Computers & Operations Research 161 (2024) 106409

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor

An improved genetic algorithm for the berth scheduling with ship-to-ship transshipment operations integrated model

Marwa Al Samrout[a], Abdelkader Sbihi[b,*], Adnan Yassine[a,c]

[a] Université Le Havre Normandie, LMAH, FR CNRS 3335, 76600, Le Havre, France
[b] University of South-Eastern Norway, Research Group CISOM, 3616, Kongsberg, Norway
[c] Université Le Havre Normandie, Institut Supérieur d'Études Logistiques, 76600, Le Havre, France

ARTICLE INFO

Keywords:
Container terminal
DBAP
Transshipment
Scheduling
Integrated
Planning

ABSTRACT

In this paper, we consider a dynamic variant of Berth Allocation Problem with ship-to-ship transshipment in a container terminal, namely (DBAP_{STS}), where ships can arrive after the start of the planning plan. A novel mixed integer linear program (MILP) is developed to optimize the berthing schedule and build a transshipment connections planning between feeder and mother vessels. The aim is to reduce vessels dwell times in the terminal, and the penalty due to late vessels and decide the mode of transshipment needed. First, we develop a packing heuristic, and later we use an improved genetic algorithm based heuristic (GA) to solve the problem efficiently. We conducted a statistical analysis to identify relative importance and effective settings for parameters control of the GA, and we applied this algorithm with the control parameters settings determined to carry out the computational experiments on random generated instances. The proposed tailored method is able to solve the problem in an acceptable computing time for medium and large instances while a commercial solver was able to solve only small size instances.

1. Introduction

Containerization is among the most applied mode of transport for merchandise today. It consists of using shipping containers, which come in a variety of standard dimensions, to move products from one point to another. Also, the COVID-19 pandemic halted whole supply chains and affected heavily container shipping industries. Amid this outbreak, the shipping company has had to struggle with port closures, lack of workers, global shipping container shortages and many others difficulties. As of the end of March 2020, the impact of COVID-19 plunged to its lowest and according to Fig. 1, container freight rates increased significantly since this period. In the third quarter of 2021, major container shipping industries had an average benefit margin of over 56%, up from about 8.5% a year earlier. As a consequence, a particularly strong steep growth in global freight rates appeared in the year 2021, and reached a record price of nearly 10,400 U.S. dollars in September 2021. In April 2022, the global freight rate index stood at about 7800 U.S. dollars.[1]

We recall that for many reasons, containerized goods may need to be moved from one vessel to another during transit to their final destination. This action is called "transshipment" (see the paper by Vis and de Koster (2003)). There are two types of transshipment: direct transshipment and indirect transshipment. The first type, also called ship-to-ship transfer (STS) is where ships are anchored beside each other and goods are moved directly from the mother vessel to the feeder vessel. The second is where cargo is unloaded and transferred from the importing vessel to the yard storage space at the transshipment hub then the exporting vessel picks cargo to the next transshipment hub or to the final destination of the cargo.

In port terminals, there are several optimization problems that arise during the processing of port operations, and they mainly concern the berths allocation and the quay cranes assignments, which are the most important. Also, the transshipment operations are of big interest in ports operations, logistics and distribution. It cuts short the time for berthing and also it becomes very economical in (STS) case as ships do not have to berth at the jetty. In this way, thus it evades trade restrictions, saves money on shipping costs and avoids tariffs. Thus, cargo undergo transshipment for many reasons. One of the main reason is the absence or the loss of direct connection between importer and exporter. Also, transshipment is undertaken when the low tide comes making the intended port of destination inaccessible by mega next generation container ships.

Nowadays, many companies adopt the practice of transshipment to enhance product distribution efficiency. However, this approach comes

* Corresponding author.
E-mail addresses: marwa.al-samrout@etu.univ-lehavre.fr (M. Al Samrout), abdelkader.sbihi@usn.no (A. Sbihi), adnan.yassine@univ-lehavre.fr (A. Yassine).

[1] https://www.statista.com/statistics/1250636/global-container-freight-index/

https://doi.org/10.1016/j.cor.2023.106409
Received 5 February 2023; Received in revised form 30 August 2023; Accepted 31 August 2023
Available online 9 September 2023

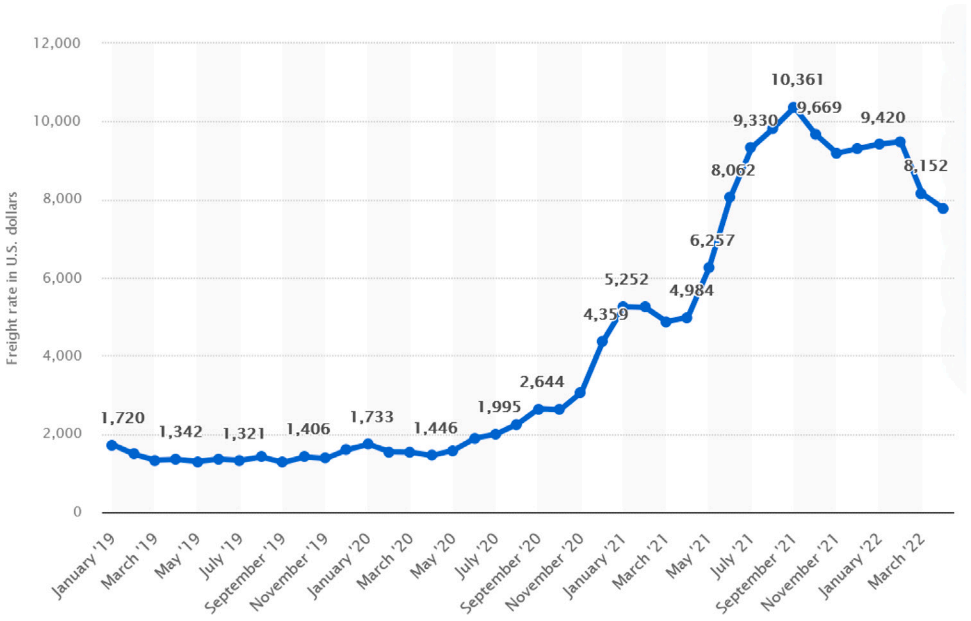

Fig. 1. International container shipping rate index from January 2019 to April 2022 (in U.S. dollars).
Source: Figure taken from https://www.statista.com/statistics/1250636/global-container-freight-index/.

with drawbacks such as increased environmental pollution and fire hazards due to potential leakage during operations in open sea. Additionally, transshipped containers are counted twice in port performance metrics, both during unloading from arriving vessels and loading onto departing vessels. Therefore, there is a significant need to develop an effective schedule plan to address the challenges associated with transshipment operations and berth allocation problems.

Although while there is a wealth of literature on the Berth Allocation Problem (BAP), it is worth noting that only few papers specifically address the concept of direct transshipment or "ship-to-ship" transshipment between vessels Nellen et al. (2022), Reda et al. (2016). In an early work by Petering and Murty (2009), the authors studied the transshipment operations and proposed an approach based simulation that determines the interaction between the terminal's long-run average quay crane amount with the length of the terminal's yard and the system that deploys yard cranes.

The literature classification by Nellen et al. (2022) indicates that there are only few research works on direct container transshipment at terminals. As evidenced by the inclusion of only 22 publications on the subject as for the BAP with direct transshipment specifically, Reda et al. (2016) and Bierwirth and Meisel (2015) indicated that only one paper addressed the seaside operational problem from a distinct perspective.

Our review collated many papers published on BAP incorporating transshipment planning (including a subset studying direct transshipment). Based on this review, we obtained the following findings:

– To the best of our knowledge, only five papers have addressed the BAP with direct transshipment. These papers include Liang et al. (2012), Lv et al. (2020), Lyu et al. (2022), Moorthy and Teo (2007) and Zeng et al. (2017).
– Unlike the five aforementioned studies, which primarily focus on minimizing operational costs, the paper by Liang et al. (2012) aims to minimize the combined handling time, waiting time of container ships at berths, and delay time of container ships.
– The research work that we develop has notable differences compared to Liang et al. (2012). Firstly, while Liang et al. (2012) focuses on a static variant of the berth allocation problem with predetermined assignments, our paper addresses the dynamic berth allocation problem where assignments can change over the time. Also, our approach uses a relative position formulation (RPF) that considers constraints on the relative positions of vessels, whereas in Liang et al. (2012), the authors used a position assignment formulation (PAF) with a binary decision variable $x_{ijk} = 1$ if ship i is serviced as the kth ship at berth j, and 0 otherwise. Moreover, Liang et al. (2012) focuses on the time and waiting involved in direct transshipment, whereas we impose constraints to ensure synchronization and simultaneous arrival for direct transshipment operations, clear and unambiguous allocation of transshipment operations, avoiding conflicts and multiple assignments (as specified in Constraints (10)–(12) of Section 3), optimizing efficiency and minimizing delays.
– Additionally, the paper by Lv et al. (2020) is the only study among the five mentioned studies that have addressed connections between mother and feeder vessels but focuses on recovery costs and disruptions, not dwell time. While Lv et al. (2020) studied post-disruption recovery strategies for BAP to set the priority and used the Squeaky Wheel Optimization (SWO) algorithm, we applied an improved genetic algorithm (GA) based on a new two-dimensional hierarchical first-fit (FF) rule to promptly allocate berths for large-scale problems.
– In the paper by Lyu et al. (2022), the authors developed a collaborative planning by examining the berth allocation recovery problem in transshipment terminals, where disruptions hinder the execution of the initial berthing plan.
– In Moorthy and Teo (2007), the authors studied the problem of allocating a home berth at a container transshipment terminal using a framework that models it as a rectangle packing problem on a cylinder. An efficient simulated annealing algorithm is applied considering data uncertainty and stochastic project scheduling techniques. Computational experiments demonstrate the effectiveness of the proposed model in constructing efficient templates for transshipment hub operations.
– In Zeng et al. (2017), the authors addressed simultaneously berth allocation, storage space assignment, and the direct transshipment plan. The objective is to minimize operational costs for trucks, yard cranes, and vessel delays. They compared approaches of direct transshipment between methods of transport at container terminal and via the application of the nearest neighbor heuristic based on GA. They concluded that the direct transshipment mode reduces considerably the operational costs for both terminal managers and carriers.

– Furthermore, we conducted a sensitivity analysis and a comparative study to determine parameters settings and evaluate different crossover operators, respectively, using statistical analysis.

Since this research work involves a novelty with this model, it is important to mention that the comparative studies in this paper are not based on the existing literature.

In the remainder, the paper is organized as follows. Section 2 reviews the related works in the literature. We give an overview of the existing integrated berth allocation and transshipment optimization problems with the different approaches and tools that were adopted. In Section 3, we describe the problem, detail it structure and give the mathematical formulation as a MILP subject to a set of constraints. We formulate the decisions about the mooring and the transshipment operations of container ships. Section 4 presents briefly the algorithms used in this paper. Section 5 detail the numerical experiments carried out to test the proposed approach. In Section 6, we present a sensitivity analysis of parameters to understand the criticality of certain parameters that regulate the feedback mechanism. Section 6.1 discusses the impact of the GA parameters. In Section 7, we test and compare three GA selection operators which are Roulette wheel Selection, Tournament Selection and Random Selection method. In Section 8, we discuss the computational experiments and the results. Finally, in Section 9, we give a conclusion that summarizes the study and propose some perspectives for future research.

2. Literature review

There have been numerous studies aiming to optimize the utilization of berthing space for vessels and the associated transshipment activities. Several notable works have addressed the integration of berth allocation, quay crane assignment, and scheduling problems in container terminals. Zheng et al. (2021) proposed a novel a MILP model based assignment for the BAP which can be seen as a liner carrier clustering problem. The authors employed a three-stage optimization based queuing, stability and clustering to solve the BAP. The paper by Cheimanoff et al. (2023) explored both exact and metaheuristic approaches to solve an integrated scheduling BAP and storage yard allocation problem by applying MILP formulation, branch-and-cut for small instances and a multi-start GRASP-ILS for large instances.

Lujan et al. (2021) introduced a fuzzy optimization model for the Berth Allocation Problem (BAP) and Quay Crane Allocation Problem (QCAP), handling imprecise arrival times and optimizing quay utilization. Xiang et al. (2023) focused on dynamic optimization in automated transshipment hubs, proposing a multi-objective model to maximize revenue, time and transportation savings, and quality of service. They developed an adaptive scheduling policy and demonstrated its superiority over benchmark policies. In another work, Xiang and Liu (2021) addressed the integrated problem of berth allocation and quay crane assignment, considering uncertainties in ship arrival delays and container quantity variations. They proposed a robust method using historical data and a decomposition approach to handle uncertainties effectively. Karam et al. (2020) introduced a planning model that integrated berth allocation, quay crane assignment, and internal truck assignment, aiming to achieve energy-saving goals. Their approach outperformed existing methods, generating reliable handling plans and achieving energy and cost savings. AbouKasm et al. (2020) proposed a mathematical formulation for integrating berth allocation, quay crane assignment, and scheduling, considering different operational policies and emphasizing the impact of choices policy on service times. Malekahmadi et al. (2020) developed an integer programming model for integrated berth allocation, quay crane assignment, and scheduling, considering water depth, tide conditions, and safety constraints. They applied the Ring Theory Particle Swarm Optimization based metaheuristic namely RTPSO to solve large instances efficiently. In the paper by Zhen et al. (2011), the authors incorporate at the tactical level both container storage space allocation problem (SSAP) comprising the assignment of quay cranes (QCs), and the berth allocation problem (BAP) in transshipment hubs. Later, Zhen et al. (2016) consider the terminal assignment (TA) problem, and the inter-modal terminal transport (ITT) with fuel consumption and storage cost economization. Lee et al. (2012) focus on reducing workload bottleneck in transshipment terminals by planning a template for feeder vessels.

