Expert Systems With Applications 179 (2021) 115014

Contents lists available at ScienceDirect

Expert Systems With Applications

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

The inbound container space allocation in the automated container terminals

Mingzhu Yu[a], Zhuobin Liang[b], Yi Teng[c], Zizhen Zhang[d,*], Xuwen Cong[b]

[a] Institute of Big Data Intelligent Management and Decision, College of Management, Shenzhen University, Shenzhen 518060, China
[b] College of Civil and Transportation Engineering, Shenzhen University, Shenzhen 518060, China
[c] Department of Computer Science, Guangdong University of Education, Guangzhou 510303, China
[d] School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China

ARTICLE INFO

Keywords:
Automated container terminal
Inbound container
Space allocation problem
Simulation-embedded optimization

ABSTRACT

In this paper, we study an inbound container space allocation problem in the automated container terminals using simulation-embedded optimization. We aim to allocate the container terminal yard space for a batch of arrived inbound containers so as to minimize the total AGV (Automated Guided Vehicle) waiting time in the space allocation process and the external truck waiting time in the future container retrieval process. We propose an integer programming model to characterize the problem and prove the NP-hardness of the problem. A simulation module is employed to estimate the container rehandle number happened in the retrieval process. A simulation-embedded genetic algorithm is developed to solve the problem. Numerical experiment results show that (1) The proposed algorithm can significantly affect the performance of the allocation decision and achieve a trade-off between fast computation and good solution quality. (2) The space allocation schemes of the inbound containers are affected by different factors, such as initial block layout, quantity of arriving containers, and containers' arriving information. (3) The automated container terminal operators who care about the external truck waiting time may consider the simulation-embedded approach when the block yard is congested.

1. Introduction

Driven by the need of maritime transportation, container terminal operators are struggling in cost control and service improvement so as to maintain competitive advantages (Buhrkal, Zuglian, Ropke, Larsen, & Lusby, 2011). They may adopt operational optimization approaches and strategic management decisions. The operational optimization in the container terminal covers almost every functional area, including the berth allocation, quay crane assignment, internal truck scheduling, and yard space allocation (Carlo, Vis, & Roodbergen, 2014). The operations in the terminal yard are quite critical for the overall performance of the container terminal. The congestion and inefficiency in the terminal yard may become the bottleneck of the whole system.

There are mainly three kinds of containers handled in container terminals: inbound containers, outbound containers and transshipment containers (Stahlbock & Voß, 2008). After arriving at the terminal, containers need to be assigned storage spaces. For outbound and transshipment containers, the information of their destinations, boarding vessels and collecting time is usually known by the terminal operator upon their arrival (Gharehgozli & Nima, 2018). For inbound containers, the future retrieval information by external trucks is unknown when they arrive at the container terminal. An appropriate space allocation for inbound containers could not only accelerate the unloading operations at the water side but also reduce the waiting time of external trucks in the future retrieval operations (Choe, Park, Oh, Kang, & Ryu, 2009). To tackle the uncertainty of inbound container's future retrieval time when being assigned yard space, this paper employs a simulation module to estimate the retrieval time. The simulation module is a part of the objective function of an optimization problem (a so-called simulation-embedded optimization method).

Under the pressure of high labor cost and high service expectation, automated container terminals are gradually built all over the world. At present, there are over thirty automated container terminals built in the world and the representative ones are Rotterdam port, Singapore port, Kawasaki port and Hamburg port. The characteristics of different types of equipment in automated terminals make the operations different from those in traditional container terminals (Yu & Qi, 2013). Fig. 1 shows the block layout of a certain kind of automated container terminals. In this

* Corresponding author.
E-mail address: zhangzzh7@mail.sysu.edu.cn (Z. Zhang).

https://doi.org/10.1016/j.eswa.2021.115014
Received 11 October 2020; Received in revised form 12 March 2021; Accepted 7 April 2021
Available online 16 April 2021

container terminal, automatic stacking cranes (ASCs) are applied to move containers among different bays in a block. Automated guided vehicles (AGVs) help to move containers inside the terminal yard. The container block's interfaces with AGVs are at its two end sides in the automated terminal. Therefore, when making container space allocation decisions in this kind of automated container terminals, operators should consider not only the container rehandle time (the rehandle operation happens when a retrieved inbound container is stored below other containers in a bay) but also the ASC horizontal moving time (Gharehgozli, Vernooij, & Zaerpour, 2017; Gharehgozli, De Koster, & Jansen, 2017). Note that, the layout of"interfaces at two ends" (end-loading) is a common design for automated container terminals handling with majority of import/export containers. Compared with the traditional container terminals, although this kind of design may lead to longer ASC horizontal moving time, it can improve the moving time and queuing time of the external/internal trucks, which in turn improve the terminal yard efficiency. Our model is defined basing on this kind of layout.

In this paper, we study an inbound container space allocation problem in the automated container terminals using simulation-embedded optimization. We aim to allocate the space for a batch of arrived inbound containers so as to minimize the total AGV waiting time in the space allocation process and the total container retrieval time (or external truck waiting time) in the future container retrieval process. An integer programming model is proposed to characterize the problem. After proving that this is an NP-hard problem, we utilize genetic algorithm to solve it. A simulation module is embedded in the algorithm to estimate the number of container rehandles happened in the retrieval process. Through computational studies, we validate the effectiveness and efficiency of our algorithm, and also compare the results of simulation-embedded approach with those of the estimation-function-based optimization method.

The contributions of this paper are summarized as follows: (1) An inbound container space allocation model in the automated container terminal is proposed by considering the efficiency of both the container unloading and retrieval processes. (2) We theoretically prove the complexity of the problem and show that it is NP-hard. (3) We employ a simulation module to estimate the number of container rehandles, which helps to tackle the randomness happened in the container retrieval operations. (4) An effective and efficient genetic algorithm is proposed and the computational experiments compare the performances of simulation-embedded approach and the estimation-function-based optimization approach.

The remaining parts of this paper are arranged as follows. Section 2 reviews the related literature. Section 3 proposes an integer programming model of the problem and proves that the problem is NP-hard. In Section 4, we introduce a genetic algorithm to solve this problem. Numerical experiments and results are presented in Section 5. Section 6 draws conclusions of this paper.

2. Literature review

Our study is an optimization problem relevant to three steams of research: automated container terminal operations, container space allocation and simulation optimization. These topics attracted the interests of the scientific community over the last few decades due to its high impact on the competitiveness of maritime container terminals around the world.

The research on the operations of automated container terminals is reviewed as follows. Automated container terminals are controlled automatically with several types of automated equipments, among which automated guided vehicles (AGVs) and automated stacking cranes (ASCs) are the most representative. The characteristics of different types of equipments in automated terminals make the operations different from those in traditional container terminals. Steenken, Voß, and Stahlbock (2004) first provide the concept and describe the equipments of automated container terminals in details. Zhen, Lee, Chew, Chang, and Xu (2012) introduce a new set of automated terminal equipment: frame bridges, rail mounted trolleys and ground trolleys. The performance of the new system and AGV in stacking capacity and transportation efficiency are compared. A dual-cycle model of the AGV dispatching and container storage allocation at automated container terminals is presented by Luo and Wu (2020). In this study, both unloading and loading processes are considered, small-sized problems can be solved optimally and the large-sized problems are solved efficiently by GA. Xie, Zhang, and Mi (2016) propose a two-stage mixed integer programming model to solve the storage allocation problem in automated container terminals. They apply a modified Particle Swarm Optimization (PSO) algorithm to solve the model. Gharehgozli, Vernooij et al. (2017) and Gharehgozli, De Koster et al. (2017) conduct a simulation study about the efficiency performance of a pair of ASCs in the automated container terminal. Lu and Wang (2019) propose a model based on graph theory to control the scheduling of two ASCs.

