


Article

Discretization-Strategy-Based Solution for Berth Allocation and Quay Crane Assignment Problem

Min Tang ¹, Bin Ji ^{1,*}, Xiaoping Fang ¹  and Samson S. Yu ²

¹ School of Traffic and Transportation Engineering, Central South University, Changsha 410075, China; tangmin1314@csu.edu.cn (M.T.); fangxp@csu.edu.cn (X.F.)

² School of Engineering, Deakin University, Waurn Ponds 3216, Australia; s.yu@ieee.org

* Correspondence: chcujiabin@csu.edu.cn; Tel.: +86-189-0846-8853

Abstract: The continuous berth allocation and quay crane assignment problem considers the size of berths and ships, the number of quay cranes, the dynamic ships and non-crossing constraints of quay cranes. In this work, a mixed-integer linear programming model of this problem is established, aiming at minimizing the total stay time and delay penalty of ships. To solve the model, the continuous berth is separated into discrete segments via a proposed discretization strategy. Thereafter, a large neighborhood search algorithm composed of the random removal operator and relaxed sorting-based insertion operator and a backtracking comparison-based constraint repair strategy are proposed. The effectiveness of the model and algorithm presented is verified via real-life instances with different characteristics, and the performances of different combinations of removal operators and insertion operators in the large neighborhood search algorithmic framework are analyzed. Numerical results show that the large neighborhood search algorithm can optimally solve the small-scale instances in a reasonable time. Meanwhile, the results of large-scale instances show that the large neighborhood search algorithm incorporating the discretization strategy is more efficient than other genetic algorithms based on continuous optimization. With the proposed approach, high-quality berth and quay crane allocation results can be obtained efficiently.

Keywords: berth allocation and quay crane assignment; continuous berth discretization; large neighborhood search; constraint handling



Citation: Tang, M.; Ji, B.; Fang, X.; Yu, S.S. Discretization-Strategy-Based Solution for Berth Allocation and Quay Crane Assignment Problem. *J. Mar. Sci. Eng.* **2022**, *10*, 495. <https://doi.org/10.3390/jmse10040495>

Academic Editor: Mihalis Golias

Received: 8 March 2022

Accepted: 30 March 2022

Published: 2 April 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Maritime container transportation is one of the most important modes of transport in international trade, and about 80 percent of the global trade volume is carried out by sea and ports around the world [1]. With the increasing internationalization of container transportation and continuous growth of port trade volume, ports have become more and more busy and the competition between ports has become increasingly fierce. Berth and quay crane (QC) are the core operation resources of the forefront of container ports and making appropriate scheduling plans is of great significance to the effectiveness of container ports. In order to improve the efficiency of berth and QC operations at container terminals, this paper studies the berth allocation and quay crane assignment specific problem (BACASP).

Over the years, scholars have paid great attention to resource scheduling in ports and have obtained rich research results. Among them, the berth allocation problem is one of the most vital problems, which aims to allocate vessels to berths restricted by certain physical and technical constraints to maximize the efficiency of berth operations [2]. Bierwirth and Meisel [3] conducted a literature review on existing research results of berth allocation problem and classified the problem into continuous, discrete and mixed berth allocation problem according to the berth layouts. In the BAPD a quay is divided into many sections, called berths, where one vessel can be in service at a time. In the BAPC, there is no partition,

and a vessel can be docked anywhere as long as there is enough space. Nishimura et al. [4] established a BAPD model by considering the waiting time and handling time of vessels. Subsequently, they further considered the vessels' berthing priority and investigated the corresponding diversification of BAPD. The BAPD is inefficient in terms of berth utilization. With the rapid growth of transportation demand, discrete berth cannot meet the actual operational requirements. Imai et al. [5] studied the continuous berth allocation problem (BAPC), and obtained the upper and lower bounds of the problem by solving the corresponding simple discrete berth allocation problem (BAPD). Cordeau et al. [6] mapped the BAPD to a multi-depot vehicle routing problem with time windows (MDVRPTW) and presented a flow-based model for the BAPD. In addition, a strategy for discretizing the continuous berth space is proposed, and the BAPC and BAPD are unified into one solution framework. A tabu search algorithm is proposed to solve the BAPC and BAPD. The BAPC has been proved to be a NP-hard problem. Therefore, the research considerations for BAPC is dominated by heuristic algorithms, such as Genetic Algorithm (GA) [7–9], Memetic algorithm [10], ant colony optimization [11], particle swarm optimization algorithm [12], tabu search algorithm [13], differential evolution algorithm [14], etc.

In practice, the handling time of a vessel at the berth is related to the QCs assigned to the vessel, and incorporating berth allocation with quay assignment is essential for maximizing the efficiency of berth and quay operation. Park and Kim [15] first proposed the berth allocation and crane assignment problem (BACAP) and proposed a two-stage approach to solve the problem, wherein the first stage used sub-gradient optimization to determine the optimal berthing position and start time of berthing and number of QCs, while the second stage utilized a dynamic programming method to assign specific QCs. Meisel et al. [16] further considered the interference between QCs and the impacts of vessels' berthing positions on its handling time, and proposed a mixed-integer linear model for the BACAP. Rodriguez-Molins et al. [17] proposed an innovative GRASP-based approach to solve the BACAP. Researchers in [18] proposed a mixed integer programming model for the BACAP and a multi-objective algorithm to achieve the balance between delayed time of vessels and handling energy consumption of QCs. Iris et al. [19] presented a BACAP model by considering the vessel speed-up cost, delayed completion penalty delayed departure penalty and QC cost. Thereafter, an algorithm based on large neighborhood search is proposed to solve the problem, which was confirmed capable of solving instances with up to 80 vessels. In their research, the number of QCs assigned to a vessel was assumed to be able to change dynamically. However, the costs of operations due to frequently moving and resetting of QCs were not considered. He et al. [20] proposed a mixed-integer programming model for BACAP and factors such as the work efficiency of QC operators, the operation cost of QCs and operation efficiency and performance-related monetary compensation between day and night, are considered.

The research mentioned above simplifies the BACAP by ignoring the restrictions that the QCs cannot cross each other along the track. Zhang et al. [21] and Türkoğulları et al. [22] argued that frequent change of QC assignment would result in considerable setup costs, and thus the QC reassignment was inhibited once the operation starts. Regarding this, Türkoğulları et al. [22] considered the non-crossing constraints of QCs, and proposed a mathematical model with a branch-cut-plane algorithm for solving small-scale berth allocation and quay crane assignment specific problems. Agra et al. [23] proposed a mixed-integer model for the BACASP, where heterogeneous QCs with different loading and unloading speeds and the non-crossing constraints of QCs are enforced. The model was solved via a branch-and-cut algorithm, which was proven to be able to effectively solve small-scale BACASPs to optimality. Thanos et al. [24] considered the movement range of QCs and established a BACASP model aiming at minimizing the transshipment distance of containers. Then a local search algorithm is proposed to solve this model. Bouzekri et al. [25] considered different water depths and multiple quays, and proposed a model integrating berth allocation and QC assignment problem.

Due to the high complexity, heuristic algorithms are usually employed for solving the BACAP [26], especially for dealing with large-scale problems. Apart from the methods mentioned above, other heuristics, especially the genetic algorithm [27], have also been applied to solve BACAP. In GA, the constraint handling strategy has a great influence on the quality of the obtained near-optimal solution. Among the studies on the BACAP, superiority of feasible solutions [28], repair method [29] and penalty function [30–32] are widely used. The superiority of feasible solutions strategy is easy implementation, but it more suitable for multi-objective models. The repair method refers to adjusting an infeasible solution to its nearest feasible solution. This strategy has high search efficiency. Moreover, the reason why the penalty function strategy has been widely used is its high flexibility and its easy implementation.

In this paper, a mixed-integer nonlinear programming model is proposed for BACASP to minimize the total port stay time of vessels and the penalties of delay with the spatial and temporal constraints such as the lengths of continuous berth, number of QCs and non-crossing of QCs, etc. In the traditional solution method, the berthing position and handling time of vessels are regarded as continuous decision variables, and the number of QCs is regarded as discrete decision variables. BACASP is regarded as a hybrid optimization problem and is solved by evolutionary algorithms such as genetic algorithm (GA) in [33]. This method contains continuous and discrete variables, making it difficult to design the coding information interaction method and to guarantee the convergence performance and stability of the algorithm [34]. To solve the problems associated with the existing methods, a strategy for discretizing the continuous berth space is proposed to solve the BAP [6,35]. The BAP is a subproblem of the BACASP to be addressed in our study.

