, 6\}$ if vessel 1 is removed from its berth, then all these possible reinsertion movements can be performed, namely, $\{3, 0, 1, 4, 2, 0, 5, 6\}$, $\{3, 0, 4, 1, 2, 0, 5, 6\}$, $\{3, 0, 4, 2, 1, 0, 5, 6\}$. $\{3, 0, 4, 2, 0, 1, 5, 6\}$, $\{3, 0, 4, 2, 0, 5, 1, 6\}$, and $\{3, 0, 4, 2, 0, 5, 6, 1\}$.

(b) *Interchange Movement.* It consists of exchanging a vessel i assigned to berth k with a vessel i' assigned to berth k' ($\forall i, i' \in V$, $i \neq i'$, $\forall k, k' \in B$, $k \neq k'$). For example, for the previous solution, $S_{\text{DBAP}} = \{1, 3, 0, 4, 2, 0, 5, 6\}$, if we select vessel 1, the possible interchange movements that can be obtained are the following: $\{4, 3, 0, 1, 2, 0, 5, 6\}$, $\{2, 3, 0, 4, 1, 0, 5, 6\}$, $\{5, 3, 0, 4, 2, 0, 1, 6\}$, and $\{6, 3, 0, 4, 2, 0, 5, 1\}$.

The generation of random neighbour solutions by the individuals is based on the reinsertion move. On the other hand, both movements are used in the improvement method.

4.1.3. Improvement Method. As discussed in the relevant section, we also analyse the capability of MBO for pointing out promising regions in the solution search space. In doing so, we applied an improvement method proposed by Lalla-Ruiz et al. [10] over the best solution provided by MBO. This method consist of the following steps: given a solution, its best neighbour solution is obtained by means of the reinsertion moment. Over that best neighbor solution, we generate its neighborhood by means of the interchange movement and return the best neighbor solution. This process is performed until no improvement in terms of the objective function value is achieved.

4.1.4. Initial Population. For generating the initial population, we use a random greedy method (R-G) proposed by Cordeau et al. [5]; that is, given a random vessel permutation, the vessels are assigned one at a time to the best possible berth following that sequence order according to their impact over the objective function value. The use of this method instead of other proposed initialization procedures reported in the literature such as First-Come First-Served Greedy (FCFS-G) or generating the solution completely at random is based on the fact that, on the one hand, FCFS-G is a deterministic approach, which use will affect the convergence of the algorithm. On the other hand, R-G provides better quality solutions than generating the initial solutions completely at random; this is due to the fact that R-G allocates the vessels within the random sequence according to the impact over the objective function value of the solution being constructed.

4.1.5. Stopping Criterion. The stopping criterion for the overall MBO search process is met when a certain number of K generated solutions by the population are reached. Moreover, as pointed out in the relevant section, for large-size instances, we included an additional stopping criterion based on a maximum number of iterations γ without improvement of the best solution obtained.

4.2. Migrating Birds Optimization for the Quay Crane Scheduling Problem

4.2.1. Solution Representation. The solutions of the QCSP are represented as sequences composed of the available tasks, that is, Ω. A given sequence includes zeros in order to delimit the subsets of tasks performed by the quay cranes. This way, the leftmost quay crane performs those tasks from the beginning of the sequence up to the first zero, the second quay crane performs those tasks from the first zero up to the second zero, and so forth. For instance, a solution for the example presented in Figure 2 with 2 quay cranes and 8 tasks could be as follows: $S_{\text{QCSP}} = (1, 2, 3, 5, 6, 7, 0, 4, 8)$. In this case, the leftmost quay crane performs the tasks $(1, 2, 3, 5, 6, 7)$, whereas the other quay crane performs the tasks $(4, 8)$.

4.2.2. Neighbourhood Structures. The neighbourhood structures used by the MBO are based on the following exploring movements.

(a) *Reinsertion Movement.* A task $t \in \Omega$ currently assigned to a quay crane $q \in Q$ is reassigned, in such a way that t is performed by another quay crane $q' \in Q$ (where $q' \neq q$).

(b) *Interchange Movement.* Given pair tasks, $t_1, t_2 \in \Omega$, assigned to two different quay cranes, $q \in Q$ and $q' \in Q$ (where $q' \neq q$), the movement exchanges the tasks. This way, t_2 is eventually assigned to q whereas t_1 is assigned to q_2.

The generation of random neighbour solutions by the individuals is based on the interchange movement.