In recent years, more and more works on container storage space allocation problem have focused on outbound containers (Wan, Liu, & Tsai, 2010; Jiang, Lee, Chew, Han, & Tan, 2012; Gharehgozli & Nima, 2018). Due to the stowage plans on the vessels, outbound containers have predetermined sequence to be loaded to vessels. The space allocation of outbound containers in the container terminal yard is to design appropriate storage places for outbound containers such that the waiting

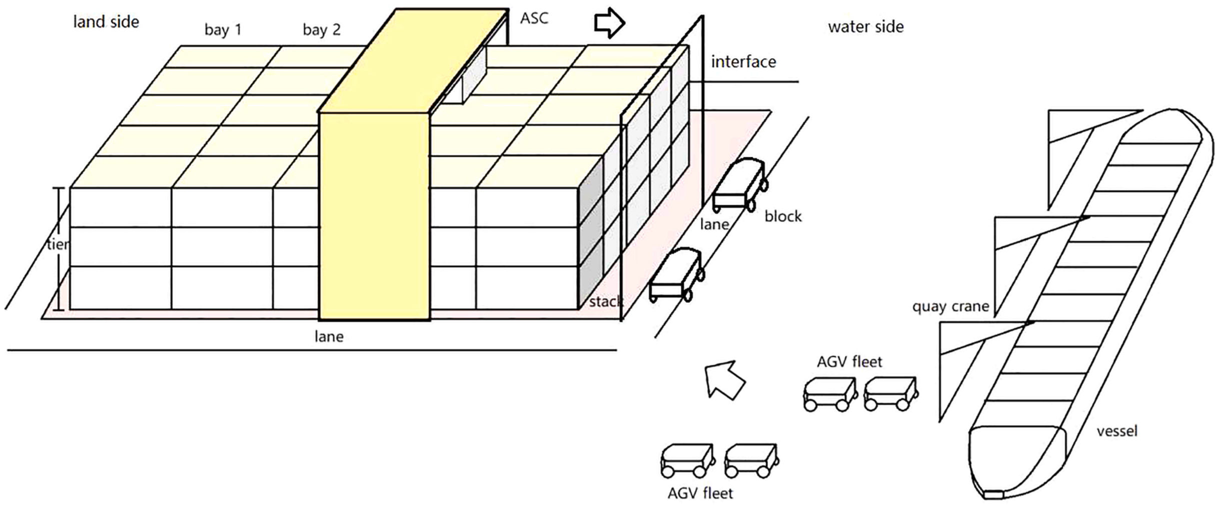

Fig. 1. The layout of a classical automated container terminals.

time of the future arriving vessels and the operation time of the yard cranes are optimized. Chen and Zhen (2012) decompose outbound container space allocation problem into two stages, and propose a mixed integer programming model to minimize travel distance and improve the loading efficiency. Lee and Jin (2013) build a mixed integer programming model in order to find a balance between workload in yard and delivery distance of outbound containers. Zhen (2016) and Tan, He, and Wang (2017) adopt stack strategies to optimize yard template among sub-blocks. Jiang and Jin (2017) obtain near-optimal solutions for yard crane deployment and container allocation in transshipment yards, respectively. Moreover, Ozcan and Eliiyi (2017) propose a reward-based algorithm for the stacking of outbound containers by taking the container's distance to the closest ASC, ASC's workload, the number of stacked containers at the neighborhood bays, and the current height of stacks into consideration.

For the inbound container space allocation, the incomplete future retrieval information makes it challenging. Kim and Kim (2007) study the inbound container space allocation problem in the traditional container terminals. They set up three mathematical models with different constraints from different viewpoints to find the optimal pricing schedule for storing inbound containers. Based on this research, Yu and Qi (2013) propose the inbound container storage space allocation models in the automated container terminals by utilizing the rehandle number estimation function in the future retrieval process. Maldonado, Gonzalez-Ramirez, Quijada, and Ramírez-Nafarrate (2019) address the stacking problem for import containers via a two-step strategy. In this study, dwell time is predicted for each container using analytic techniques. This prediction is used as an input for a mathematical programming model that minimizes the container rehandles heuristically. Other research concerning heuristic algorithm for inbound container allocation problem include Chang and Zhu (2019),Zhou, Wang, and Li (2020) and Zhu, Ji, and Guo (2020). In these approaches, however, the rehandle number estimation all assume that the retrieval probabilities of future inbound containers are the same, which is not suitable in reality. In this regard, we develop a simulation module to estimate the number of rehandles more precisely, which fits the reality better.

Additionally, container stacking optimization problems involving randomness. It is quite complicated to handle the random factors. Existing researches usually handle these issues by adding assumptions about the random factors or conducting estimations through theoretical or numerical approaches. In recent years, the simulation-based optimization has developed to be a preferable method tackling the random factors which are widely existed in traditional optimization problems. Modelling and simulation are used to estimate the impact of key parameters on operational efficiency in container terminal (Longo, 2010). At the same year, Legato, Mazza, and Trunfio (2010) employ a simulation-based optimization framework to estimate diversity and randomness of loading and unloading capabilities. Nguyen, Reiter, and Rigo (2014) provide an overview of simulation method, and introduce an optimization method that combines simulation with a heuristic algorithm. Simulation has also been used to access the quality of solutions (He, Huang, & Chang, 2015). One outstanding proposal to solve the container stacking problem is presented by Gaete, González-Araya, González-Ramírez, and Astudillo (2017). In this paper, a simulation approach is used as the stacking strategy, which is quite different from the mathematical programming and heuristic strategies.

Our study has some strengths over the previous works. We initiate the study about the inbound container space allocation problem in the automated container terminals by adopting a simulation-embedded optimization approach to estimate the number of rehandles in the container retrieval process. Since the existing studies are not applicable for our problem, we prove the complexity of the problem and adopt simulation-embedded genetic algorithm to solve the problem. Through computational experiments, we validate the effectiveness and efficiency of the proposed algorithm, which can provide a promising solution of the large-scale instances.

3. Model formulations

We consider the inbound container operation process in an automated container terminal as shown in Fig. 2. Blocks are perpendicular to the waterside. The AGV interface point and the external truck interface point are at the two ends of the block. After arriving at the automated container terminals, inbound containers will be unloaded from vessels by quay cranes to AGVs. The AGV will move the inbound container to the waterside interface point of the block. Then, the ASC will move the inbound container from the AGV to its assigned storage bay. The operation time of the ASC mainly includes the ASC's horizontal moving time (if its vertical loading/unloading time is ignored). When a certain inbound container is called by the consignee, an external truck will come to the landside interface point of the block. The ASC will move the target container from its storage bay to the external truck. The retrieval time of an inbound container mainly includes the ASC's horizontal moving time and the ASC's rehandle operation time.

For a batch of arriving inbound containers, we aim to assign a suitable space in a given yard block for them. Upon their arrival, the future retrieval information of the inbound containers (e.g., when will a certain container be collected by an external truck) is unknown. Therefore, there may exist rehandle operations in the future retrieval process. A reasonable space allocation will not only accelerate the unloading operation of the vessel, but also help to reduce the waiting time of external trucks in the future retrieval process.

In this paper, given the initial layout of a block and a batch of inbound containers, we aim to assign the block space to these inbound containers so as to (1) minimize the total waiting time of AGVs in the space allocation process, which will directly reflect the efficiency and utilization of AGVs; (2) minimize the total expected waiting time of external trucks in the future container retrieval process, which determines the service quality for external trucks.

We now present an integer program (IP) for inbound container allocation problem in an automated container terminal, the assumptions and mathematical notations including the indices, parameters and decision variables used in the IP as follow.

3.1. Assumptions and notations

Assumption 1. *The block only stores the inbound containers. To simplify the analysis, we assume that the container space allocation process and the container retrieval process are separated.*

Assumption 2. *It is assumed that as soon as the inbound containers are unloaded from the vessel, they can directly be moved by AGVs to the target block. The unloaded time of each container from the vessel (namely, its time to be loaded onto an AGV) is known.*

Assumption 3. *The basic vertical loading/unloading time of ASC is not considered here (even if this time is considered, the solution procedures and basic results will not be affected). But if there exist rehandle operations in the retrieval process, the rehandle operation time is not negligible. Note that, although the vertical moving time of the ASC is ignored, the horizontal moving time of the ASC is not ingored.*

Assumption 4. *There is no buffer area at the waterside of the block. The ASC serves inbound containers one after another according to their arrival order. Each AGV has to wait at the interface area until the ASC removes the container on it.*

Assumption 5. *In this paper, we utilize the non-segregation strategy to store the inbound containers. Container storage allocation operation usually complies with different stacking strategies, e.g., segregation strategy and non-segregation strategy (*Yu & Qi, 2013*). In the segregation strategy, the arriving containers can only be allocated in the empty bays of a block. Namely, the*

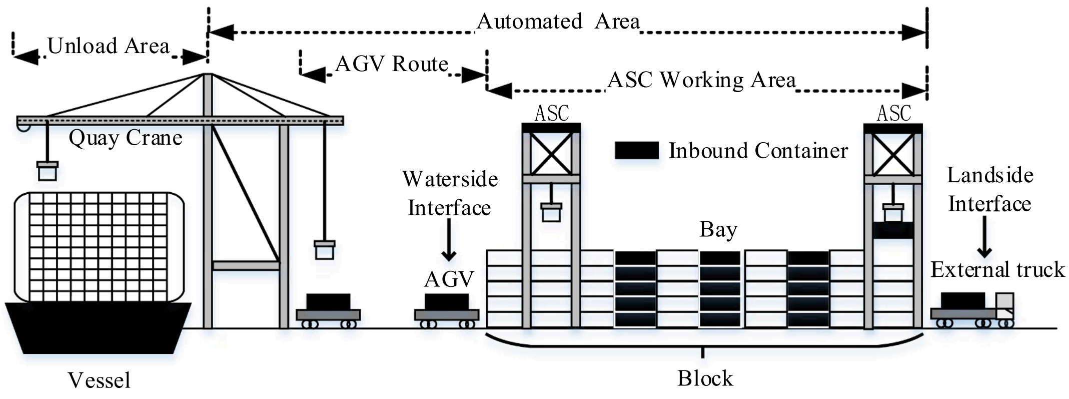

Fig. 2. Inbound containers unloading processes in automated container terminal.

containers arrived in different periods could not be stored in the same bay in the segregation strategy. While, arriving containers could be stored in any bays in the non-segregation strategy as long as the capacity of the bay is not violated.

