SN Computer Science (2022) 3:325
https://doi.org/10.1007/s42979-022-01211-z

ORIGINAL RESEARCH

Enhanced Berth Allocation Using the Cuckoo Search Algorithm

Sheraz Aslam[1] · **Michalis P. Michaelides**[1] · **Herodotos Herodotou**[1]

Received: 13 September 2021 / Accepted: 15 May 2022

Abstract
Berth allocation is one of the most important optimization problems in container terminals at ports worldwide. From both the port operator's and the shipping lines' point of view, minimizing the time a vessel spends at berth and minimizing the total cost of berth operations are considered fundamental objectives with respect to terminal operations. In this study, we focus on the berth allocation problem (BAP), where berth positions are assigned to arriving ships with the objective of reducing the total service cost, which includes waiting cost, handling cost, and several penalties, such as a penalty for late departure and a penalty for non-optimal berth allocation. First, the BAP is formulated as a mixed-integer linear programming (MILP) model. Since BAP is an NP-hard problem and cannot be solved by mathematical approaches in a reasonable time, a metaheuristic approach, namely, a cuckoo search algorithm (CSA), is proposed in this study to solve the BAP. To validate the performance of the proposed CSA-based method, we use two benchmark approaches, namely, the genetic algorithm (GA) and the optimal MILP solution. Next, we conduct several experiments using a benchmark data set as well as a randomly-generated larger data set. Simulation results show that the proposed CSA algorithm has higher efficiency in allocating berths within a reasonable computation time than its counterparts.

Keywords Berth allocation problem · Intelligent sea transportation · Cuckoo search algorithm · Metaheuristic optimization · Maritime container terminal

Introduction

Background and Motivations

The shipping industry covers 90% of the world seaborne trade movements and 74% of the total goods that are imported or exported in Europe travel with ships [1]. According to Hsu et al. [2], 60% of the total sea transport is based on containers, which is also growing every year by 6.4%. Hence, the maritime container terminal (MCT) serves as an important node in the shipping industry to deal with increasing sea trade. Another recent report [3] stated that worldwide ports have handled almost 701 million twenty-foot equivalent units (TEUs) of containers in 2016. At the same time, the throughput of container ports is also continuously increasing, and the management of MCTs' operations is becoming a challenging task. As a critical and integral part of the global transportation network, the MCTs serve the cost-efficient delivery of various products in different markets. Linear shipping companies use mega-ships in order to carry large containers up to 20,000 TEUs [4]. Since the MCTs are of such high value in the maritime industry, there is an exigent need to enhance the operational efficiency of MCTs by minimizing the total turnaround service times of vessels and achieving competitive strategy along with customer satisfaction. Moreover, port authorities always try to optimize MCT operations by employing various strategies for the efficient utilization of all the port resources.

MCT operations can be categorized into three major operational areas, namely, seaside, land-side, and yard-side operations, as presented in Fig. 1. Among all MCT operations,

This article is part of the topical collection "Vehicle Technology and Intelligent Transport Systems" guest edited by Oleg Gusikhin and Markus Helfert.

✉ Herodotos Herodotou
herodotos.herodotou@cut.ac.cy

Sheraz Aslam
sheraz.aslam@cut.ac.cy

Michalis P. Michaelides
michalis.michaelides@cut.ac.cy

[1] Department of Electrical Engineering, Computer Engineering and Informatics, Cyprus University of Technology, Limassol, Cyprus

Published online: 10 June 2022

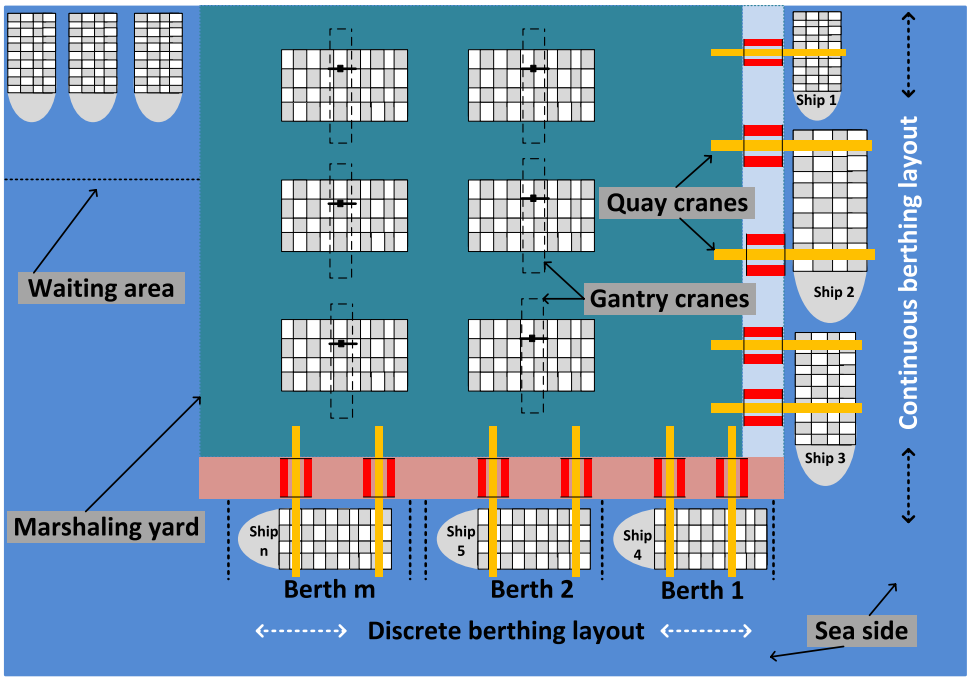

Fig. 1 Illustration of MCT with multiple berthing positions assuming discrete and continuous berthing layout

the seaside operations are the most important as they affect the overall performance of MCTs. Inefficient planning and improper utilization of port resources may create several issues, including congestion, long waiting times, and late departures. For instance, 13,647 vessels arrived from Jan to Sep 2019 at Port of Shanghai, China, from which almost 57% of vessels arrived late (more than 12 h) [5]. According to another recent report presented in [6], the average waiting times for vessels from port-to-berth are 2.2, 2.4, and 2.7 h in Malaysia, Dubai, and China, respectively. Michaelides et al. [7] investigate the factors influencing the various waiting times at the Port of Limassol, Cyprus, both from a quantitative and a qualitative perspective. For shipping, and particularly for short sea shipping, there are obvious and immediate benefits from improving efficiency by supporting all actors involved in the port call process to engage more easily, to give shipping companies, port service providers, and ship agents better information and decision support systems to boost their efficiency and that of their port [8]. Hence, MCTs' operators need to employ suitable strategies and approaches for proper utilization of the port resources and to avoid the aforementioned issues.

These challenges have motivated us to focus on enhancing seaside operations, and more specifically, the *berth allocation problem* (*BAP*). The BAP is a well-known problem that aims to assigning berthing positions to arriving vessels at the port in order to minimize or maximize a given objective function (e.g., minimize total waiting time, reduce late departures, or maximize terminal performance). Before dealing with the BAP, it is necessary to understand the problem environment. Based on the current literature, there are two major factors affecting the BAP, i.e., the configuration of quay/wharf and the arrival time of ships. Quays can be configured in three different ways: (1) *continuous* berthing layout, where arriving vessels can be moored at any location along the wharf; (2) *discrete* berthing layout, where the wharf is divided into a fixed number of berths; and (3) *hybrid* berthing layout, where a mix of continuous and discrete berthing layouts is encountered [9]. In terms of vessel arrivals, there are two main types: (1) *static arrivals*, where all the vessels are assumed to be at the MCT before berth planning and (2) *dynamic arrivals* meaning that vessels are not at the MCT before berth planning but instead the expected time of arrival (ETA) is known for each vessel. Static arrivals are a simple special case of dynamic arrivals (where all ships are expected to arrive at the same time), while the discrete berthing layout is a basic variation of the more-complex continuous layout scenario. Thus, this study focuses primarily on the continuous berthing layout together with dynamic vessel arrivals (i.e., DC-BAP).