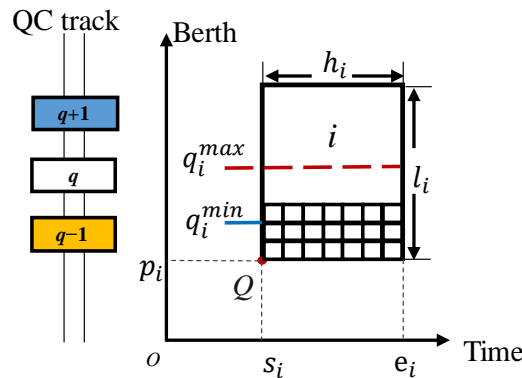
In this work, an MINLP model is presented for the BACASP. The BACASP is converted to a discrete combinatorial optimization problem via a discretization strategy, and a large neighborhood search (LNS) incorporating a specific constraint-handling technique is proposed to efficiently solve it. The main contributions of this work are three-folds and summarized as follows: (1) A discretization-strategy-based method to solve the BACASP based on its combinatorial optimization property. (2) A specific LNS algorithm is designed to solve the BACASP. Under the framework of LNS algorithm, some destroy and repair operators are designed according to the problem's characteristics to update the solution and a backtracking comparison-based (BCB) constraint-handling strategy is proposed to enhance the performance of the LNS for achieving feasible solutions of the BACASP, so as to achieve an efficient solution to the BACASP. (3) A large number of experiments are conducted to demonstrate the effectiveness of the proposed algorithm and constraint-handling strategy in solving the BACASP.

2. Problem Description

The problem studied in this paper is described below. Section 2.1 describes the notations and Section 2.2 introduces the problem model.

To facilitate the analysis, generally the BACASP can be described within a space-time two-dimensional coordinate, which is described in Figure 1. The horizontal axis represents the time, the vertical axis represents the berth, and the rectangles represent the vessels, the lengths of which in the vertical and horizontal directions (l_i and h_i) represent the length of the vessel and the handling time, respectively. The vessel can be berthed at any position in the continuous berth. The handling time of a vessel is affected by the number of assigned QCs and its berthing position when considering the interference between the QCs. As a result, it is assumed that each vessel corresponds to an optimal berthing position in which the interference between the QCs is minimal. The QC track is distributed along the coastline, where the QC can move to carry out loading and unloading operations on any vessel. However, due to the limitation of the track structure, the QCs cannot pass each other. Since different vessels cannot be served in the same berth at the same time, any two rectangles in the figure cannot overlap.

The objectives of this paper are to obtain the optimal berth allocation and QC assignment theme, so as to minimize the port stay times and delayed departure time of vessels.



- The cranes of vessel i
- The rightmost crane number assigned to vessel i (Rc_i)
- The leftmost crane number assigned to vessel i (Lc_i)

Figure 1. Diagram of the BACASP.

For BACASP, the following assumptions are made in this paper.

- There is no limitation of vessel draft and water depth.
- Since the handling time of the QC is much greater than its movement time, the QC’s movement time is negligible.
- The QC assigned to a vessel cannot serve other vessels during the berthing period of the vessel.
- The safety distance between vessels is included in the length of the vessels.

2.1. Notation

Table 1 lists the variables used in our model for the BACASP along with their types and descriptions.

Table 1. Variables used in model. - is a dimensionless value.

| Variables | Type | Definition | Unit |
|--------------|---------|--|------|
| s_i | Integer | Berthing start time of vessel $\forall i \in N$ | h |
| p_i | Integer | Berthing position of vessel $\forall i \in N$ | m |
| c_i | Integer | The number of QCs serving vessel $\forall i \in N$ | - |
| Lc_i | Integer | The leftmost QC of the service vessel $\forall i \in N$ | - |
| Rc_i | Integer | The rightmost QC of the service vessel $\forall i \in N$ | - |
| h_i | Integer | The handling time of vessel $\forall i \in N$ | h |
| e_i | Integer | Departure time of vessel $\forall i \in N$ | h |
| x_{ij} | Binary | Set to 1 if the berthing start time of vessel $\forall j \in N$ is later than the departure time of vessel $\forall i \in N$, 0 otherwise | - |
| y_{ij} | Binary | Set to 1 if vessel $\forall i \in N$ is fully berthed on the left side of vessel $\forall j \in N$, 0 otherwise | - |
| Δb_i | Integer | Deviation between the desired and the berthing position of vessel $\forall i \in N$ | m |

2.2. Model of BACASP

This section presents a model for the BACASP which is extended from Türkoğulları et al. (2014). In paper [22], the model assumes that the handling time of vessels is only related to the number of QCs. To be more realistic, we also consider the impacts of berthing

preference and interference between QCs on the handling time of vessels. The objective and constraints of the model are as follows:

$$\text{Min } \sum_{i \in N} \{ (e_i - a_i) + (e_i - d_i)^+ \} \tag{1}$$

s.t.

$$c_i^\alpha h_i \geq (1 + \Delta b_i \beta) m_i, \forall i \in N \tag{2}$$

$$e_i = s_i + h_i, \forall i \in N \tag{3}$$

$$\Delta b_i \geq p_i - b_i, \forall i \in N \tag{4}$$

$$\Delta b_i \geq b_i - p_i, \forall i \in N \tag{5}$$

$$e_i \leq s_j + M(1 - x_{ij}), \forall i, j \in N, i \neq j \tag{6}$$

$$p_i + l_i \leq p_j + M(1 - y_{ij}), \forall i, j \in N, i \neq j \tag{7}$$

$$x_{ij} + x_{ji} + y_{ij} + y_{ji} \geq 1, \forall i, j \in N, i \neq j \tag{8}$$

$$Lc_i \leq Rc_i, \forall i \in N \tag{9}$$

$$c_i = Rc_i - Lc_i + 1, \forall i \in N \tag{10}$$

$$Rc_i \leq Lc_j + M(1 - y_{ij}) + Mx_{ij} + Mx_{ji}, \forall i, j \in N, i \neq j \tag{11}$$

$$s_i \geq a_i, \forall i \in N \tag{12}$$

$$p_i + l_i \leq L, \forall i \in N \tag{13}$$

$$c_i \leq q_i^{max}, \forall i \in N \tag{14}$$

$$c_i \geq q_i^{min}, \forall i \in N \tag{15}$$

$$Rc_i \leq g, \forall i \in N \tag{16}$$

$$x_{ij}, y_{ij} \in \{0, 1\}, \forall i, j \in N \tag{17}$$

$$s_i, e_i, p_i, c_i, Rc_i, Lc_i \in \{0, 1, \dots, \infty\}, \forall i \in N \tag{18}$$

The objective function (1) minimizes the port stay time and the delayed departure time, whose component for each vessel are: (i) the waiting time and handling time of vessels, (ii) the delayed departure time when the vessel is delayed to leave the port.

Constraint set (2) ensures that every vessel receives required QC hours considering productivity losses by QC interference and the chosen berthing position. It is worth noting that the constraint (2) in this paper is different from the constraint in [22]. They assume that the handling time of vessels is only related to the number of QCs. Considering the realistic scenario, the model also considers the impacts of berthing preference and productivity lost by QC interference. Constraint set (3) describes that the departure time of vessel is determined by the berthing start time and the handling time of vessel. Constraints (4) and (5) determine the deviations of the berthing position from the desired berthing position. Constraints (6)–(8) prevent any two vessels from overlapping in the space-time-diagram. Constraints (9) and (10) indicate the relationship of the specific number of QCs assigned to the same vessel. Constraint (11) means that for any two ships being served at the same time, the assigned QCs cannot cross each other. Constraints (12) indicate the vessels berthing after their arrival time. Constraints (13)–(18) are types and boundaries of each variable.

3. Methodology

This section describes the solution method for the BACASP. The framework of solution method is presented in Section 3.1. The main techniques of the method are described in Section 3.2.

3.1. Framework of Solution Method

LNS is a neighborhood search algorithm with strong global search ability, which is widely used in solving discrete optimization problems, and has achieved excellent performance in solving discrete problems [36,37]. In this study, a discretization strategy [6] is applied to divide the continuous berth into discrete segments, which transforms the BACASP into a discrete combinatorial optimization problem, and is further solved by the LNS algorithm. The frame of the LNS is shown in Algorithm 1.

