Computers & Industrial Engineering 173 (2022) 108671

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

GMM-based online optimization for container stacking in port container terminals

Sung Won Cho [a,*], Hyun Ji Park [a], Armi Kim [b], Jin Hyoung Park [a]

[a] Maritime Safety and Environmental Research Division, Korea Research Institute of Ships and Ocean Engineering, Daejeon, Republic of Korea
[b] Department of Transportation and Logistics Engineering, Hanyang University, Gyeonggi-do, Republic of Korea

ARTICLE INFO

Keywords:
Online optimization
Container stacking problem
Gaussian mixture model
Vertical stacking policy
Port management

ABSTRACT

Online optimization has a limitation in that it creates a policy that is unrelated to the actual data by not characterizing the problem data uncertainty. In this study, to overcome the limitations of the vertical stacking policy which is a conventional online optimization approach utilized in the container stacking problem (CSP), we propose a GMM-based online optimization. The container weight is classified into data-driven weight classes based on the Gaussian mixture model (GMM), and our stacking policy is updated in response to problem data. When comparing the weight variance of the other existing stacking policies with the proposed stacking policy, the proposed stacking policy showed smaller values of weight variance on average. Based on this study, containers can be stacked to facilitate flexible responses to various situations with uncertainties and reduce the time taken for container relocation movements.

1. Introduction

The COVID-19 pandemic led to the second global crisis of the 21st century since the 2008 financial crisis, causing a shock to the global economy and to supply chains, which was followed by a recession in OECD countries and most developing countries (Notteboom, Pallis & Rodrigue, 2021). Because the economy of South Korea is highly dependent on imports and exports, ports serve as the major gateway for import and export logistics, and the importance of the operational capabilities of the ports has gained increasing attention in response to the risk of a recession (Kim et al., 2021). Container terminals should be able to efficiently link containers between maritime and inland transport; to this end, secure advanced technical equipment and manpower are required to improve the productivity and provide a high-quality service for shipping companies that use these ports (Park et al., 2021). However, securing necessary resources alone is insufficient to predict the effect of productivity improvement due to the complexity of terminal operations, and without an advancement in the terminal operating system, the operational efficiency may drop considerably against the desired ideal handling capability (Cho et al., 2021). As listed in Table 1, various containers enter the yard of a terminal, and the method of storing the loaded containers in the yard has a direct impact on both the seaside and landside operations when the containers are loaded. Therefore, an efficient operation of the port terminals based on a well-established operational strategy is imperative (Zhang et al., 2003).

A primary task in yard management is determining the storage space for containers. In the yard, numerous yard blocks are placed, as shown in Fig. 1. Through the following three-step decision process, the stacks to store the arriving containers are determined: (1) a yard block or sub-block is allocated to the outbound and transshipment containers to be loaded onto the same vessel, (2) a yard bay is allocated to containers of similar attributes (size, type, and destination), and (3) the position for each of the newly arriving containers is determined within a single yard bay (Borgman et al., 2010; Zhang et al., 2014a, 2014b; He et al., 2020). This study focuses on the problem of determining the lowest level, which is referred to as the container stacking problem (CSP).

In the CSP for outbound and transshipment containers, the stowage plan and ship stability, which are constraints of loading operations, should be considered. The shipping company provides a stowage plan as an instruction to the terminal operator, where the information on the attributes (size, type, and destination) of the containers that can be stacked in each slot of ship bay is included. Containers to be loaded on a vessel are grouped based on the container attributes described in the stowage plan, and this is called a container group (Carlo et al., 2014). The terminal operator loads the containers considering ship stability, and for this purpose, the heavier containers are loaded at the bottom of

* Corresponding author.
E-mail addresses: gungnir@kriso.re.kr (S.W. Cho), qgw159@kriso.re.kr (H.J. Park), 0718kar@naver.com (A. Kim), jin.h.park@kriso.re.kr (J.H. Park).

https://doi.org/10.1016/j.cie.2022.108671
Received 12 February 2022; Received in revised form 26 July 2022; Accepted 15 September 2022
Available online 22 September 2022

Table 1
Container types with container flow.

Container type	Inbound mode of transport	Outbound mode of transport
Outbound	Truck	Ship
Inbound	Ship	Truck
Transshipment	Ship	Ship

the vessel. Therefore, in the yard, the outbound and transshipment containers in the same container group should be placed together and the lighter containers should be stored at the bottom so that heavy containers can be loaded into the ship first and the time required for the relocation of the containers can be reduced.