Chargui et al. (2023) have proposed to optimize berth and quay crane allocation while considering the uncertainty associated with renewable energy availability. They employed a robust exact decomposition approach to tackle this complex problem while incorporating renewable energy uncertainty into the scheduling process. In Zhen et al. (2019), the authors focused on the holistic optimization of maritime logistics by considering several costs and penalties. They used three approaches to solve the problem efficiently. Similarly, another exact approach has been proposed by Correcher et al. (2019) for an integrated BAP and QCAP. The authors proposed a new MILP and solved the problem by employing valid inequalities and iterative and branch-and-cut procedures.

A memetic heuristic was also presented to handle large instances. In the paper by Tao and Lee (2015), the authors integrated the problems of berths and yard storage areas allocation to minimize the total distance of container movements between mother vessels and feeders during transshipment processes. In another work, Schepler et al. (2017) developed a MILP to model the container transport vehicles assignment to terminals, which includes trucks that are characterized by time windows for their pick-ups. The objective is to minimize turnaround times for long-haul vessels, coastal vessels, river barges and trains in a multi-modal terminal system. Also, several heuristics based on mathematical programming were proposed as well as a rolling horizon planning method for dynamic management that takes into account uncertainties. Numerical experiments were conducted with thousands of realistic instances. The results demonstrated that allowing cooperation between terminals significantly increases the system performance.

Despite the many published papers related to dynamic berth allocation problem Park and Kim (2002), Seyedalizadeh et al. (2010), Lee et al. (2010); studies that deal with this topic still remains under the transshipment circumstances. In Lv et al. (2020), the authors developed a decision support system for recovering berth allocation in transshipment terminals during disruptions. The authors used a (MILP) model to adjust the original berthing plan and minimize the cost of deviations. The study also considered quay crane assignment and incorporated transshipment connections between feeder and mother vessels. To solve large-scale problems, they introduced a Squeaky Wheel Optimization heuristic. Extensive computational experiments validated the effectiveness and efficiency of the method, emphasizing the practical benefits of incorporating transshipment considerations in rescheduling berth allocation.

The study developed in Lyu et al. (2022) examines the berth allocation recovery problem in transshipment terminals, where disruptions hinder the execution of the initial berthing plan. Terminal operators can mitigate the negative impact by collaborating with other terminals and optimizing transshipment connections. The authors proposed a MILP model to re-optimize the berth and quay crane allocation plan. They also used a Squeaky Wheel Optimization metaheuristic for large-scale instances. Extensive computational experiments demonstrate consistent cost reductions of up to 40% for terminal operators when implementing the collaborative strategy.

Also, an integrated model has been proposed by Zeng et al. (2017) to simultaneously address berth allocation, storage space assignment, and the direct transshipment plan. The objective is to minimize operational costs for trucks, yard cranes, and vessel delays. They compared approaches of direct transshipment between methods of transport at container terminal and via the application of the nearest neighbors heuristic based on GA. They concluded that the direct transshipment mode reduces considerably the operational costs for both terminal managers and carriers.

In the study by Moorthy and Teo (2007), the authors addressed the problem of allocating a home berth at a container transshipment terminal using a framework that models it as a rectangle packing problem on a cylinder. An efficient simulated annealing algorithm is applied considering data uncertainty and stochastic project scheduling techniques. Computational experiments demonstrate the effectiveness of the proposed model in constructing efficient templates for transshipment hub operations.

Liang et al. (2012) restricted the direct transshipment to ships that arrive at the same time. Nevertheless, additional berthing conditions were considered. To solve the transshipment transport problem, they developed a hybrid multi-stage operation-based genetic algorithm (GA) and the obtained results showed that direct transshipments can speed up the Mega Ship service process. Park and Kim (2002) presented a BAPC where the objective is to reduce the costs of late ship departures due to an inadequate service order and the additional complexity of handling containers. They assumed that the quay locations of ships and the factors estimated in the objective are dependent. Seyedalizadeh et al. (2010) developed another GA-based heuristic. The approach was able to obtain close-to-optimal solutions for small and large-sized instances. Lee et al. (2010) applied two GRASP procedures to solve a continuous BAP where vessels were allowed to berth anywhere along the quay.

Other approaches such as multi-objective were considered. Prayogo et al. (2021) integrated a bi-objective mathematical formulation of operational and tactical planning in container terminal. In Dkhil et al. (2021), the authors addressed a bi-objective optimization problem for automotive terminals which seeks to enhance handling times and the crossing of car flows simultaneously. The capability and locations of storage yards constraints, are not taken into account despite their importance. They provide a complexity analysis. The assumptions used in the numerical tests differ from those used to develop the integer programming formulations. To solve the problem, they used the Cplex solver. In Lv et al. (2020), the authors formulated the real-time rescheduling for the BACAP considering the transshipment links between feeder and mother vessels during the recovery process. A SWO-based construction heuristic was developed to manage the real-time disruptions due to uncertainties at the functional level.

Pang and Liu (2014) proposed an iterative decomposition heuristic method to solve different integrated berth allocation and transshipping routing problems. They considered the berthing time crash escaping between the vessels. They carried the computational simulation of the routine of a feeder service enterprise that operates around the Pearl River Delta region. Their results showed the important profit compared with classical approaches in which independent and sequential decisions are made. Zhen et al. (2016) studied the vessel allocation problem in transshipment hubs by considering arrival times and bunker consumption costs. A model is developed to minimize these costs while accounting for port resource limitations and quay crane assignments. Two solution methods are proposed for large-scale problems. Numerical experiments show the model's effectiveness, achieving 14% cost savings compared to the "First Come First Served" rule, with near-optimal results and a 0.1–0.7% relative gap.

For more details on the BAP and its variants, the reader may refer to the survey by Rodrigues and Agra (2022) that delved into the complexities arising from uncertain factors such as arrival times, processing times, and external disruptions and an overview of the main methodologies proposed that include and deterministic, robust optimization, stochastic programming, and fuzzy programming approaches.

3. Model and solving approach

In this section, we suggest a mixed integer linear programming formulation to manage docking and transshipment operations. Prior to that, we insert some terminology, that will help us to develop the formulation.

3.1. Problem description

Upon its arrival at a seaport, a container ship undergoes several operations. The huge amount of cargo piled up on its deck and which usually have to be discharged at different major hubs with strategic geographic locations, gave rise to the principle of transshipment. This principle consists in transferring a part of large ships containers to "feeder" vessels and in interchanging between transoceanic ships serving different lines, alternatives to the direct delivery of goods from port to port. Consequently, some of the mega-ship merchandise will not be discharged in the terminal storage space but will be transferred directly to one or more feeder ships. The planned mathematical model extends the BAP position formulation used in Ak (2008), with some additional constraints to deal with transshipment tasks.

The quay is partitioned into D equal-sized docks. A dock cannot receive more than one ship a time. Furthermore, since mega-ships can hold a large number of containers, the processing time of one ship may exceed one day. Therefore, the time axis is divided using relatively long segments of 3 or 4 h. The duration of treatment encompasses activities like unloading, loading, maintenance, and inspections, and plays a critical role in determining the overall time required for completing ship operations.

If we consider that we have N ships, regardless of whether they are feeder ships or mother ships, we can consider the features of a ship $i = 1, \ldots, N$ as follows:

- Arrival time of ship i.
- Due date of ship i.
- Delay penalty of ship i.
- Length of ship i calculated in number of needed berth sections.
- Processing time of ship i.
- Estimated leaving time of ship i.
- Transshipment ships pairs that involve the ship i.

Given the features of a ship i landing at the terminal and the list of transshipment ships connected to i, the aim is then to determine the berthing position POS_i of this vessel ($1 \leq POS_i \leq D-1$; where D is the set of berths), the berthing time TIM_i, and to choose the most suitable transshipment mode to apply between each pair of ships. We require the following assumptions:

- The ship must be served without disruption from its arrival at the terminal until its departure.
- Transshipment can be fulfilled exclusively by vessels of distinct types.
- Mother ships can exchange containers with a maximum of six feeder ships, while feeder ships can be assigned to a maximum of three mother ships.
- Incoming ships can dock at the quay without any physical restraints.
- The setup and waiting times of vessels are ignored.

Thus the feature of a ship i for berth scheduling model with ship-to-ship transshipment operations integration involves several components and considerations to ensure efficient and effective operations.

3.2. Sets

The sets used to build the model are:

$\mathcal{F}$: Set of feeder ships ($|\mathcal{F}| = N_1$ number of feeder ships).
$\mathcal{M}$: Set of mother ships ($|\mathcal{M}| = N_2$ number of mother ships).
$\mathcal{V}$: Set of all ships ($\mathcal{V} = \mathcal{F} \cup \mathcal{M}$ and $|\mathcal{V}| = N$ total number of ships $i = 1, \ldots, N$).

3.3. Parameters

We set the parameters featuring a ship i as follows :

ARR_i: arrival time of ship i.
DUE_i: due date of ship i (where $DUE_i \geq ARR_i + PRO_i$).
PEN_i: delay penalty of ship i.
LEN_i: length of ship i calculated in number of needed berth sections.
PRO_i: processing time of ship i.
DEP_i: the estimated leaving time of ship i.

3.4. Decision variables

We have identified both continuous and binary decision variables for the proposed model:

POS_i: scheduled berthing position of ship i.
TIM_i: berthing time of ship i.
EAR_i: earliest time that ship i can depart (its termination time $TIM_i + PRO_i$).

$$r_{ij} = \begin{cases} 1 & \text{if ship } j \text{ moors after ship } i \text{ departs} \\ 0 & \text{otherwise;} \end{cases}$$

$$u_{ij} = \begin{cases} 1 & \text{if ship } j \text{ moors completely above ship } i \\ & \quad \text{on the time-space diagram} \\ 0 & \text{otherwise;} \end{cases}$$

$$D_{ij} = \begin{cases} 1 & \text{if the transshipment process between} \\ & \quad \text{ship } i \text{ and ship } j \text{ exists and it is direct} \\ 0 & \text{otherwise;} \end{cases}$$

$$I_{ij} = \begin{cases} 1 & \text{if the transshipment process between} \\ & \quad \text{ship } i \text{ and ship } j \text{ exists and it is indirect} \\ 0 & \text{otherwise;} \end{cases}$$

3.5. Mathematical model

3.5.1. Constraints

$$r_{ij} + r_{ji} + u_{ij} + u_{ji} \geq 1 \qquad \forall i, j \in \mathcal{V} \qquad and \qquad i < j \tag{1}$$

$$r_{ij} + r_{ji} \leq 1 \qquad \forall i, j \in \mathcal{V} \qquad and \qquad i < j \tag{2}$$

$$u_{ij} + u_{ji} \leq 1 \qquad \forall i, j \in \mathcal{V} \qquad and \qquad i < j \tag{3}$$

$$TIM_j \geq EAR_i + (r_{ij} - 1).M \qquad \forall i, j \in \mathcal{V} \qquad and \qquad i \neq j \tag{4}$$

$$POS_j \geq POS_i + LEN_i + (u_{ij} - 1).M \qquad \forall i, j \in \mathcal{V} \qquad and \qquad i \neq j \tag{5}$$

$$TIM_i \geq ARR_i \qquad \forall i \in \mathcal{V} \tag{6}$$

$$EAR_i \geq TIM_i + PRO_i \qquad \forall i \in \mathcal{V} \tag{7}$$

$$POS_i \leq D - (LEN_i + 1) \qquad \forall i \in \mathcal{V} \tag{8}$$

$$POS_i \geq 1 \qquad \forall i \in \mathcal{V} \tag{9}$$

$$D_{ij} + I_{ij} = 1 \qquad \forall i \ \forall j \in \mathcal{V} \qquad and \quad i \neq j \tag{10}$$

$$| TIM_i - TIM_j | \geq I_{ij} \qquad \forall i \ \forall j \in \mathcal{V} \qquad and \quad i \neq j \tag{11}$$

$$| TIM_i - TIM_j | \leq I_{ij}.M \qquad \forall i \ \forall j \in \mathcal{V} \qquad \text{and} \quad i \neq j \tag{12}$$

Constraints (1)–(3) prevent ship rectangles from overlapping. Constraints (4) and (5) assure that the preferred docking times and positions are in compliance with the definitions of r_{ij} and u_{ij}, where M corresponds to a large positive scalar. Constraint (6) and (7) ensure that ships are not allowed to dock before their scheduled arrival time and cannot leave before their end of service time. Constraints (8) and (9) ensure that all ships fit on the dock. Constraint (10) guarantees that each pair of vessels can perform one and only one type of transshipment operations either direct or indirect transshipment. Constraints (11) and (12) are the new added constraints which aim at preventing ships arriving at different times from performing the direct transshipment mode.