The BACASP optimal solution needs to determine the order of services at the berths and the QCs allocated to the vessels. Let K be the set of berths after discretization and $|K|$ denote the total number of berths (line 1 in Algorithm 1). For a particular berth k , let P_k^s and P_k^f be the left and right boundaries of berth k , respectively. We define a solution structure as follows in function (19):

$$X = \begin{pmatrix} S_1(B_1) & S_1(B_2) & \cdots & S_1(B_{n1}) \\ S_2(B_1) & S_2(B_2) & \cdots & S_2(B_{n2}) \\ \vdots & \vdots & \vdots & \vdots \\ S_{|K|}(B_1) & S_{|K|}(B_2) & \cdots & S_{|K|}(B_{n|K|}) \end{pmatrix} \tag{19}$$

The columns of the solution structure represent the berth, and the rows represent the service order of vessels. X_0 , X_{now} , X_{accept} and X_{best} are the initial solution, the current solution, the accepted solution and the best solution, respectively. Set term $S_k(B_u)$ denotes the u -th served vessel on berth k and the number of QCs assigned to the vessel. Hitherto, we prove that the optimal berthing position, berthing start time and number of cranes assigned to each vessel can be obtained as long as the solution structure $S_k(B_u)$ is determined.

The berthing start time s_{B_1} of the first vessel on the berth is set as the arrival time of the vessel, and s_{B_u} of subsequent vessels must be later than the departure time of the previous one. The berthing start time of the vessel on berth k is:

$$s_{B_u} = \begin{cases} a_{B_u}, & u = 1 \\ \max(a_{B_u}, a_{B_{u-1}} + h_{B_{u-1}}), & 1 < u \leq nk \end{cases} \tag{20}$$

Destroy operators remove τ vessels from the current solution structure (line 5 in Algorithm 1). τ is a random value in $[1, N \cdot \Phi]$ where φ controls the number of vessels to be removed (%). A new solution structure can be obtained by implementing the destroy (Z^-) and repair (R^+) operations (line 6 in Algorithm 1). There is conflict among vessels on different berths due to the continuity of the berths. The BCB strategy is used to adjust the infeasible solutions under the berthing sequence to form new solutions (line 7 in Algorithm 1). Afterwards, the QC allocation algorithm proposed by [22] is adopted to achieve optimal QC assignment results (line 8 in Algorithm 1). The basic idea is to first determine the number of QCs allocated to the vessels, and then the specific QCs are allocated in the order from left to right of the berthing position of the vessels. These processes are repeated until the stop criteria is met. The details of destroy and repair operators in Algorithm 1 are further discussed in the following subsections.

Algorithm 1: Framework of the LNS algorithm.

Input: Parameters of LNS
Output: X_{best}

- 1 Determine the number of berth segments $|K|$, and P_k^s, P_k^f
- 2 Generate an initial solution X_0
- 3 $X_{best} \leftarrow X_0, X_{accept} \leftarrow X_0, X_{now} \leftarrow X_0$
- 4 **While** the stop criteria are not met **do**
- 5 Determine $\tau = Random[1, N \cdot \Phi]$
- 6 $X_{now} = R^+(Z^-(X_{now}))$
- 7 Adjust X_{now} using the strategy BCB
- 8 Cranes assignment algorithm
- 9 Acceptance criteria for solution
- 10 **End while**

3.2. Main Technique

3.2.1. Discretization Strategy

The discretization strategy was originally proposed by [6] to solve the BAP. This work extends the discretization strategy solution of BAP to address and solve the BACASP. The discretization strategy we proposed is designed to divide the continuous berth into discrete segments, and convert the BACASP to a discrete combinatorial optimization problem, which is efficiently solved by the proposed LNS. Figure 2a is a schematic diagram of the discretized berths. The short solid line divides the berths and the dotted line represents the berth boundary, indicating such as $k, k - 1$, and $k + 1$. Each berth shares part of the adjacent berths.

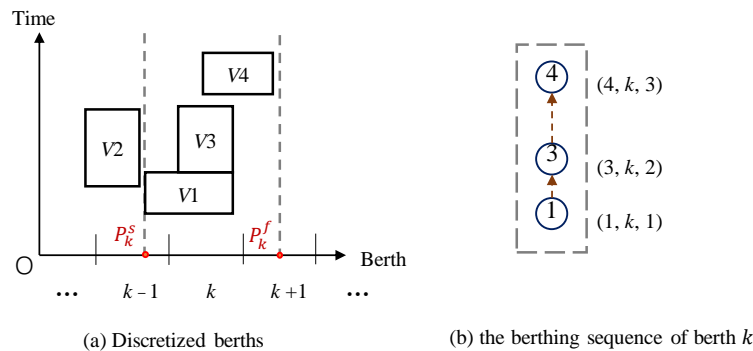


Figure 2. Diagram of discrete berths and berthing order. (a) Description of the discretized berths. (b) Description of the berthing sequence of berth k .

Take berth k as an example. Berth k occupies the right half of the left berth and the left half of the right berth. The discretization strategy divides the berth segment, and the continuous docking of vessels has no effect. Figure 2b is the berthing sequence of berth k .

The discretization strategy has an impact on the final solution. There are some limitations in discretization strategy. The relationship between the length of discrete berths and vessels affects the scheduling solutions. If the discrete berth is too long, the occupancy rate of the berth will be reduced. If the discrete berth is too short, the discretization strategy will caused a fragmentation of the decision space. All berth segments should allow all types of vessels to dock during the course of the algorithm. Since the berths share a part of the adjacent berths, vessels allocated to different berths may violate the space constraints. Therefore, a constraint-handling strategy is proposed in this paper to solve this problem.

3.2.2. Destroy Operators

(1) Related destroy operator

The related destroy operator was first proposed in [38] for vehicle routing problem to remove similar nodes. In this study, the degree of similarity between vessel i and j is computed through a relatedness function $R(i, j)$, where a higher value corresponds to less similar vessels. Since the arrival time and berthing position of the vessel have a greater impact on the objective value, the following Equation (21) is constructed.

$$R(i, j) = \delta \cdot |a_i - a_j| + \eta \cdot |p_i - p_j| \tag{21}$$

where δ and η are the correlation coefficients of the arrival time and berthing position of the vessel, respectively. The operator randomly destroys a vessel i , calculates the $R(i, i')$ ($i' \in$ current solution X), and destroys vessel j with the maximum value of the relatedness function. This procedure is repeated until Φ vessels are removed.

(2) Random destroy operator

The random destroy operator removes τ vessels randomly from the current solution. This operator expands the range of neighborhood search and avoids the LNS sitting at the local optimum.

3.2.3. Repair Operators

(1) Deep greedy repair operator

The deep greedy repair operator was proposed by [39] for solving vehicle routing problems. In this study, we propose an operator suitable for this problem based on the idea of deep greedy repair operator. The deep greedy repair operator intends to repair the vessels to the positions with minimal objective value increment among all feasible positions. Let R_p be the set of vessels that have been destroyed and X_{+i} be the solution with vessel i reinserted. Let $O_b(X)$ be the objective value of current solution X . The implementation of this operator with respect to the BACASP is described in the following three steps. Firstly, examine whether set R_p is empty and where the vessel (that belongs to R_p) can be repaired. For each vessel $i \in R_p$, the deep greedy repair operator calculates and records its best insertion position $p_{i,1}$ with the lowest cost $O_b^1(X_{+i}, p_{i,1})$ (line 2 in Algorithm 2). Afterwards, vessel $j = \operatorname{argmin}_{j \in R_p} O_b^1(X_{+j}, p_{j,1})$ is inserted into its best insertion position $p_{j,1}$ of X . This procedure is repeated until R_p is empty.

Algorithm 2: Deep greedy repair operator.