4.2.3. Local Search. A local search process based on the best improvement strategy is proposed in order to find local optima solutions during the search. This way, given a certain feasible solution of the QCSP, at each step, the set of neighbour solutions found by means of reinsertion movement is generated. The best neighbour solution replaces the current solution until a local optimum is achieved.

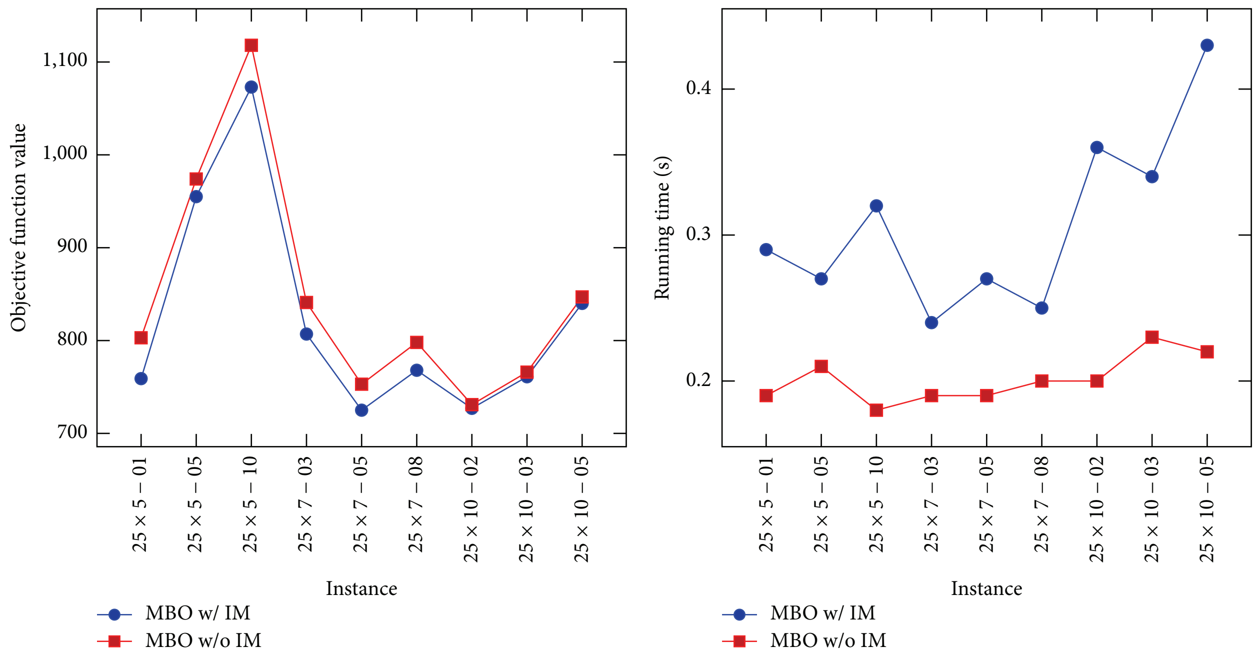


Figure 4: Performance of MBO with and without an improvement procedure when solving the DBAP small-size instances.

4.2.4. Initial Population. The solutions included into the initial population have been generated at random. This means that each task is assigned to one of the available quay cranes randomly. It is worth mentioning that the tasks are selected to be assigned from the leftmost up to the rightmost within the container vessel.

4.2.5. Stopping Criterion. The stopping criterion for the overall MBO search process is met when a certain number of K neighbour solutions have been already generated by the individuals.

5. Computational Experiments

This section is devoted to assessing the performance of the Migrating Birds Optimization (MBO) introduced in the previous section. All the reported computational experiments have been conducted on a computer equipped with an Intel Dual Core 3.16 GHz and 4 GB of RAM.

5.1. Computational Experiments for the DBAP. The problem instances used for evaluating the performance of our MBO approach are those provided by Cordeau et al. [5]. According to the authors, their instances were generated by taking into account a statistical analysis of the traffic and berth allocation data at the maritime container terminal of Gioia Tauro (Italy). The instances are grouped into sets of 10 instances, whose sizes range from 25 vessels and 5 berths up to 60 vessels and 13 berths. Moreover, in order to fit the space for this work, for the small- and medium-size problem instances we have selected the 3 hardest solvable instances of each set with regard to the time required to provide the optimal solution by the implementation of the mathematical formulation (Buhrkal et al. [11]) in CPLEX (http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/) to provide the optimal solution. For the large instance set, we selected a representative set of instances. By taking into account the experiments carried out in this work, we identified the following parameter values for MBO, $n_{\text{birds}} = 31$, $\delta = 3$, $\lambda = 20$, $\text{iter}_{\max} = 3$, and $K = |N|^3$ for the small- and medium-size instances. For the large-size instances, we set $K = |N|^{2.5}$ and an additional stopping criterion of a maximum number of $\gamma = 10$ consecutive iterations without improvement of the best solution obtained.