Contributions

The present study is an extended version of [10] and solves the BAP with the aim of reducing the total service cost of arriving ships, which includes waiting costs, handling costs, as well as penalties for late departures. We first formulate BAP as a mixed-integer linear programming (MILP) problem and solve it using the cuckoo search algorithm (CSA), a metaheuristic computational intelligence approach that we applied to BAP for the first time in [10]. Furthermore, we enhance the BAP mathematical formulation with some new

practical constraints and features compared to the current literature and our previous study [10], which include: (1) a penalty for non-optimal berthing positions that is proportional to the distance from the preferred berthing position, (2) a safety time interval between consecutive berth arrivals due to practical constraints present at the port entrance, and (3) account for smaller time intervals, which offer better, more fine-grained berth allocation decisions. We also implement two benchmark algorithms to verify the effectiveness of our proposed algorithm, namely, genetic algorithm (GA) and an exact approach (MILP). Extensive simulations were performed on two types of data instances; the first one was obtained from the existing literature (benchmark data) and the second data set (it includes up to 100 arriving ships) was generated based on real-world data (by uniform distribution). The simulation results show that the proposed CSA method outperforms its counterparts in terms of reducing service cost within a reasonable computation time.

Organization

The remainder of the paper is organized as follows. Literature review is presented in Section "Literature Review". Section "Problem Description" explains the investigated problem and provides its mathematical formulation. Our proposed CSA method along with the compared approach is described in detail in Section "Proposed and Benchmark Approaches" and simulation settings along with results are presented in Section "Experimental Results". Finally, Section "Conclusions" concludes the paper.

Literature Review

Several approaches are reported in the literature that deal with the BAP [11, 12]. These approaches may provide exact solutions [13] or approximate (based on heuristic or metaheuristic) solutions [14, 15]. However, approximate approaches are more popular over exact methods due to their efficiency in terms of computational complexity. The authors of [14] present a solution of the BAP by employing evolutionary algorithms (EAs), particle swarm optimization (PSO), and differential evolution (DE). An EA-based solution is developed in [15] to deal with BAP, while the study presented in [16] proposed a simulated annealing (SA) algorithm for the same problem. [13] developed a mixed-integer linear programming (MILP) model to deal with the BAP and a genetic algorithm (GA) is developed in [17] to solve the BAP.

For the dynamic and continuous BAP (DC-BAP), there have been significant efforts in the recent literature to solve the problem using mainly metaheuristic and evolutionary approaches. A GA-based approach is developed in [18] to deal with the DC-BAP to minimize penalty costs for late departures. In [19], the objectives of this study are to minimize the total service cost and the total ship stay time at the port. To solve the DC-BAP, a variant of the neighborhood search method called adaptive large neighborhood search (ALNS) algorithm is developed. The algorithm works based on the principle of destroy and recreate, where at each iteration some solutions are destroyed and new ones are generated in different ways to find the best solution according to fitness criteria. The authors of [20] discuss DC-BAP with the aim of minimizing total delays in departures. A GA-based hybrid algorithm is developed by including some features of the branch & bound (B&B) method. For comparison purposes, the hybrid method along with counterparts (i.e., standard GA and CPLEX) are implemented on different data sets, which include small and large data instances. The results from experiments show the effectiveness of the hybrid method. Another DC-BAP study is carried out in [21] with the primary objective of reducing delays in departures. This study considers some new tidal constraints to make the problem more realistic. The DC-BAP is formulated in three different ways, i.e., standard MILP, time-indexed variables-based MILP (TI-MILP), and MILP based on sequence variables (S-MILP). Furthermore, the authors develop some data sets based on real-time data, which can be utilized in the future as benchmark data sets. Subsequently, these three models are solved on a CPLEX solver and simulation results demonstrate that the time-indexed-based MILP achieves higher efficiency with lower computation time. The DC-BAP is investigated in another study [22], where the major aim of the study is to perform a robust berth allocation and to achieve minimum turnaround-time. A recently developed metaheuristic, namely, grey wolf optimization (GWO) is adopted to handle DC-BAP. Moreover, several uncertainties are also taken into account to make the problem more practical, i.e., uncertainty in arrival times and operational times of vessels. The results from experiments confirm the efficiency of GWO in solving the BAP with uncertainties over state-of-the-art methods, such as GA and CPLEX. Another study [23] solves a new version of BAP, where multiple quays instead of a single quay and continuous berthing layouts are assumed. They formulate this problem as an integer linear model and then solve it using GA. To confirm the effectiveness of the proposed method, several simulations are performed with different sized data sets and the results show the effectiveness against the compared approaches.

For the discrete and dynamic BAP (DD-BAP), there have also been significant efforts in the recent literature to solve the problem using mainly metaheuristic and evolutionary approaches. A genetic algorithm (GA) is adopted to solve the DD-BAP in [24], where the fundamental motive is to minimize service time of arriving ships that includes waiting time and handling time. Furthermore, unlike existing

studies, the authors of this work modeled BAP such that each berth has different handling productivity. The results from simulations show that their proposed GA-based approach can solve larger instances as compared to a CPLEX solver. Another study presented in [25] also develops a heuristic-based GA to solve the same problem, i.e., DD-BAP. The authors construct a mixed-integer programming (MIP) model that considers a discrete berthing layout along with dynamic vessel arrivals with the aim of reducing delays in departures as well as shifting workload in night-times. Several experiments were also conducted to affirm the performance of the proposed GA method and results show the efficacy of the newly developed algorithm over counterparts. The authors of [26] proposed an evolutionary-based method, i.e., an enhanced differential evolution (EDE) algorithm, to deal with DD-BAP. This study has two primary objectives: (i) reducing delays in ships' departure times and (ii) total handling time minimization. Furthermore, to check the productivity of the proposed EDE method, several benchmark approaches are also implemented, i.e., GSSP, tabu search, and particle swarm optimization (PSO). Results are provided demonstrating the superiority of the proposed method over counterparts in terms of the aforementioned objectives.

In this paper, we propose using a cuckoo search algorithm (CSA) to solve the berth allocation problem. The CSA was first introduced in our previous work [10] for solving the dynamic and continuous berth allocation problem (DC-BAP). In this study, we extend the BAP mathematical formulation by considering penalty for non-optimal berth position, safety time interval between consecutive berth arrivals, and account for smaller time intervals (30-min time interval), which offer better, more fine-grained berth allocation decisions.

Problem Description

This section first describes in detail the BAP considered in this work, followed by a mathematical formulation as a mixed-integer linear programming problem. Table 1 lists all abbreviations and notations used in this section and throughout the paper.

In the dynamic and continuous berth allocation problem, the MCT has one or more continuous berthing layouts of known lengths that serve vessels arriving at different points in time (i.e., in a dynamic fashion). Let $B = \{1, 2, \ldots, M\}$ denote the set of all possible berthing positions on the wharf of the port. Typically, the BAP considers a particular time period of vessel arrivals, such as the next 48 h. Hence, time is modeled as a set of time intervals $T = \{1, 2, \ldots, K\}$ that can represent some time duration of interest (e.g., an hour or a 30-min interval). Finally, let $S = \{1, 2, \ldots, N\}$ denote the set of ships arriving at the terminal. For each ship, the estimated time of arrival (ETA), the preferred berthing position (PBP), the ship's length, and the estimated (or requested) time of departure (ETD) are known in advance.

In the ideal scenario, as soon as a vessel arrives at the MCT, it should be moored at its preferred berthing position. If the MCT cannot serve the vessel at the time of arrival, the vessel must be towed to the waiting area of the terminal, as shown in Fig. 1, and/or berth at a non-optimal berthing position. In the first scenario, the number of ships in the waiting area increases, causing congestion and navigational challenges to arise at the seaside of the terminal. In this

Table 1 Nomenclature

Name	Explanation
Acronyms	
BAP	Berth allocation problem
BP	Berthing position
CSA	Cuckoo search algorithm
DC-BAP	Dynamic and continuous BAP
ETA	Estimated time of arrival
ETD	Estimated time of departure
GA	Genetic algorithm
HT	Handling time
LoS	Length of ship
MCT	Maritime container terminal
PBP	Preferred berthing position
QCs	Quay cranes
WC	Waiting cost
WT	Waiting time
Indices	
$b \in B = \{1, 2, \ldots, M\}$	Berthing position
$s \in S = \{1, 2, \ldots, N\}$	Individual ship
$t \in T = \{1, 2, \ldots, K\}$	Single time period
Notations	
BP_s	Berthing position of ship s
BT_s	Berthing time of s
ETA_s	Estimated time of arrival of s
ETD_s	Estimated time of departure of s
HC_s	Handling cost of s per time period
HT_s	Handling time of s
L_b	Length of berth b (only for discrete berthing layout)
L_s	Length of ship s
LDC_s	Late departure cost of s per time period
LDT_s	Late departure time of s
NBC_s	Non-optimal berthing cost of s
PBP_s	Preferred berthing position of s
SET	Safety entrance time
W	Length of wharf
WC_s	Waiting cost of s per time period
WT_s	Waiting time of s

case, the MCT incurs an extra waiting cost WC_s against the ship s for the duration of s' waiting time (e.g., calculated in EURO/hour).