Input: The set of vessels removed R_p
Output: The current solution X

- 1 **While** $R_p \neq \emptyset$ **do**
- 2 Calculate $O_b^1(X_{+i}, p_{i,1}), O_b^2(X_{+i}, p_{i,2}), \dots$ of structure X_{+i} with vessel i inserted to position $p_{i,l}$
- 3 Insert vessel $j = \operatorname{argmin}_{j \in R_p} O_b^1(X_{+j}, p_{j,1})$ to its best insertion position $p_{j,1}$ into X
- 4 Remove vessel j from R_p
- 5 **End while**

(2) Slack sorted repair operator

Mauri et al. [35] proposed the sorted greedy operator, in which vessels berthing in the same berth can only be served in a certain order, which may lead the search trapped in local optimum. As such, we propose the slack sorted repair operator considering the QC capacity demand and the remaining time to the due time of vessel. This operator first sorts the vessels in set R_p according to the slack value V_i in ascending order. Then, given

the current solution, the operator repairs the first vessel into its best position in every berth. We define V_i as follows:

$$V_i = m_i / (d_i - a_i) \tag{22}$$

where m_i is required QC capacity of vessel i , and $d_i - a_i$ is the difference between the latest departure time and the arrival time of vessel. The handling time of the vessel is related to m_i , and a larger m_i indicates that the vessel needs longer time to departure.

3.2.4. BCB Strategy

Due to the limitations of the discretization strategy, the vessel allocation to these positions may not always satisfy the spatial constraints when the distances between the berthing positions are too small. Therefore, the BCB strategy is proposed in this paper to solve this problem.

We use LNS to solve the model combining the constraint handling strategy, which include the BCB strategy. The BCB strategy can solve the infeasible solutions caused by discretization strategy and reduce the waste of berths. Let E_k be the set of unadjusted vessels on berth k . Let k' be the berth on the left of berth k . The specific steps of BCB strategy are as follows:

- Step 1: Select the leftmost berth of the coastline and mark it as k .
- Step 2: If $E_k \neq \emptyset$, continue; Otherwise, go to Step 7.
- Step 3: Find the first vessel $i \in E_k$ and determine the berthing position p_i of vessel i .
- Step 4: If vessel i overlaps with vessel j ($j \in E_k$ or $j \in E_{k'}$, and $j \neq i$), continue; otherwise, go to Step 6.
- Step 5: Eliminate the overlap of vessel i with j and return to Step 4.
- Step 6: Remove vessel i from E_k and return to Step 2.
- Step 7: If berth k is the rightmost berth of the coastline, stop; Otherwise, Continue.
- Step 8: $k = k + 1$. Return to Step 2.

Figure 3 shows the setting method of berthing position. The berthing position is determined in Step 3. Assume that Vessel 1 is at the berth k and b_1 is the best berthing position of vessel 1. If $b_1 < P_k^s$, then $p_1 = P_k^s$, as shown in Figure 3a. If $P_k^s \leq b_1 \leq P_k^f$ and $b_1 + l_1 \leq P_k^f$, then $p_1 = b_1$, as shown in Figure 3b. Otherwise, $p_1 = P_k^f - l_1$, as shown in Figure 3c.

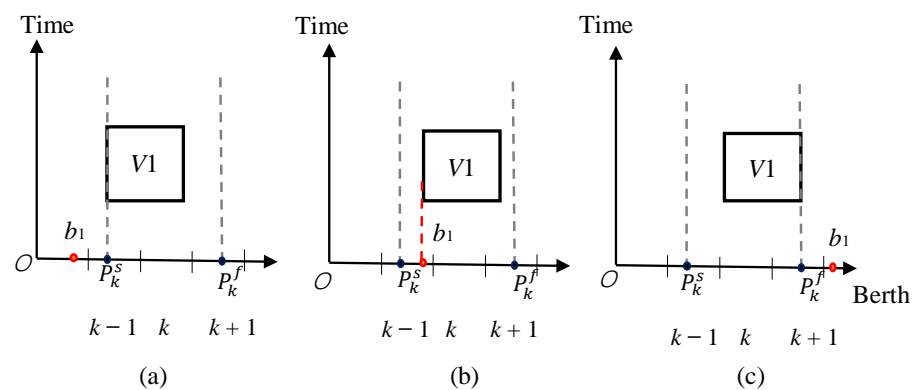


Figure 3. The setting method of berthing position.

The overlap can be eliminated by delaying the berthing start time of the vessel and pushing back the berthing position of the vessel, and the influence of the two methods on the value of the objective function is calculated. Finally, the BCB strategy compares the objective value increments and chooses a way to make the increments smaller to eliminate the overlap.

3.2.5. Acceptance Criteria

With respect to the acceptance criteria (line 9 in Algorithm 1), a commonly used acceptance criteria (line 9 in Algorithm 1) in SA is applied in this paper [35]. Specifically, for a new solution X' , a neighbor solution X with smaller objective value ($O_b(X') < O_b(X)$) is accepted, and with probability $e^{(O_b(X) - O_b(X'))/T}$ otherwise. The temperature T starts at T_{start} and decreased by ΔT in each iteration and resets to T_{start} when the freezing temperature T_{end} is reached.

4. Numerical Experiments

4.1. Generation of Instances

Two sets of problem instances are employed in this study for computational experiments. The first set of instances is the real data of 21 vessels reported by [22] at the Tianjin Five Continents International Container Terminal in Tianjin during a 200-h planning period. The terminal has a berth 1200 m long with 12 QCs. This set consists of 7 instances with different scales, denoted as $i01-i07$, and the numbers of vessels are 3, 6, 9, 12, 15, 18, and 21, respectively.

At present, there is no public large-scale example set of the BACASP. The second set of instances include vessels following the generation rules proposed by Türkoğulları et al. [22]. We generate the number of vessels N from 24 to 60. The length l_i and the QC capacity demand m_i of each vessel i are generated from uniform distribution $U(3, 8)$ and $U(10, 120)$. The desired berthing position of vessel is generated by uniform distributions $U(1, L - l_i + 1)$. The length of the vessel directly determines q_i^{min} and q_i^{max} . Vessels with length of 3, 4, 5, 6, 7, and 8 correspond to their q_i^{min} and q_i^{max} in the range $[2, 2]$, $[2, 3]$, $[2, 4]$, $[3, 5]$, $[3, 6]$, and $[3, 7]$. The due time is generated according to $s_i + \lceil m_i/q_i^{max} \rceil + U(5, 15)$ where $\lceil m_i/q_i^{max} \rceil$ is rounded up to the nearest integer (m_i/q_i^{max}). Table 2 shows the means and variances of the Parameters of the sample.

Table 2. The means and variances of the Parameter of the sample.

| Parameter | l_i | m_i | q_i^{min} | q_i^{max} | b_i | a_i | d_i |
|-----------|-------|--------|-------------|-------------|-------|----------|----------|
| Mean | 5.4 | 66.4 | 2.4 | 4.4 | 10.6 | 294.7 | 320.6 |
| Variance | 1.9 | 1119.6 | 0.2 | 1.9 | 30.3 | 26,827.8 | 26,934.1 |

All the experimental study is coded in C language and carried out on a computer with a 3.00 GHz CPU and 8 GB RAM. For each instance, the analysis is run for 20 consecutive times.

4.2. Parameter Setting

All parameters are used within the appropriate range, and extensive experiments based on the orthogonal design have been performed on different parameter combinations. Table 3 shows the parameters of the LNS algorithm, which include α , β , Φ , Nt , T_{start} , T_{end} and ΔT . When the parameter changes within a range, and the most appropriate quantity is selected according to the influence of the variation of the parameter combination on the solution. Afterwards, a certain parameter combination is selected based on the best average result of the testing instances. All experiments are run for 20 consecutive times with different randomness seeds with the same parameter combination, as shown in Table 3.