3.5.2. Objective function

To solve the problem, we set the objective of simultaneously minimizing the amount of time that container vessels stay in the marine terminal and the total penalty imposed on tardy vessels. We call the problem berth scheduling with ship-to-ship transshipment and we note it DBAP_{STS}. Formally, DBAP_{STS} can be written as follows:

$$(\text{DBAP}_{\text{STS}}) \begin{cases} min \sum_{i \in \mathcal{V}} (EAR_i - ARR_i) + \sum_{i \in \mathcal{V}} PEN_i.(EAR_i - DUE_i)^+ \\ \text{Subject to:} \\ (1) - (12) \\ r_{ij}, \ u_{ij}, \ D_{ij}, \ I_{ij} \in \{0, 1\} \ \forall i \ \forall j \in \mathcal{V} \text{ and } i \neq j \end{cases}$$

3.6. Complexity study of the problem

This model is an extension of the BAP position formulation proposed by Ak (2008). We have added two new constraints to take into account the transshipment operations. In Ak (2008), the BAP problem has been proven to be NP-hard, then it is clear that the DBAP_{STS} is also NP-hard as one of the variant of the BAP. Also, the problem can be easily seen as a dynamic scheduling problem on parallel machines Ak (2008), while for one problem instance, one can assume that the set of B quays are simply a set of B parallel machines and assigned for each ship i, there are $p_i \times h_i$ tasks to process and each with a corresponding unit length and a unit processing time, hence the complexity of the problem.

4. Genetic algorithm

Designed for the purpose of optimization, the genetic algorithm (GA) is a generalized computationally version of Fisher's evolutionary formulation Holl (1995). The most important mechanisms that are at the basis of evolution in this algorithm are as follows: First of all, the conceivable solutions of the optimization scheme are delineated in the form of a genome. Next, a population of probable candidate solutions is generated. Then, the most efficient individuals are chosen with respect to the task to be optimized. A new population is then created by copying long sequences of the selected solutions' genetic code. Finally, variation operators are applied to the genomes of the individuals of the new population, in order to create different solutions. The child population in turn becomes the parent population, and the same procedure is iterated until a satisfactory solution is found.

Generally, when treating scheduling problems, the most targeted objective of GA approaches is to minimize as much as possible the makespan (or Cmax, the time at which the last product is manufactured) for a m-stages system with parallel machines, setup times and machines eligibility Ruiz and Maroto (2006), Sbihi and Chemangui (2018). In another study, Simrin and Diabat (2015) have proposed a GA for the dynamic BAP (DBAP). They formulated the problem as a

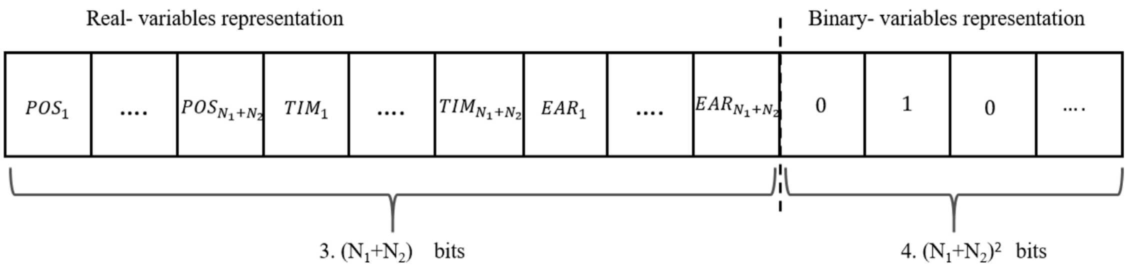

Fig. 2. Representation of an individual's chromosome encoding (genotype).

non-linear mixed integer program, then they linearized the problem as an equivalent mixed integer program (MIP).

Also, for scheduling problems, most of the GA approaches usually separate assignment and sequencing Rajendran and Chaudhuri (1992), Sherali et al. (1990), Sbihi and Chemangui (2018).

Since the DBAP formulation used here has been proven to be NP-hard in Ak (2008), it becomes clearer that the proposed problem is NP-hard since the DBAP$_{STS}$ model can be addressed as a dynamic scheduling problem on parallel machines Ak (2008). Therefore, it is interesting to tailor a GA based algorithm to solve the problem that we detail in the next sections. In the literature, various scheduling problems such as flowshop, jobshop, and related ones have been successfully solved using tailored GA approaches. The scheduling aspect of our problem aligns well with the characteristics of a metaheuristic approach like GA. Furthermore, GA offers a more manageable solution structure and representation for handling a large number of constraints and variable types compared to alternative methods GAs are effective in solving resource allocation challenges, including berth assignment for terminal resources. Goel and Tiwari (2022) compared optimization algorithms for Resource Allocation Problems and found that GAs outperformed other methods. GAs' success can be attributed to their ability to explore diverse solutions, handle large search spaces, and adapt to changing constraints, making them a favorable choice for Resource Allocation Optimization.

The GA is written in MATLAB 2018b. The overall scheme is based on an implementation by Yarpiz.[2] The original code is modified and adjusted to the scheduling problem considered. The process continues iteratively generation after generation, and it is terminated when the limit on the number of iterations is reached.

4.1. Encoding chromosomes representation

The population is represented as a two dimensional array, possessing two main qualities: the position of the solutions in the search space and its costs. Each member (row) of the population consists of a set of N genes ($N = N_1 + N_2$) denoting the place of each ship on a time-space diagram besides its transshipment links. On the side of binary variables (r_{ij}, u_{ij}, D_{ij} and I_{ij}) where the genotype consists of a bit strings of $4 \times (N_1 + N_2)^2$, a binary encoding is employed while on the other side, the genes of continuous variables (namely POS_i, TIM_i and EAR_i), are defined through a real valued representation of $3\times(N_1+N_2)$ components.

Fig. 2 illustrates an example of the proposed coding. Such a portrayal brings certain facilities since the genotype can be dilapidated into diverse recognizable portions permitting its exploitation by the reproduction operators.

4.2. Initial solution

When dealing with genetic algorithms, a premature convergence can occur due to the fact that, after a few runs, similar traits arise independently in different individuals and lead to a less diversified population. Thus, the diversity is critical for a population to adapt to changing environments. Previous research works have shown that the initialization of the entire population with the mean of a heuristic, leads to a low diversity since it affects the initial fitness of the population while random solutions can lead to a high level of population diversity. However, reaching randomly a feasible solution subject to several constraints is not always a simple task. Therefore, to improve the performance of the used GA, we seed the population with some initial packed solutions and then, we fill up the rest with random solutions.

To do that, a new heuristic algorithm based on the two-dimensional hierarchical first-fit (FF) rule is applied, where the vessels rectangles are arranged on the time space diagram one by one at the earliest possible berthing time with h_k available contiguous berth sections, and at the lowest index berth section at that time Ak (2008).

The problem data is encoded as an ordered list L of vessels, representing a feasible solution to the problem. L is of size N. Each vessel is represented by its characteristics in a specific format: (i) the length of vessel, (ii) the processing time, (iii) the arrival time, (iv) the berthing position and, (v) the berthing time. The procedure of this algorithm is detailed in Algorithm 1. The packing heuristic algorithm is simulated using Python3.8 and the IDE Pycharm.

Initially, the solution is sorted by ascending order of vessel arrival times. Then, a feasibility checking mechanism is applied. For each produced individual, we examine the objective function such that:

- To prevent the collision status between vessels rectangles.
- To respect the condition of dynamic arrivals of vessels and the time needed to serve it.
- To satisfy the time-space diagram layout constraints.
- To repair the solution infeasibility.

More solutions are generated based on the initial one by applying randomness to the berthing positions and the berthing times. If the new obtained solution list is not valid, then the repairing step is performed.

In Algorithm 1, the list of vessel is called (solution). Initially, the berthing position of each vessel was attribute to 0 while the berthing time was attribute to its arrival time and the list (solution) was sorted by increasing arrival time. Afterwards, the algorithm detects the collision between vessel i and all its previous vessels. For instance, if vessel i is the 5th vessel, then the algorithm check the collisions of i with 1st, 2nd, 3rd and 4th vessels. Next, it increases the vessels length one unit by one unit of length and, then the berthing time attribute one unit by one unit of time until there is no possible collision. In this case, we consider that the vessel fits at the berth section at that time period and we start to create a population with a copy of the initial solution. In the main steps, the for loop, applies randomness to the candidate solution. It randomly assigns a value to the docking position and docking time.

Then, we increase the berth section. If the last berth section is reached, the algorithm restarts at berth section 0, but in the next time period. Finally, the code generates more solutions based on the initial one but using random berthing position.

[2] https://yarpiz.com/23/ypea101-genetic-algorithms

Algorithm 1 Heuristic algorithm based on the two-dimensional hierarchical first-fit (FF) rule

```
for $i$ in range(len(solution)) do
    $v$ ← solution[$i$]
    while True do
        col ← False
                                ▷ Check the collision between vessel $i$ and all its previous vessels.
        for $j$ in range($i$) do
            $w$ ← solution[$j$]
            if collision is detected between vessels then
                col ← True
            end if
                                ▷ No collision, the vessel fits at positions $b_i$ and $t_i$.
            if col is False then
                break
            else if
                then    ▷ Increase the berth section. If reached the last berth section, then restart at berth section 0, but in the next $t$ period.
            end if
        end for
    end while
    Main steps:
    while len(population) ≤ PopSize do
        candidate_solution ← copy_solution(initial_solution)
                                ▷ A possible candidate solution
        for $i$ in range(3) do
                                ▷ Apply randomness to the candidate solution
            index ← random.randint(0, len(candidate_solution) - 1)
            vessel_length ← candidate_solution[index][0]
            vessel_arrival_time ← candidate_solution[index][2]
            vessel_processing_time ← candidate_solution[index][1]
                                ▷ Randomly assign a value to the berthing position
            candidate_solution[index][2] ← random.randint(vessel_arrival_time, TIME_PERIODS - vessel_processing_time - 1)
                                ▷ Randomly assign a value to the berthing time
            candidate_solution[index][3] ← random.randint(0, BERTH_SIZE - vessel_length - 1)
            repair_infeasibility(candidate_solution)
        end for
        if candidate_solution not in population then
                                ▷ Add the new candidate solution to the population
            population.append(candidate_solution)
        end if
    end while
end for
print_solution(solution)
```

4.3. Constraints handling technique and fitness evaluation

We use a simple mechanism based on pushing the algorithm towards the feasible search space to adequately handle the constraints over the generations and to enforce the constraint set and improve the performance of the GA. In the following, we describe the technique of constraint handling used.

Penalty function

Penalty-function methods are part of the sequential unconstrained minimization technique (SUMT) which was initially proposed by Carroll (1961) and later developed by Fiacco and McCormick (1964a,b), Pomentale (1965), Stong (1965) and Fiacco and McCormick (1966). This type of approaches convert constrained optimization problems into unconstrained problems by adding penalty terms to the objective functions. Generally, the penalty term is resolved from the degree of constraint violation of the solution. It can be expressed by:

$$p(x) = \sum_{i=1}^{I} \eta_i \times max(0, g_i(x))^2 + \sum_{j=1}^{E} \rho_j.|h_j(x)|^2 \tag{13}$$

where $p(x)$ is the penalty term, $g_i(x)$ and $h_j(x)$ are the inequality and equality constraints functions respectively, η_i and ρ_j are positive constants called "penalty factors". I and E are the numbers of inequality and equality restraints respectively. The aim of Eq. (13), is to diminish

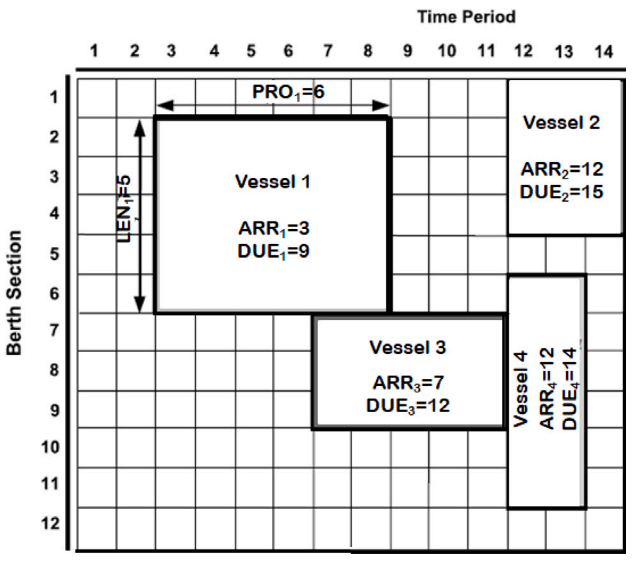

Fig. 3. A space–time grid of the proposed DBAP$_{STS}$ model.
Source: The figure is taken from Ak (2008) and readjusted.

the fitness of non feasible solutions, in favor of the selection of feasible ones El Samrout (2018). The measure of violation is null when the constraints are not violated and is nonzero in the region where constraints are violated. The penalty factors of this equation require careful tuning to determine the severity of the penalty to be utilized. In this implementation, the penalty is dynamic which means that the coefficients change with the iteration number.