In practice, owing to the uncertain arrival schedule of the transportation, the arrival and departure times of containers change in real-time. Subsequently, the workload of the yard crane is not known in advance when containers enter the terminal so that the stack to store the container is determined in real-time. Therefore, the CSP is a real-time decision-making problem with only limited available information in advance. In this study, to address this problem, an online optimization is employed in which incomplete input information is available, and decisions are made sequentially based on a piece-by-piece input.

The approaches for solving practical problems include deterministic optimization and stochastic optimization techniques. The former has a limitation in that accurate information on the problem data must be provided. The latter has the limitation of heavily depending on the distribution used to characterize the uncertainty of the problem. These limitations result from the tendency of optimization approaches to simplify the problem to facilitate handling (Jaillet & Wagner, 2010). However, the online optimization approach assumes that there is uncertainty in the problem data and the information on the problem data incrementally becomes available, but makes no assumptions about the distribution or set that characterizes this uncertainty. That is, the online optimization aims to create an optimization policy that performs reasonably well irrespective of the actual generated data. Various online optimization policies exist to solve the CSP (Dekker et al., 2007; Borgman et al., 2010; Park et al., 2011; Chen & Lu, 2012; Güven & Eliiyi, 2014, 2019; Gunawardhana et al., 2021; Park et al., 2022), and an efficient method is developed according to the research purpose.

Many studies assume that there are no containers loaded on the ship, such as ship bay i in Fig. 2, so heavy containers will be loaded first. However, in the actual environment, there is a stowage plan in which the container is loaded on the ship, such as ship bay j. In this case, lightweight containers can be loaded first. Therefore, it is advantageous to create an environment in the yard so that a container of the required weight can be selected according to the situation of the stowage plan. In this study, we employed a vertical stacking policy, also known as a category stacking policy (Dekker et al., 2007).

In the vertical stacking policy, a weight class is designated for each stack in a single bay so that containers with similar weights can be stacked together, as shown in Fig. 3. In general, the terminal operator classifies the containers to be loaded onto the vessel into knowledge-based weight classes and applies the same stacking criteria to multiple bays irrespective of the actual generated data. However, because containers are randomly allocated to the bay according to the situation at the time when the containers are incoming, the proportion of containers per weight class in each bay cannot be the same.

The motivation for this study stemmed from the limitations of the online optimization approach (i.e., vertical stacking policy) and the characteristics of the CSP. The limitation of the online optimization mentioned above is that it creates a policy that is not related to the actual data by not characterizing the uncertainty of the problem data. In the past, it was difficult to utilize the data, even with large amounts of available data, but currently, techniques such as machine learning are used to represent the system characteristics better and improve conventional optimization approaches (Corne et al., 2012).

In addition, the static weight class, such as three classes: heavy, medium, and light (Kim et al., 2000; Kang et al., 2006; Zhang et al., 2010; Zhang et al., 2014a; Zhang et al., 2014b), has a limitation in which it cannot respond to the uncertain proportion of containers per weight class in each bay. This is critical to the vertical stacking policy that requires containers with similar weights to be stacked on the same stack. To overcome these limitations, we utilize a Gaussian mixture model (GMM). GMM is a semi-parametric probability density function, which can represent a complex distribution of observed variables that cannot be represented by a Gaussian distribution (Reynolds, 2009). Therefore, we propose a GMM-based online optimization, which can define weight classes based on the historical data and update the parameters of GMM for each bay by dynamically responding to problem data. Our method overcomes the limitations of existing online optimization by stacking containers with similar weights in stacks for each bay.

The remainder of this paper is organized as follows. Section 2 presents a literature review related to CSP. Section 3 describes the problem definition, assumptions, and GMM-based vertical stacking policy. In