Once the ships are moored at their assigned berthing position, the quay cranes (QCs) start working in order to load/unload containers. Container handling resources (e.g., number of QCs, gantry cranes) are allocated to ships based on the handling rate that is negotiated between the MCT operator and the shipping company. The handling time for ship s at the assigned berthing position is calculated based on the total number of containers loaded on that ship and the requested handling productivity. Note that this study adopts a data set for implementation with precomputed handling times for all arriving vessels. However, the handling productivity is reduced if the vessel is assigned to a berth position other than its preferred berthing position (PBP) [11, 12]. The PBP typically depends on vessel characteristics, such as the vessel length or vessel load as well as port-related considerations, such as the number of available quay cranes of the berthing area allocated to a particular ship. Hence, the major cause of handling productivity reduction is the increased loading/unloading and transfer time of containers from the assigned (suboptimal) berth to storage.

Finally, each ship s specifies its own estimated (or requested) time of departure ETD_s and the MCT is supposed to complete the tasks (loading/unloading) of s before the ETD_s, $\forall s \in S$. Otherwise, the MCT is liable to pay a late departure penalty cost LDC_s for the duration of the delay (e.g., calculated in EURO/hour) to the shipping companies. Overall, the aim of the MCT is to minimize the total waiting, handling, and late departure costs for all arriving vessels at the port.

Mathematical Formulation

Before disclosing the mathematical formulation of BAP, we list the assumptions that are considered in our work.

- The total number of arriving ships at the planning horizon is known.
- Each berth position is able to handle only one vessel at a particular time.
- A ship takes consecutive time intervals until loading/unloading completes (i.e., no shifting).
- The ETA and ETD for each vessel are known and will not change.
- Estimated processing time for each vessel is known or can be easily calculated.
- Each ship has a preferred berthing position and it is known.
- Each ship can berth and be served at a set of known berthing positions.
- All berths are idle at the start of the time horizon.
- The length of the wharf is known.

The total processing cost of a vessel s that is scheduled for berthing at position BP_s at time BT_s includes a waiting cost, a handling cost, and late departure penalty, expressed by the following function:

$$\begin{aligned}&\mathrm{Cost}\left(s, \mathrm{BP}_s, \mathrm{BT}_s\right)\\&= \mathrm{WT}_s \cdot \mathrm{WC}_s + \mathrm{HT}_s \cdot \left(\mathrm{HC}_s + f(s, \mathrm{BP}_s)\right) + \mathrm{LDT}_s \cdot \mathrm{LDC}_s.\end{aligned} \tag{1}$$

The first term in Eq. (1), $\mathrm{WT}_s \cdot \mathrm{WC}_s$, represents the waiting cost when a vessel has to wait for berthing. The waiting time WT_s of vessel s is calculated as the difference between the berthing time BT_s and the planned time of arrival ETA_s:

$$\mathrm{WT}_s = \mathrm{BT}_s - \mathrm{ETA}_s, \quad \forall\, s \in S. \tag{2}$$

The second term $(\mathrm{HT}_s \cdot (\mathrm{HC}_s + f(s, \mathrm{BP}_s))$ in Eq. (1) corresponds to the total handling cost for loading or unloading containers, which is proportional to the handling time. Basically, the handling time HT_s of any ship s depends on the total volume of containers to be loaded or unloaded on the vessel, the number of quay cranes available at this berth, and the average handling productivity of the cranes. However, in this study, we assume handling time as an input for solving BAP, as considered in [27]. Even though we consider the handling time as input in this work, we can easily extend our formulation to compute the handling time based on the vessel's load and quay crane characteristics.

Furthermore, unlike our previous work [10], this study considers a new aspect of the handling cost. Without loss of generality, we also introduce the function $f(s, \mathrm{BP}_s)$, which will penalize the handling cost based on the assigned berthing position BP_s. In this work, we use the following formulation for f:

$$f\left(s, \mathrm{BP}_s\right) = \left|BP_s - PBP_s\right| \cdot NBC_s, \tag{3}$$

which will penalize the total service cost based on the absolute difference between the assigned berthing position BP_s and the preferred berthing position PBP_s for the entire duration of handling that a ship spends at the non-optimal position. The factor NBC_s defines the penalty cost for ship s against non-optimal berthing position per unit (meter) distance. In order to NBC_s to be comparable to the handling cost HC_s, we set NBC_s as the ratio of HC_s to the length of the wharf W. However, the function f can be easily adjusted to account for other, more complex practical scenarios. For example, if a particular ship s cannot be accommodated at some berthing positions, f can return an infinite cost to ensure that s will not be assigned to one of those positions.

The final term in Eq. (1), $\mathrm{LDT}_s \cdot \mathrm{LDC}_s$, computes the late departure penalty when a vessel departs after its estimated time of departure. The delayed departure time LDT_s

of vessel s (if any) is calculated as the difference between the time s completes its operations and the estimated time of departure ETD_s:

$$\mathrm{LDT}_s = max\{\mathrm{BT}_s + \mathrm{HT}_s - \mathrm{ETD}_s, 0\}, \quad \forall\, s \in S. \tag{4}$$

The goal of the dynamic and continuous berth allocation problem is to find the optimal berthing positions and times for all vessels such that the total processing cost is minimized, as shown by the following objective function:

$$\textbf{minimize} \sum_{s \in S} \sum_{b \in B} \sum_{t \in T} x_{sbt} \cdot \mathrm{Cost}\left(s, \mathrm{BP}_s, \mathrm{BT}_s\right), \tag{5}$$

subject to the following set of **constraints**:

$$x_{sbt} \in \{0,\ 1\}, \qquad \forall\, s \in S,\ b \in B,\ t \in T \tag{6}$$

$$\sum_{b \in B} \sum_{t \in T} x_{sbt} = 1, \qquad \forall\, s \in S \tag{7}$$

$$\mathrm{BT}s \geq ETA_s, \qquad \forall\, s \in S \tag{8}$$

$$\left|\mathrm{BT}_s - \mathrm{BT}_{s'}\right| \geq \mathrm{SET}, \qquad \forall\, s, s' \in S \tag{9}$$

$$\mathrm{BP}_s + L_s \leq W, \qquad \forall\, s \in S \tag{10}$$

$$\sum_{s' \neq s \in S} \sum_{b=\mathrm{BP}_s - L_{s'}+1}^{\mathrm{BP}_s + L_s} \sum_{t=\mathrm{BT}_s - HT_{s'}+1}^{\mathrm{BT}_s + \mathrm{HT}_s} x_{s'bt} = 0, \quad \forall\, s \in S. \tag{11}$$

In Eq. (6), the variable x_{sbt} is 1 if vessel s is assigned to berthing position b at berthing time t, and 0 otherwise. Constraint (7) ensures that each arrived ship at the MCT will be assigned at a particular berthing position only once during the planning time. Constraint (8) warrants that the scheduled berthing time BT_s of ship s must always be later than or equal to its planned time of arrival ETA_s. Constraint (9) ensures a minimum safety entrance time between berthing times of any two ships, s and s', since most ports will only berth one ship at a time due to physical constraints at the port's entrance. Constraint (10) guarantees that the berthing position BP_s of ship s plus its length L_s will always be less than or equal to the total length W of the wharf. Finally, constraint (11) ensures that no two ships can share (part of) the same berth during the handling times of the two ships. For instance, suppose a ship s is planned to be berthed at time 5 h, has handling time equal to 4 h, utilizes berthing position 500 m, and its length is 300 m as shown in Fig. 2. According to constraint (11), no other ship can use berthing positions from 500 to 800 m (as length of ship s is 300 m) in the time interval 5–9 h. In addition, a second ship s' with length 200 m and handling time 3 h cannot use the berthing positions from 301 to 800 m in the time interval 3–9 h as it would overlap with ship s. Visually, this constraint ensures that the two rectangles denoting the time intervals and berthing positions allocated to the two vessels shown in Fig. 2 can never overlap.