Assumption 6. *If the target container is berried under other container(s) in the bay, rehandle operations are needed to remove the obstructing container(s). The obstructing container(s) can be put in any other stack in the bay while meeting the height limit of the stack.*

The parameters and decision variables are defined as follows.

Parameters

K: the number of bays in the block. The bays are indexed from the land side to the water side. Namely, bay 1 is the one at the land side and bay K is the one at the water side.
B: the number of arriving inbound containers.
i: index of the inbound containers, $i \in \{1, 2, \ldots, B\}$. Inbound containers are sequentially indexed according to the time they are loaded on the AGV from the vessel.
D: the distance from berth area to the waterside of the block.
v: the speed of AGVs.
s_i: the time when inbound container i is loaded on the AGV, $i \in \{1,2, \ldots, B\}$. Here, $s_{i+1} > s_i, i \in \{1, 2, \ldots, B-1\}$.
a_i: the time when AGV loaded with container i arrives at the waterside of the block. $a_i = s_i + D/v, i \in \{1, 2, \ldots, B\}$.
r: the time of a rehandle operation.
m: the ASC's unit horizontal moving time.
$\overline{h}$: the capacity of a bay.
h_k: the number of initial containers in bay $k, k \in \{1, 2, \ldots, K\}$.

Decision variable

x_{ik}: is equal to 1, if container i is allocated to bay k; is equal to 0, otherwise. $i \in \{1, 2, \ldots, B\}, k \in \{1, 2, \ldots, K\}$.

Dependent variables

l_i: the time when AGV unloads container i at the waterside of the block and turns around, $i \in \{1, 2, \ldots, B\}$.
y_k: the total number of allocated containers in bay $k, k \in \{1, 2, \ldots, K\}$.

3.2. Rehandle number estimation

For inbound containers, their future retrieval information is unknown upon their arrival. When we make the space allocation decisions, we assume that they will be randomly retrieved in the future. Therefore, the space allocation will inevitably lead to rehandle operations in the future retrieve operations. The number of rehandles needed to retrieve all containers in a bay is determined by the initial layout of the bay and the number of allocated arriving inbound containers. We define $R(x)$ as the number of rehandles needed to retrieve all containers in a bay when the bay contains x inbound containers. In this paper, we adopt two kinds of approaches to estimate the number of rehandles: the estimation function and the simulation module.

By assuming that the probability of each container to be retrieved by a future external truck is the same, Kim and Kim (1999),im (1997) had tried to estimate the number of rehandles which is affected by the number of stacks and containers in a bay. It was summarized in the following equation:

$$R(x) = \left(\frac{1}{4s} + \frac{1}{16s^2}\right)x^2 + \left(\frac{1}{8s} - \frac{1}{4}\right)x \tag{1}$$

Here, s is the number of stacks in a bay. This function indicates that when there are x containers in a bay, the expected number of rehandles needed to remove all of them is $R(x)$. Yu and Qi (2013) modified this formula to make it more precise:

$$R(x) = \begin{cases} 0, & x \leqslant s \\ \gamma x - \delta, & s + 1 \leqslant x \leqslant 2s \\ \mu x^2 + \eta x, & x > 2s \end{cases} \tag{2}$$

Here, s is the number of stacks in a bay, $\mu = \frac{1}{4s} + \frac{1}{16s^2}$ and $\eta = \frac{1}{8s} - \frac{1}{4}$. $\gamma = \frac{s+2}{2s+2}$ and $\delta = \frac{s(s+2)}{2s+2}$.

By using these formulas, we can estimate the number of rehandles in the objective function of the proposed model. We call this method an estimation-function-based optimization method.

In this paper, we propose a simulation module to estimate $R(x)$ in a more precise way. The simulation module will be embedded to the optimization problem so as to estimate the objective function that includes random factor (a so-called simulation-embedded optimization method). Simulation is a better way to model the real situations and fits the reality better than the estimation function (2). We will compare the solutions under these two approaches in the computational experiments (refer to Section 5).

3.3. Mathematical formulation

$$\min \alpha \sum_{i=1}^{B}(l_i - a_i) + \beta\left(r\sum_{k=1}^{K}(R(h_k + y_k) - R(h_k)) + 2\sum_{k=1}^{K} mky_k\right) \tag{3}$$

$$\text{s.t.} y_k = \sum_{i=1}^{B} x_{ik}, \quad \forall k = 1, 2, \ldots, K \tag{4}$$

$$\sum_{k=1}^{K} x_{ik} = 1, \quad \forall i = 1, 2, \ldots, B \tag{5}$$

$$l_1 = a_1 \tag{6}$$

$$l_i \geqslant a_i, \quad \forall i = 2, 3, \dots, B \tag{7}$$

$$\begin{gathered} l_{i+1} \geqslant max\{l_i + 2m(K + 1 - k), a_{i+1}\} - (1 - x_{ik})M \\ \forall i = 1, 2, \dots, B - 1, \forall k = 1, 2, \dots, K \end{gathered} \tag{8}$$

$$y_k + h_k \leqslant \bar{h}, \quad \forall k = 1, 2, \dots, K \tag{9}$$

$$x_{i,k} \in \{0, 1\}, \quad \forall i = 1, 2, \dots, B, \forall k = 1, 2, \dots, K \tag{10}$$

The objective function minimizes the weighted sum of the total AGV waiting time in the container allocation process and the total expected external truck waiting time in the container retrieval process, $\alpha + \beta = 1$. The total AGV waiting time directly reflects the efficiency and utilization of AGVs, and the total expected external truck waiting time reflects the service quality for the external logistics customers. Specifically, $\sum_{i=1}^{B}(l_i - a_i)$ indicates the total AGV waiting time and $r\sum_{k=1}^{K}(R(h_k + y_k) - R(h_k)) + 2\sum_{k=1}^{K} mky_k$ is the total expected external truck waiting time in the retrieval process (including the total rehandle time $r\sum_{k=1}^{K}(R(h_k + y_k) - R(h_k))$ and the total ASC horizontal moving time $2\sum_{k=1}^{K} mky_k$). Here, $R(x)$ is a function which specifies the expected total number of rehandles needed to retrieve x containers in a bay. Therefore, for a certain bay k, $R(h_k + y_k) - R(h_k)$ is the total increased number of rehandles caused by the allocation.

Constraints (4) specify the number of containers allocated to a bay. Constraints (5) ensure that each inbound container needs to be assigned to a certain bay. Constraints (6)-(8) describe the relationship of the AGV's arrival time at the block and its leaving time from the block. Here, M is a real number that is large enough. Constraints (9) indicate that the number of containers in each bay cannot exceed the bay's capacity. Constraints (10) define the decision variables.

3.4. The complexity of the inbound container space allocation problem

We prove that the proposed inbound container space allocation problem is NP-hard, which means that currently it does not exist a polynomial-time algorithm to solve the problem to optimality. Before that, we show the following defined Fixed-size Subset-sum Problem is NP-hard.

Fixed-size Subset-sum Problem: Given $c + 2$ positive integers $w_1, \dots, w_c, e$ and B, is there a subset $M \subseteq C = \{1, \dots, c\}$ such that $\sum_{i \in M} w_i = e$ and $|M| = B$?

According to Yu and Qi (2013), we can directly conclude that the Fixed-size Subset-sum Problem is NP-hard.

Theorem 1. *The proposed inbound container space allocation problem of the automated container terminal is NP-hard.*

Proof. We use a reduction from the Fixed-size Subset-sum Problem to prove the theorem. Let the Fixed-size Subset-sum problem be $P1$, assuming that $w_1 > \dots > w_c$. We define the proposed inbound container space allocation problem as $P2$. There are B inbound containers that need to be allocated in the block. As shown in Fig. 3, in the initial block, each bay has a capacity of s (s is the number of stacks in a bay), and there are c bays, each of which contains $s - 1$ containers initially. Other bays in the block are fully occupied. Assuming that in the container retrieval process, the ASC's initial and destination position is at the landside access point and ASC's round-trip traveling time between landside and the other bays is $w_1, w_2, \dots, w_c$. We define w_0 as the ASC's round-trip traveling time of the whole block. It is assumed that the arrival times of AGV at the block are 0, $(w_0 - w_c)$, $2(w_0 - w_c)$, …, $(c - 1)(w_0 - w_c)$. Namely, except the first container, other containers' AGVs arrive at the block relatively late such that the waiting time of each AGV is 0. So the total AGV waiting time is 0. Since the total capacity of each bay is s (namely the bay can only contain one tier of containers), there will be no rehandle operations during the retrieval process. Therefore, the objective of $P2$ only includes the total ASC horizontal moving time in the future retrieval process. We aim to find positions for the B arrived inbound containers in the block such that the ASC moving time in the future retrieval process of them is exactly e.