Table 3. Parameters of the LNS algorithm.

| Parameter | Description | Range | Tuned Value |
|-------------|--------------------------------------|----------------------|-------------|
| α | Interference exponent for the cranes | (0.9, 0.8) | 0.9 |
| β | Berth deviation factor | (0.01, 0.02) | 0.01 |
| Φ | The number of vessels to be removed | (0.3, 0.35, 0.4) | 0.3 |
| Nt | Number of iterations | (2000, 5000, 10,000) | 5000 |
| T_{start} | Start temperature | 1000 | 1000 |
| T_{end} | Freezing temperature | 0.01 | 0.01 |
| ΔT | Cooling rate | 0.975 | 0.975 |

4.3. Computational Results on the First Set of Instances

4.3.1. Analysis of Operators

To illustrate the performance of slack sorted repair operator, we increase pairwise combinations of different removal and insertion operators and the results are shown in Figures 4 and 5. Figure 4 is the average and the best objective values of the solution obtained by different combinations of destroy–repair operators. Figure 5 depicts the standard deviation and CPU time of the solution obtained by different combinations of destroy–repair operators. Different operators are represented by the related destroy operator (Re), random destroy operator (Ra), deep greedy repair operator (Gr) and slack sorted repair operator (Sl). Figure 4 shows that the Ra-Sl combination obtains the best average and best values in most cases. Figure 5 The standard deviation shows an increasing trend with the increase of vessels’ size. In general, the value of the standard deviation of the Re-Gr combination is the largest, and the value of the standard deviation of the Ra-Sl combination is the smallest. The CPU time in Figure 5 shows that the combinations using Sl operator have shorter running time than the combinations using Gr operator. For scenario *i07*, the CPU time of the Re-Gr operator is 79.6 s, and the CPU time of the Re-Sl operator is 31.9 s. Compared with the Gr operator, the Sl operator saves $((79.6 - 31.9)/79.6) \times 100\% = 59.9\%$ of the computing time. The reason is that the Sl operator can obtain the insertion order of the vessels by calculating the slack variables of the removed vessels. The Gr operator calculates the insertion cost of the removed vessels, and only obtains the currently best inserted vessels, and then continues the above calculation process until the insertion sequence of all vessels is completed.

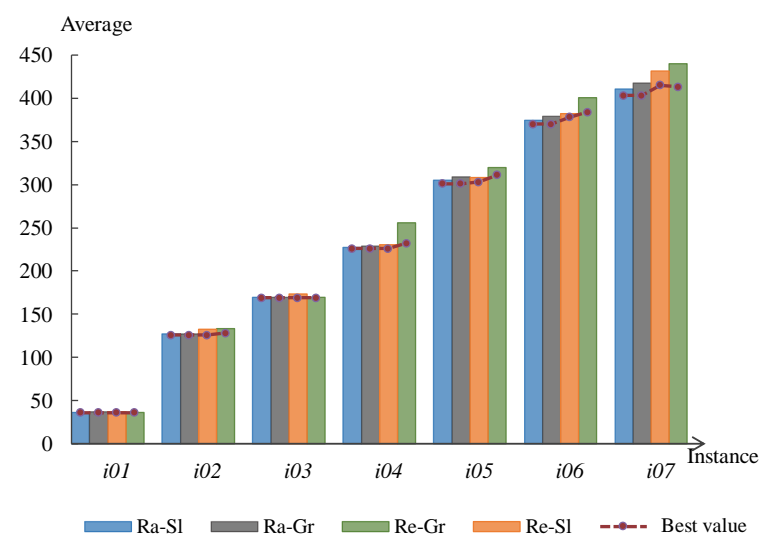


Figure 4. Average and the best objective values obtained by different combinations of destroy–repair operators.

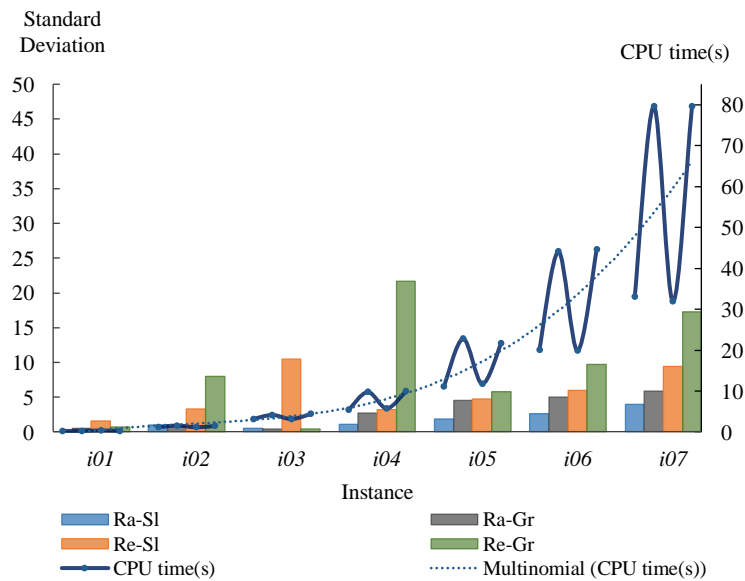


Figure 5. Standard deviation and CPU times obtained by different combinations of destroy-repair operators.

In Figure 4, the difference among the best solutions obtained by different operator combinations is very small. To further investigate the difference between operator combinations, we implement them, respectively, based on the observed value obtained by wilcoxon rank sum (WRS) test [40]. At the significance level of 5%, this method detects whether there is a significant difference between the two groups. If yes, it indicates the bilateral detection probability $p < 0.05$, rejects the null hypothesis, and the observed value h returns 1; If not, 0 is returned. Table 4 shows the WRS test of different combinatorial operators, where WRS_1 , WRS_2 and WRS_3 are the rank sum statistics of operators Ra-Sl and Ra-Gr, Re-Sl and Re-Gr, respectively. The results of Ra-Sl and Ra-Gr are significantly different in more than half of the examples, the results of Ra-Sl and Re-Sl are significantly different in all examples, and the results of Ra-Sl and Re-Gr are significantly different in most examples.

Table 4. The WRS test of different combinatorial operators.

| Instances | WRS ₁ | | WRS ₂ | | WRS ₃ | |
|-----------|-----------------------|---|-----------------------|---|-----------------------|---|
| | p | h | p | h | p | h |
| i01 | 3.11×10^{-4} | 1 | 1.08×10^{-2} | 1 | 5.88×10^{-2} | 0 |
| i02 | 7.39×10^{-1} | 0 | 9.90×10^{-5} | 1 | 2.54×10^{-4} | 1 |
| i03 | 5.78×10^{-2} | 0 | 1.83×10^{-2} | 1 | 1.32×10^{-1} | 0 |
| i04 | 5.18×10^{-2} | 0 | 6.24×10^{-4} | 1 | 8.70×10^{-5} | 1 |
| i05 | 4.77×10^{-3} | 1 | 9.96×10^{-3} | 1 | 8.70×10^{-5} | 1 |
| i06 | 6.38×10^{-3} | 1 | 8.60×10^{-5} | 1 | 8.80×10^{-5} | 1 |
| i07 | 3.48×10^{-3} | 1 | 1.31×10^{-4} | 1 | 1.03×10^{-4} | 1 |

4.3.2. Analysis of BCB Strategy

Figure 6 shows the comparison results obtained under the BCB strategy and time-adjustment (TA) strategy, where the BCB strategy performs better in mean and variance overall. The time-adjustment strategy is lack of flexibility and cannot get better results in the large or complex scale instances. The BCB strategy combines adjustment of berthing start time and berthing position to ensure the stability and efficiency of the algorithm.

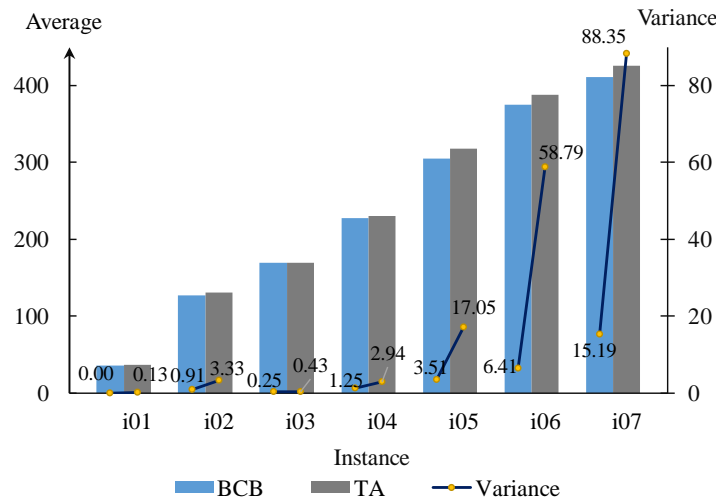


Figure 6. The results of BCB and time adjustment strategy.

4.3.3. Analysis of LNS Algorithm

In order to confirm the merit of the proposed LNS for solving the BACASP, the results obtained by the LNS are compared with those obtained in Türkoğulları et al. [22], where the necessary and sufficient condition is proposed for generating an optimal solution of the BACASP, which is obtained by commercial solver CPLEX 12.2. In case this condition is not satisfied, the cutting plane algorithm is used until the condition holds.