As discussed earlier in Section "Introduction", the case of the the discrete berthing layout is a basic variation of the

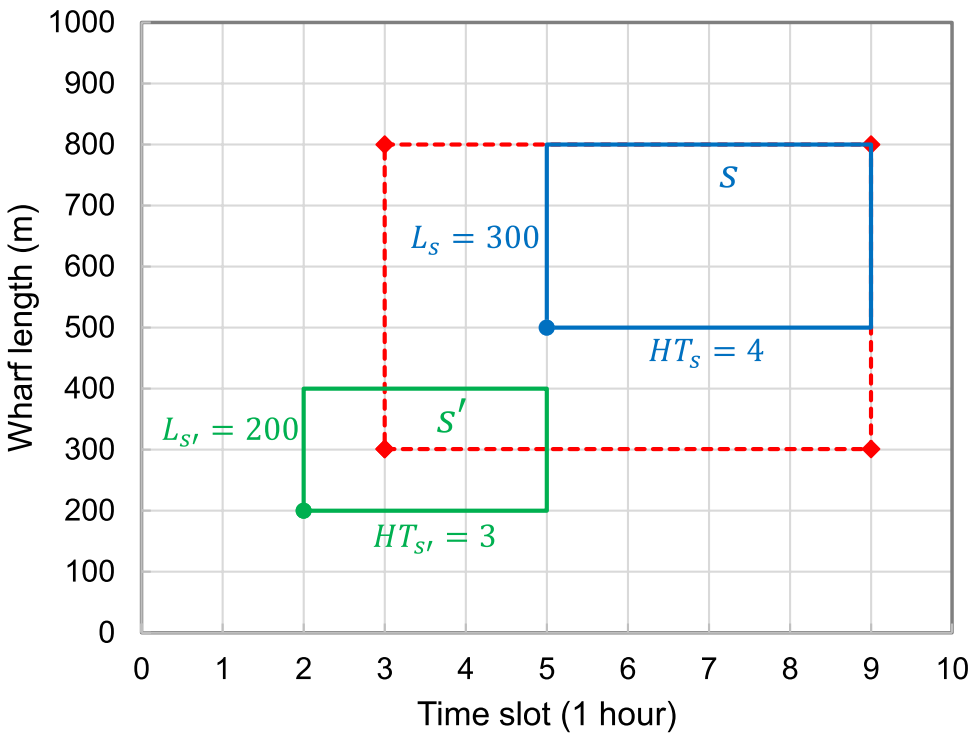

Fig. 2 Berth allocation example for ship s (blue square) with $\mathrm{BP}_s = 500$ and $\mathrm{BT}_s = 5$ and ship s' (green square) with $\mathrm{BP}_{s'} = 200$ and $\mathrm{BT}_{s'} = 2$. The red dotted square shows the solution area that is not valid for ship s' according to constraint (11)

continuous layout scenario. If the berthing layout is discrete, then each berthing position b represents a specific berth. The objective function shown in Eq. 5 remains the same and constraints (6)–(9) still apply, whereas constraints (10) and (11) are, respectively, replaced by the following two constraints:

$$L_s \leq L_b, \qquad \forall\, s \in S,\ b = \mathrm{BP}_s \tag{12}$$

$$\sum_{s' \neq s \in S} \sum_{t=\mathrm{BT}_s - \mathrm{HT}_{s'}+1}^{\mathrm{BT}_s+\mathrm{HT}_s} x_{s'bt} = 0, \qquad \forall\, s \in S,\ b = \mathrm{BP}_s. \tag{13}$$

Specifically, constraint (12) ensures that the length L_s of ship s is less than or equal to the length L_b of berth b that is assigned to ship s. Finally, constraint (13) guarantees that a berth is never assigned to two ships during the same time intervals.

Proposed and Benchmark Approaches

In this section, we uncover the details of the proposed method (i.e., CSA) and state-of-the-art heuristic-based popular approach, i.e., GA.

Cuckoo Search Algorithm

CSA is a swarm-based metaheuristic optimization algorithm that was developed by Yang et al. [28]. The CSA emulates the breeding behavior of some cuckoo species, which have a fascinating reproduction mechanism. In particular, some cuckoos lay their eggs in nests of other birds (often nests of other species' nests), where they may discard eggs of other birds in order to enhance the hatching ratio of their own eggs. Then, the host birds take care of cuckoo eggs as they presume that the eggs belong to them. Nonetheless, sometimes the host birds distinguish between their own eggs and the alien eggs. Accordingly, either the discovered alien eggs are thrown out of the current nest or new nests are built in new locations. Inspired by this particular mechanism of laying eggs by the cuckoo birds, the following three standard rules are adopted to employ CSA for optimization problems [28]:

1. each cuckoo lays one egg at a time at a randomly chosen nest;
2. the best nests with high-quality eggs will not be removed and will be carried over to the next generation;
3. the quantity of host nests is fixed and the egg dumped by a cuckoo is discovered by a host bird with a probability $p_a \in (0, 1)$.

In this study, each nest denotes a solution set that includes the berthing times and berthing positions for all arriving vessels. An egg represents either a berthing position or time, while a cuckoo egg represents a new (and better) berthing position or time. The total number of host nests reflects the total search space at each iteration of the algorithm. In this work, 100 host nests are considered and each nest contains $2N$ eggs, where N is the total number of vessels. Hence, the total number of eggs in a nest is double the total number of arriving vessels. Overall, the high-level goal of the algorithm is to use cuckoo eggs (better solutions) to replace not-so-good eggs in the nests.

Algorithm 1 Cuckoo Search Algorithm for BAP

```
X[1..k] = Generate initial population of host nests
for t = 1 to max number of iterations do
    for i = 1 to k do
        x_new = X[i] + α ⊕ Levy(λ)
        if (fitness(x_new) < fitness(X[i])) then
            X[i] = x_new
        end if
    end for
    for i = 1 to k do
        if (rand(0,1) < p_a) then
            X[i] = Generate new host nest
        end if
    end for
    x_best = Find nest with lowest fitness value in X
end for
```

Algorithm 1 presents the pseudocode of the Cuckoo Search Algorithm, which begins with a randomly distributed initial population of k=100 host nests over the search space (line #1). In each iteration of the algorithm, the reproduction step is performed first, where new solutions are generated by replacing some existing eggs with cuckoo eggs in randomly selected nests (lines #3–8). The rationale for the egg replacements is that if a cuckoo egg is very similar to a host egg, then this egg has lesser chances to be discovered. Thus, a random walk is performed through Lévy flights in order to generate new nests (i.e., new solutions):

$$X_i^{(t+1)} = X_i^{(t)} + \alpha \oplus \text{Levy}(\lambda), \tag{14}$$

where t denotes the current iteration number, X_i the solution for nest i, and α ($\alpha > 0$) the step size. The $\oplus$ operation denotes entrywise multiplication. A random walk in Lévy flights is performed from a Lévy distribution with a scale parameter λ [29]. The primary aim of performing random steps is to increase the possibility of finding the global solution instead of becoming stuck in a local optimum. A new solution replaces a current solution if its fitness score is lower than the fitness score of the current solution (lines #5–7). The fitness of each possible solution is evaluated using the objective function of the BAP presented in Eq. (5) and accounts for the total processing cost, which includes waiting, handling, and late departure penalty costs. In addition, it is possible for some cuckoo eggs to be discovered by host birds with a discovering probability p_α. In our work, p_α is set to 0.45 as reported in [28]. In this case, the nests with the discovered cuckoo eggs are abandoned and new ones are built; as a result, the exploration of the search space is enhanced (lines #9–13). Finally, the best solution across all iterations is kept (line #14). The above steps repeat until either the total number of iterations is reached (which equals 100 in this work) or there has been no fitness improvement for some iterations.

Genetic Algorithm

GA is one of the most popular algorithms from the metaheuristic family and is based on the evolution process in natural systems, i.e., Darwin's principles of survival of the fittest individuals [30]. Since GA has a high convergence rate compared to most metaheuristics, it can solve big and high complexity problems relatively quickly. GA is a population-based approach that finds a better solution by managing a population that contains various possible solutions, which are revised generation to generation by employing several genetic operators, including selection, crossover, and mutation. The complete working procedure of GA consists of the following six steps [31].

Initial population generation: A random population of different possible solutions (chromosomes) is generated at the first step. A single solution is known as a gene, a solution set is known as a chromosome, and all solution sets form a population.