Now we prove that $P2$ also has a feasible solution if and only if $P1$ has a feasible solution.

First, if $P1$ has a subset $M \subseteq C = \{1, \dots, c\}$ such that $\sum_{i \in M} w_i = e$ and $|M| = B$, then there is a corresponding space allocation scheme for the B inbound containers in $P2$, which satisfies that the ASC moving time in the future retrieval process of them is exactly e.

Second, if there is a feasible solution for $P2$ in which the retrieval time for the B inbound containers is e, then this space allocation scheme corresponds to a subset $M \subseteq C = \{1, \dots, c\}$ such that $|M| = B$ and $\sum_{i \in M} w_i = e$. Therefore, $P1$ has a feasible solution.

4. Solution approaches

Due to the difficulty of designing an exact approach to solve the large-scale instances of the inbound container space allocation problem, we develop a genetic algorithm (GA) to effectively find an excellent near-optimal solution.

GA is a well-known meta-heuristic and its efficiency is verified for many multi-objective optimization problems in the literatures (Kumar & Omkar, 2008, Bazzazi, Safaei, & Javadian, 2009; Ganji, Babazadeh, & Arabshahi, 2010; Guerrero, Lera, & Juiz, 2018). In order to avoid being trapped in local optimum, we extract the characteristics of the proposed problem, and modify GA to make it more adaptable to solve the proposed space allocation model. According to the characteristics of mathematical formulation (e.g., there are many equations in the formulation that need to guarantee in the algorithm's search process), solving the proposed space allocation model needs a population-based approach such as GA for better exploration of the solution space. The space allocation problem of automated container terminal includes random factors in the objective function. Several related researches explore GA approach, which makes it an attractive method for the studies of maritime transportation (Theofanis, Boile, & Golias, 2007; Lee, Wang, & Miao, 2008; Golias, Portal, Konur, Kaisar, & Kolomvos, 2014).

4.1. Chromosome representation

The encoding mode affects not only the landscape of the solution space, but also affects the efficiency of the solution evaluation. This paper employs binary coding so that each chromosome can intuitively and accurately represent a solution. And a variation in a dimension of a chromosome can represent a substantial change in the allocation plan. If there are B containers waiting to be assigned into K bays, the length of a chromosome is $K * B$. As an example, when both B and K equal to 3, the individuality's structure is shown in Fig. 4. Each container has K bays as alternative target destinations. Chromosome 1 in Fig. 4 means that container 1 is going to be sent to bay 2, container 2 is assigned to bay 3 and container 3 will be at bay 1. On the other hand, given a solution, we extract the information so that the solution is transformed into decision variables and depending variables. For example, chromosome 1 in Fig. 4 can be interpreted as: $x_{11} = 0, x_{12} = 1, x_{13} = 0; x_{21} = 0, x_{22} = 0, x_{23} = 1; x_{31} = 1, x_{32} = 0, x_{33} = 0; y_1 = 1, y_2 = 1, y_3 = 1$. Here, $x_{ij} = 1$ means that container j is allocated to the ith bay.

4.2. Crossover and mutation operators

In the evolution process of classical GA, the first 50% best individuals will survive after each individual in the population produces offspring through crossing-over operation (according to our testing, we set the survive rate as 50% to guarantee the diversity and convergence speed of

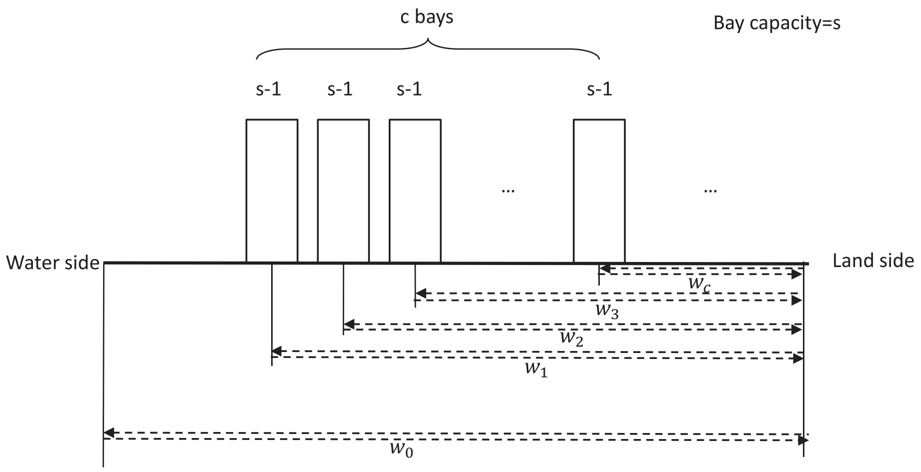

Fig. 3. The instance of the inbound container space allocation problem in the proof of Theorem 1.

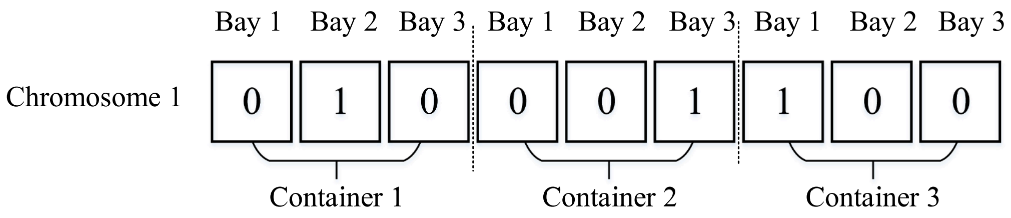

Fig. 4. The encoding of the proposed algorithm.

the search process). Therefore, the iteration based on this mechanism makes the whole population more and more convergent, and then makes the algorithm easy to fall into the local optimal. In order to eliminate the impact of this mechanism, we apply the self-crossover operation besides crossover and mutation operators.

In the crossover operation, a point is generated randomly and all bits after this point of two selected chromosomes are exchanged. For each chromosome selected to mutate, a bit is randomly selected, and the value of the bit is changed to the opposite value of the original one (that is, from 1 to 0 or from 0 to 1). In the self-crossover operation, we randomly select two bits in a chromosome and exchange the values of these two digits. Unlike traditional crossover operations, new individuals obtained from self-crossover do not have to calculate the objective value and are directly preserved in the population (if feasibility check are satisfied). Although this strategy may omit some of the good solutions in a short period, in the long run, it is beneficial to increase the diversity of the population in order to get rid of the local optimal solution.

4.3. *Offspring acceptance strategy*

We use a semi-greedy strategy to accept the offspring generated. In this strategy, an offspring is accepted for the new generation if its objective value is less than the average objective value of its parent(s). This strategy leads to a monotonous convergence toward the optimum solution.

To ensure the feasibility of chromosomes, we first need to ensure that no more than one code value equals to 1 in the coding of one container. Otherwise, the generated offspring will be abandoned. In addition, the total containers allocated to a bay should not exceed the bay capacity. The container that leads to the violation of bay capacity constraint will be removed and randomly placed in other available bays.

4.4. *Simulation-embedded optimization*

As mentioned in Section 3, besides using the rehandle number estimation function, we also employ the simulation module to evaluate the number of rehandles in the retrieval process. Once a solution is determined, it is difficult to calculate the time taken to retrieve all containers in a bay, due to the uncertainty of the order in which the containers are recalled.

In the simulation module, we randomly define a retrieval sequence for all containers in a bay. The simulation module will randomly number all containers in a bay which indicate their retrieved sequence. Then the containers are retrieved out of the bay one by one following the sequence. If the target container is buried under other containers, the emulator automatically moves the obstructing containers above the target container to another stack in accordance with the retrieve rule (as stated in Assumption 6 of Section 3.1). The simulation module continues till all containers in the bay are retrieved. The emulator then outputs the total number of rehandles.

The simulation module is embedded in the genetic algorithm framework (as shown in Fig. 5). In order to eliminate sampling error, we will run the simulation module multiple times. For example, in the computational studies mentioned below, we run the simulation for 10 times, 100 times and 500 times, respectively. It is found that a 10 times simulation experiment is enough to approximate the expected value.