The handling time of vessels in Türkoğulları et al. [22] is obtained by dividing the QC capacity demand of each vessel according to the number of cranes assigned, which is rounded up to the nearest integer. The costs attracted by vessels’ deviations from their best berthing positions and the impacts of interference among QCs have not been included in the handling time. Instead, the offset of the berthing position is considered in the objective function. To compare with the quality of solutions presented in Türkoğulları et al. [22], the handling time and objective in this set of instances are set to be the same as those in Türkoğulları et al. [22]. Specifically, we set parameters α and β in constraint (2) as 1 and 0, respectively, and the objective function then written as:

$$\min \sum_{i=0}^n \{ \gamma_1 |s_i - a_i| + \gamma_2 |p_i - b_i| + \gamma_3 \max(0, a_i + h_i - 1 - d_i) \} \quad (23)$$

The objective function contains the cost of waiting time of berthing, deviation from the desired berthing position and delayed departures of vessels. The corresponding coefficients γ_1 , γ_2 , and γ_3 are 1000, 1000 and 2000, respectively.

Table 5 shows the comparison results of the LNS algorithm and CPLEX solver for solving the BACASP. The “Best”, “Average” and “RT” in Table 5 denote the best objective value, average objective value and the average CPU time for 20 consecutive runs obtained by the LNS. The table shows that the LNS can solve all instances to optimality. The results also infer that the proposed LNS has good stability. For the instances *i06* and *i07*, the method proposed in Türkoğulları et al. [22] is not able to obtain a feasible solution of the BACASP within the specified time (76,110 s) by CPLEX solver. As a result, the BACASP is relaxed to a BACAP, which is optimally solved by CPLEX. Afterwards, Türkoğulları et al. [22] checked the optimal solutions of the relaxed BACAP in *i06* and *i07*, and found the solutions in aforementioned condition. In contrast, the proposed LNS can find the optimal solutions for all instances without relaxation within 41 s.

Table 5. Comparison results of the LNS algorithm and CPLEX solver for solving the BACASP.

| Instance | Türkoğulları et al. [22] | | | LNS | | |
|----------|--------------------------|-------------|-------------------------------|--------|---------|-------|
| | Best | CPLEX RT(s) | Cutting Plane Algorithm RT(s) | Best | Average | RT(s) |
| i01 | 2000 | 372 | 11.3 | 2000 | 2000 | 0.55 |
| i02 | 21,000 | 53,689 | 35.3 | 21,000 | 21,000 | 2.05 |
| i03 | 21,000 | 21,383 | 40.3 | 21,000 | 21,000 | 4.85 |
| i04 | 21,000 | 57,818 | 44.9 | 21,000 | 21,000 | 8.40 |
| i05 | 35,000 | 76,110 | 83.5 | 35,000 | 40,750 | 15.35 |
| i06 | 43,000 | – | 105.3 | 43,000 | 44,500 | 25.60 |
| i07 | 43,000 | – | 89.3 | 43,000 | 45,200 | 40.25 |

4.4. Computational Results on the Second Set of Instances

Due to the complexity of BACASP, exact methods are difficult to find the optimal in the efficient time, especially for dealing with a large-scale problem. Traditionally, BACASP is regarded as a continuous and discrete hybrid optimization problem, which is solved by the evolutionary algorithms, such as GA [33]. The genetic algorithm encodes the solution, interacts with the encoded information of different solutions, and solves the problem by combining the constraint processing method. Among them, the repair method (GA-RM) and the penalty function (GA-PF) strategy are used widely. GA-RM delays the berthing start time of overlapping vessels to get a feasible solution. The GA-PF constructs a fitness function that balances the objective function and constraints violation to obtain a feasible solution.

In order to further verify the discretization strategy and the performance of LNS algorithm in solving large-scale BACASP, the GA-RM, GA-PF and LNS algorithms are tested with the second set of examples, respectively. The results of the two algorithms are summarized in Table 6, where “BE”, “ME”, “SD” and “FS” denote the best objective value, the mean, the standard deviation of the objective value and the times feasible solutions are obtained in the 20 runs, respectively. When the scale of vessels is less than 40, GA-PF can obtain better solutions than GA-RM. When the number of vessels reaches 40, the quality of the solution obtained by GA-PF decreases significantly. The “SD” of GA-PF is the largest among all instances. The GA-PF has the advantage of high flexibility but disadvantage of difficulties of the parameters of the objective and the penalty. The penalty function of the GA-PF cannot effectively solve large-scale cases. The GA-RM has high feasible solution search efficiency, but there is a large systematic deviation in the search process. The average solution, the best solution and the standard deviation of the objective function obtained by the LNS algorithm are better than those obtained by GA-RM and GA-PF. Although this paper considers a few factors such as QCs interference and berthing deviation in the handling time, the LNS algorithm satisfies all constraints and obtains a feasible solution in each operation. In comparison, GA-PF cannot often obtain feasible solutions while the proposed method can.

Table 6. The results of LNS algorithm and GA.

| Instance | N | LNS | | | | GA-RM | | | | GA-PF | | | |
|----------|----|------|--------|------|----|-------|--------|------|----|-------|--------|-------|----|
| | | BE | ME | SD | FS | BE | ME | SD | FS | BE | ME | SD | FS |
| i08 | 24 | 375 | 379.0 | 3.8 | 20 | 547 | 581.4 | 30.9 | 20 | 504 | 692.5 | 120.6 | 16 |
| i09 | 28 | 493 | 497.8 | 4.1 | 20 | 739 | 790.7 | 33.4 | 20 | 573 | 822.4 | 133.6 | 13 |
| i10 | 32 | 563 | 570.7 | 4.1 | 20 | 855 | 888.7 | 36.2 | 20 | 676 | 897.6 | 133.5 | 16 |
| i11 | 36 | 618 | 626.5 | 5.1 | 20 | 922 | 1008.6 | 80.2 | 20 | 790 | 936.5 | 137.0 | 9 |
| i12 | 40 | 898 | 913.2 | 6.3 | 20 | 980 | 1060.2 | 63.0 | 20 | 1774 | 2278.4 | 308.3 | 10 |
| i13 | 50 | 1239 | 1278.8 | 17.9 | 20 | 1448 | 1501.2 | 42.0 | 20 | 1981 | 2673.3 | 550.1 | 19 |
| i14 | 60 | 1482 | 1535.8 | 24.7 | 20 | 1898 | 1993.2 | 71.9 | 20 | 2073 | 3249.6 | 735.3 | 10 |

The BACASP allocation scheme obtained by the above method can reflect the stay times of vessels, delayed departure times of vessels and the berth occupancy rate. The objective value of the model includes two parts: the total stay time and the delayed departure time of vessels. Taking the best solution obtained by LNS and GA-RM as an example. Table 7 shows the statistics of the vessel’s stay time in port and delayed departure time, as well as the difference of stay time Δst and the difference of delayed departure time Δdt . The LNS algorithm can effectively reduce the stay time of vessels in port for 6 out of 7 instances. Furthermore, the LNS algorithm reduces the delayed departure time of vessels in all cases. In instance *i08*, the total times at port and delayed times of the 24 vessels decreased by 117 h and 72 h, respectively.

Table 7. The statistical results of stay time and delayed departure time of vessels.

| Instance | N | LNS | | | GA-RM | | | Δst | Δdt |
|------------|----|-----------|------------------------|--------|-----------|------------------------|-------|-------------|-------------|
| | | Stay Time | Delayed Departure Time | RT(s) | Stay Time | Delayed Departure Time | RT(s) | | |
| <i>i08</i> | 24 | 362 | 13 | 51.55 | 479 | 85 | 82.3 | 117 | 72 |
| <i>i09</i> | 28 | 480 | 13 | 84.1 | 626 | 113 | 127.1 | 146 | 100 |
| <i>i10</i> | 32 | 550 | 13 | 134.25 | 722 | 133 | 165.6 | 172 | 120 |
| <i>i11</i> | 36 | 605 | 13 | 213.9 | 783 | 139 | 232.1 | 178 | 126 |
| <i>i12</i> | 40 | 883 | 15 | 295.8 | 844 | 136 | 293.7 | −39 | 121 |
| <i>i13</i> | 50 | 1167 | 72 | 736.15 | 1191 | 257 | 555.2 | 24 | 185 |
| <i>i14</i> | 60 | 1406 | 76 | 1500.8 | 1535 | 363 | 876.9 | 129 | 287 |

Berth occupancy rate reflects the degree of occupancy of vessels at berths. The calculation of the berth occupancy rate in this paper is as follows:

$$BOR = \frac{\sum_{i=0}^n (Ocl_i \times Oct_i)}{L \times T} \tag{24}$$

where *BOR* is berth occupancy rate. *Ocl_i* and *Oct_i* are the lengths of berths occupied by vessel *i* and occupied time of vessel *i*. *L* is the length of berth and *T* is the planning period.