Since this study seeks to minimize the makespan Cmax which is defined as vessels residence times in the terminal and the total penalty for tardy vessels, the lower time spent at the terminal, the higher the fitness of an individual selection. Then, solutions will be ranked-ordered based on the dwell time obtained from objective function. A chromosome is said to be healthier if it has the lowest Cmax value. Consequently, the fitness of a chromosome F is identical to the objective function value of the solution it corresponds to, plus the penalty function value resulting from the constraints violation by adopting this solution:

$$\begin{aligned} F &= min(\sum_{i\in\mathcal{V}}(EAR_i - ARR_i) + \sum_{i\in\mathcal{V}} PEN_i.(EAR_i - DUE_i)^+ + p(x)) \\ &= min\ (C_{max} + p(x)) \end{aligned} \tag{14}$$

4.4. GA operators

4.4.1. Parent selection method

Basically, selecting good parents lead to fitter *offsprings*, and eliminate non-elite solutions. This makes GA depends greatly on the parents selection phase. In this implementation, three types of selection techniques are included for comparative trials, namely: (i) roulette wheel, (ii) tournament, and (iii) random selection. Once the fitness values have been set from the operational evaluation, parents from the current generation are put as candidates for the next generation.

(i) The roulette wheel selection: maintains the fitness of a population while including some diverse parents. This method uses fitness to calculate a probability of selection for each individual. Assume that F_{worst} is the highest error attained, F_i the fitness of the individual i, and pre is the selection pressure. The probability to select an individual i is given by[3]:

$$probability_i = \frac{exp(-pre.F_i)}{F_{worst}}$$

(ii) The tournament selection: In the current implementation, the tournament selection method returns m candidates from the overall population for the tournament and the top two candidates with the best fitness are chosen as the parents for the next generation.

(iii) The random selection: Select parents randomly from the population based on some probability distribution.

Each selection method presented in one of the above three types that was applied to solve the problem. We evaluated the performance of each selection method and have chosen the ones that obtained the best results. We give in the next, the details of the analytical results of this comparison.

4.4.2. Crossover operators

The main basic method to approach a feasible solution of DBAP$_{\text{STS}}$ is to depict this solution as a berth scheduling using a time-space diagram where the horizontal axis represents time and the vertical axis represents berth sections Ak (2008) (Fig. 3). To model vessels, we use rectangular shapes since they provide easier and more effective collision detecting algorithms. It is worth noting that rectangles cannot fully surround vessels as they are only approximations of the vessels. Their lengths symbolize the time needed to process the vessels and their heights represent the lengths of the vessels.

Once the feasible solution representation is explained, the next step is to apply the GA crossover operator and explain how the information contained in the parents' chromosomes is combined and mixed to produce the children's chromosomes during the mating step.

For the binary coded part of the selected parents, three different conventional crossover operators are examined:

(i) Single-Point Crossover: a cutting site is randomly chosen between the two parents. This leads us to take a number m (site index). We then obtain two children by taking the first m genes of the second parent and the (G-m) last genes of the first parent where G is the number of genes in an individual.

(ii) Double-Point Crossover: two cutting sites are randomly introduced between the two parents, and the segments between these two positions are swapped.

(iii) Uniform Crossover: this technique is completely different from the previous two techniques. Its operation is illustrated in Fig. 4. As before, two exchange positions are randomly chosen and then an arithmetic weighted average is performed by a coefficient $\alpha \in \{0, 1\}$ as follows:

$$x'_{1i} = \alpha_i.x_{1i} + (1-\alpha_i).x_{2i}$$

$$x'_{2i} = \alpha_i.x_{2i} + (1-\alpha_i).x_{1i}$$

where $x_1 = (x_{11}, x_{12}, \dots, x_{1n})$ and $x_2 = (x_{21}, x_{22}, \dots, x_{2n})$ are two parents and $x'_1 = (x_{11}, x_{12}, \dots, x'_{1i}, \dots, x_{1n})$ and $x'_2 = (x_{21}, x_{22}, \dots, x'_{2i}, \dots, x_{2n})$ are their *descendants*.

For the continuous chromosomes, the next crossing technique is applied: Suppose $x_1 = (x_{11}, x_{12}, \dots, x_{1n})$ and $x_2 = (x_{21}, x_{22}, \dots, x_{2n})$ are the two parents and suppose that $x'_1 = (x_{11}, x_{12}, \dots, x'_{1i}, \dots, x_{1n})$ and $x'_2 = (x_{21}, x_{22}, \dots, x'_{2i}, \dots, x_{2n})$ are the two *childs* obtained from this action. is described by the following formulas:

$$x'_{1i} = \alpha_i.x_{1i} + (1-\alpha_i).x_{2i}$$

$$x'_{2i} = \alpha_i.x_{2i} + (1-\alpha_i).x_{1i}$$

where $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$ and $\alpha_i \in [0, 1]$. Later, these different techniques are statistically evaluated in order to determine the most efficient one in terms of the time spent to find an acceptable solution, as well as the quality of the solution obtained.

4.4.3. Mutation operators

The mutations help to prevent the genetic algorithm from converging too prematurely on a solution. To avoid the population from falling in solutions in a local optimum, we resort a mutation strategy for each kind of variables:

- For binary variables, let $x = (x_1, x_2, \dots, x_n)$ be the affected chromosome. The mutated solution can be obtained as Alp et al. (2003):

$$\begin{cases} x_i & \text{if } i \neq j \\ 1 - x_i & \text{if } i = j \end{cases}$$

- For continuous variables:
let $x = (x_1, \dots, x_n)$ be the concerned chromosome and $j \in \{1, \dots, n\}$ an arbitrary integer index. The used mutation operator choose randomly a gene x_i from x and add to it an aleatory noise adapted from normal distribution $\mathcal{N}(m,\ \sigma^2)$. In other words, this works by appending some Δx_i to x_i, where the Δx_i values are identified using the given $\mathcal{N}(0,\ \sigma^2)$ distribution as follows:

$$x'_i = x_i + \mathcal{N}(0, \sigma^2) \tag{15}$$

[3] https://yarpiz.com/23/ypea101-genetic-algorithms

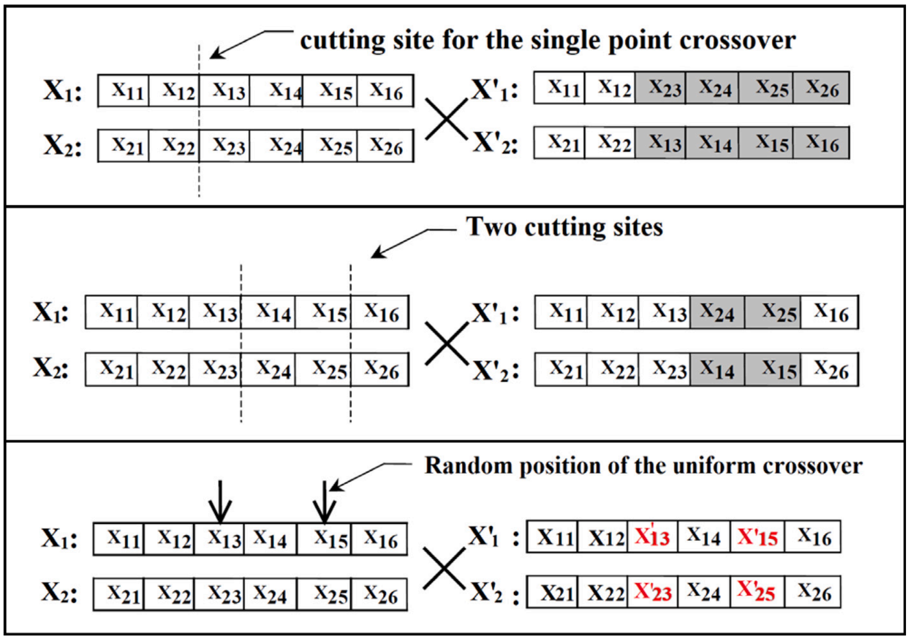
cutting site for the single point crossover
X1: X11 X12 X13 X14 X15 X16
X2: X21 X22 X23 X24 X25 X26
X'1: X11 X12 X23 X24 X25 X26
X'2: X21 X22 X13 X14 X15 X16
Two cutting sites
X1: X11 X12 X13 X14 X15 X16
X2: X21 X22 X23 X24 X25 X26
X'1: X11 X12 X13 X24 X25 X16
X'2: X21 X22 X23 X14 X15 X26
Random position of the uniform crossover
X1: X11 X12 X13 X14 X15 X16
X2: X21 X22 X23 X24 X25 X26
X'1: X11 X12 X'13 X14 X'15 X16
X'2: X21 X22 X'23 X24 X'25 X26

Fig. 4. A general view of the single point crossover, the double point crossover and the uniform crossover process is shown from top to bottom (example of an individual with six genes).

The average of Δx_i is null ($m = 0$) and the standard deviation is σ^2 which stands for the mutation step size.

5. Experimental tests

In this section, we detail the numerical experiments carried out to test the proposed approach. In the following, we present all performed experiments as well as a detailed discussion of the obtained solutions. The experiments were conducted on a PC laptop `intel CORE i5-4570 CPU`, with `3.20 GHz` and `RAM 4 GB`. For the model presented in Section 3, we have developed and tested the GA based heuristic on a set of DBAP$_{\mathrm{STS}}$ instances randomly generated to simulate real terminal containers berth operations.

The mathematical model we present provides a better understanding of the structure of the problem and its complexity. However, to our knowledge, the model presented here has not been studied to date. Thus, we have not succeed to get access to real instance problems or similar one form the literature. Consequently, no existing results are available in the literature for comparison with our numerical results. We have based the design of our benchmark on the work by Ak (2008) and other large instances were randomly generated with the same industrial context.

The population size `PopSize` defines the number of chromosomes that shaped the population. If we have population sizes too small, the algorithm converges more quickly to a local optimum. In the case of a large population size, the computation time becomes penalizing. The number of populations generated represents the number of evolution steps to be applied, and a small number of generations generally leads to a premature optimization. In addition, a large number of generations takes too much time and no possible improvements can be made after a certain number of generations. Thus, in the following, we will detail our strategy for a best trade-off parameters settings.

5.1. Instances generation

We recall that the studied problem is a new variant of BAP, and that there exist no instances data set available in the literature. To test the proposed model, we have randomly generated 3 sets of realistic problem instances Table 2, and have set to 200 the number of generations for each tested instance.

Table 1
The intervals of parameters employed in the GA.

Parameters	Intervals
ARR_i	[1,10] for small instances
	[1,20] for large instances
DUE_i	$ARR_i + K.PRO_i$ where $K \in \{1,3\}$
PEN_i	[3,5]
LEN_i	[2,6]
PRO_i	[1,4] if $LEN_i = 2$
	[1,5] if $LEN_i \in \{3,4\}$
	[1,6] if $LEN_i \in \{5,6\}$

The first two sets consist of nine small and medium size instances noted Inst 1 to Inst 9 where the number of berths $D = 12$, and the number of vessels $N = N_1 + N_2 \in \{3; 4; 7; 9; 10; 12; 13; 15; 16\}$. The third set consists of six large size instances, noted Inst 10 to Inst 15 where $D = 20$ and $N \in \{19; 41; 43; 44\}$.

The parameter intervals of Ak (2008) (Table 1) are used to run the numerical tests (Table 2).

5.2. Experimental environment

The mixed integer programming model developed is solved optimally for different sizes by Cplex V. 12.7.0, integrated to the Java programming language using Netbeans IDE 8.2. This enables us to endorse the mathematical formulation empirically. Later, the obtained results were compared to those of the GA employed to validate the effectiveness of this GA approach. Extensive experiments have been conducted for large-size problems. Due to the extreme complexity of the formulated problem, Cplex was unable to obtain the exact solutions within a reasonable amount of time when considering about 30 vessels. Therefore, the solver was run using default settings and with a time limit of 3600 s for each large instance and results were compared to the ones of GA.

Table 2
Detailed descriptions of the ship's features.