5. Numerical Experiments

In this section, we present the computational results. The commercial solver CPLEX[1] with standard settings is adopted to solve the IP formulation proposed in Section 3.3. Our aims are to (1) validate the effectiveness and efficiency of our algorithm by comparing our results with those obtained by CPLEX; (2) conduct some sensitivity analysis to

[1] https://www.ibm.com/analytics/cplex-optimizer

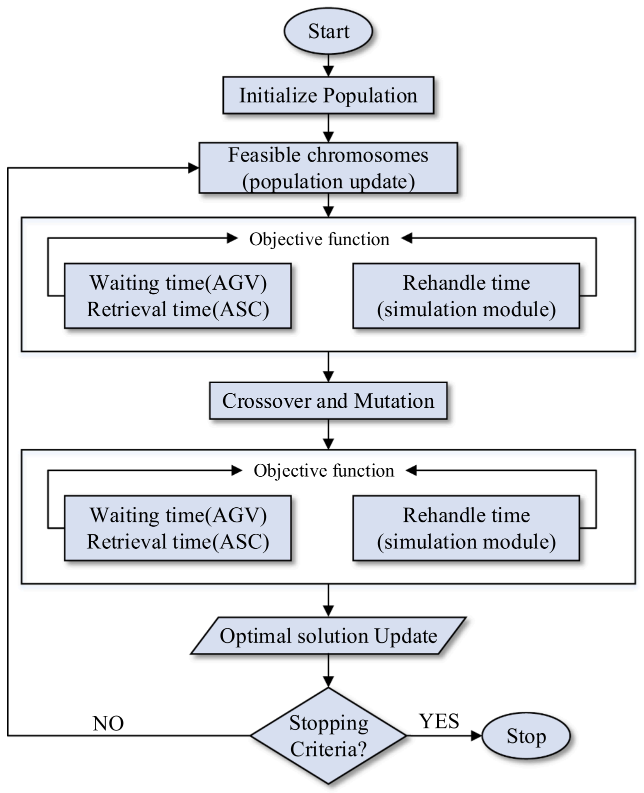
Start
Initialize Population
Feasible chromosomes (population update)
Objective function
Waiting time(AGV) Retrieval time(ASC)
Rehandle time (simulation module)
Crossover and Mutation
Objective function
Waiting time(AGV) Retrieval time(ASC)
Rehandle time (simulation module)
Optimal solution Update
NO
Stopping Criteria?
YES
Stop

Fig. 5. The framework of the simulation-embedded GA.

investigate the impact of different parameters; (3) compare the results of the simulation-embedded optimization approach and the estimation-function-based optimization method. The experiments are run on a PC with Intel Core i5–5200U, 2.20 GHz and 4G RAM. The improved GA is implemented by Matlab 2017a.

In the simulation model, the total rehandle time to of retrieving all containers in a bay is the average time of 10 simulation results. The parameter values of the model and the GA are set as in Table 1. If a bay is fully occupied, it will not be able to conduct any rehandle, so the actual stack height is not equal to the designed safety height of the bay. If the safety hight of the stack is H, then we should leave $H-1$ empty positions in the bay to allow rehandles. Namely, the capacity of the bay is $H{\cdot}s-(H-1)$ instead of $H{\cdot}s$. To simplify the experimental setting, in all experiments, the stack height is limited at $H-1$. Here, H is the safety stack high and s is the number of stacks in a bay.

5.1. The comparison with the results of CPLEX

In order to validate the effectiveness and efficiency of our proposed GA, we compare the results of our algorithm with those obtained by CPLEX based on three types of instances (small-, medium- and large-scale instances). Since CPLEX cannot directly calculate nonlinear segmented functions, we use Eq. (1) rather than Eq. (2) to estimate the number of rehandles when compared with CPLEX.

Table 1
Parameter values of the model and the GA algorithm.

Parameter	Value
r: ASC operation time of one rehandle	66s/TEU
m: ASC gantry horizontal moving speed	3 m/s
α: the weight of the AGV waiting time	0.6
β: the weight of the external truck waiting time	0.4
N: population scale	50
E: maximum iteration	500
p_m: mutation probability	0.09
p_c: crossover probability	0.8

We summarize the results in Table 2. We change the number of arriving containers (denoted by B in the instance) from 10 to 400; the number of bays (denoted by K) from 5 to 30; the number of stacks (denoted by s) from 3 to 7; the number of tiers (denoted by H) from 3 to 5. In Table 2, we record the objective values and the running time of different instances. The column “gap” records the improved objective value percentage by GA from CPLEX. As shown in Table 2, for the small-scale instances (e.g., Instance I-1-a to Instance I-3-b), CPLEX can obtain the optimal solutions within reasonable running time. However, as the scale of the problem continues to grow, CPLEX is not comparable with the proposed GA. When the number of arriving inbound containers reaches 200 or 400, CPLEX may not get a solution within 3,600 s. In contrary, the proposed GA can produce satisfactory solutions in reasonable running time even for the large-scale instances.

Fig. 6 shows the computational results of the algorithm in 500 iterations for one instance. It starts from a random initial solution for the given instance parameters as 400 containers, 15 bays, 5 stacks and 5 tiers. As shown in Fig. 6, the simulation-embedded GA converges fast when solving the model.

5.2. The comparison between the simulation-embedded optimization and estimation-based optimization

In this subsection, we run a 15-period (e.g., 15 days) simulation so as to compare the system performance of the simulation-embedded optimization with that of the estimation-function-based optimization. We conduct the off-line container space allocation upon the arrival of a batch of inbound containers using both two methods. After the space allocation process, a daily container retrieval process happens. The space allocation and container retrieval processes are simulated alternately.

The initial number of containers in the target block is 0 TEU. The block’s capacity is 1600 TEUs (with the dimensions 40 bays, 8 stacks and 5 tiers). We consider different daily arrival rates of the inbound containers, e.g., 50, 80 and 100 TEUs per day (namely, 750, 1200, and 1500 TEUs in total). The daily retrieval rate of containers varies from 15% to 25%. The weight parameters in the objective are set as $\beta=0.8, \alpha=0.2$. In the experiment of simulation-embedded optimization approach, the simulation module is run for 10 times so as to return the estimated objective function in each iteration. We have compared the mean objective value by running the simulation for 10 times, 100 times and 500 times, respectively. It is found that a 10 times simulation experiment is enough to approximate the expected value. To concentrate on the ability of estimating the retrieval time,α is 0.2 and β is 0.8 in these experiments.

In Table 3, the second column is the daily arrival container number and the third column shows different retrieval rate. We record the performance of the retrieval process (column 4–6). Column 4 and 5 record the total retrieval time of the retrieval process (namely, the time that the ASC loads and moves the containers from the allocated bay to the external truck) under both the simulation-embedded optimization model and the estimation-function based model. Each recorded retrieval time is the mean of 10 simulation results. The ”Gap” (column 6) records difference ratio between the two approaches.

$$gap=\frac{\textit{retrieval time under estimation}-\textit{retrieval time under simulation}}{\textit{retrieval time under estimation}}\times 100\% \tag{11}$$

As shown in Table 3, the average expected retrieval time got by the simulation-embedded approach is less than that got by the estimated-based approach. In each instance group (e.g., instance group I-1, I-2, I-3), as the block utilization increases (namely, the retrieval rate

Table 2
The comparison between proposed GA and CPLEX.

	Parameters	Result			Computation Time(s)	
Instance	B-K-s-H	CPLEX	Proposed GA	Gap	CPLEX	Proposed GA
I-1-a	10-5-3-3	581.97	581.97	0	2.33	11.31
I-1-b	10-5-4-3	613.20	613.20	0	2.00	11.57
I-1-c	10-8-3-3	677.90	677.90	0	205.00	11.64
I-2-a	20-5-3-3	1905.90	1905.90	0	3600	15.26
I-2-b	20-6-3-3	2119.33	2119.33	0	3600	17.30
I-2-c	20-6-4-3	1817.91	1817.91	0	3600	17.74
I-3-a	30-6-4-4	3838.16	3838.16	0	3600	22.18
I-3-b	30-7-4-3	4315.48	4315.48	0	3600	25.20
I-3-c	30-8-4-4	6069.17	4412.68	27.29%	3600	27.78
I-4-a	100-12-6-4	80727.31	70667.82	12.46%	3600	103.83
I-4-b	100-15-6-4	95400.68	62926.28	34.04%	3600	128.39
I-4-c	100-20-6-4	85573.78	49756.58	41.86%	3600	224.32
I-5-a	200-15-5-5	479206.63	331955.59	30.73%	3600	239.09
I-5-b	200-15-6-4	450772.65	365243.55	18.97%	3600	239.13
I-5-c	200-20-6-4	–	528215.99	–	3600	360.22
I-6-a	400-15-7-5	–	1525438.92	–	3600	982.71
I-6-b	400-20-7-5	–	1976555.80	–	3600	697.76
I-6-c	400-30-7-5	–	2991096.59	–	3600	1565.72

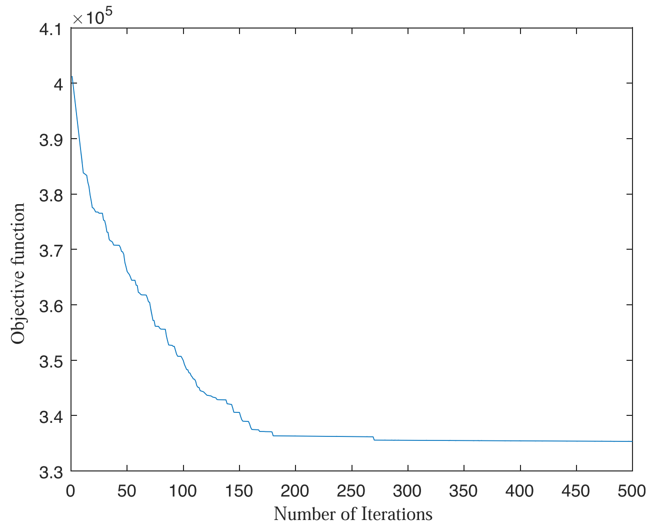

Fig. 6. The convergence curve of simulation-embedded GA.