The berth occupancy rate can be derived from information such as berthing time, start time, vessel length and handling time of the solution. As shown in Figure 7, the LNS algorithm can effectively improve the berth occupancy rate. Due to the limited resources of continuous berths and QCs, there are fluctuations in berth occupancy. Figure 8 shows the solution obtained by the LNS for the instance *i13*. It can be seen that the resources of the port are effectively utilized.

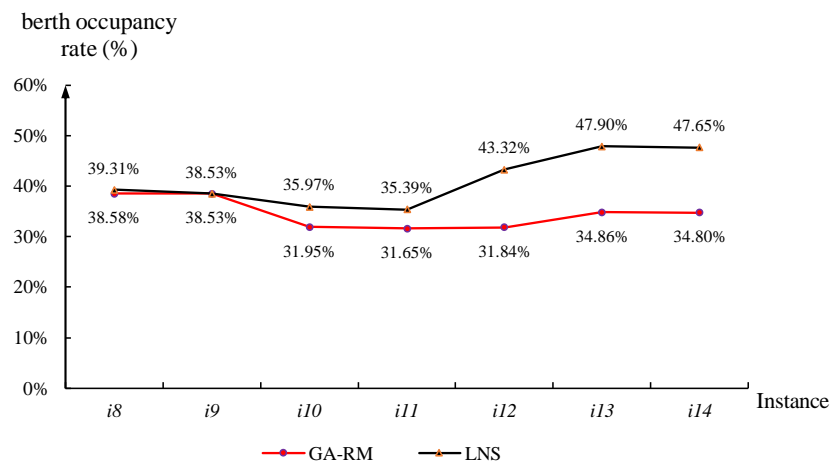


Figure 7. Comparison of the berth occupancy rate of the LNS and GA-RM.

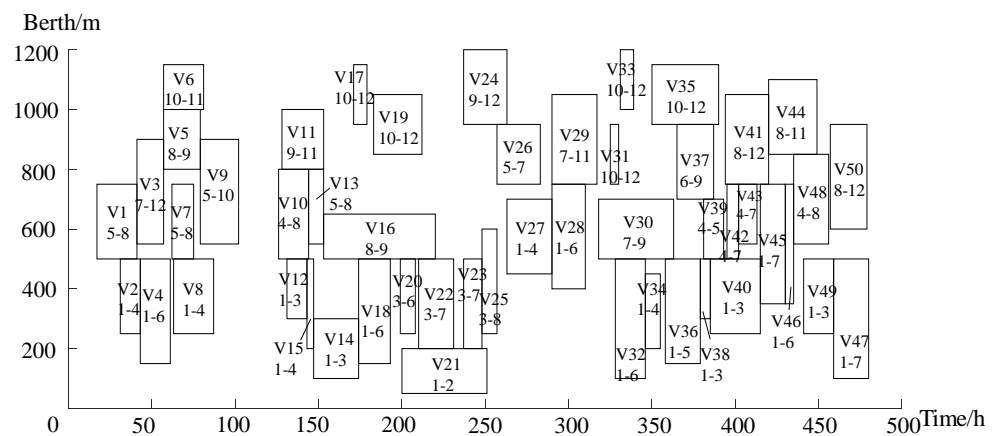


Figure 8. Best solution obtained by the LNS for instance *i13*.

5. Conclusions

From a practical point of view, the BACASP allocation solution solved in this paper realizes the full utilization of port berth and QC resources on the premise of ensuring the service level and idleness reduction, which is conducive to avoiding ship congestion, improving the competitiveness of the port and exerting the maximum economic benefits for ports.

In this paper, the BACASP considers the non-crossing constraints of QCs, the mutual interference between QCs and the impact of berthing position on handling time. The final solution meets all constraints, and the allocation scheme is close to real-time operations.

The effect of different operator combinations has significant differences in solving the BACASP within the LNS algorithm in this work. Compared with the deep greedy repair operator, slack sorted repair operator can obtain higher-quality solutions in a shorter time. From the CPU time with different operator combinations in Section 4.3.1. The slack sorted repair operator saves 59.9% of the computing time. In this paper, the BACASP is transformed into a discrete optimization problem by a new discretization strategy. The effective search of solution can be obtained by the LNS with the BCB strategy. Compared with the traditional continuous optimization combined with GA, the proposed method is more effective and can ensure the efficiency and stability of the solution. In addition, the berth and quay crane allocation scheme obtained by the LNS reduces the stay time of vessels and improves the occupancy rate of berth resources.

This study has opened up several directions for future research. First, this study mainly focuses on the BACASP in a certain environment, but there are some uncertain factors in port such as unscheduled vessels and breakdown of QCs, which deserves further investigations. Second, although the LNS has been confirmed to be able to achieve high-quality solutions for the BACASP, the gap between the obtained solution and the optimal solution is unknown. As such, it will be interesting to develop accurate and efficient exact methods, such as branch-and-price and branch-and-cut, to achieve optimal solutions for the BACASP.

Author Contributions: Conceptualization, M.T. and B.J.; methodology, M.T. and B.J.; software, M.T.; validation, M.T., B.J. and X.F.; formal analysis, B.J. and X.F.; investigation, M.T. and B.J.; resources, B.J.; data curation, M.T. and B.J.; writing—original draft preparation, M.T.; writing—review and editing, B.J., X.F. and S.S.Y.; visualization, M.T.; supervision, B.J. and S.S.Y.; project administration, B.J.; funding acquisition, B.J. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Natural Science Foundation of Hunan Province, China under Grant No. 2020JJ5780, and Central South University under Grant No. 202045007.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

| | |
|--------------|---|
| BACAP | Berth Allocation and Crane Assignment Problem |
| BACASP | Berth Allocation and Crane Assignment Specific Problem |
| BCB | Backtracking comparison-based |
| LNS | Large neighborhood search |
| Re | Related destroy operator |
| Ra | Random destroy operator |
| De | Deep greedy repair operator |
| Sl | Slack sorted repair operator |
| QC | Quay crane |
| N | Set of vessels, the number of vessels $n = N $ |
| G | Set of QCs, the number of quay cranes $g = G $ |
| L | Length of the quay given as a multitude of 1-m sections |
| a_i | Expected time of arrival of vessel $i \in N$ |
| l_i | Length of vessel $i \in N$ |
| d_i | Due time of vessel $i \in N$. |
| b_i | Desired berthing position of vessel $i \in N$ |
| m_i | Quay crane hours demand of vessel $i \in N$ |
| q_i^{\min} | Lower bound on the number of QCs that can serve vessel $i \in N$ simultaneously |
| q_i^{\max} | Upper bound on the number of QCs that can serve vessel $i \in N$ simultaneously |
| α | Interference exponent of cranes. |
| β | Berth deviation factor. |
| M | A large positive number |