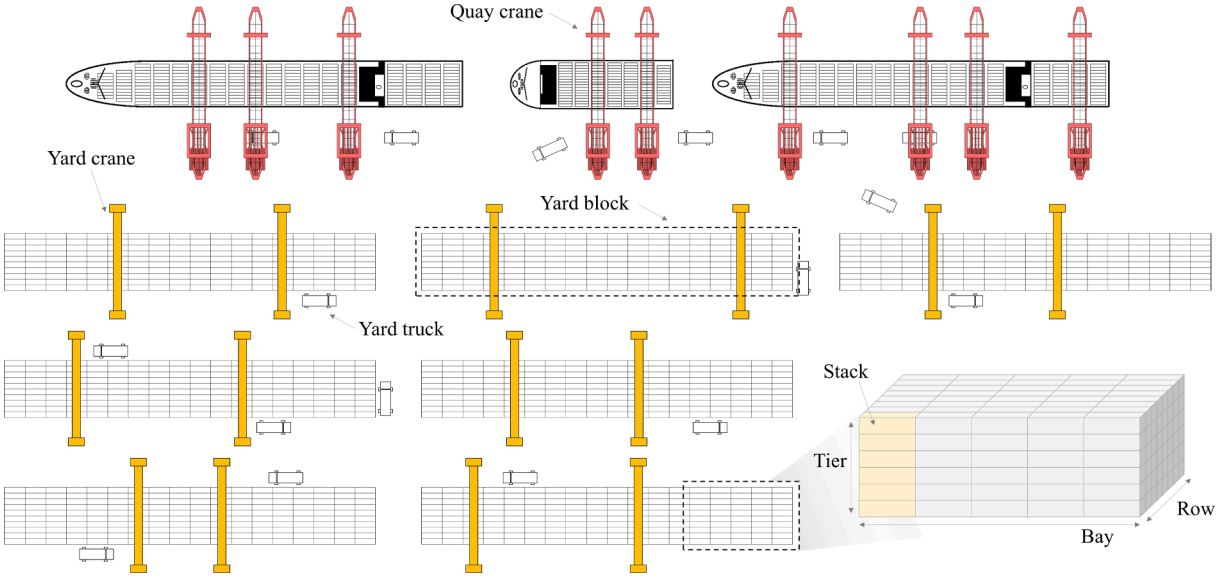


Fig. 1. Typical configuration of container terminals and yard blocks.

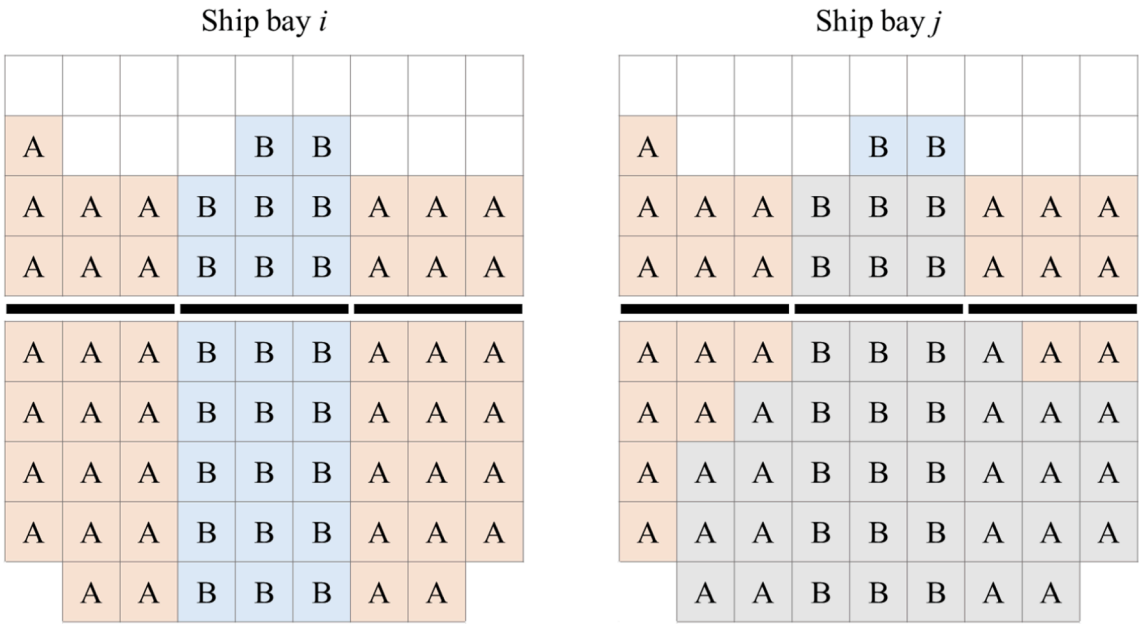


Fig. 2. Stowage plan of the ship.