Table 3
The comparison between estimation-based and simulation-embedded optimization.

	Arriving	Retrieval	Expected Retrieval time		
Instance	containers/day	rate	Estimation	Simulation	Gap
I-1-A	50	25%	36015.6	32311.8	10.28%
I-1-B	50	20%	26730.6	23056.8	13.74%
I-1-C	50	15%	18904.8	15346.2	18.82%
I-2-A	80	25%	64259.6	55977.4	12.87%
I-2-B	80	20%	46548.7	40810.4	12.47%
I-2-C	80	15%	32781.6	26658	18.68%
I-3-A	100	25%	82192.2	74547	9.30%
I-3-B	100	20%	62239.8	54262.8	12.82%
I-3-C	100	15%	42957	35765.4	16.74%

decreases), the retrieval time saving percentage by simulation-embedded optimization increases accordingly. This is because the estimation-based approach utilizes Eq. (2) which could estimate the rehandle number quite well when the number of containers in a bay is relatively small, e.g., less than two tiers (the first two segments of Eq. (2) are exact functions of rehandle number). Therefore, in the container terminal where the space utilization is not very high, the advantage of simulation-embedded optimization approach is not so significant. The automated container terminal operators who care about the external truck waiting time may consider the simulation-embedded approach when the block yard is relatively congested.

5.3. Sensitivity analysis

The previous experiments have demonstrated the effectiveness and efficiency of the proposed algorithm, we now further explore the impacts of the arriving container number, the initial block space utilization and other factors on the system outcomes.

5.3.1. The impacts of arriving container number and initial block layout

In this section, we investigate the impacts of the number of arriving inbound containers and the initial block layout on the system outcomes. The capacity of the block is set as 1600 TEUs (namely $40\ bays \times 8\ stacks \times 5\ tiers$). The initial number of containers in the block is set as 800, 1000, and 1200, respectively. The number of arrival containers is set as 50, 100, 150, 200, 250, 300, 350, 400.

As shown in Fig. 7, the total AGV waiting time increases with the number of arriving inbound containers and the initial number of containers in the block. In addition, the slopes of the three curves in Fig. 7 increase with the number of arriving containers and the initial number of containers. Namely, the total AGV waiting time increases faster with the number of arriving containers when the block utilization is relatively high. Fig. 8 indicates that the overall retrieval time also increases with the number of arriving inbound containers. As shown in Fig. 8, when the number of initial containers is not quite large (e.g., 800 and 1000), the differences between the curves are not quite significant. The curves also have intersections due to the randomness of container needed calls. When the block utilization is not very high, the change of the initial container number has not so significant impact on the external truck waiting time. If the block utilization is relatively high (e.g., initial container number is 1200), containers are intended to be allocated to the land side, which leads to shorter external truck waiting time.

5.3.2. The trade-off between the allocation and retrieval processes

In this section, we intend to investigate the impacts of allocation and retrieval process weights (namely, α and β) in the objective function.

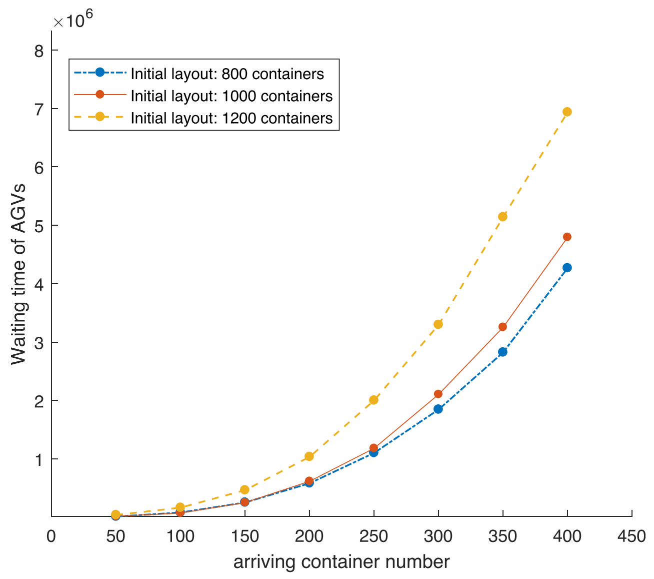

Fig. 7. Waiting time of AGVs under different situation.

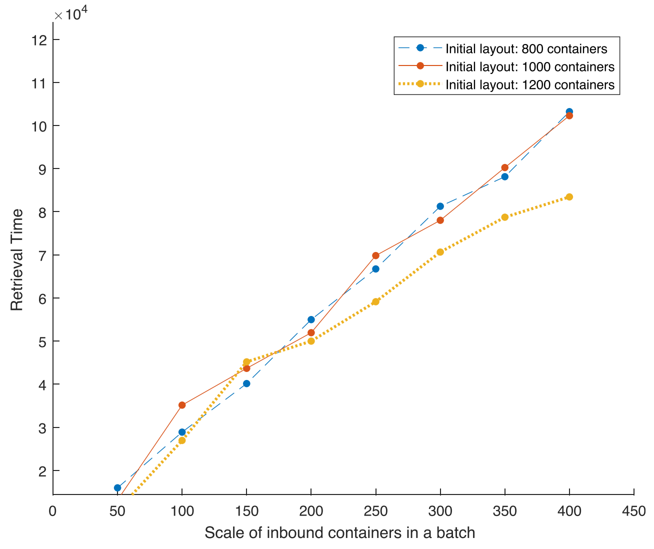

Fig. 8. Waiting time of external trucks under different situation.

Recall that, α is the weight of the water side allocation operation time, and β is the weight of the land side retrieval time. The initial number of containers in the block is 0, and the number of arrival containers is set as 500 and 600, respectively. The space allocation schemes of the arriving containers are shown in Fig. 9.

Fig. 9 shows the block layouts after the space allocation. As shown, the change of the weights (α and β) has little impact on the container distributions. The space allocation schemes give priorities to the bays close to the water side in order to minimize the objective function, even though α equals 0.1 and β equals 0.9. This is most likely due to the fact that we set one of the objectives as the summation of the AGV waiting time. A batch of containers arrive at the terminal at the same time and are sent to the yard subsequently. In this case, there will be many AGVs queuing at the water side interface of the block, which directly produces quite long total waiting time at the water side. While, for the retrieval process at the land side, the time intervals between different external trucks' arrivals are not so tight as those of the AGVs. Therefore, there is almost no queuing time for the external trucks. In this case, the water side priority will be higher than that at the landside in the solutions and the change of the weight parameters has little impact on the results.

5.3.3. *Different measurements of water side efficiency*

Based on the above experiments and analysis, we find that the water side efficiency (namely, the total waiting time of the AGVs at the waterside of the block) plays an important role in the objective function. In this section, we aim to compare two different measurements of the waterside efficiency. Namely, the total waiting time of the AGVs (as described in the objective function in Section 3)) and the allocation finishing time of the last inbound containers (namely, the allocation finish time of the last container). If we consider the allocation finish time of the last container, the objective function can be rewritten as Eq. (14):

$$C(x) = \alpha l_B + \beta \left(r \sum_{k=1}^{K} (R(h_k^0 + y_k) - R(h_k^0)) + 2 \sum_{k=1}^{K} mky_k \right) \tag{12}$$

In the comparison experiments, we set the block initially empty. We study different kinds of block configuration. In the first set of instances, the number of bays is set as 20, the number of stacks is set as 2, 4 and 10, and the number of tiers is set as 2, 5, and 10, respectively. In another set of instances, the number of bays is set as 40, the number of stacks is set as 2 and 5, and the number of tiers is set as 5 and 2, respectively. The number of arriving inbound containers is set as 200.

As shown in Figs. 10-(a) and 11-(a), if we focus on the total AGV waiting time, the space allocation scheme will give priority to the waterside spaces. If the objective function emphasizes the makespan rather than the total AGV waiting time, the distribution of the inbound containers in the block tend to be smoothly spread among the bays (as shown in Figs. 10-(b) and 11-(b)).

Therefore, if the arriving containers come from a single vessel (namely, the allocation operation makespan which affects the vessel turn-around time matters), they will be allocated more evenly in the block than the case when they are unloaded from different vessels (namely, the total AGV waiting time in the allocation operation which affects the total vessel turn-around time matters).