References

- Global Shipping Has Been Hit by the Coronavirus. Now Goods Are Getting Stranded. Available online: <https://edition.cnn.com/2020/02/05/business/shipping-coronavirus-impact/index.html> (accessed on 26 March 2022).
- Lim, A. The berth planning problem. *Oper. Res. Lett.* **1998**, *22*, 105–110. [CrossRef]
- Bierwirth, C.; Meisel, F. A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. *Eur. J. Oper. Res.* **2015**, *244*, 675–689. [CrossRef]
- Nishimura, E.; Imai, A.; Papadimitriou, S. Berth allocation planning in the public berth system by genetic algorithms. *Eur. J. Oper. Res.* **2001**, *131*, 282–292. [CrossRef]
- Imai, A.; Sun, X.; Nishimura, E.; Papadimitriou, S. Berth allocation in a container port: Using a continuous location space approach. *Transp. Res. Part B Methodol.* **2005**, *39*, 199–221. [CrossRef]
- Cordeau, J.F.; Laporte, G.; Legato, P.; Moccia, L. Models and tabu search heuristics for the berth-allocation problem. *Transp. Sci.* **2005**, *39*, 526–538. [CrossRef]
- Tsai, A.H.; Lee, C.N.; Wu, J.S.; Chang, F.S. Novel wharf-based genetic algorithm for berth allocation planning. *Soft. Comput.* **2017**, *21*, 2897–2910. [CrossRef]
- Seyedalizadeh Ganji, S.R.; Babazadeh, A.; Arabshahi, N. Analysis of the continuous berth allocation problem in container ports using a genetic algorithm. *J. Mar. Sci. Technol.* **2010**, *15*, 408–416. [CrossRef]
- Frojan, P.; Correcher, J.F.; Alvarez-Valdes, R.; Koulouris, G.; Tamarit, J.M. The continuous Berth Allocation Problem in a container terminal with multiple quays. *Expert Syst. Appl.* **2015**, *42*, 7356–7366. [CrossRef]
- Mauri, G.R.; de Andrade, L.N.; Lorena, L.A.N. A Memetic Algorithm for a Continuous Case of the Berth Allocation Problem. *IJCCI* **2011**, *25*, 105–113.
- Cheong, C.Y.; Tan, K.C. A multi-objective multi-colony ant algorithm for solving the berth allocation problem. In *Advances of Computational Intelligence in Industrial Systems*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 333–350.
- Ting, C.J.; Wu, K.C.; Chou, H. Particle swarm optimization algorithm for the berth allocation problem. *Expert Syst. Appl.* **2014**, *41*, 1543–1550. [CrossRef]
- Tong, J.; Nachtmann, H. A tabu search approach to the cargo prioritisation and terminal allocation problem. *OR Spectr.* **2020**, *12*, 147–173.
- Şahin, C.; Kuvvetli, Y. Differential evolution based meta-heuristic algorithm for dynamic continuous berth allocation problem. *Appl. Math. Model.* **2016**, *40*, 10679–10688. [CrossRef]
- Park, Y.M.; Kim, K.H. A scheduling method for Berth and Quay cranes. *OR Spectr.* **2003**, *25*, 1–23. [CrossRef]

16. Meisel, F.; Bierwirth, C. Heuristics for the integration of crane productivity in the berth allocation problem. *Transp. Res. Part E Logist. Transp. Rev.* **2009**, *45*, 196–209. [[CrossRef](#)]
17. Rodriguez-Molins, M.; Salido, M.A.; Barber, F. A GRASP-based metaheuristic for the Berth Allocation Problem and the Quay Crane Assignment Problem by managing vessel cargo holds. *Appl. Intell.* **2003**, *40*, 273–290. [[CrossRef](#)]
18. He, J. Berth allocation and quay crane assignment in a container terminal for the trade-off between time-saving and energy-saving. *Adv. Eng. Inform.* **2016**, *30*, 390–405. [[CrossRef](#)]
19. Iris, Ç.; Pacino, D.; Ropke, S. Improved formulations and an adaptive large neighborhood search heuristic for the integrated berth allocation and quay crane assignment problem. *Transp. Res. Part E Logist. Transp. Rev.* **2017**, *105*, 123–147. [[CrossRef](#)]
20. He, J.; Wang, Y.; Tan, C.; Yu, H. Modeling berth allocation and quay crane assignment considering QC driver cost and operating efficiency. *Adv. Eng. Inform.* **2021**, *47*, 101252. [[CrossRef](#)]
21. Zhang, C.; Zheng, L.; Zhang, Z.; Shi, L.; Armstrong, A.J. The allocation of berths and quay cranes by using a sub-gradient optimization technique. *Comput. Ind. Eng.* **2010**, *58*, 40–50. [[CrossRef](#)]
22. Türkoğulları, Y.B.; Taşkın, Z.C.; Aras, N.; Altinel, İ.K. Optimal berth allocation and time-invariant quay crane assignment in container terminals. *Eur. J. Oper. Res.* **2014**, *235*, 88–101. [[CrossRef](#)]
23. Agra, A.; Oliveira, M. MIP approaches for the integrated berth allocation and quay crane assignment and scheduling problem. *Eur. J. Oper. Res.* **2018**, *264*, 138–148. [[CrossRef](#)]
24. Thanos, E.; Toffolo, T.; Santos, H.G.; Vancroonenburg, W.; Berghe, G.V. The tactical berth allocation problem with time-variant specific quay crane assignments. *Comput. Ind. Eng.* **2021**, *155*, 107168. [[CrossRef](#)]
25. Bouzekri, H.; Alpan, G.; Giard, V. Integrated Laycan and Berth Allocation and time-invariant Quay Crane Assignment Problem in tidal ports with multiple quays. *Eur. J. Oper. Res.* **2021**, *293*, 892–909. [[CrossRef](#)]
26. Ayvaz, D.; Topcuoglu, H.R.; Gurgun, F. Performance evaluation of evolutionary heuristics in dynamic environments. *Eur. J. Oper. Res.* **2012**, *37*, 130–144. [[CrossRef](#)]
27. Duan, J.; Liu, Y.; Zhang, Q.; Qin, J. Combined Configuration of Container Terminal Berth and Quay Crane considering Carbon Cost. *Math. Probl. Eng.* **2021**, *2021*, 6043846. [[CrossRef](#)]
28. Ji, B.; Yuan, X.; Yuan, Y. Modified NSGA-II for solving continuous berth allocation problem: Using multiobjective constraint-handling strategy. *IEEE Trans. Cybern.* **2017**, *47*, 2885–2895. [[CrossRef](#)]
29. Han, X.L.; Lu, Z.Q.; Xi, L.F. A proactive approach for simultaneous berth and quay crane scheduling problem with stochastic arrival and handling time. *Eur. J. Oper. Res.* **2010**, *207*, 1327–1340. [[CrossRef](#)]
30. Zhang, X.; Sun, B.; Sun, J.; Gou, Z. The berth and quay cranes integrated scheduling based on redundancy policy. In Proceedings of the 2014 33rd Chinese Control Conference (CCC), Nanjing, China, 28–30 July 2014; Volume 2014, pp. 7595–7600.
31. Rodriguez-Molins, M.; Ingolotti, L.; Barber, F.; Salido, M.A.; Sierra, M.R.; Puente, J. A genetic algorithm for robust berth allocation and quay crane assignment. *Prog. Artif. Intell.* **2014**, *2*, 177–192. [[CrossRef](#)]
32. Shang, X.T.; Cao, J.X.; Ren, J. A robust optimization approach to the integrated berth allocation and quay crane assignment problem. *Transp. Res. Part E Logist. Transp. Rev.* **2016**, *94*, 44–65. [[CrossRef](#)]
33. Correcher, J.F.; Alvarez-Valdes, R. A biased random-key genetic algorithm for the time-invariant berth allocation and quay crane assignment problem. *Expert Syst. Appl.* **2017**, *89*, 112–128. [[CrossRef](#)]
34. Chen, L.; Shen, J.; Qin, L.; Chen, H. An improved ant colony algorithm in continuous optimization. *J. Syst. Sci. Syst. Eng.* **2003**, *12*, 224–235. [[CrossRef](#)]
35. Mauri, G.R.; Ribeiro, G.M.; Lorena, L.A.N.; Laporte, G. An adaptive large neighborhood search for the discrete and continuous berth allocation problem. *Comput. Oper. Res.* **2016**, *70*, 140–154. [[CrossRef](#)]
36. Pacino, D.; Van Hentenryck, P. Large neighborhood search and adaptive randomized decompositions for flexible jobshop scheduling. In Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence, Barcelona, Spain, 16–22 July 2011; Volume 102, pp. 1997–2002.
37. Grangier, P.; Gendreau, M.; Lehuédé, F.; Rousseau, L.M. A matheuristic based on large neighborhood search for the vehicle routing problem with cross-docking. *Comput. Oper. Res.* **2017**, *84*, 116–126. [[CrossRef](#)]
38. Shaw, P. A New Local Search Algorithm Providing High Quality Solutions to Vehicle Routing Problems. 1997, pp. 1–12. Available online: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.51.1273&rep=rep1&type=pdf> (accessed on 5 March 2022).
39. Ropke, S.; Pisinger, D. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transp. Sci.* **2006**, *40*, 455–472. [[CrossRef](#)]
40. Chakraborti, S.; Van der Laan, P.; Bakir, S.T. Nonparametric control charts: An overview and some results. *J. Qual. Technol.* **2001**, *33*, 304–315. [[CrossRef](#)]