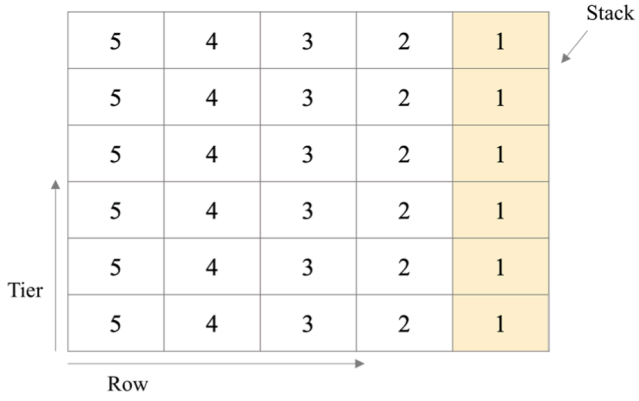


Fig. 3. Vertical stacking policy (five weight classes).

Section 4, numerical experiments based on empirical data are conducted to evaluate the performance of the proposed model, and interpretations of the experimental results are discussed. Finally, Section 5 presents the conclusions and implications of the study.

2. Literature review

Yard space management can be divided into three levels of decision-making: strategic, tactical, and operational (Zhen et al., 2013; Filom et al., 2022). At the strategic level, equipment selection and layout design are determined in consideration of the size of the terminal and the container type to be handled during the construction phase of the yard space (Carlo et al., 2014). After that, the aim is to find the best storage location by considering the characteristics of inbound and outbound containers. The problem types are the storage space allocation problem (SSAP) and CSP (Covic, 2018), and the process of stacking containers is divided into the block (or sub-block), bay, and stack levels. The SSAP is represented at the tactical level and contains the block to bay levels. At the operational level, the CSP is representative and determines the stack on which containers will be stacked in a single bay or multiple bays. This paper focuses on the operational decision to determine the stacking positions of the outbound and transshipment containers within the allocated yard bays, which belongs to the CSP. Thus, this paper mainly provides a literature review for the CSP.

The CSP was investigated by dividing the containers into inbound and outbound containers, according to the characteristics of each type. De Castillo & Daganzo (1993) proposed a segregation strategy for stacking separately inbound containers arriving from different ships. Kim (1997) developed a regression model to derive the expected number of rehandles, and based on the model, proposed a method to determine the stacking positions of the inbound containers entering the terminal so that the number of rehandles is minimized. Kim & Kim (1999) investigated the space allocation method for inbound containers. They presented a mathematical model for the relationship between the stack height and number of rehandles and used the Lagrangian relaxation technique to minimize the latter. The storage period of the inbound containers varies depending on the arrival time of the external truck (ET). Because the arrival time of the ET changes in real-time, there is a limit to addressing the problem of container relocation solely by determining the stacking location. Instead, in the remarshalling problem, after container stacking, the number of rehandles can be minimized in advance through an integration with the truck appointment system (Covic, 2017). Ting et al. (2010) proposed a category stacking for inbound containers based on analytical results from historical data and developed a container pick-up booking system. Sauri & Martin (2011) developed three new stacking strategies for the inbound containers, considering that the probability of a container leaving the terminal varies depending on the time each container arrives. Gaete et al. (2017) developed a dwell time segregated container stacking policy that segregates the inbound containers based on dwell time intervals. They used the classification algorithms, such as Naive Bayes, K-Nearest Neighbor and rules induction learning, to classify the dwell time. Maldonado et al. (2019) proposed a two-step strategy for the inbound container. In the first step, they predicted the dwell times for each container using multiple linear regression, decision trees, and the random forest method. Then, they assigned the containers to a yard block and determined their location within the block using three heuristics: sequential stacking based on nominal prediction, stacking based on nominal and numerical prediction, and stacking based on numerical prediction. Zhu et al. (2020) proposed an integer model and two-stage search algorithm to solve the inbound containers unloading and stacking problem, which is defined as pre-processing the container relocation problem. Feng et al. (2022) developed a smart stacking (SS) strategy using customer

information to improve the inbound container retrieval efficiency at container terminals. They proposed non-split policy and split policy under the SS strategy. In addition, they developed two mixed-integer programming models for analyzing the non-split policy and a divide-and-conquer heuristic algorithm to efficiently solve the models.