Instance	N_1, N_2, B	Parameters values for a ship i	transshipment ships pairs
Inst 1	(6,3,20)	$ARR_i = \{5, 5, 17, 17, 1, 1, 20, 20, 9\}$ $LEN_i = \{5, 3, 2, 6, 3, 4, 4, 3, 5\}$ $PRO_i = \{1, 1, 3, 1, 5, 4, 4, 5, 5\}$ $DUE_i = \{6, 6, 20, 18, 16, 24, 24, 14, 12\}$ $PEN_i = \{3, 4, 5, 4, 4, 4, 3, 4, 5\}$	[(6,8)]
Inst 2	(2,1,20)	$ARR_i = \{3, 17, 7\}$ $LEN_i = \{3, 2, 3\}$ $PRO_i = \{5, 4, 5\}$ $DUE_i = \{13, 25, 17\}$ $PEN_i = \{4, 5, 4\}$	[(2,3)]
Inst 3	(8,4,20)	$ARR_i = \{3, 4, 9, 1, 18, 4, 6, 3, 13, 18, 1, 10\}$ $LEN_i = \{3, 5, 4, 3, 6, 6, 4, 4, 2, 6, 6, 5\}$ $PRO_i = \{1, 3, 2, 2, 2, 6, 4, 1, 2, 5, 2, 5\}$ $DUE_i = \{6, 13, 15, 7, 24, 22, 18, 6, 19, 33, 7, 25\}$ $PEN_i = \{3, 4, 5, 5, 4, 3, 4, 3, 4, 4, 4, 4\}$	[(2,8)(2,7)(3,8) (3,9)(4,11)]
Inst 4	(10,5,20)	$ARR_i = \{11, 12, 3, 1, 19, 4, 7, 17, 7, 14, 12, 15, 15, 2, 4\}$ $LEN_i = \{3, 3, 6, 4, 3, 4, 4, 4, 5, 3, 4, 6, 3, 3, 6\}$ $PRO_i = \{2, 4, 5, 3, 1, 4, 3, 3, 3, 3, 1, 5, 5, 4, 6\}$ $DUE_i = \{15, 20, 13, 7, 21, 12, 13, 23, 13, 20, 14, 25, 25, 10, 16\}$ $PEN_i = \{3, 5, 5, 4, 5, 3, 4, 5, 4, 3, 3, 3, 4, 5, 3\}$	[(5,12)(6,10)(6,12)]
Inst 5	(7,3,12)	$ARR_i = \{1, 3, 7, 7, 9, 13, 18, 18, 22, 24\}$ $LEN_i = \{5, 5, 2, 3, 2, 5, 2, 3, 5, 2\}$ $PRO_i = \{2, 4, 2, 4, 4, 5, 4, 3, 2, 2\}$ $DUE_i = \{3, 7, 9, 11, 13, 18, 22, 21, 24, 26\}$ $PEN_i = \{3, 3, 3, 3, 3, 3, 3, 3, 3, 3\}$	[(1,3)(1,4)(2,4) (2,6)(2,7)(2,8) (3,5)(3,9)(3,10) (6,8)(4,8)(4,10) (7,10)]
Inst 6	(6,1,12)	$ARR_i = \{2, 4, 8, 8, 10, 14, 19\}$ $LEN_i = \{5, 5, 2, 3, 2, 5, 2\}$ $PRO_i = \{3, 5, 3, 5, 5, 6, 5\}$ $DUE_i = \{5, 9, 11, 13, 15, 20, 24\}$ $PEN_i = \{3, 4, 4, 4, 4, 5, 5\}$	[(1,3)(1,4)(2,4) (2,6)(2,7)(3,5)]
Inst 7	(3,1,12)	$ARR_i = \{1, 3, 7, 7\}$ $LEN_i = \{5, 5, 2, 3\}$ $PRO_i = \{2, 4, 2, 4\}$ $DUE_i = \{3, 7, 9, 11\}$ $PEN_i = \{3, 3, 3, 3\}$	[(1,2)(1,3)(2,3)(3,4)]
Inst 8	(9,4,12)	$ARR_i = \{3, 5, 9, 9, 11, 15, 20, 20, 24, 26, 27, 30, 31\}$ $LEN_i = \{5, 5, 2, 3, 2, 5, 2, 3, 5, 2, 5, 5, 2\}$ $PRO_i = \{4, 6, 4, 6, 6, 7, 6, 5, 4, 4, 4, 6, 4\}$ $DUE_i = \{7, 11, 13, 15, 17, 22, 26, 25, 28, 30, 31, 36, 35\}$ $PEN_i = \{4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5\}$	[(5,12)(6,10)(6,12)]
Inst 9	(12,4,12)	$ARR_i = \{1, 3, 7, 7, 9, 13, 18, 18, 22, 24, 25, 28, 29, 30, 31, 32\}$ $LEN_i = \{5, 5, 2, 3, 2, 5, 2, 3, 5, 2, 5, 5, 2, 5, 5, 2\}$ $PRO_i = \{2, 4, 2, 4, 4, 5, 4, 3, 2, 2, 2, 4, 2, 2, 4, 2\}$ $DUE_i = \{3, 7, 9, 11, 13, 18, 22, 21, 24, 26, 27, 32, 31, 32, 35, 34\}$ $PEN_i = \{3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3\}$	[(5,12)(6,10)(6,12)]
Inst 10	(13,6,12)	$ARR_i = \{2, 4, 8, 8, 10, 14, 19, 19, 23, 25, 26, 29, 31, 33, 35, 37, 38, 40, 41\}$ $LEN_i = \{5, 5, 2, 3, 2, 5, 2, 3, 5, 2, 5, 5, 2, 5, 5, 2, 2, 2, 2\}$ $PRO_i = \{2, 4, 2, 4, 4, 5, 4, 3, 2, 2, 2, 4, 2, 2, 4, 2, 2, 2, 2\}$ $DUE_i = \{4, 8, 10, 12, 14, 19, 23, 22, 25, 27, 28, 33, 33, 35, 39, 39, 40, 42, 43\}$ $PEN_i = \{3, 3, 3, 3, 3, 4, 4, 4, 4, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4\}$	[(5,12)(6,10)(6,12)]
Inst 11	(20,21,9)	$ARR_i =$ $\{11, 12, 3, 1, 19, 4, 7, 17, 7, 14, 12, 15, 15, 2, 4, 12, 17, 10, 10, 16, 12, 4, 15, 2, 8, 4, 10, 18, 16, 1\}$ $LEN_i =$ $\{3, 3, 6, 4, 3, 4, 4, 4, 5, 3, 4, 6, 3, 3, 6, 4, 6, 3, 3, 3, 5, 2, 3, 5, 4, 5, 5, 3, 2, 3\}$ $PRO_i =$ $\{2, 4, 5, 3, 1, 4, 3, 3, 3, 3, 1, 5, 5, 4, 6, 3, 4, 2, 4, 3, 4, 1, 4, 1, 2, 3, 1, 2, 1, 5\}$ $DUE_i =$ $\{15, 20, 13, 7, 21, 12, 13, 23, 13, 20, 14, 25, 25, 10, 16, 18, 25, 14, 18, 22, 20, 6, 23, 4, 12, 10, 12, 22, 18, 11\}$ $PEN_i =$ $\{3, 5, 5, 4, 5, 3, 4, 5, 4, 3, 3, 3, 4, 5, 3, 3, 5, 4, 5, 5, 3, 5, 4, 5, 4, 5, 4, 5, 5, 3\}$	(5,11)(6,9)(6,11)

(*continued on next page*)

6. GA parameters settings

One of the main challenges through parameter setting is to compare parameters sets of the evolutionary algorithm and examine whether that some of them give consistently better results than others. Obviously, there are many parameters which affect GAs performance and need to be tuned to balance between the convergence reliability and the convergence velocity. The main control parameters are: the population size `PopSize`, the selection probability, the crossover probability χ, mutation probability μ and the mutation rate $\mathtt{R}$ (see Table 2).

In this section, a statistical analysis is carried out to determine the most relevant configurations of factors used in the GA. For each of the

Table 2 (*continued*).

Inst 12	(20,21,9)	$ARR_i = \{13, 11, 15, 19, 17, 10, 4, 10, 4, 14, 9, 17, 6, 11, 14, 1, 2, 15, 1, 17, 9, 3, 15, 9, 6, 15, 18, 18, 14, 12\}$ $LEN_i = \{4, 5, 2, 6, 2, 2, 6, 2, 2, 4, 6, 3, 2, 5, 5, 2, 6, 3, 5, 3, 5, 3, 2, 2, 2, 3, 2, 3, 5, 5\}$ $PRO_i = \{4, 1, 2, 5, 4, 4, 3, 4, 3, 1, 5, 5, 2, 3, 1, 1, 2, 3, 5, 4, 6, 3, 2, 3, 1, 1, 4, 1, 6, 3\}$ $DUE_i = \{21, 13, 19, 29, 25, 18, 10, 18, 10, 16, 19, 27, 10, 17, 16, 3, 6, 21, 11, 25, 21, 9, 19, 15, 8, 17, 26, 20, 26, 18\}$ $PEN_i = \{4, 3, 3, 5, 3, 4, 5, 5, 5, 4, 3, 5, 4, 5, 3, 5, 4, 5, 5, 4, 5, 5, 3, 3, 4, 4, 3, 4, 5, 5\}$	(5,11)(6,9)(6,11)
Inst 13	(20,21,9)	$ARR_i = \{3, 6, 15, 12, 1, 15, 8, 13, 16, 7, 2, 5, 3, 12, 2, 13, 11, 16, 13, 15, 17, 3, 2, 3, 13, 19, 10, 7, 1, 5\}$ $LEN_i = \{2, 3, 4, 4, 4, 5, 3, 3, 3, 5, 3, 3, 5, 4, 4, 6, 2, 6, 2, 2, 3, 2, 3, 5, 2, 4, 2, 5, 2, 2\}$ $PRO_i = \{1, 3, 1, 2, 1, 1, 5, 2, 3, 6, 2, 4, 2, 4, 3, 2, 3, 3, 2, 4, 1, 3, 4, 3, 2, 1, 1, 1, 3, 1\}$ $DUE_i = \{4, 9, 16, 14, 2, 16, 13, 15, 19, 13, 4, 9, 5, 16, 5, 15, 14, 19, 15, 19, 18, 6, 6, 6, 15, 20, 11, 8, 4, 6\}$ $PEN_i = \{3, 3, 5, 3, 5, 5, 3, 4, 3, 4, 3, 4, 5, 3, 3, 5, 3, 5, 3, 5, 3, 5, 5, 3, 3, 5, 3, 5, 4, 4\}$	(5,11)(6,9)(6,11)
Inst 14	(24,17,8)	$ARR_i = \{2, 4, 8, 8, 10, 14, 19, 19, 23, 25, 26, 29, 31, 33, 35, 37, 38, 40, 41, 42, 44, 48, 48, 50, 54\}$ $LEN_i = \{5, 5, 2, 3, 2, 5, 2, 3, 5, 2, 5, 5, 2, 5, 5, 2, 2, 2, 2, 5, 5, 2, 3, 2, 5\}$ $PRO_i = \{2, 4, 2, 4, 4, 5, 4, 3, 2, 2, 2, 4, 2, 2, 4, 2, 2, 2, 2, 2, 4, 2, 4, 4, 5\}$ $DUE_i = \{4, 8, 10, 12, 14, 19, 23, 22, 25, 27, 28, 33, 33, 35, 39, 39, 40, 42, 43, 44, 48, 50, 52, 54, 59\}$ $PEN_i = \{3, 3, 3, 3, 3, 4, 4, 4, 4, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 3, 3, 3, 3, 3, 4\}$	(5,11)(6,9)(6,11)
Inst 15	(24,20,10)	$ARR_i = \{2, 4, 8, 8, 10, 14, 19, 19, 23, 25, 26, 29, 31, 33, 35, 37, 38, 40, 41, 42, 44, 48, 48, 50, 54, 55, 57, 61, 61, 63\}$ $LEN_i = \{5, 5, 2, 3, 2, 5, 2, 3, 5, 2, 5, 5, 2, 5, 5, 2, 2, 2, 2, 5, 5, 2, 3, 2, 5, 5, 5, 2, 3, 2\}$ $PRO_i = \{2, 4, 2, 4, 4, 5, 4, 3, 2, 2, 2, 4, 2, 2, 4, 2, 2, 2, 2, 2, 4, 2, 4, 4, 5, 2, 4, 2, 4, 4\}$ $DUE_i = \{4, 8, 10, 12, 14, 19, 23, 22, 25, 27, 28, 33, 33, 35, 39, 39, 40, 42, 43, 44, 48, 50, 52, 54, 59, 57, 61, 63, 65, 67\}$ $PEN_i = \{3, 3, 3, 3, 3, 4, 4, 4, 4, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 3, 3, 3, 3, 3, 4, 3, 3, 3, 3, 3\}$	(5,11)(6,9)(6,11)

Table 3
Evolutionary algorithm's parameter values tested in Section 6.1.

Population size (PopSize)	Crossover probability (χ)	Mutation probability (μ)	Mutation rate (R)
500	0.70	0.80	0.900
1000	0.80	0.20	0.010
1500	0.90	0.50	0.001

Table 4
Best 10 tested combinations for parameter selection considering the fitness.

Combination #	χ	μ	R	PopSize	Avg. position
Comb1f	0.9	0.5	0.010	1000	122.446
Comb2f	0.8	0.5	0.010	1500	122.446
Comb3f	0.9	0.5	0.001	1500	122.450
Comb4f	0.9	0.5	0.010	500	122.459
Comb5f	0.8	0.5	0.001	1000	122.459
Comb6f	0.9	0.5	0.010	1500	122.473
Comb7f	0.8	0.5	0.001	1500	122.476
Comb8f	0.9	0.5	0.001	1000	122.484
Comb9f	0.8	0.5	0.010	1000	122.490
Comb10f	0.8	0.5	0.001	500	122.491

Table 5
Best parameter combinations among all datasets considering the CPU time.