Fitness evaluation: The fitness values of all solutions are evaluated to demonstrate the goodness of solutions.

Selection: The selection phase aims to select a couple of fittest individuals (parents), which are used for the next generation. There are more chances for the selection of individuals for reproduction having the best fitness value. A few of the fittest individuals are selected as parents for the next generation.

Crossover: Crossover is employed to produce offsprings, where, a child adopts one portion of its characteristics from one parent and the other part from the second parent.

Mutation: A mutation operator is also applied to some portion of the solutions in the new generation to avoid premature convergence and maintain diversity in the population. In addition, it also ensures that the probability of any solution is never zero.

Termination of algorithm: The fitness values of the new population are calculated and the same process repeats until conditions of termination are met. Termination conditions include maximum available computation time, maximum iterations, and a maximum number of generations.

Experimental Results

In this section, we present the settings, data sets, and results of our extensive berth allocation simulations. In addition to the CSA method, we implemented a popular population-based heuristic method (i.e., GA) proposed in the recent literature [32], as well as the exact MILP approach. The implemented algorithms are coded in MATLAB 2019b on a Windows 10 PC with COREi7 processor and 8GB RAM.

Table 2 One-hour interval data set employed for experiments [27]

Ship #	ETA	HT	ETD	PBP	LoS
1	4	3	8	778	128
2	5	5	11	1416	113
3	10	5	15	957	334
4	4	2	8	1437	423
5	13	3	16	362	173
6	15	1	18	1015	391
7	11	3	15	434	338
8	6	2	9	1008	140
9	9	1	11	1043	302
10	2	2	5	102	194

For our experiments, two data sets were used. The first problem data set is taken from Şahin et al. [27] and shown in Table 2. The data set contains a number of arriving ships in a day along with the estimated arrival time, handling time, estimated departure time, preferred berth position, and length for each arriving ship. This data set was adapted to generate more ships within the same planning horizon of 1 day. For instance, the minimum and maximum handling times for any ship are 2 h and 5 h, respectively (see column #3 in Table 2); so, we generate randomly handling times for all ships in between 2 and 5 h; the same policy is used for the other parameters. In order to stress-test the algorithms and also to test other planning horizons, a second larger data set was generated randomly by employing uniform distribution with realistic settings for all parameters, and a planning horizon of up to 1 week.

The quay is continuous and has a length of 2000 m in both data sets. For the cost calculations of the different algorithms, we have used the values 10, 5, and 5 Euros per hour handling cost, waiting cost, and late departure cost, respectively [27]. When a vessel is moored at a berth position other than its optimal berthing position, a penalty based on the absolute difference between the assigned berth position and the optimal berthing position is added to the handling cost. The non-optimal berthing cost (NBC) is calculated as follows: $\mathrm{NBC} = \mathrm{HC}_s/W = 0.005$ Euros per meter.

Furthermore, unlike our previous [10] and many other studies (e.g., [27, 32]), we use a time interval of less than an hour for the experiments, i.e., 30 min, to make the problem more practical and to reduce the time loss in the simulations. For example, if a ship takes 5 h and 30 min to load and unload, a time interval of 1-h wastes 30 min, since the

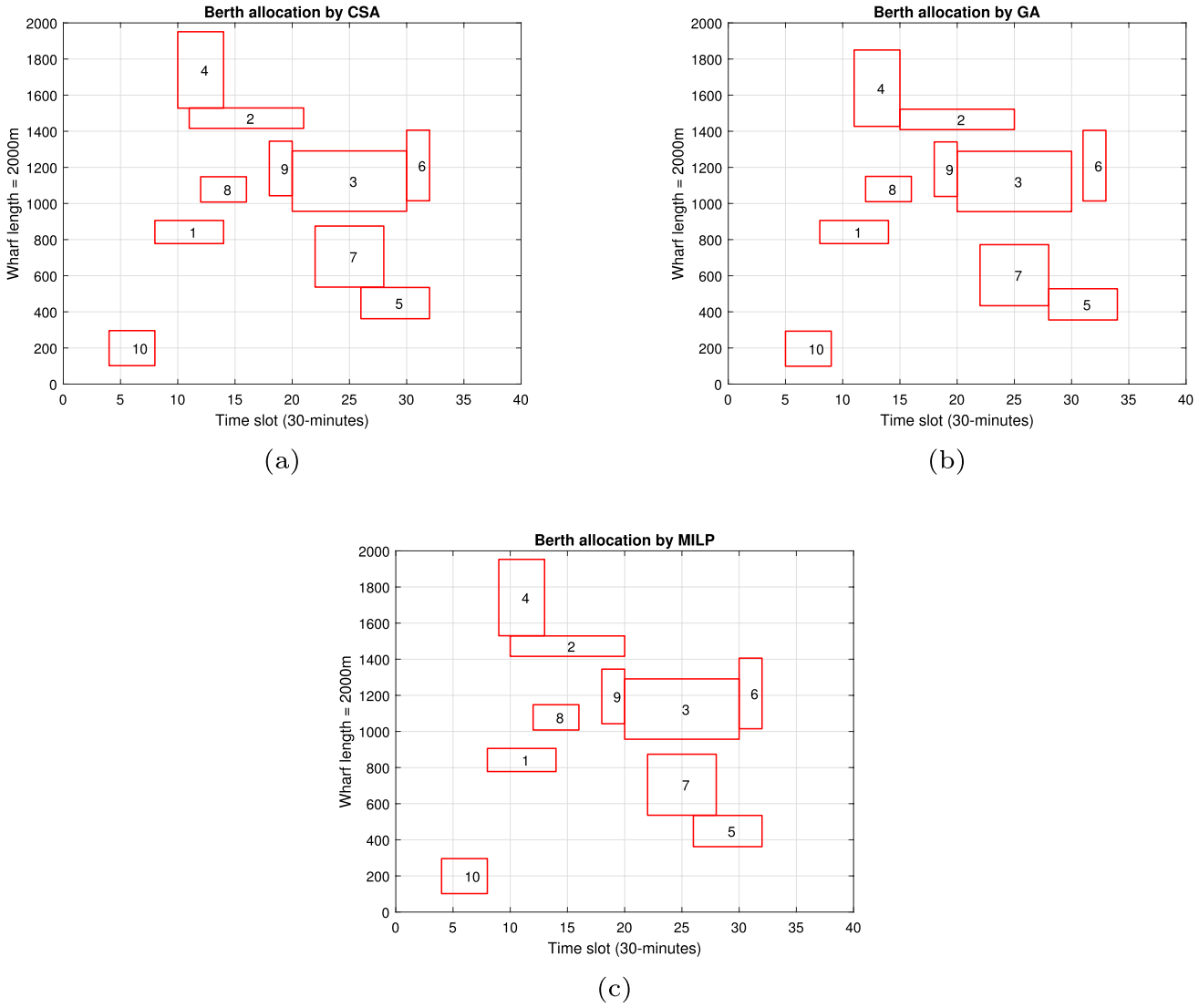

Fig. 3 Berth allocation solution, when we consider 30-min time interval, generated by **a** CSA, **b** GA, and **c** MILP

algorithms work with a time interval of 1 h and cannot schedule a ship before the next hour, i.e., 6. A more fine-grained time interval of 30 min avoids this situation. The same problem instances were used in the case of the 30-min time intervals; however, all times in Table 2 are multiplied by two to generate 30-min values, since the original data contain values based on 1-h time interval.

Next, we present simulation results when our proposed and benchmark methods are implemented considering a 30-min time interval. When we implement the 30-min time interval, this study considers a new constraint, shown in Eq. (9), which guarantees a minimum and realistic safety entrance time (SET) of one 30-min time interval between the berthing times of any two vessels. This constraint is yet another benefit derived from using a smaller time interval as using a 1-h SET would be too long and wasteful. The results presented in Fig. 3 show the three solutions for berth assignment when we consider a 30-min time interval and the ten arriving ships shown in Table 2. All three approaches allocate ships to the available berth positions and time slots based on the primary objective of this study, which is to minimize the total processing cost, as shown in Eq. (5). In Fig. 3, the vertical axis shows the berth positions, while the horizontal axis shows time divided into 30-min time intervals. Each rectangle in this figure denotes the berthing periods and berthing positions assigned to an arriving vessel. The label within a rectangle indicates the ship index. Unlike the previous study, a safety entrance time can be noticed as no two ships berth at the same time. By comparing the sub-figures, it becomes evident that the three different algorithms offer three different solutions to the problem, with some vessels berthed at different positions and/or berthing times. For example, the estimated arrival time of ship 5 is the 26th time slot (30-min each time slot). The berthing time proposed by MILP and CSA is the 26th slot, while GA chose the 28th slot because of the berth placement of ship 7. Hence, ship 5 does not have to wait before berthing when using the CSA and MILP approaches. On the contrary, ship 5 has to wait for 1 h (2 slots of 30 min) if GA is employed.