In the CSP for outbound containers, the research was based on the evaluation criterion in which relocation movement occurs when a heavy container is stacked under a light container. Kim et al. (2000) and Zhang et al. (2010) proposed a dynamic programming model to determine the stacking position of outbound containers, which guaranteed an optimal solution that was impractical owing to the time-consuming. Therefore, a classification procedure based on the decision tree was developed for real-time decision-making. Kang et al. (2006) classified the weight of containers into three groups and used the simulated annealing technique to derive the stacking policy to minimize the relocation movement. Furthermore, an advanced stacking policy that can overcome the uncertainty in container weight information by a machine learning-based methodology was proposed. Dekker et al. (2007) proposed a category stacking policy considering the workload for an automated stacking crane in an automated container terminal (ACT). To verify the proposed methodology, the effect of the proposed policy was compared with that of the random stacking policy through simulation analysis. The proposed stacking policy, which stacked the containers based on the stowage plan, showed better performance than the random stacking policy. Park et al. (2011), as in the study of Dekker et al. (2007), assumed an ACT in their research and proposed a container stacking policy with dynamic adjustment to determine the stacking position according to changes in the environment. An online search algorithm was proposed by setting the weights of indicators through preliminary experiments. Chen & Lu (2012) investigated the stacking policy of stacking containers with similar weights in the same tier or stack. They proposed a hybrid stacking policy, which is an improvement of the vertical stacking policy, and showed that it had a higher reduction in the number of rehandles than the random and vertical stacking policies. Zhang et al. (2014a) proposed a conservative stacking policy as an extension to the dynamic programming model proposed by Kim et al. (2000) and Zhang et al. (2010). Two penalty methods were proposed depending on the containers to be stacked in the future. Zhang et al. (2014b) considered the uncertainty of the weight information considered by Kang et al. (2006) in the dynamic programming model proposed by Kim et al. (2000) and Zhang et al. (2010); the change in the proportion of the weight group of containers that have not arrived yet is considered. Güven and Eliiyi (2014) proposed a category stacking strategy in which outbound and inbound containers are categorized by those destined vessel and the owner company. For the outbound container stacking strategy, they additionally considered the port of destination sequence and the weight of the containers. Furthermore, Güven and Eliiyi (2019) included empty containers in the existing research scope and compared random stacking, attribute-based stacking, and weight-relaxed stacking using operational data from the port. He et al. (2020) investigated a stacking policy for an environment in which containers to be loaded on multiple vessels are stacked in the same bay. They investigated the stacking policy considering the uncertainty of the berthing sequence of vessels at the terminal, assuming that the weight of the containers and the arrival sequence of each container are known. Based on three stacking rules, namely least reshuffle rule, lowest stack rule, and nearest stack rule, five stacking policies were proposed according to the combination of these rules. Gunawardhana et al. (2021) developed a rule-based dynamic container stacking model consisting of a three-step methodology to minimize container handling costs and consider uncertainty. Park et al. (2022) proposed a data-driven dynamic stacking strategy that utilizes GMM to determine weight class-to-stack assignment. They developed a mechanism to adjust the stack configuration in response to container arrival.

3. GMM-based vertical stacking policy

In this section, we present a method for real-time decision-making of the stacking positions of incoming containers to the yard in the operation stage of container terminals. In the following section, the assumptions and definitions of the problem and proposed methodology are described in detail.

3.1. Assumptions and definitions of the problem

1. In the yard, the outbound and transshipment containers do not share the same storage space with the inbound containers (Kim et al., 2000).
2. Each outbound and transshipment container is classified into one of the container groups according to container-specific attributes such as the vessel to be loaded, destination, size, and container type.
3. Yard information expressed as a storage space of each container group is given in advance. A unit of storage space adjacent to the same container group is referred to as a 'sub area'. Furthermore, a sub area comprises a fixed number of stacks and has a size of at most one bay.
4. Each container group has one or more sub areas according to the number of containers, and arriving containers are allocated to sub areas considering the workload of the yard crane responsible for each sub area.
5. The total number of containers to be loaded on a vessel is given in a rough large number. Additionally, the weight of the container of each container group is assumed to have similar distribution for each voyage (Kim et al., 2000; Kang et al., 2006).
6. Each container group of outbound and transshipment containers is classified into a specific number of weight classes.

3.2. Proposed methodology

The workflow of the proposed GMM-based vertical stacking policy has the following three steps: (1) GMM creation to define the weight class of containers, (2) container stacking according to the vertical stacking policy, and (3) updating of the GMM parameters. As shown in Fig. 4, the algorithm of the proposed method creates the GMM for the container weight of each container group, and then performs the iterative step of stacking the container and updating the GMM parameters in response to the problem data. In the following section, detailed information on the three steps and a pseudo code of the entire algorithm are presented.