5.1.1. Improvement Method. As previously indicated in Section 4, an improvement phase (based on that proposed by Lalla-Ruiz et al. [10]) is applied over the best solution provided by MBO. Figures 4, 5, and 6 show the computational performance in terms of objective function value and computational time of MBO with (MBO w/IM) and without (MBO w/o IM) improvement method for the small-, medium-, and large-size instances proposed by Cordeau et al. [5]. As can be checked, the performance exhibited by MBO is similar regardless of size of the instance. Furthermore, the use of a improvement procedure leads to an enhancement of the best-known solution through a small increase of the computational time. This indicates that the use of the improvement to this method enhances the convergence to the best solution within our complete approach proposed in this work.

At the light of this analysis, in the following results we report the computational results provided by the joint use of MBO with the improvement method.

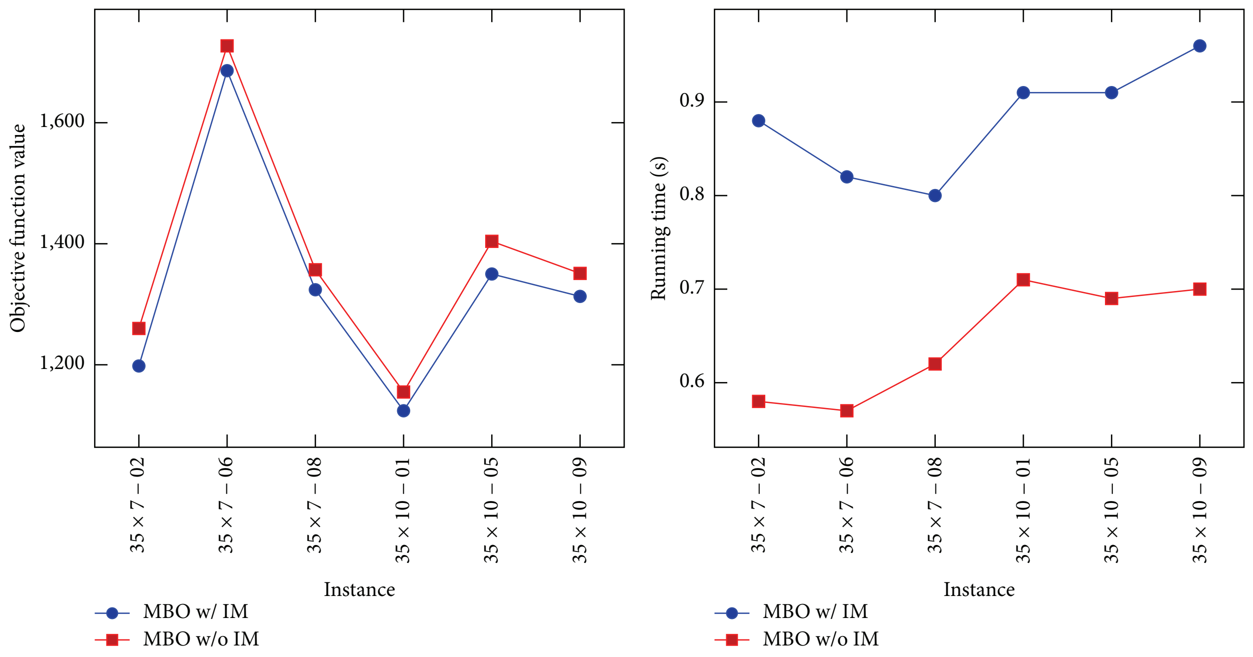


Figure 5: Performance of MBO with and without an improvement procedure when solving the DBAP medium-size instances.