Figure 4 shows the waiting times incurred by the ten ships when using the three approaches. From Fig. 4, it can be seen that only one vessel has to wait (for 30 min only) when MILP is used. However, our newly proposed CSA method provides a solution with a maximum waiting time of two time periods (30 min each) for any ship and only two ships have to wait before berthing. In contrast, when using GA, five ships have to wait for optimal berthing and the maximum waiting time for ship 2 is five time slots (i.e., 2.5 h). Such waiting times are very high and in turn can cause late departures. From these results, we can conclude that the CSA method outperforms GA in terms of reducing the waiting times.

Figure 5 shows the requested departure times of all arriving ships and the proposed departure times by our developed

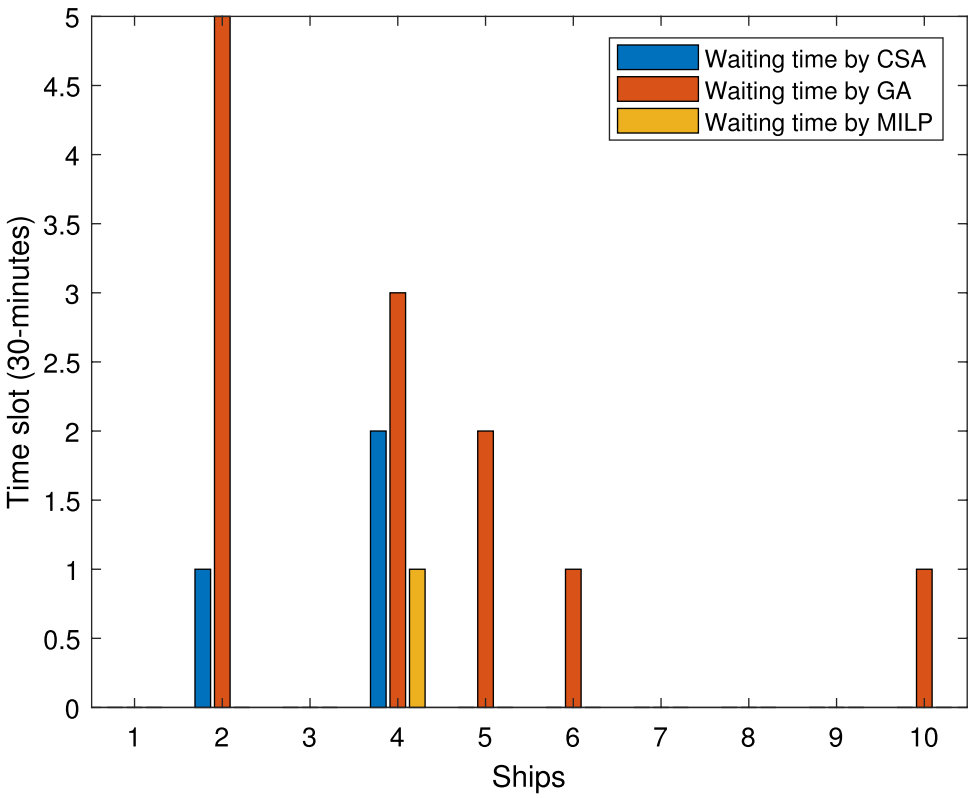

Fig. 4 Ships' waiting times when using CSA, GA, and MILP approaches

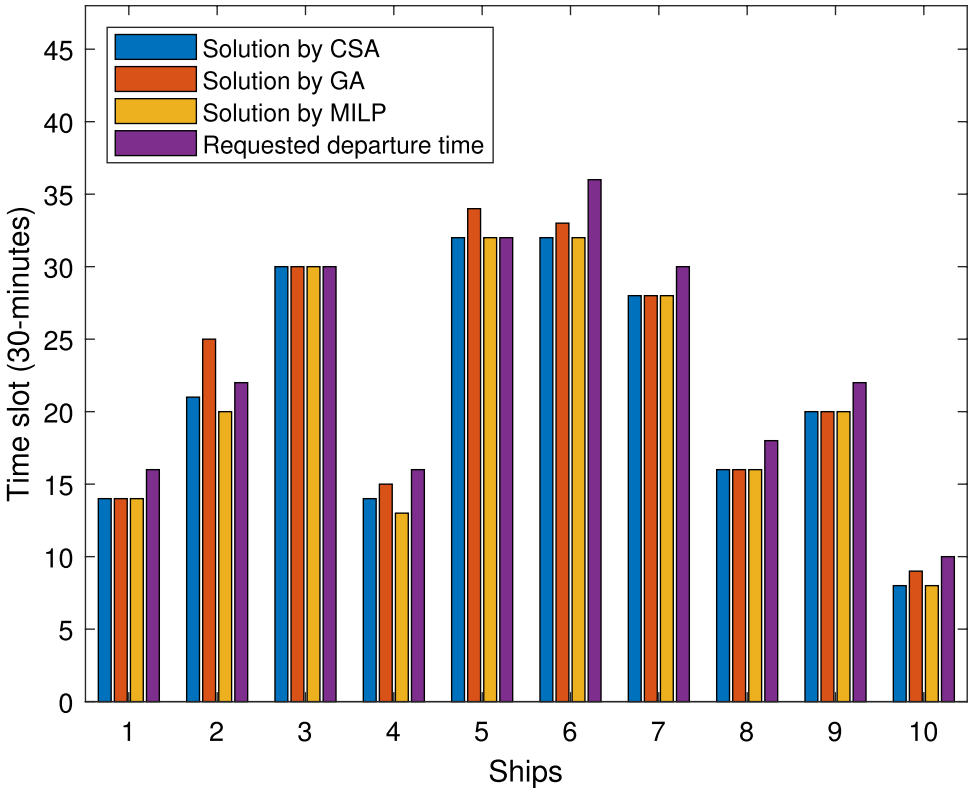

Fig. 5 Requested and planned departure times for each ship using CSA, GA, and MILP

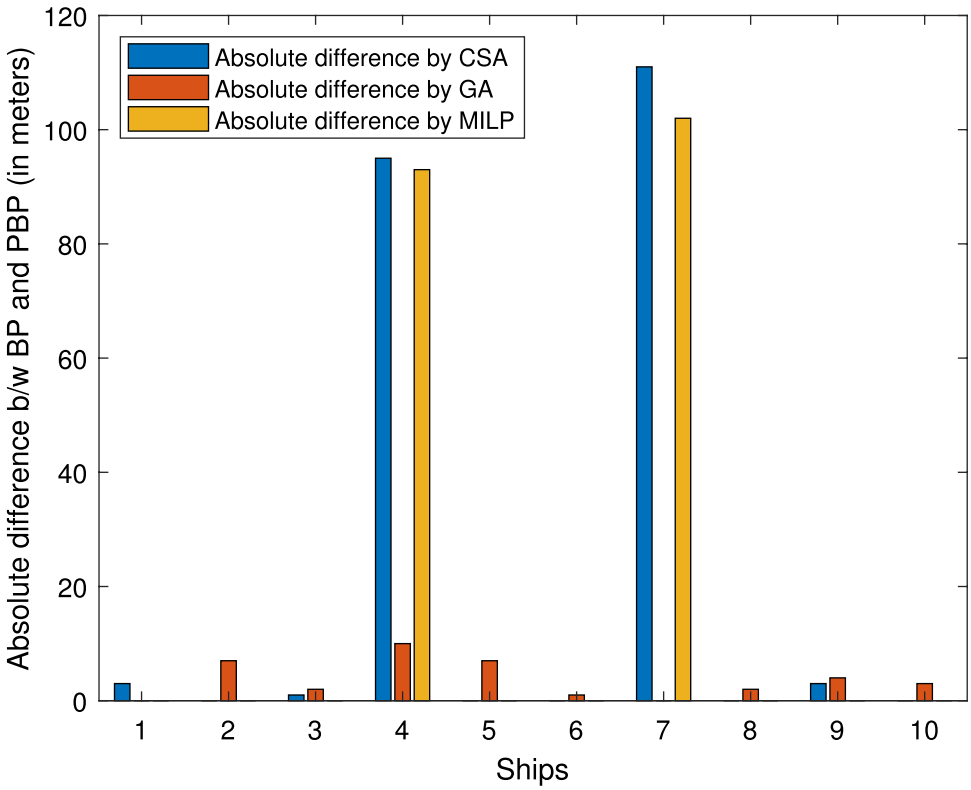

Fig. 6 Absolute difference between the berthing position and the preferred berthing position of the ten ships provided by CSA, GA, and MILP

and compared algorithms, i.e., CSA, GA, and MILP. From this figure, it can be seen that no vessel departs late using either MILP or our proposed CSA approach. On the other hand, with GA, two ships depart late: ships 2 and 5 are three slots and two slots late, respectively. Once again, we conclude that CSA shows higher performance in minimizing late departures compared to GA.