Combination #	χ	μ	R	PopSize	Avg. position
Comb1t	0.7	0.001	0.001	500	150.555
Comb2t	0.7	0.001	0.100	1500	155.947
Comb3t	0.8	0.001	0.100	500	174.354
Comb4t	0.7	0.001	0.001	500	174.946
Comb5t	0.8	0.001	0.010	500	176.540
Comb6t	0.8	0.001	0.001	500	178.084
Comb7t	0.9	0.001	0.010	500	179.371
Comb8t	0.9	0.001	0.001	500	180.542
Comb9t	0.9	0.001	0.100	500	185.233
Comb10t	0.7	0.200	0.010	500	188.437

afore-mentioned parameters, we choose three values and we conduct tests for several runs of the GA over all possible combinations (81) of those values (Table 3).

These values were selected based on the literature review, trial and error method and the most appropriate values for this instance. We examine the dynamics of GA from two viewpoints. The first is to investigate the best solution found by the system (Table 4). The second viewpoint is the less computational time needed to find that solution (Table 5).

The best mutation probability among trial values, seems to be $\mu = 0.50$ to obtain fitter solution. Two most efficient crossover probabilities are shown in Table 4 which are $\chi = 0.90$ and $\chi = 0.80$ respectively, and two most efficient mutation rates $R = 0.010$ and $R = 0.001$ respectively. It appears like the more populations the most efficient the result in Table 4. However, the increase in the number of chromosomes in a population has an impact on the computing time, which leads to a decrease in the number of generations performed in a given time frame (Table 5). In this case, the maximum value was not really the best but the computing time was reduced. As for the mutation probability, the values of the latter remain uniform. It is common to use a negligible ($\ll 0.0001$) value of this parameter. On the other hand, the three values chosen for the other three parameters tested are able to provide satisfactory results in terms of CPU time.

These parameter combinations (Tables 4–5) were determined in two steps:

1. The first step is to calculate the average position of the results obtained using a given set of parameters over all data sets.
2. The second step is to establish a ranking where the higher the average position, the better the result. The average position obtained, mentioned above, is illustrated in the last column of both Tables 4–5.

6.1. GA parameters statistical analysis

The Analysis Of the Variance (ANOVA) is one of the most broadly used statistical techniques to investigate the relevancy and relative importance of the parameters involved in GA design. The analysis of variance table comprising the sum of squares, degrees of freedom, mean square, test statistics, constitutes the initial analysis in a compact form. This type of tabular representation is commonly used to set out the results of ANOVA calculations.

6.1.1. Best solution analysis

In this section, we firstly test if the response is the same for all factor levels. Secondly, we test each two-factor interaction. Table 6 gives the four-way variance analysis for the whole groups of problems studied. The considered factors corresponds to χ, PopSize, μ and R, respectively. The *p-values* equal 0.061, 0.573 and 0.387 indicate that the mean responses for the levels of the factors χ, PopSize and R are not significantly different.

Table 6
Two-way ANOVA with repeated experiments for the performance of the selection methods based on fitness while adopting three different type of crossover operators.

Source	Sum Sq.	d.f.	Mean Sq.	F	*p-value* > *F*
χ	0.681	1	0.681	3.540	0.061
PopSize	0.061	1	0.061	0.320	0.573
μ	38.588	1	38.588	200.430	0
R	0.144	1	0.144	0.750	0.387
Error	45.822	238	0.192		
Total	85.287	242			

Table 7
ANOVA of the interactions of the parameters of GA considering the best fitness.

Source	Sum Sq.	d.f.	Mean Sq.	F	*p-value* > *F*
χ	0.008	1	0.008	0.04	0.836
μ	0.543	1	0.543	2.85	0.092
$\chi \bowtie$ PopSize	0.144	1	0.144	0.76	0.384
$\chi \bowtie \mu$	0.222	1	0.222	1.17	0.281
$\chi \bowtie$ R	0.004	1	0.004	0.02	0.880
PopSize $\bowtie \mu$	0.061	1	0.061	0.32	0.571
$\mu \bowtie$ R	0.019	1	0.019	0.10	0.752
Error	44.433	233	0.190		
Total	85.287	242			

Table 8
ANOVA of the parameters of GA considering the computational time.

Source	Sum Sq.	d.f.	Mean Sq.	F	*p-value* > *F*
χ	6.22482e+07	1	6.224e+07	1.27	0.260
PopSize	6.088e+07	1	6.088e+07	1.24	0.266
μ	2.98615e+07	1	2.98615e+07	0.61	0.435
R	9.97825e+07	1	9.97825e+07	2.04	0.154
Error	1.16652e+10	238	4.90134e+07		
Total	1.18654e+10	242			

Table 9
ANOVA of the interactions of the parameters of GA considering the computational time.

Source	Sum Sq.	d.f.	Mean Sq.	F	*p-value* > *F*
χ	194849794.801	1	194849794.801	4.02	0.046
μ	228739728.362	1	228739728.362	4.72	0.030
$\chi \bowtie$ PopSize	125334142.572	1	125334142.572	2.59	0.109
$\chi \bowtie \mu$	101945971.807	1	101945971.807	2.1	0.148
$\chi \bowtie$ R	149402717.872	1	149402717.872	3.08	0.126
PopSize $\bowtie \mu$	127081706.855	1	127081706.855	2.62	0.106
Error	11285307109.679	233	48434794.462		
Total	11865367093.8056	242			

However, the *p-value* equals 0 is small enough to conclude that the mean responses are significantly different for the levels of the factor μ. The population size factor PopSize presents the lowest statistical relevance and has the lowest mean square while the mutation probability has the higher mean square. Obviously, this is because that the latter is associated with diversity among the population, with high fitness individuals increasing the average fitness value.

An important analysis that must be conducted is that of the interaction between the main effects considered. As it can be seen from Table 7, there is no variable from the ones analyzed in the statistical process that has a significant effect on the execution of GA with regards to the best solution. Note that the mutation percentage occurring exhibits the biggest statistical relevance because the higher the F-ratio or the smaller the significance level, the greater the relevance of the corresponding factor.

The interaction property is represented by the symbol $\bowtie$ and all the interactions between parameters are: $\chi \bowtie$ PopSize, $\chi \bowtie \mu$, $\chi \bowtie$ R, PopSize $\bowtie \mu$ and $\mu \bowtie$ R in the ANOVA table. An interaction is defined as when the difference in response means between the levels of one factor is not the same for all levels of another factor. The *p-values* listed in the last column in Table 7 indicate that the corresponding interactions are not significant. The crossover and mutation percentages, the population size and the mutation rate are not so significant, even though several of the levels employed in this study produce statistically distinct behavior patterns concerning the output variable.

6.1.2. Computational time analysis

The technique used is the same as that employed in Section 6.1.1. In this case, we assess whether the four factors have a significant influence on the CPU time by GA or not. Table 8 detail the variance analysis for the CPU time.

It is important to note that the different levels of each parameter produce only one set; thus we decide to discard it from the study of computational time. Table 9 shows that all the interactions between the factors are insignificant. The higher F-ratio is between the crossover probability and the mutation rate.

7. Statistical comparison of the parent selection methods

In this section, we summarize the performance evaluation showing strength and weakness of the well known GA selection processes: the Roulette Wheel Selection (RWS), the Tournament Selection (TS) and the Random Selection (RS). A descriptive statistical analysis represented by the variability of the generated data and the measures of central tendency is indicated in Figs. 5–6–7, where the solutions are shown in relation to their final solutions (Figs. 5(a)–6(a)–7(a)) and to their computation time (sec) (Figs. 5(b)–6(b)–7(b)).

7.1. Comparison of the three distributions based on the fitness

This comparison study involves three distributions, we have identified three cases: The first case uses Single Point Crossover. The second applies Double point Crossover and the last case uses Uniform Crossover. We note that the TS has the best median in the three cases mentioned above and there are no outliers detected in any of these cases.

However, as the data overlapped, it is not possible to visually dissect which version performed best. Therefore, it is necessary to perform a statistical test to estimate the existence of differences between the processes. The two-way ANOVA statistical test is used to evaluate the population means and to determine whether the independent variable GA has a statistically significant effect on the solution fitness (Table 10) and completion time (Table 11) under the three null hypotheses:

$\mathcal{H}0$: The means of observations grouped by one factor are the same.
$\mathcal{H}1$: The means of observations grouped by the other factor are the same.
$\mathcal{H}2$: There is no interaction between the two factors.

- Taking the significance level $\alpha = 0.05$, we performed a two-way ANOVA test with replication resulting in a *p-value* of 0.379 (Table 10). As this exceeds the 0.05 Alpha level, the result has no statistical significance. We can conclude that the crossover operator chosen has no significant impact on the execution time. Moreover, we assess whether the selection method has a notable impact.
- The *p-value* for this factor is $0.0023 < 0.05$. Because of that, we consider parent selection method to be statistically significant. We can remark a substantial difference in fitness values between roulette wheel selection, tournament selection and random selection method.
- The last F-test will examine the interaction test. The *p-value* is 0.406, which means the result is not statistically significant. There is no significant interaction between crossover operators and selection method in terms of fitness.

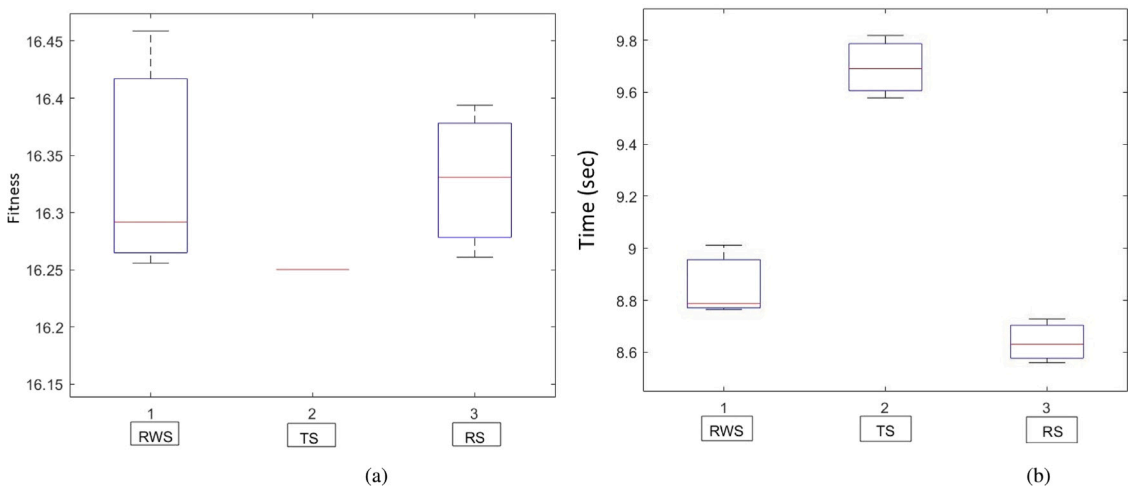

Fig. 5. Performance of the selection operators RWS, TS, RS using the Single Point Crossover, Inst 2 test and the parameters combination Comb1f (Tables 4 and 12).

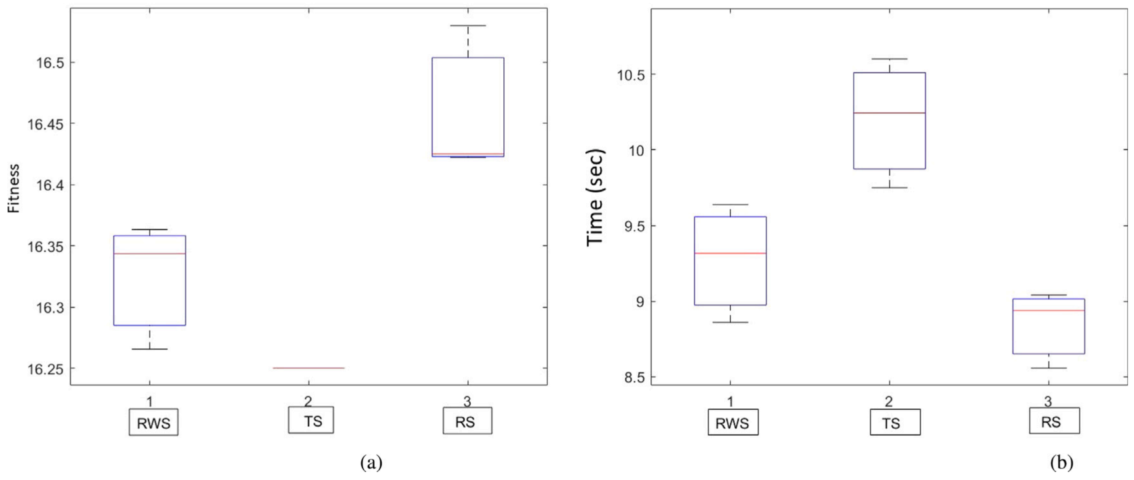

Fig. 6. Performance of the selection operators RWS, TS, RS using the Double Point Crossover, Inst 2 test and the parameters combination Comb1f (Tables 4 and 12).