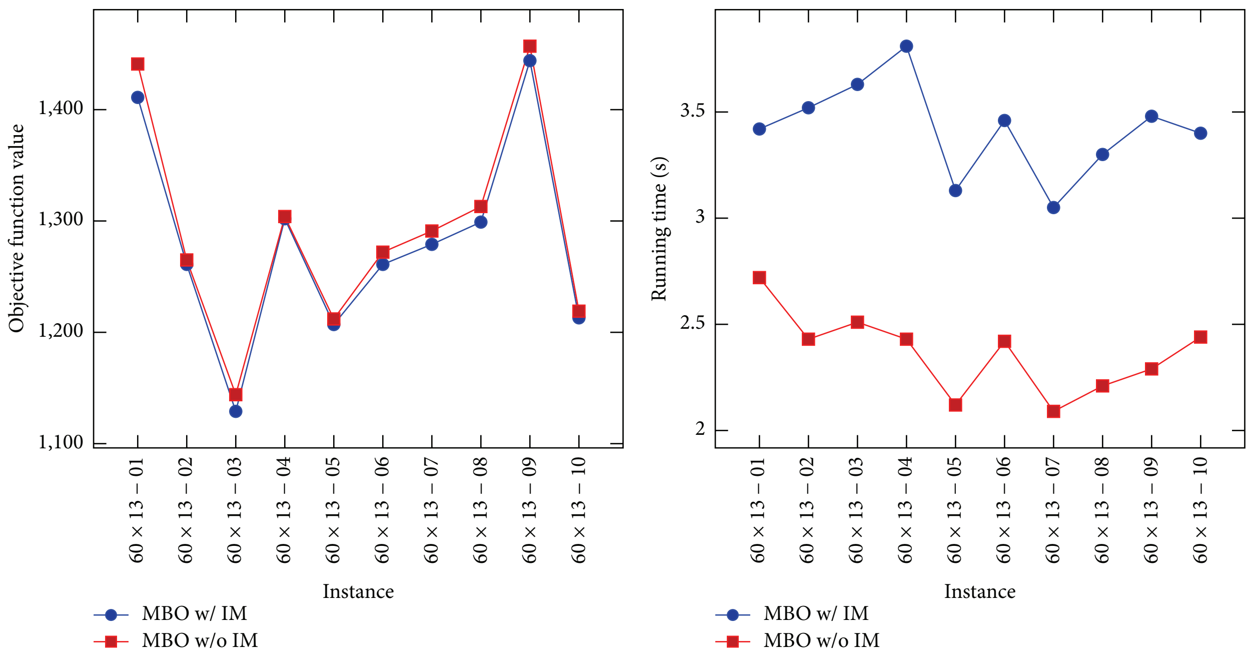


Figure 6: Performance of MBO with and without an improvement procedure when solving the DBAP large-size instances.

5.1.2. Comparison with the Best Literature Approach. Tables 1 and 2 present a comparison among the best published approaches for the DBAP, namely, the mathematical model presented by Buhrkal et al. [11], the best approximate approach for this problem consisting of a Particle Swarm Optimization algorithm (PSO) proposed by Ting et al. [13], and our MBO. The first column shows the characteristics of the instances to solve, that is, the number of vessels ($|V|$) and berths ($|B|$), and the instance identifier (id). For each instance, the best objective value (Obj.), relative error (Gap (%)) with regard to the optimal value provided by CPLEX, and the computational time measured in seconds (t (s)) are presented. Furthermore, with the aim of assessing the time improvement reported by MBO in comparison with PSO, the percentage of time improvement (t_{impr}) is also reported.

As shown in Table 1, MBO reports high-quality solutions in shorter computational times than the other approaches. It reaches the optimal value for the majority of the problem

Table 1: Computational results for a representative set of small- and medium-size instances proposed by Cordeau et al. [5].

Instance			CPLEX		PSO			MBO			
$\|V\|$	$\|B\|$	id	Obj.	t (s)	Obj.	Gap (%)	t (s)	Obj.	Gap (%)	t (s)	t_{impr} (%)
25	5	1	759	5.99	759	0.00	0.75	759	0.00	0.29	−61.33
		5	955	6.97	955	0.00	0.86	955	0.00	0.27	−69.77
		10	1073	6.38	1073	0.00	0.73	1073	0.00	0.32	−54.79
25	7	3	807	4.28	807	0.00	0.97	807	0.00	0.24	−75.26
		5	725	3.85	725	0.00	0.44	725	0.00	0.27	−38.64
		8	768	3.93	768	0.00	1.05	768	0.00	0.25	−76.19
25	10	2	727	6.99	727	0.00	0.75	727	0.00	0.36	−52.00
		3	761	6.12	761	0.00	0.56	761	0.00	0.34	−39.29
		5	840	6.77	840	0.00	0.45	840	0.00	0.29	−35.56
35	7	2	1192	15.93	1192	0.00	4.91	1198	0.50	0.81	−83.50
		6	1686	29.16	1686	0.00	3.28	1692	0.36	0.77	−76.52
		8	1318	17.52	1318	0.00	2.39	1324	0.46	0.73	−69.46
35	10	1	1124	19.98	1124	0.00	1.58	1124	0.00	0.91	−42.41
		5	1349	22.31	1349	0.00	1.53	1350	0.07	0.91	−40.52
		9	1311	29.45	1311	0.00	2.81	1313	0.15	0.97	−65.48
Average			1026.33	12.38	1026.33	0.00	1.54	1027.73	0.10	0.52	−58.71