The results presented in Fig. 6 show the absolute difference between the optimal berthing positions and berthing positions of the 10 ships provided by the implemented algorithms, i.e., CSA, GA, and MILP. We can observe from this figure that MILP always provides optimal berthing positions except for only two ships, namely, ships 4 and 7. CSA's behavior in terms of berth placement is similar to MILP as it also uses non-optimal berthing positions for ships 4 and 7, plus some very small deviations for three more ships. On the contrary, 8 out of the 10 ships are moored to other than optimal berthing positions when using GA, albeit with small differences. Even though ships 4 and 7 berth at (near) optimal positions with GA, they cause major waiting times and departure delays for ships 2 and 5, respectively, leading to higher cost solutions. Overall, by considering all results presented so far, it is evident that it is not possible to achieve both zero waiting times and optimal berthing positions for all ships from Table 2. The best solutions comes from delaying ship 4 by 30 min (due to the safety entrance time constraint) and using non-optimal berthing positions from ships 4 and 7 in order to avoid delays for other ships. In this regard, the solution provided by CSA is much closer to the optimal MILP solution compared to the GA solution.

For comparison purposes, the total processing cost along with the computation times for all implemented algorithms, i.e., CSA, GA, and MILP, are shown in Table 3. In addition, we have also varied the number of arriving ships to study the scalability of the proposed method. We tested the three approaches on multiple instances, where 10–30 ships and a 30-min time interval are considered, while all other parameters are the same, i.e., the length of the quay, the arrival pattern of the ships, and the berth layout. It can be seen from Table 3 that MILP provides the optimal solution in terms of minimum processing cost in all cases; however, our proposed CSA-based algorithm also provides a near-optimal solution in all cases when we compare it with the alternative GA approach. For example, the results of the first instance show that MILP achieves the lowest cost of 285 Euros, as expected, CSA also has a lower cost as compared to GA, which is 322 Euros; however, total processing cost with the GA is significantly higher (347 Euros). A similar pattern is observed for the other instances; the solutions proposed by CSA are only slightly costlier (ranging from 4.9 to 14.1%) than the optimal MILP solution, while the solutions of GA are more expensive (ranging from 11.9 to 20.5%). Furthermore, Table 3 also presents the computational time of all three algorithms when using 30-min time interval. It can be noticed from this table that GA has the minimum computational time and MILP has higher computational time. Our

Table 3 Comparative analysis in terms of cost and computation time when using data instances from [27] with 30-min time intervals

Method		CSA		GA		MILP	
No.	No. ships	Cost (€)	Time (s)	Cost (€)	Time (s)	Cost (€)	Time (s)
1	10	302	0.94	347	0.21	288	207.26
2	15	434	1.26	470	0.30	395	318.84
3	20	595	1.43	620	0.21	535	404.59
4	25	680	2.01	688	0.57	615	530.96
5	30	827	3.08	832	0.98	725	644.01

Table 4 Comparative analysis in terms of cost and computational time when using randomly generated data instances with 30-min time intervals

Method		CSA		GA		MILP	
No.	No. ships	Cost (€)	Time (s)	Cost (€)	Time (s)	Cost (€)	Time (s)
1	10	355	1.16	380	0.40	305	195.25
2	20	702	1.71	722	0.42	620	333.29
3	30	1005	2.83	1020	0.56	895	557.70
4	40	1322	4.99	1785	0.89	1080	719.98
5	50	1730	6.92	1792	1.56	1460	957.05
6	60	2102	5.07	2137	1.22	–	–
7	70	2527	6.49	2597	1.91	–	–
8	80	2990	1.29	3060	2.42	–	–
9	90	3092	17.89	3215	4.94	–	–
10	100	3635	12.82	3692	4.07	–	–

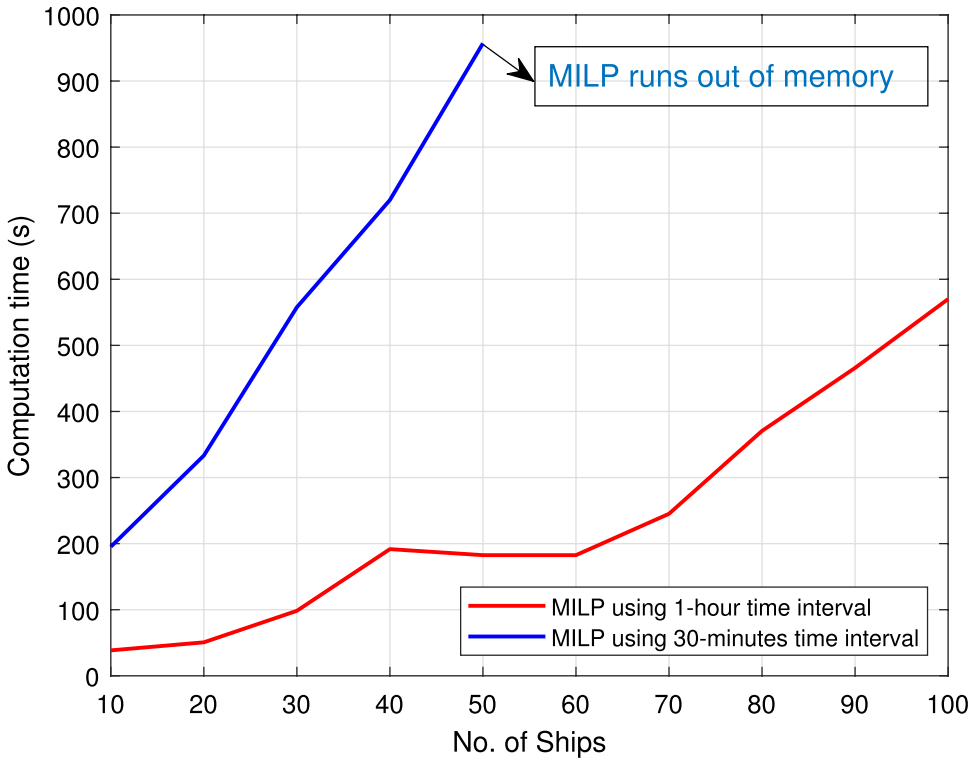

Fig. 7 Computation time of MILP using random data (10–100 ships) when we consider both time intervals, i.e., 1 h and 30 min

proposed CSA-based method takes only slightly higher time compared to GA (up to 3 s vs. 1 s) but provides better cost reduction. Compared to CSA, the MILP is orders of magnitude slower and takes on average 250× more computational time to solve the berth allocation problem.

In order to stress-test the algorithms, we also generated a larger synthetic data set with 20 uniformly random instances (10–100 vessels), 10 with 1-h interval and 10 with a 30-min interval. In addition, we increased the planning period from 1 day to 1 week when the number of ships is greater than 40. Table 4 shows a comparative analysis when using the data instances that were randomly generated with 30-min time intervals. The overall trends in terms of cost are the same as before: MILP offers the lowest cost, followed closely by CSA (with 12.3–22.4% higher cost), while GA lead to the highest cost with up to 65.3% worse cost than the minimum. In terms of computation times, both GA and CSA scale well with very low running times, less than 5 and 18 s, respectively. MILP, on the other hand, leads to much longer running times (3–16 min) that are about *two orders of magnitude* higher compared to CSA and grow super-linearly. To make matters worse, when the number of ships in the problem set becomes greater than 50, the MILP runs out of memory and thus it is unable to solve large problem instances. Overall, CSA is a very efficient approach that leads to near-optimal solutions in terms of total processing costs.