6. Conclusions

In this paper, we study the inbound container storage space allocation problem in automated container terminals by using a simulation-embedded optimization method. We propose an integer programming model to characterize the space allocation of inbound containers in the automated container terminal and prove that this problem is NP-hard. To handle the random factors in the future retrieval process, we utilize a simulation module to estimate the future rehandle operation time in the objective function. By embedding the simulation module into a proposed GA, we provide a heuristic approach to solve the problem.

We test our approach with a number of sample instances with different container numbers, initial layouts and other factors. Our computational results evidence the excellent performance of the proposed algorithm in comparison to the approach with commercial solver CPLEX. Although we cannot guarantee that the simulation-embedded GA is always better, according to our computational study results, the simulation-embedded GA leads to shorter external truck waiting time, especially when the yard block utilization is high. The automated container terminal operators who care about the external truck waiting time may consider the simulation-embedded approach when the block yard is congested. We also conduct some sensitivity analysis to investigate the impacts of the initial block layout, the number of arriving containers, the trade off between the allocation and retrieval processes, and different measurements of water side efficiency. The problem we proposed may shed lights on the operations in automated container terminals by specifically pointing out the random factors in the rehandle number estimation. In addition, by integrating historical data to estimate the container retrieval probability, the proposed simulation-embedded heuristic approach could also be utilized by the practitioners to improve the efficiency of the container yard operations.

Although our research contributes to the existing literature and

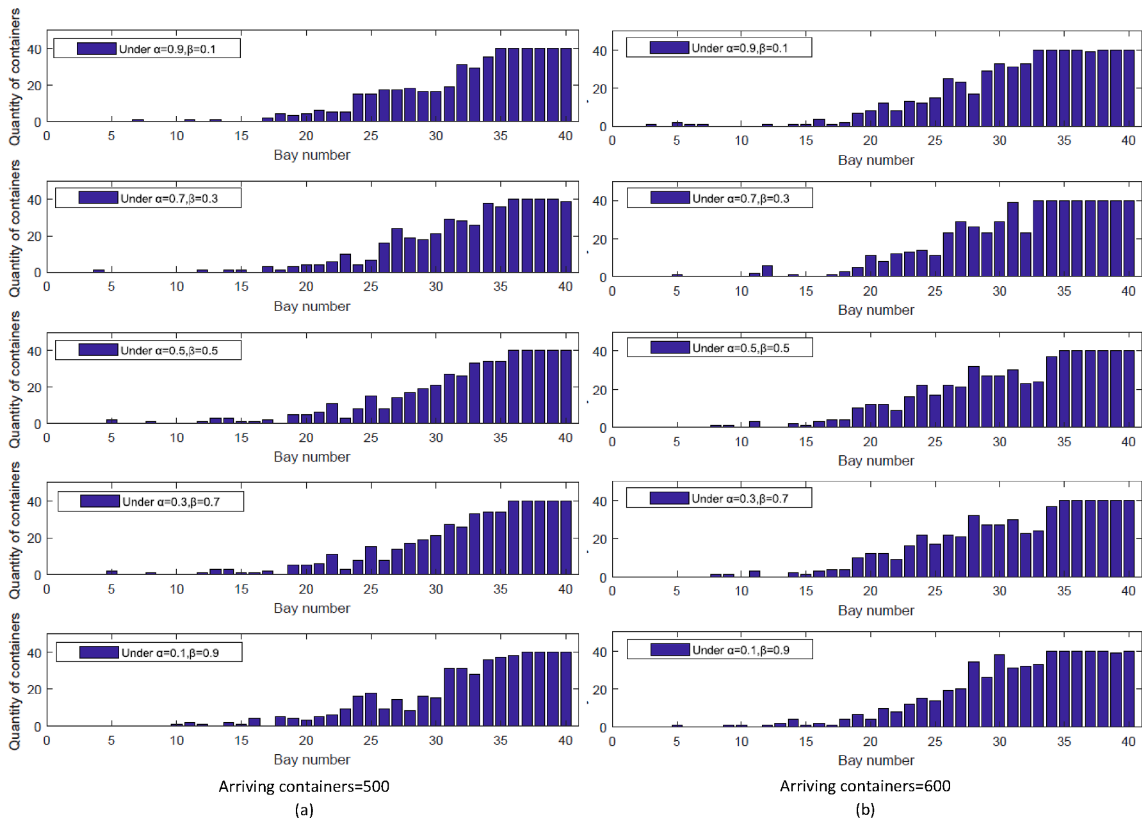

Fig. 9. Layout of 40 bays under different weights.

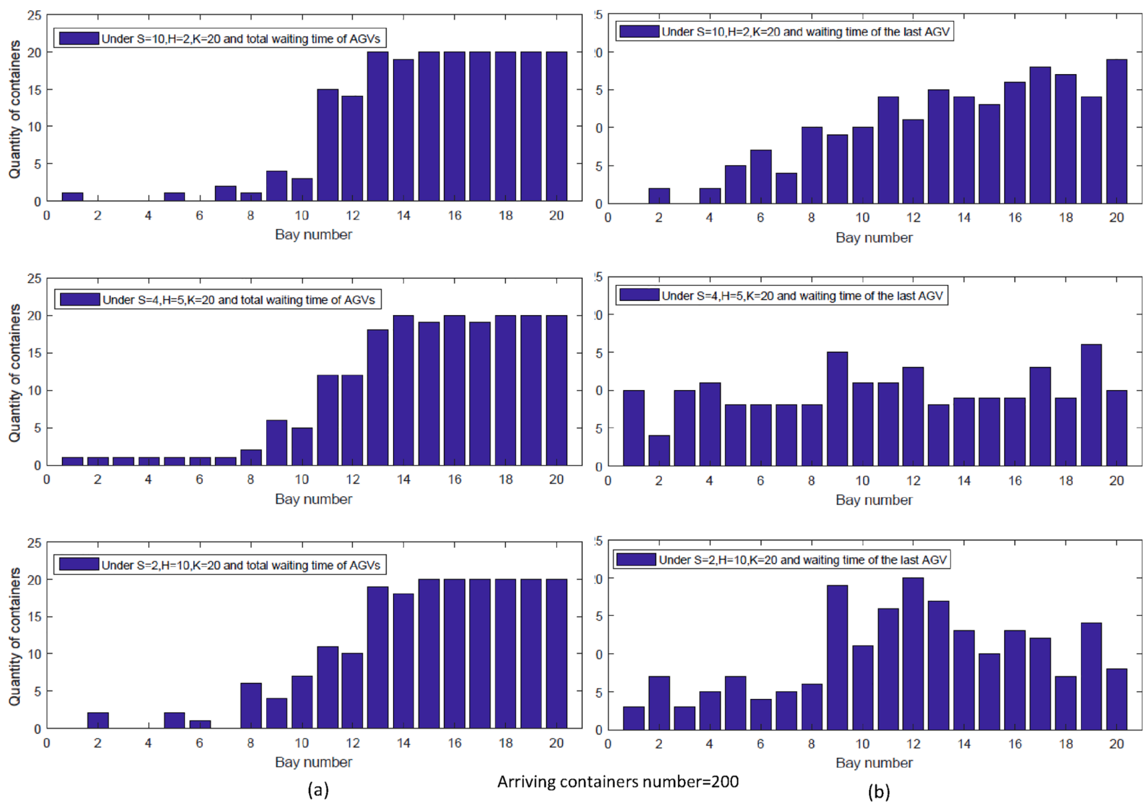

Fig. 10. Layout of 20 bays under different measurements.

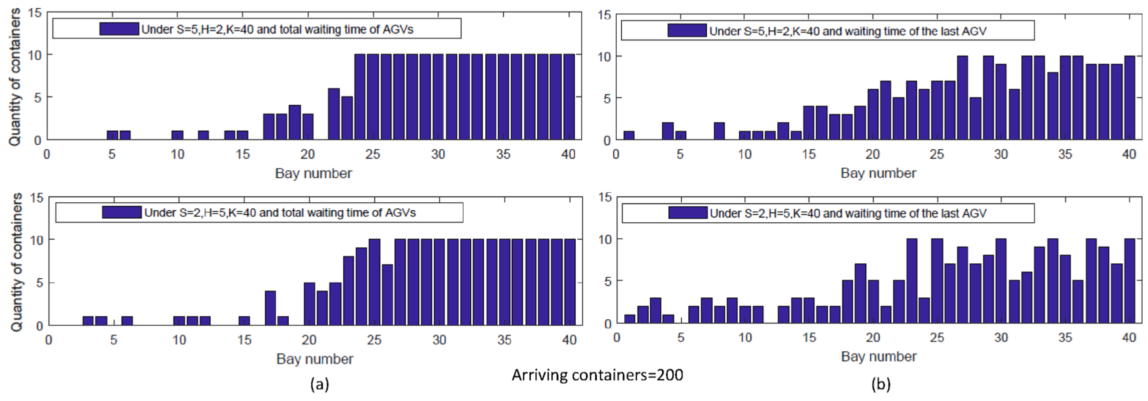

Fig. 11. Layout of 40 bays under different measurements.

provides some management suggestions for the practical managers, there are some limitations in our research which may be worth further studying. Firstly, since CPLEX cannot handle the piecewise non-linear function (Eq. (2)), we use the less precise quadratic function (Eq. (1)) to evaluate the rehandle numbers in the comparison with CPLEX. Secondly, the inbound container space allocation in this paper is simplified as an off-line problem. However, the problem of storage allocation is a dynamic problem in reality. The simultaneous allocation and retrieval operations could be future research directions.