Table 2: Computational results for a representative set of large-size instances proposed by Cordeau et al. [5].

Instance			CPLEX		PSO			MBO			
$\|V\|$	$\|B\|$	id	Opt.	t (s)	Obj.	Gap (%)	t (s)	Obj.	Gap (%)	t (s)	t_{impr} (%)
60	13	1	1409	17.92	1409	0.00	11.11	1411	0.14	3.42	−69.22
		2	1261	15.77	1261	0.00	7.89	1261	0.00	3.52	−55.39
		3	1129	13.54	1129	0.00	7.48	1129	0.00	3.63	−51.47
		4	1302	14.48	1302	0.00	6.03	1302	0.00	3.81	−36.82
		5	1207	17.21	1207	0.00	5.84	1207	0.00	3.13	−46.40
		6	1261	13.85	1261	0.00	7.67	1261	0.00	3.46	−54.89
		7	1279	14.60	1279	0.00	7.5	1279	0.00	3.05	−59.33
		8	1299	14.21	1299	0.00	9.94	1299	0.00	3.30	−66.80
		9	1444	16.51	1444	0.00	4.25	1444	0.00	3.48	−18.12
		10	1213	14.16	1213	0.00	5.2	1213	0.00	3.40	−34.62
Average			1280.40	15.23	1280.40	0.00	7.29	1280.60	0.01	3.42	−49.31

instances. Although MBO is not able to provide the optimal solutions in five cases (i.e., $35 \times 7 - 2$, $35 \times 7 - 6$, $35 \times 7 - 8$, $35 \times 10 - 1$, and $35 \times 10 - 9$), it presents a very competitive performance with a time range enough for improvement. In this regard, the maximum gap in those cases is 0.50%. Furthermore, MBO is able to reduce, on average, about the 58% of the time required by PSO.

Moreover, when we evaluate larger problem instances, as those reported in Table 2, we can point out the relevant time improvement reported by MBO over PSO, which is, on average, of almost 50%. In this regard, as shown in this table, the quality of the solutions is similar to the PSO. MBO is able to provide the optimal solutions in the majority of the cases. In the unique case where MBO does not provide the optimal solution, it is able to provide a solution with a gap of 0.07%. It should be also pointed out that the time benefit presented by MBO makes it suitable as a solution method for being applied either individually or included into integrated schemes in which the berth allocation is required and has to be executed frequently.

5.2. Computational Experiments for the QCSP. In order to assess the suitability of MBO when solving the QCSP, we have considered 40 instances (*k13–k52*) of those proposed by Bierwirth and Meisel [7]. These instances have different number of tasks (from 10 up to 25) and quay cranes (from 2 up to 3) which allow encompassing real-world scenarios. It is worth mentioning that, as done in previous works, in this experiment we have established that the quay cranes are available from the beginning of the service time (i.e., $r^q = 0$, $\forall q \in Q$) and have to keep a safety distance $\delta = 1$ during the service. Moreover, by preliminary tests the following