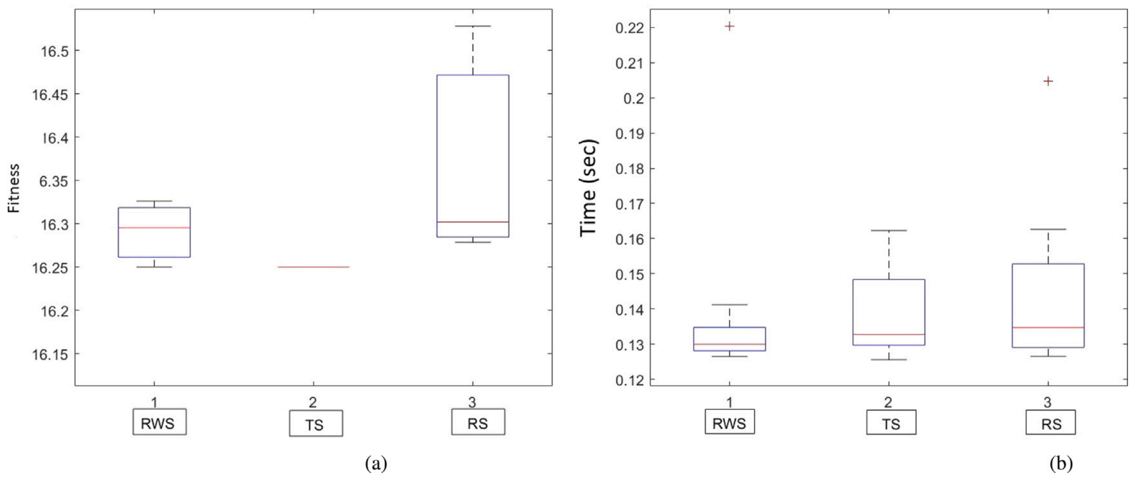

Fig. 7. Performance of the selection operators RWS, TS, RS using the Uniform Crossover, Inst 2 test and the parameters combination Comb1f (Tables 4 and 12).

Comparing the mean for each group, we see that the averages fitness values in the RWS group are 16.2905, 16.3241 and 16.3354 while for the TS all are identical equal to 16.25 and for the RS group it is more, sitting at 16.3696, 16.4591 and 16.3284. This comparison is not enough to assess whether the difference is statistically significant, but it allows to learn more from the data.

Table 10
Two-way ANOVA with repeated experiments for the performance of the selection methods based on fitness and applying the 3 different crossover operators.

Source of variations	Sum of squares	Degree of freedom	Mean of squares	F	*p-value*	Critical value for F
Sample	0.009	2	0.004	1.022	0.379	3.554
Columns	0.082	2	0.041	8.653	0.002	3.554
Interaction	0.020	4	0.005	1.056	0.406	2.927
Inside the group Sample	0.086	18	0.004			
Total	0.1992	26				

Table 11
Two-way ANOVA table with repeated experiments for the performance of the selection methods considering CPU time and applying the 3 different crossover operators.

Source of variations	Sum of squares	Degree of freedom	Mean of squares	F	*p-value*	Critical value for F
Sample	0.647	2	0.323	3.3384	0.058	3.554
Columns	5.176	2	2.588	26.7028	4.1101E−06	3.554
Interaction	0.346	4	0.086	0.8934	0.488	2.927
Inside the group Sample	1.744	18	0.096			
Total	7.914	26				

7.2. Comparison of the three distributions based on the computing time

In the single and double point crossover cases, the RS has the best median while in the case of the uniform crossover the RWS leads to lower computational time and optimal median. Two outliers occur when adopting the RWS and the RS in the case of choosing the uniform crossover operator. The RWS and the RS are left skewed in only one case which is the double point crossover while in the two other these distributions are right skewed. The TS has the worst median when the single and double point crossover where selected. The TS and the RS are similar in the case where the uniform crossover operator was chosen, but the upper whisker of TS is better (Fig. 7(b)).

Based on the confidence level 0.05 generated, we know that the crossover operators do not influence the GA solutions in terms of convergence time since the *p-value* obtained is 0.0584 greater than α (Table 11) while the selection methods has a substantial impact on the completion time since the *p-value* for this factor is 4.1101E−06. Therefore, we expect the RWS, the TS and the RS groups to differ from each other.

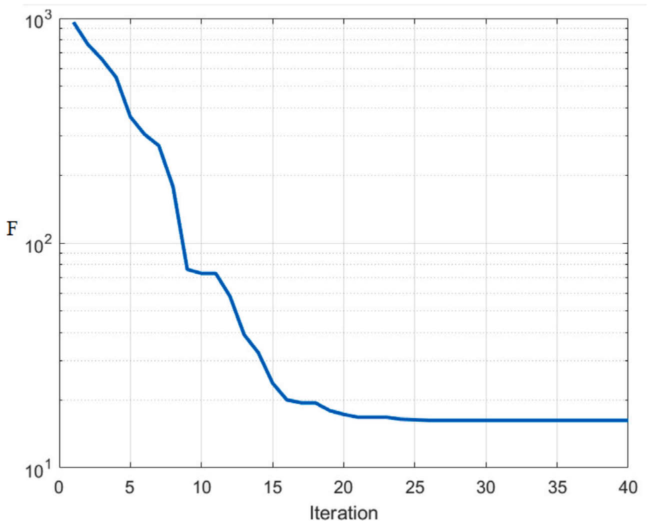

Fig. 8. Convergence of the instance "Inst2" using the best parameters combinations in terms of cost (Combination Comb1f).

8. Results and discussion

8.1. Results of GA meta-heuristic and cplex

To test the model's efficacy, we use the solver Cplex V. 12.7.0. We have succeed to solve the instances with up to 19 vessels. Tables 12 and 13 detail the results. On the one hand, for small size instances, Table 12 columns `Cplex` and `GA` show that the solutions (both columns $ObjVal$) have been obtained with a CPU time less than one minute by Cplex, while GA was able to compute the solution in less than two minutes. Column "% Dev" gives the percentage of deviation of GA solution from the optimal ot the best feasible solution obtained by Cplex, and computed as $\% \text{ Dev} := 100 \times \frac{(GA - Optimal\ (or\ Best\ Feasible\ Solution))}{Optimal\ (or\ Best\ Feasible\ Solution)}$ where $Optimal$ and $Best\ Feasible\ Solution$ represent the solution status obtained by Cplex. This column shows that on average % Dev ranges between 0.195% and 16.071%.

On the other hand, for large size instances, Cplex was not able to reach the optimal solution for instances Inst 11, Inst 12 and Inst 13. We have set to a maximum of 3600 s the run time for Cplex.

Cplex was able to compute the optimal solution for only two instances in less than one second (Inst 14 and Inst 15) while GA was able to solve all the large instances with a CPU time less that two minutes ten seconds and with an average % Dev that ranges between 0.320% and 4.633%.

The Table 14 provides an overview of the transshipment connections between vessels in the specified instances, distinguishing between direct and indirect relationships. Column `Instance` lists the instances being considered. Column `DT vessel pairs` indicates the pairs of vessels that have a direct transshipment relationship with each other. The term `NDT` (No direct transshipment vessel pairs) is used when there are no direct transshipment connections between vessels. Column `IT vessel pairs` shows the pairs of vessels that have an indirect transshipment relationship with each other. The format [(a,b)(c,d)...] is used to represent multiple indirect transshipment connections between vessels. `NDT` does not indicate that there will be no transshipment operation at all, which means the containers will not necessarily remain in the port without being transferred to another vessel and staying there until further handling or transportation arrangements are made. `NDT` can imply two possibilities: (i) the absence of direct transshipment, and (ii) the presence of indirect transshipment. In the case of indirect transshipment, the containers will be temporarily unloaded at the terminal until the arrival of the second concerned vessel. Subsequently, the containers will undergo an intermediate transshipment process at the port before being loaded onto the second vessel.

Table 12
Computational results of small and medium size instances using the parameters combination Comb1f of Table 4.

Instance	$Cplex$			GA		
	$ObjVal$	CPU(s) ≤ 3600 s	Solution status	$ObjVal$	CPU(s)	% Dev
Inst 1	29.0	0.03	Optimal	30.066	28.638	3.675
Inst 2	14.0	0.00	Optimal	16.250	10.611	16.071
Inst 3	35.0	0.03	Optimal	37.661	107.953	7.604
Inst 4	56.0	0.66	Optimal	59.010	59.584	5.376
Inst 5	32.0	0.03	Optimal	32.062	43.386	0.195
Inst 6	32.0	0.03	Optimal	32.550	51.242	1.720
Inst 7	12.0	0.02	Optimal	13.400	17.787	11.670
Inst 8	40.0	0.05	Optimal	41.891	55.989	4.728
Inst 9	60.0	0.08	Optimal	64.061	64.251	6.768
Inst 10	54.0	0.11	Optimal	56.127	78.376	3.938

Table 13
Computational results of large-scale problems using the parameters combination Comb1f of Table 4.

Instance	$Cplex$			GA		
	$ObjVal$	CPU(s) ≤ 3600 s	Solution status	$ObjVal$	CPU(s)	% Dev
Inst 11	128.0	3600	Best feasible solution	130.462	239.162	1.923
Inst 12	122.0	3600	Best feasible solution	122.619	478.805	0.507
Inst 13	107.0	3600	Best feasible solution	110.531	389.081	3.300
Inst 14	75.0	0.13	Optimal	78.475	59.069	4.633
Inst 15	91.0	0.14	Optimal	91.291	125.902	0.320

Table 14
Transshipment connections results of the small and medium size instances. (DT): direct transshipment, (IT): indirect transshipment, (NDT): No direct transshipment vessel pairs.

Instance	DT vessel pairs	IT vessel pairs
Inst 1	NDT	[(6,8)]
Inst 2	NDT	[(2,3)]
Inst 3	[(4,11)]	[(2,7)(2,8)(3,8)(3,9)]
Inst 4	NDT	[(5,12)(6,10)(6,12)]
Inst 5	NDT	[(1,3)(1,4)(2,4)(2,6)(2,7) (2,8)(3,5),(3,9)(3,10)(4,8) (4,10)(6,8)(7,10)]
Inst 6	NDT	[(1,3) (1,4)(2,4)(2,6)(2,7) (3,5)]
Inst 7	[(3,4)]	[(1,2)(1,3)(2,3)]
Inst 8	NDT	[(5,12)(6,10)(6,12)]
Inst 9	NDT	[(5,12)(6,10)(6,12)]
Inst 10	NDT	[(5,12)(6,10)(6,12)]

From Fig. 8, we can see that the total value of the fitness for the instance "Inst2" converges after 25 iterations.

8.2. The impact of the transshipment constraints

This section provides a detailed analysis of the results obtained from our experiments and simulations, highlighting the strengths and limitations of the model in light of these constraints.

Tables 15–17 provide a summary of the results obtained for 7 different runs of three instances Inst11, Inst12 and Inst13, comparing Ak's model Ak (2008) with our proposed model. Columns 2 and 3 show objective values and consuming times for Ak's model, while Columns 4 and 5 display the performance of our model. Columns 6 and 7 show the percentage deviations of both the obtained objective value and the run time with regards to those for Ak's model.

On average, our model demonstrates better solutions obtained compared to Ak's model (Column 6) and similar slightly bigger consuming times (Column 7). The values with ‡ symbol correspond to Ak's model objective values that outperform our model objective values. Also the model shows a good accuracy in estimating dwell time and penalties that is 2.1598% on average (the average of the 3 Columns 6). However, we remark an increase in computational time due to the additional constraints which requires more processing time. The addition of constraints related to ship arrival times and direct transshipment introduces more intricate relationships between ship schedules. These constraints lead to complex scheduling conflicts and require the algorithm to explore a more constrained solution space, which can result in longer computation times.

These results highlight the advantages of incorporating transshipment connections into berth allocation recovery, particularly for busy container transshipment terminals and situations involving significant delays.

9. Conclusions and future work

This work involves a comprehensive study on a berth planning optimization problem in transshipment container terminals. To organize the loading and unloading of ships arriving or leaving the port, the scope of the planning covers two main processes (position allocation and development of transshipment activities connection design). The model attributes are basically: a continuous quay, dynamic arrival times for ships, i.e., ships arrive individually with fixed arrival dates, and an objective function which minimizes the berthing time taking into account the arrival delay penalties.

A new mixed integer programming (MIP) formulation is offered and a genetic algorithm GA for solving large problems is proposed. To create initial population for GA, a packing heuristic algorithm based on the two-dimensional hierarchical first-fit (FF) rule is developed. Instances with 3 to 15 vessels were solved optimally by the exact method Cplex under 1 s. More realistic instances with 20 to 30 vessels are solved using Cplex where a time limit is set. With the increase of problem size, the computational experiments show that the performance of the evolutionary GA increases and the results were obtained in significantly computational times.

This integrated model takes into account the specific requirements of ships and their suitability for berths and transshipment operations. It aims to minimize the berthing time taking into account the arrival delay

Table 15
Comparison between Ak's model Ak (2008) and our model for 7 runs for `Inst 11`.

Run	The Ak's model		Our model		% of deviation Dev.	
	Objective value	Run time	Objective value	Run time	Objective value	Run time
1	137.0426	198.6330	130.463	221.7085	4.8011	−11.6171
2	137.0701	200.0486	130.4631	237.1503	4.8202	−18.5463
3	136.736	198.3515	130.4631	224.7171	4.5877	−18.2097
4	138.6646	198.2019	130.4631	221.9589	5.9147	−18.2097
5	133.462	199.8998	130.4631	222.7703	2.2471	−17.5302
6	133.3935	200.1124	130.4631	222.3159	2.1968	−17.0189
7	129.258	209.8626	130.4631	223.1226	−0.9322‡	−10.1638
Average	**135.0895**	**200.7300**	**130.4631**	**224.8205**	**3.3765**	**−15.8994**

Table 16
Comparison between Ak's model Ak (2008) and our model for 7 runs for `Inst 12`.