Finally, Fig. 7 shows the computation time of MILP when using a 1-h or a 30-min time interval as we increase the number of ships in the problem data set. In both scenarios, the computation time grows super-linearly, with the 30-min time interval case growing much more aggressively, even though the number of intervals is only double compared to the 1-h interval case. As previously mentioned, MILP runs out of memory and cannot solve problem instances with more than 50 cases when using 30-min intervals. Therefore, MILP cannot be used for larger, more realistic problem sizes; an observation that has been reported previously in other studies, such as requiring over 100 h of CPU time for real-world instances [32]. Such times are certainly not acceptable in the context of MCT operations.

Conclusions

This study focuses on the berth allocation problem with dynamic ship arrivals, where a metaheuristic-based cuckoo search algorithm (CSA) is proposed to solve the BAP. In addition, we implemented two benchmark methods for comparison, a well-known metaheuristic genetic algorithm (GA) and an exact approach (MILP). Unlike existing studies, and to make the problem more practical, a fine-grained time interval of 30 min is used in this study along with other practical constraints, such as a safety time distance between berths. The proposed and compared approaches

are implemented on multiple data instances generated from a benchmark data set. In addition, randomly (uniformly) generated data instances with up to 100 vessels and a planning horizon up to a week are used for experiments to test the flexibility and scalability of the proposed method. The results show that our proposed algorithm (CSA) has higher efficiency in terms of minimum processing cost for all incoming ships compared to GA. Compared to MILP, our proposed CSA algorithm provides a near-optimal solution at a fraction of the computation time. Moreover, when we implement all algorithms on large data sets, the MILP algorithm runs out of memory and cannot provide an optimal solution in reasonable time. Overall, CSA beats GA in terms of processing cost and outperforms MILP in terms of computation time, and thus provides near-optimal solutions in affordable computation times.

Funding This work was supported by the European Regional Development Fund and the Republic of Cyprus through the Cyprus Research and Innovation Foundation (STEAM Project: INTEGRATED/0916/0063).

Availability of data and materials Not applicable.

Declarations

Conflict of interest/Competing interests Not applicable.

Code availability Not applicable.

References

1. Aslam S, Michaelides MP, Herodotou H. Internet of ships: a survey on architectures, emerging applications, and challenges. IEEE Internet of Things J. 2020;7:9714–27.
2. Hsu H-P, Wang C-N, Chou C-C, Lee Y, Wen Y-F. Modeling and solving the three seaside operational problems using an object-oriented and timed predicate/transition net. Appl Sci. 2017;7(3):218.
3. Barbosa F, Rampazzo PCB, Yamakami A, Camanho AS. The use of frontier techniques to identify efficient solutions for the berth allocation problem solved with a hybrid evolutionary algorithm. Comput Oper Res. 2019;107:43–60.
4. De A, Pratap S, Kumar A, Tiwari M. A hybrid dynamic berth allocation planning problem with fuel costs considerations for container terminal port using chemical reaction optimization approach. Ann Oper Res. 2020;290(1):783–811.
5. Vessels arrivals at China Ports. 2019. https://www.cargosmart.ai/en/blog/vessels-arrive-at-top-china-ports-with-shorter-delays-in-2019/. Accessed 2 May 2022.
6. UNCTAD: Ports in 2017. 2017. https://unctad.org/system/files/official-document/rmt2018ch4_en.pdf. Accessed 2 May 2022.
7. Michaelides MP, Herodotou H, Lind M, Watson RT. Port-2-port communication enhancing short sea shipping performance: the case study of Cyprus and the eastern Mediterranean. Sustainability. 2019;11(7):1912.
8. Lind M, Michaelides M, Ward R, Herodotou H, Watson R. Boosting data-sharing to improve short sea shipping performance: evidence from Limassol port calls analysis. Tech. Rep. 35, UNCTAD Transport and Trade Facilitation Newsletter No. 82-Second Quarter 2019.
9. Carlo HJ, Vis IF, Roodbergen KJ. Seaside operations in container terminals: literature overview, trends, and research directions. Flex Serv Manuf J. 2015;27(2–3):224–62.
10. Aslam S, Michaelides MP, Herodotou H. Dynamic and continuous berth allocation using cuckoo search optimization. In: Proceedings of the 7th International Conference on vehicle technology and intelligent transport systems (VEHITS), 2021; p. 72–81.
11. Bierwirth C, Meisel F. A survey of berth allocation and quay crane scheduling problems in container terminals. Eur J Oper Res. 2010;202(3):615–27.
12. Bierwirth C, Meisel F. A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. Eur J Oper Res. 2015;244(3):675–89.
13. Jos BC, Harimanikandan M, Rajendran C, Ziegler H. Minimum cost berth allocation problem in maritime logistics: new mixed integer programming models. Sādhanā. 2019;44(6):149.
14. Kavoosi M, et al. Berth scheduling at marine container terminals. Marit Bus Rev. 2019;5(1):30–66.
15. Dulebenets MA. Application of evolutionary computation for berth scheduling at marine container terminals: Parameter tuning versus parameter control. IEEE Trans Intell Transp Syst. 2017;19(1):25–37.
16. Xu Y, Xue K, Du Y. Berth scheduling problem considering traffic limitations in the navigation channel. Sustainability. 2018;10(12):4795–816.
17. Hsu H-P, Chiang T-L, Wang C-N, Fu H-P, Chou C-C. A hybrid GA with variable quay crane assignment for solving berth allocation problem and quay crane assignment problem simultaneously. Sustainability. 2019;11(7):2018–38.
18. Chen L, Huang Y. A dynamic continuous berth allocation method based on genetic algorithm. In: Proceedings of the 3rd IEEE International Conference on control science and systems engineering (ICCSSE), (IEEE) 2017; p. 770–73.
19. Mauri GR, Ribeiro GM, Lorena LAN, Laporte G. An adaptive large neighborhood search for the discrete and continuous berth allocation problem. Comput Oper Res. 2016;70:140–54.
20. Alsoufi G, Yang X, Salhi A. Robust berth allocation using a hybrid approach combining branch-and-cut and the genetic algorithm. In: International Workshop on hybrid metaheuristics, (Springer). 2016; p. 187–201.
21. Ernst AT, Oğuz C, Singh G, Taherkhani G. Mathematical models for the berth allocation problem in dry bulk terminals. J Sched. 2017;20(5):459–73.
22. Xiang X, Liu C, Miao L. A bi-objective robust model for berth allocation scheduling under uncertainty. Transp Res Part E Logist Transp Rev. 2017;106:294–319.
23. Frojan P, Correcher JF, Alvarez-Valdes R, Koulouris G, Tamarit JM. The continuous berth allocation problem in a container terminal with multiple quays. Expert Syst Appl. 2015;42(21):7356–66.
24. Simrin A, Diabat A. The dynamic berth allocation problem: a linearized formulation. RAIRO-Oper Res. 2015;49(3):473–94.
25. Hu Z-H. Multi-objective genetic algorithm for berth allocation problem considering daytime preference. Comput Ind Eng. 2015;89:2–14.
26. Sabar NR, Chong SY, Kendall G. A hybrid differential evolution algorithm–game theory for the berth allocation problem. In: Proceedings of the 18th Asia Pacific Symposium on intelligent and evolutionary systems-volume 2, (Springer). 2015; p. 77–87.
27. Şahin C, Kuvvetli Y. Differential evolution based meta-heuristic algorithm for dynamic continuous berth allocation problem. Appl Math Model. 2016;40(23–24):10679–88.

28. Yang XS, Deb S. Cuckoo search via Lévy flights. In: World Congress on Nature & Biologically Inspired Computing (NaBIC), (IEEE), 2009; p. 210–14.
29. Sanajaoba S, Fernandez E. Maiden application of cuckoo search algorithm for optimal sizing of a remote hybrid renewable energy system. Renew Energy. 2016;96:1–10.
30. Holland JH. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. Cambridge: MIT Press; 1992.
31. Popov A. Genetic algorithms for optimization. User Manual, Hamburg. 2005; (**2013**).
32. Salhi A, Alsoufi G, Yang X. An evolutionary approach to a combined mixed integer programming model of seaside operations as arise in container ports. Ann Oper Res. 2019;272(1–2):69–98.