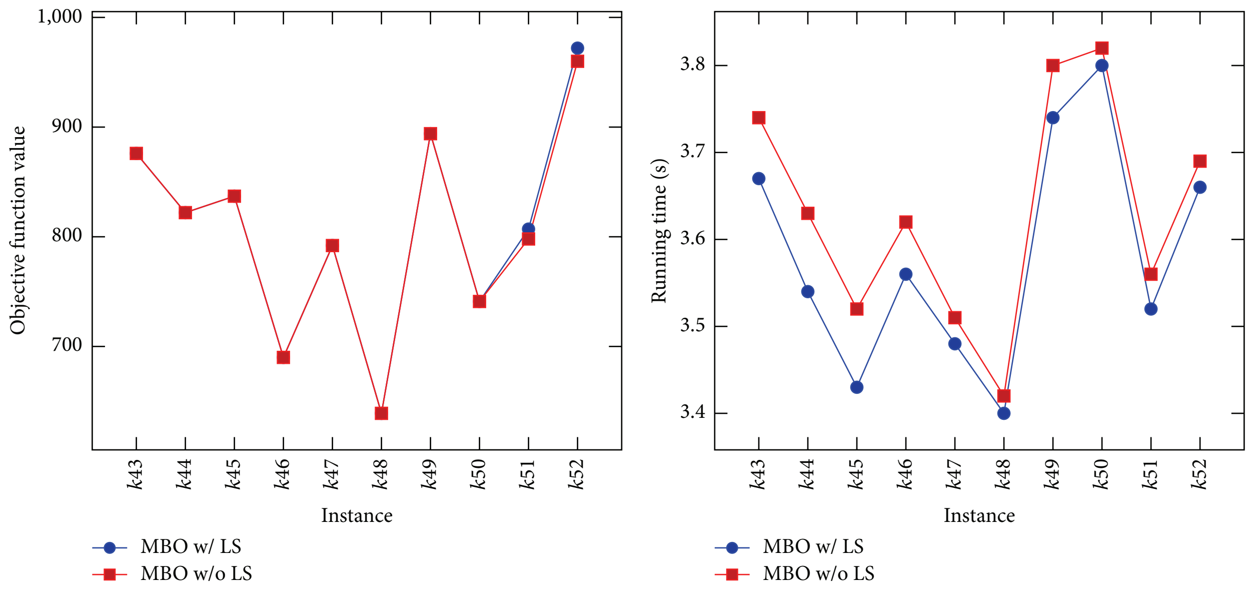


Figure 7: Performance of MBO with and without a local search procedure when solving the QCSP *k43*–*k52* problem instances.

parameter values have been used in the execution of MBO: $n_{\text{birds}} = 31$, $\delta = 1$, $\lambda = 10$, $\text{iter}_{\max} = 5$, and $K = |N|^3$.

5.2.1. Local Search. As done in previous works (e.g., Sammarra et al. [17], Expósito-Izquierdo et al. [8]), a local search process is applied to the best solution provided by MBO. The rationale behind this is to assess the capacity of MBO to point out promising regions in the search space. Figure 7 shows the computational performance in terms of objective function value and computational time of MBO with (MBO w/LS) and without (MBO w/o LS) local search for the instances *k43*–*k52* (for the other instances, *k13*–*k42*, regardless of the use or not of the local search, our algorithm provides the best known solution). As shown in the figure, the performance exhibited by the MBO with and without LS is similar independently of the instance tackled. In this regard, the use of a local search leads to a very small improvement of some solutions in some cases (see *k51* and *k52*) requiring only an slightly increase of the computational time. This fact may indicate that, in some cases, the solution provided by MBO is already a local optimum. Finally, although LS contributes to improving the quality of the solution, not using it may not affect substantively the quality of the solution. Nevertheless, due to small computational time required, in the following, the MBO computational results reported for the QCSP instances are the ones obtained with local search.

5.2.2. Comparison with the Best Literature Approach. Table 3 shows a comparison between the optimal solutions reported by Bierwirth and Meisel [7], the Estimation Distribution Algorithm (EDA) proposed by Expósito-Izquierdo et al. [8], and our MBO. In each case, we report the objective function value of the best found solution and the computational time measured in seconds. In the case of the MBO, we also report the gap in the objective function value compared with the optimal solution and computational time compared with those reported by the EDA.

As can be checked in Table 3, our MBO has reported (near-)optimal solutions for all the instances under analysis. Only in one instance (*k45*), the optimal solution was not reached. In those cases, MBO reports a gap of 0.36% and an overall gap of only 0.01% for all the instances considered. The quality of the solutions reported by MBO indicates that our approach is highly effective in realistic scenarios. Finally, when carrying out an analysis of the computational times, we realize that MBO requires short computational times, requiring at most 3.84 seconds. This fact constitutes a relevant improvement in comparison with the EDA, which requires more than 16 seconds in some instance (*k50*). This time advantage must be suitably considered when addressing practical scenarios where the QCSP has to be solved dynamically.