Run	The Ak's model		Our model		% of deviation Dev.	
	Objective value	Run time	Objective value	Run time	Objective value	Run time
1	128.6698	403.4317	123.063	432.4840	4.3575	−7.2013
2	123.1427	392.7036	123.0198	441.2437	0.0998	−12.3605
3	129.0562	396.5504	123.1946	433.9197	4,.5419	−9.4236
4	122.8360	396.9029	122.9605	430.2377	0.1014	−8.3987
5	128.1590	393.6398	123.0819	434.4291	3.9616	−10.3621
6	124.8481	395.7635	123.0195	430.4751	1.4647	−8.7708
7	124.2928	396.4959	122.9905	438.5502	1.0478	−10.6065
Average	**125.8578**	**396.4983**	**123.0471**	**434.4771**	**2.2249**	**−9.5891**

Table 17
Comparison between Ak's model Ak (2008) and our model for 7 runs for `Inst 13`.

Run	The Ak's model		Our model		% of deviation Dev.	
	Objective value	Run time	Objective value	Run time	Objective value	Run time
1	113.3374	390.8481	109.9336	444.8793	3.0032	−13.8241
2	110.7586	393.8093	111.0628	441.8732	−0.2747 ‡	−12.2048
3	115.2033	391.1387	111.2964	496.8523	3,.913	−27.0271
4	110.8896	391.0447	111.9038	444.2126	−0.9146 ‡	−13.5964
5	110.8853	390.4015	111.0998	440.0711	−0.1934 ‡	−12.7227
6	108.0728	390.1778	111.1938	438.9652	−2.8879‡	−12.5039
7	116.2206	408.2820	111.5466	484.9616	4.0217	−18.7810
Average	**112.1953**	**393.6717**	**111.1481**	**455.9736**	**0.8780**	**−15.8086**

penalties and enhance overall port productivity. By optimizing the allocation of berths, scheduling ship arrivals and departures, coordinating ship-to-ship transshipment operations, and managing port resources, the model ensures efficient utilization of resources and reduces idle time.

Comparative result analysis for fitness value using roulette wheel, tournament selection and random selection method clearly show that tournament method has generated very fit populations generation compare to former both methods. The fastest convergence was reached by roulette wheel selection method. Then, GA was performed to obtain the optimal processing parameters. Based on the result, the problem has been optimized by 37.880% as compared to the recommended setting obtained through simulation analysis. The convergence speed is minimized by 4.5267%. For further work, we suggest proposing other important criteria to minimize ship berthing time and extending the model to a multi-objective optimization problem by considering multiple objectives and applying appropriate multiobjective algorithms.

The integration of constraints preventing ships arriving at different times from performing direct transshipment can lead to solutions that are more aligned with real-world operational scenarios, better resource utilization, and improved overall efficiency. This integration optimizes cargo handling, reduces waiting time, and improves overall port productivity. While the computational time required to solve the model might be higher, the enhanced quality of the solutions can justify the investment in terms of both operational effectiveness and customer satisfaction.

Also, we intend to improve the proposed model by considering environmental factors, such as energy efficiency and emissions reduction, aligns with sustainable practices and promotes environmentally friendly port operations. Other perspective is to take advantage of systems like the Automated Identification System (AIS), where vessels can transmit accurate position and ETA (Estimated Time of Arrival) data. This allows port authorities to receive timely information and make informed decisions to minimize berthing time and optimize resource allocation.

CRediT authorship contribution statement

Marwa Al Samrout: Conceptualization, Methodology, Software, Data curation, Writing – revised draft preparation. **Abdelkader Sbihi:** Visualization, Investigation, Validation, Revision, Writing – review & editing. **Adnan Yassine:** Visualization, Validation, Reviewing and editing.

Data availability

Data will be made available on request.

References

AbouKasm, O., Diabat, A., Cheng, T., 2020. The integrated berth allocation, quay crane assignment and scheduling problem: mathematical formulations and a case study. Ann. Oper. Res. 291, 435–461.
Ak, A., 2008. Berth and Quay Crane Scheduling: Problems, Models and Solution Methods. Georgia Institute of Technology.
Alp, O., Erkut, E., Drezner, Z., 2003. An efficient genetic algorithm for the p-median problem. Ann. Oper. Res. 122 (1), 21–42.
Bierwirth, C., Meisel, F., 2015. A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. European J. Oper. Res. 244 (3), 675–689.
Carroll, C.W., 1961. The created response surface technique for optimizing nonlinear, restrained systems. Oper. Res. 9 (2), 169–184.
Chargui, K., Tarik Zouadi, T., Sreedharan, V.R., 2023. Berth and quay crane allocation and scheduling problem with renewable energy uncertainty: A robust exact decomposition. Comput. Oper. Res. 156, 10625.
Cheimanoff, N., Féniès, P., Nour Kitri, M., Tchernev, N., 2023. Exact and metaheuristic approaches to solve the integrated production scheduling, berth allocation and storage yard allocation problem. Comput. Oper. Res. 153, 106174.
Correcher, J.F., Alvarez-Valdes, R., Tamarit, J.M., 2019. New exact methods for the time-invariant berth allocation and quay crane assignment problem. European J. Oper. Res. 275 (1), 80–92.
Dkhil, H., Diarrassouba, I., Benmansour, S., Yassine, A., 2021. Modelling and solving a berth allocation problem in an automotive transshipment terminal. J. Oper. Res. Soc. 72 (3), 580–593.
El Samrout, A., 2018. Hybridization of multicriteria metaheuristic optimization methods for mechanical problems. (Ph.D. thesis). Université de Technologie de Troyes.
Fiacco, A.V., McCormick, G.P., 1964a. Computational algorithm for the sequential unconstrained minimization technique for nonlinear programming. Manage. Sci. 10 (4), 601–617.
Fiacco, A.V., McCormick, G.P., 1964b. The sequential unconstrained minimization technique for nonlinear programing, a primal–dual method. Manage. Sci. 10 (2), 360–366.
Fiacco, A.V., McCormick, G.P., 1966. Extensions of sumt for nonlinear programming: equality constraints and extrapolation. Manag. Sci. 12 (11), 816–828.
Goel, G., Tiwari, R., 2022. Task management in IoT-fog-cloud environment employing static scheduling techniques. ENP Eng. Sci. J. 2 (1), 13–20.
Holl, J.H., 1995. Hidden Order, how Adaptation Builds Complexity. California, Addison-Wesley Publishing Company, Redwood City.
Karam, A., Eltawil, A., HegnerReinau, K., 2020. Energy-efficient and integrated allocation of berths, quay cranes, and internal trucks in container terminals. Sustainability 12 (8), 3202.
Lee, D.-H., Chen, J.H., Cao, J.X., 2010. The continuous berth allocation problem: A greedy randomized adaptive search solution. Transp. Res. E 46 (6), 1017–1029.
Lee, D.-H., Jin, J.G., Chen, J.H., 2012. Terminal and yard allocation problem for a container transshipment hub with multiple terminals. Transp. Res. E 48 (2), 516–528.
Liang, C., Hwang, H., Gen, M., 2012. A berth allocation planning problem with direct transshipment consideration. J. Intell. Manuf. 23, 2207–2214.
Lujan, E., Vergara, E., Rodriguez-Melquiades, J., Jiménez-Carrión, M., Sabino-Escobar, C., Gutierrez, F., 2021. A fuzzy optimization model for the berth allocation problem and quay crane allocation problem (bap+ qcap) with n quays. J. Mar. Sci. Eng. 9 (2), 152.
Lv, X., Jin, J.G., Hu, H., 2020. Berth allocation recovery for container transshipment terminals. Marit. Policy Manag. 47 (4), 558–574.
Lyu, X., Negenborn, R.R., Shi, X., Schulte, F., 2022. A collaborative berth planning approach for disruption recovery. IEEE Open J. Intell. Transp. Syst. 3, 153–164.
Malekahmadi, A., Alinaghian, M., Hejazi, S.R., Saidipour, M.A.A., 2020. Integrated continuous berth allocation and quay crane assignment and scheduling problem with time-dependent physical constraints in container terminals. Comput. Ind. Eng. 147, 106672.
Moorthy, R., Teo, C.-P., 2007. Berth management in container terminal: the template design problem. In: Container Terminals and Cargo Systems: Design, Operations Management, and Logistics Control Issues. pp. 63–86.
Nellen, N., Lange, A.-K., Jahn, C., 2022. Potentials of direct container transshipment at container terminals. In: Changing Tides: The New Role of Resilience and Sustainability in Logistics and Supply Chain Management–Innovative Approaches for the Shift to a New Era. In: Proceedings of the Hamburg International Conference of Logistics (HICL), Vol. 33, epubli GmbH, Berlin, pp. 679–705.
Pang, K.-W., Liu, J., 2014. An integrated model for ship routing with transshipment and berth allocation. IIE Trans. 46 (12), 1357–1370.
Park, K., Kim, K.H., 2002. Berth scheduling for container terminals by using a sub-gradient optimization technique. J. Oper. Res. Soc. 53 (9), 1054–1062.
Petering, M.E.H., Murty, K.G., 2009. Effect of block length and yard crane deployment systems on overall performance at a seaport container transshipment terminal. Comput. Oper. Res. 36 (5), 1711–1725.
Pomentale, T., 1965. A new method for solving conditioned maxima problems. J. Math. Anal. Appl. 10 (1), 216–220.
Prayogo, D., Hidayatno, A., et al., 2021. Bi-objective optimization model for integrated planning in container terminal operations. In: IOP Conference Series: Materials Science and Engineering. Vol. 1072, IOP Publishing, 012022.
Rajendran, C., Chaudhuri, D., 1992. A multi-stage parallel-processor flowshop problem with minimum flowtime. European J. Oper. Res. 57 (1), 111–122.
Reda, O., Harraz, N., El-Tawil, A., 2016. Terminal operations in container transshipment hubs: Literature review and research directions. In: Conference: The International Conference of Engineering Sciences & Applications (ICESA 2016). Aswan, Egypt.
Rodrigues, F., Agra, A., 2022. Berth allocation and quay crane assignment/scheduling problem under uncertainty: A survey. European J. Oper. Res. 303 (2), 501–524.
Ruiz, R., Maroto, C., 2006. A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility. European J. Oper. Res. 169 (3), 781–800.
Sbihi, A., Chemangui, M., 2018. A genetic algorithm for the steel continuous casting with inter-sequence dependent setups and dedicated machines. RAIRO-Oper. Res. 52 (4–5), 1351–1376.
Schepler, X., Balev, S., Michel, S., Sanlaville, É., 2017. Global planning in a multi-terminal and multi-modal maritime container port. Transp. Res. E 100, 38–62.
Seyedalizadeh, G.S., Babazadeh, A., Arabshahi, N., 2010. Analysis of the continuous berth allocation problem in container ports using a genetic algorithm. J. Mar. Sci. Technol. 15 (4), 408–416.
Sherali, H.D., Sarin, S.C., Kodialam, M.S., 1990. Models and algorithms for a two-stage production process. Prod. Plan. Control 1 (1), 27–39.
Simrin, A., Diabat, A., 2015. The dynamic berth allocation problem: A linearized formulation. RAIRO-Oper. Res. 49 (3), 473–494.
Stong, R.E., 1965. A note on the sequential unconstrained minimization technique for non-linear programming. Manage. Sci. 12 (1), 142–144.
Tao, Y., Lee, C.-Y., 2015. Joint planning of berth and yard allocation in transshipment terminals using multi-cluster stacking strategy. Transp. Res. E 83, 34–50.
Vis, I., de Koster, R., 2003. Transshipment of containers at a container terminal: An overview. European J. Oper. Res. 147, 1–16.
Xiang, X., Lee, L.H., Chew, E.P., 2023. An adaptive dynamic scheduling policy for the integrated optimization problem in automated transshipment hubs. IEEE Trans. Autom. Sci. Eng. http://dx.doi.org/10.1109/TASE.2023.3267448.
Xiang, X., Liu, C., 2021. An almost robust optimization model for integrated berth allocation and quay crane assignment problem. Omega 104, 102455.
Zeng, Q., Feng, Y., Chen, Z., 2017. Optimizing berth allocation and storage space in direct transshipment operations at container terminals. Marit. Econ. Logist. 19 (3), 474–503.
Zhen, L., Chew, E.P., Lee, L.H., 2011. An integrated model for berth template and yard template planning in transshipment hubs. Transp. Sci. 45 (4), 483–504.
Zhen, L., Wang, S., Laporte, G., Hu, Y., 2019. Integrated planning of ship deployment, service schedule and container routing. Comput. Oper. Res. 104, 304–318.
Zhen, L., Wang, S., Wang, K., 2016. Terminal allocation problem in a transshipment hub considering bunker consumption. Nav. Res. Logist. 63 (7), 529–548.
Zheng, J., Yang, L., Han, W., Sun, Y., Meng, F., Zhen, L., 2021. Berth assignment for liner carrier clusters under a cooperative environment. Comput. Oper. Res. 136, 105486.