CRediT authorship contribution statement

Mingzhu Yu: Methodology, Conceptualization, Software. **Zhuobin Liang:** Methodology, Conceptualization, Software. **Yi Teng:** Writing - review & editing. **Zizhen Zhang:** Conceptualization, Validation. **Xuwen Cong:** Methodology, Software.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

The work described in this article was supported by National Natural Science Foundation of China (No. 71771154), Guangdong Basic and Applied Basic Research Foundation (No. 2020A1515011272) and Guangdong Natural Science Funds (No. 2019A1515011169, 2021A1515011301).

References

Bazzazi, M., Safaei, N., & Javadian, N. (2009). A genetic algorithm to solve the storage space allocation problem in a container terminal. *Computers & Industrial Engineering, 56*(1), 44–52.

Buhrkal, K., Zuglian, S., Ropke, S., Larsen, J., & Lusby, R. (2011). Models for the discrete berth allocation problem: a computational comparison. *Transportation Research Part E: Logistics and Transportation Review, 47*(4), 461–473.

Carlo, H. J., Vis, I. F., & Roodbergen, K. J. (2014). Storage yard operations in container terminals: Literature overview, trends, and research directions. *European Journal of Operational Research, 235*(2), 415–432.

Chang, Y., & Zhu, X. (2019). A novel two-Stage heuristic for solving storage space allocation problems in rail–water intermodal container terminals. *Symmetry, 11*(10), 1229.

Chen, L., & Zhen, L. (2012). The storage location assignment problem for outbound containers in a maritime terminal. *International Journal of Production Economics, 135*(1), 73–80.

Choe, R., Park, T., Oh, M.-S., Kang, J., & Ryu, K. R. (2009). Generating a rehandling-free-intra-block remarshaling plan for an automated container yard. *Journal of Intelligent Manufacturing, 22*(2), 201–217.

Ganji, S. R., Babazadeh, A., & Arabshahi, N. (2010). Analysis of the continuous berth allocation problem in container ports using a genetic algorithm. *Journal of Marine ence and Technology, 15*(4), 408–416.

Gaete, M., González-Araya, M.C., González-Ramírez, R.G. & Astudillo, C. (2017). A novel storage space allocation policy for import containers, International Conference on Operations Research and Enterprise Systems, Vol. 884, Springer pp. 293–316.

Gharehgozli, A., & Nima, Z. (2018). Stacking outbound barge containers in an automated deep-sea terminal. *European Journal of Operational Research, 267*(1), 975–977.

Gharehgozli, A. H., Vernooij, F. G., & Zaerpour, N. (2017). A simulation study of the performance of twin automated stacking cranes at a seaport container terminal. *European Journal of Operational Research, 261*(1), 108–128.

Gharehgozli, A. H., De Koster, R., & Jansen, R. (2017). Collaborative solutions for inter terminal transport. *International Journal of Production Research, 55*(21), 6527–6546.

Golias, M., Portal, I., Konur, D., Kaisar, E., & Kolomvos, G. (2014). Robust berth scheduling at marine container terminals via hierarchical optimization. *Computers & Operations Research, 41*, 412–422.

Guerrero, C., Lera, I., & Juiz, C. (2018). Genetic algorithm for multi-objective optimization of container allocation in cloud architecture. *Journal of Grid Computing, 16*, 113–135.

He, J., Huang, Y., & Chang, D. (2015). Simulation-based heuristic method for container supply chain network optimization. *Advanced Engineering Informatics, 29*(3), 339–354.

Jiang, X.J. & Jin, J.G. (2017). A branch-and-price method for integrated yard crane deployment and container allocation in transshipment yards. Transportation Research Part B: Methodological, 98, 62-75.

Jiang, X., Lee, L. H., Chew, E. P., Han, Y., & Tan, K. C. (2012). A container yard storage strategy for improving land utilization and operation efficiency in a transshipment hub port. *European Journal of Operational Research, 221*(1), 64–73.

Kim, K. H. (1997). Evaluation of the number of rehandles in container yards. *Computers and Industrial Engineering, 32*(4), 701–711.

Kim, K. H., & Kim, H. B. (1999). Segregating space allocation models for container inventories in port container terminals. *International Journal of Production Economics, 59*(1–3), 415–423.

Kim, K.H., & Kim, K.Y. (2007). Optimal price schedules for storage of inbound containers. Transportation Research Part B: Methodological, 41(8), 0-905.

Kumar, M. M., & Omkar, S. N. (2008). Optimization of yard crane scheduling using particle swarm optimization with genetic algorithm operators. *Journal Ofentific and Industrial Research, 67*(5), 335–339.

Lee, D. H., Wang, H. Q., & Miao, L. (2008). Quay crane scheduling with noninterference constraints in port container terminals. *Transportation Research Part E: Logistics and Transportation Review, 44*(1), 124–135.

Lee, D.H.,& Jin, J.G. (2013). Feeder vessel management at container transshipment terminals. Transportation Research Part E: Logistics and Transportation Review, 49 (1), 201-216.

Legato, P., Mazza, R. M., & Trunfio, R. (2010). Simulation-based optimization for discharge/loading operations at a maritime container terminal. *OR Spectrum, 32*(3), 543–567.

Longo, F. (2010). Design and integration of the containers inspection activities in the container terminal operations. *International Journal of Production Economics, 125*(2), 272–283.

Lu, H. J., & Wang, S. (2019). A study on multi-ASC scheduling method of automated container terminals based on graph theory. *Computers & Industrial Engineering, 129*, 404–416.

Luo, J., & Wu, Y. (2020). Scheduling of container-handling equipment during the loading process at an automated container terminal. *Computers & Industrial Engineering, 149*, 106848.

Maldonado, S., Gonzalez-Ramirez, R. G., Quijada, F., & Ramírez-Nafarrate, A. (2019). Analytics meets port logistics: A decision support system for container stacking operations. *Decision Support Systems, 121*, 84–93.

Nguyen, A. T., Reiter, S., & Rigo, P. (2014). A review on simulation-based optimization methods applied to building performance analysis. *Applied Energy, 113*, 1043–1058.

Ozcan, S., & Eliiyi, D. T. (2017). A reward-based algorithm for the stacking of outbound containers. *Transportation Research Procedia, 22*(1), 213–221.

Stahlbock, R., & Voß, S. (2008). Operations research at container terminals: a literature update. *OR Spectrum, 30*(1), 1–52.

Steenken, D., Voß, S., & Stahlbock, R. (2004). Container terminal operation and operations research: a classification and literature review. *OR Spectrum, 26*(1), 3–49.

Tan, C., He, J., & Wang, Y. (2017). Storage yard management based on flexible yard template in container terminal. *Advanced Engineering Informatics, 34*, 101–113.

Theofanis, S., Boile, M., & Golias, M. (2007). An optimization based genetic algorithm heuristic for the berth allocation problem. 2007 IEEE congress on evolutionary computation, pp. 4439–4445, doi:10.1109/CEC.2007.4425052.

Wan, Y., Liu, J., & Tsai, P. C. (2010). The assignment of storage locations to containers for a container stack. *Naval Research Logistics, 56*(8), 699–713.

Xie, M., Zhang, N., & Mi, W. (2016). Storage allocation in automated container terminals: the upper level. *Polish Maritime Research, 23*, 160–174.

Yu, M., & Qi, X. (2013). Storage space allocation models for inbound containers in an automatic container terminal. *European Journal of Operational Research, 226*(1), 32–45.

Zhen, L. (2016). Modeling of yard congestion and optimization of yard template in container ports. *Transportation Research Part B: Methodological, 90*, 83–104.

Zhen, L., Lee, L. H., Chew, E. P., Chang, D. F., & Xu, Z. X. (2012). A comparative study on two types of automated container terminal systems. *IEEE Transactions on Automation Science and Engineering*. https://doi.org/10.1109/TASE.2011.2165539

Zhou, C., Wang, W., & Li, H. (2020). Container reshuffling considered space allocation problem in container terminals. *Transportation Research Part E: Logistics and Transportation Review, 136*, 101869.

Zhu, H., Ji, M., & Guo, W. (2020). Two-stage search algorithm for the inbound container unloading and stacking problem. *Applied Mathematical Modelling, 77*, 1000–1024.