6. Conclusions and Further Research

In this paper, we have presented a Migrating Birds Optimization-based approach for addressing two essential seaside problems at maritime container terminals: the Dynamic Berth Allocation Problem (DBAP) and Quay Crane Scheduling Problem (QCSP). It is noticeable from the computational experiments that the proposed algorithm is able to report high-quality solutions by means of short computational times. In this regard, the time advantage makes MBO promising and competitive as solution method when tackling seaside operations either individually or embedded into real decision-support systems where this problem has to be solved frequently. Moreover, since our approach includes the use of an improvement method (in the case of DBAP) and a local search (in the case of QCSP) over the best solution provided by MBO, we have assessed the contribution of them to the quality of the solution provided. In this regard, their use

Table 3: Comparison among the optimal solutions (UDS, Bierwirth and Meisel [7]), the Estimation Distribution Algorithm (EDA, Expósito-Izquierdo et al. [8]), and our MBO when solving the QCSP.

	UDS		EDA		MBO		
	Obj.	t (s)	Obj.	t (s)	Obj.	t (s)	Gap (%)
$k13$	453	—	453	0.08	453	0.17	0.00
$k14$	546	—	546	0.09	546	0.18	0.00
$k15$	513	—	513	0.08	513	0.17	0.00
$k16$	312	—	312	0.56	312	0.17	0.00
$k17$	453	—	453	0.08	453	0.17	0.00
$k18$	375	—	375	0.07	375	0.16	0.00
$k19$	543	—	543	0.08	543	0.18	0.00
$k20$	399	—	399	0.09	399	0.19	0.00
$k21$	465	—	465	0.07	465	0.17	0.00
$k22$	540	—	540	0.13	540	0.20	0.00
$k23$	576	—	576	0.27	576	0.40	0.00
$k24$	666	—	666	0.39	666	0.38	0.00
$k25$	738	—	738	0.25	738	0.35	0.00
$k26$	639	—	639	0.33	639	0.37	0.00
$k27$	657	—	657	0.29	657	0.35	0.00
$k28$	531	—	531	0.27	531	0.34	0.00
$k29$	807	—	807	0.31	807	0.36	0.00
$k30$	891	—	891	0.22	891	0.40	0.00
$k31$	570	—	570	0.26	570	0.37	0.00
$k32$	591	—	591	0.37	591	0.38	0.00
$k33$	603	—	603	9.12	603	1.13	0.00
$k34$	717	—	717	9.24	717	1.16	0.00
$k35$	684	—	684	4.48	684	1.09	0.00
$k36$	678	—	678	7.62	678	1.13	0.00
$k37$	510	—	510	4.09	510	1.11	0.00
$k38$	618	—	618	6.66	618	1.09	0.00
$k39$	513	—	513	6.54	513	1.14	0.00
$k40$	564	—	564	7.14	564	1.11	0.00
$k41$	588	—	588	6.66	588	1.13	0.00
$k42$	573	—	573	6.30	573	1.18	0.00
$k43$	876	12.6	876	12.60	876	3.74	0.00
$k44$	822	12.0	822	11.40	822	3.63	0.00
$k45$	834	10.8	834	8.40	837	3.52	0.36
$k46$	690	11.4	690	9.60	690	3.66	0.00
$k47$	792	10.2	792	10.20	792	3.51	0.00
$k48$	639	11.4	639	8.40	639	3.44	0.00
$k49$	894	10.8	894	13.20	894	3.80	0.00
$k50$	741	10.2	741	16.80	741	3.84	0.00
$k51$	798	10.2	798	12.00	798	3.52	0.00
$k52$	960	10.2	960	13.20	960	3.69	0.00
Avg.	633.98	10.98	633.98	4.70	634.05	1.33	0.01

allows an enhancement in the quality of the solutions in terms of objective function value through a small increase of the computational time.

Furthermore, the inherent dynamism of the seaside operations at maritime container terminals highly impacts on the performance of the technical equipment and, consequently, on the involved transportation modes. Thus, having effective and fast algorithms to reach high-quality solutions is aspired by terminal managers. In this context, at the light of the computational results presented along this paper, we can claim that using MBO is suitable and advisable to be used in practical contexts with the goal of providing an adequate service to the incoming container vessels.

A multitude of lines are open for further research. In the future, we are going to test the performance of MBO in other heterogeneous transportation problems, such as Vehicle Routing Problem, due to its generalist standpoint. In this regard, we are also going to study how different interaction schemes impact on the performance of MBO.

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 funded by the Spanish Ministry of Economy and Competitiveness (Project TIN2012-32608). The authors thank the funding granted to the University of La Laguna (ULL) by the Canary Agency of Investigation, Innovation, and Information Society (ACIISI), 85% cofinanced with the help of European Social Fund. Eduardo Lalla-Ruiz and Christopher Expósito-Izquierdo would like to thank the Canary Government for the financial support they receive through their doctoral grants.

References

[1] L. Nicoletti, A. Chiurco, C. Arango, and R. Diaz, "Hybrid approach for container terminals performances evaluation and analysis," *International Journal of Simulation and Process Modelling*, vol. 9, no. 1-2, pp. 104–112, 2014.

[2] C. Expósito-Izquierdo, E. Lalla-Ruiz, B. Melián Batista, and J. M. Moreno-Vega, "Optimization of container terminal problems: an integrated solution approach," in *Computer Aided Systems Theory—EUROCAST 2013*, R. Moreno-Díaz, F. Pichler, and A. Quesada-Arencibia, Eds., vol. 8111 of *Lecture Notes in Computer Science*, pp. 324–331, Springer, Berlin, Germany, 2013.

[3] H.-J. Yeo, "Competitiveness of Asian container terminals," *The Asian Journal of Shipping and Logistics*, vol. 26, no. 2, pp. 225–246, 2010.

[4] 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.

[5] J.-F. Cordeau, G. Laporte, P. Legato, and L. Moccia, "Models and tabu search heuristics for the berth-allocation problem," *Transportation Science*, vol. 39, no. 4, pp. 526–538, 2005.

[6] E. Duman, M. Uysal, and A. F. Alkaya, "Migrating birds optimization: a new metaheuristic approach and its performance on quadratic assignment problem," *Information Sciences*, vol. 217, pp. 65–77, 2012.

[7] C. Bierwirth and F. Meisel, “A fast heuristic for quay crane scheduling with interference constraints,” *Journal of Scheduling*, vol. 12, no. 4, pp. 345–360, 2009.

[8] C. Expósito-Izquierdo, J. L. González-Velarde, B. Melián-Batista, and J. M. Moreno-Vega, “Hybrid estimation of distribution algorithm for the quay crane scheduling problem,” *Applied Soft Computing Journal*, vol. 13, no. 10, pp. 4063–4076, 2013.

[9] A. Imai, E. Nishimura, and S. Papadimitriou, “The dynamic berth allocation problem for a container port,” *Transportation Research Part B: Methodological*, vol. 35, no. 4, pp. 401–417, 2001.

[10] E. Lalla-Ruiz, B. Melián-Batista, and J. Marcos Moreno-Vega, “Artificial intelligence hybrid heuristic based on tabu search for the dynamic berth allocation problem,” *Engineering Applications of Artificial Intelligence*, vol. 25, no. 6, pp. 1132–1141, 2012.

[11] K. Buhrkal, S. Zuglian, S. Ropke, J. Larsen, and R. Lusby, “Models for the discrete berth allocation problem: a computational comparison,” *Transportation Research Part E: Logistics and Transportation Review*, vol. 47, no. 4, pp. 461–473, 2011.

[12] R. M. de Oliveira, G. R. Mauri, and L. A. N. Lorena, “Clustering search for the berth allocation problem,” *Expert Systems with Applications*, vol. 39, no. 5, pp. 5499–5505, 2012.

[13] C.-J. Ting, K.-C. Wu, and H. Chou, “Particle swarm optimization algorithm for the berth allocation problem,” *Expert Systems with Applications*, vol. 41, no. 4, pp. 1543–1550, 2014.

[14] E. Lalla-Ruiz and S. Voß, “Popmusic as a matheuristic for the berth allocation problem,” *Annals of Mathematics and Artificial Intelligence*, pp. 1–17, 2014.

[15] F. Meisel and C. Bierwirth, “A unified approach for the evaluation of quay crane scheduling models and algorithms,” *Computers & Operations Research*, vol. 38, no. 3, pp. 683–693, 2011.

[16] K. H. Kim and Y.-M. Park, “A crane scheduling method for port container terminals,” *European Journal of Operational Research*, vol. 156, no. 3, pp. 752–768, 2004.

[17] M. Sammarra, J.-F. Cordeau, G. Laporte, and M. F. Monaco, “A tabu search heuristic for the quay crane scheduling problem,” *Journal of Scheduling*, vol. 10, no. 4-5, pp. 327–336, 2007.

[18] P. Legato, R. Trunfio, and F. Meisel, “Modeling and solving rich quay crane scheduling problems,” *Computers & Operations Research*, vol. 39, no. 9, pp. 2063–2078, 2012.

[19] C. F. Daganzo, “The crane scheduling problem,” *Transportation Research Part B: Methodological*, vol. 23, no. 3, pp. 159–175, 1989.