3.2.1. Step 1: GMM creation

In this study, the weight class of containers is defined based on the GMM. A mixture model is a probabilistic model that represents subpopulations within an overall population, and a GMM is a model that represents a subpopulation with a Gaussian distribution. The GMM can be obtained by marginalizing all latent variables in the form of a linear combination of several Gaussian distributions, which allows for the presentation of a complex distribution of observed variables that cannot be represented with a Gaussian distribution. Mixture models are used to estimate complex distributions by assuming these latent variables, and they are also used for clustering (McLachlan et al., 2019). When using GMM clustering for container weight data, the weight class of containers

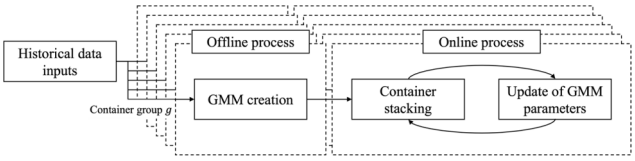


Fig. 4. Framework of the proposed methodology.

is represented in terms of each Gaussian distribution of the GMM, and it is possible to predict which weight class the new input data (container weight) will be classified into. The notations used in the GMM creation step is presented in Table 2.

The GMM is represented by three parameters of the mean, covariance, and weight of each Gaussian, and the probability density function of the container weight based these parameters is given as follows:

$$p(x|g) = \sum_{k \in K_g} \pi_k^g N(x|\mu_k^g, \Sigma_k^g) \tag{1}$$

For a maximum likelihood estimation of the GMM, an expectation–maximization (EM) algorithm is used. The EM algorithm comprises the expectation step (E step) for evaluating responsibility (see Equation (2)) from randomly initialized parameter values and a step for re-estimating the parameters (Maximization step; M step) based on the responsibility value obtained in the E step, and these two steps are iteratively carried out until there is the convergence of the parameters. The trained GMM can determine the weight class for the new container weight data by Equation (3).

$$\gamma(z_k^g|g) = \frac{\pi_k^g N(x|\mu_k^g, \Sigma_k^g)}{\sum_{j \in K_g} \pi_j^g N\left(x|\mu_j^g, \Sigma_j^g\right)} \tag{2}$$

$$k(i) = \underset{k \in K_g}{\operatorname{argmax}} \gamma(z_k^g|g) \tag{3}$$

3.2.2. Step 2: Container stacking

In this study, the GMM-based online optimization is used for real-time decision making of the stacking positions for the arriving containers. Generally, in vertical stacking, the preferred weight class is designated per stack of each sub area; thus, the container is placed on a yard stack according to the preferred weight class. If the yard stack is not available, the container is placed on a yard stack designated with a weight class closest to the original weight class. This study aims to store containers with similar weights in the same stack, and the vertical stacking policy is suitable for this purpose. The novelty of this study compared to the vertical stacking policy is that the weight class is derived using GMM clustering, one or more of the preferred weight classes are set for each yard stack, and the GMM-based stacking rule is applied to determine the container stacking position. The notations used in the container stacking step are listed in Table 3, and subsequently, the proposed container stacking method is described in detail.

First, the process of setting the preferred weight class (K_{cs}) for each yard stack is described. We consider a Gaussian weight parameter (π_k^g) so that a similar number of containers can be loaded for each yard stack.

Table 2
Notations in the GMM creation step.

Symbol	Definition
i	Index of container
g	Index of container group
x	Index of container weight
k	Index of the Gaussian distribution (container weight class)
G	Set of the container group g
K_g	Set of Gaussian k in container group g
$g(i)$	Container group where container i belongs
$x(i)$	Weight of container i
$k(i)$	Weight class of arriving container i predicted by the GMM
μ_k^g	Mean of Gaussian k in the GMM for container group g
Σ_k^g	Covariance of Gaussian k in GMM for container group g
π_k^g	Weight of Gaussian k in the GMM for container group g (weight parameter), $\sum_{k \in K_g} \pi_k^g = 1$
z_k^g	Latent variable that returns 1 if container weight x is created in the Gaussian (weight class) k, or 0 otherwise
$p(x\|g)$	Probability density function of container weight x in the GMM for the container group g
$\gamma(z_k^g\|g)$	Probability that container weight x is generated in Gaussian (weight class) k in the GMM for container group g (That is, responsibility)

Table 3
Notations in the container stacking step.

Symbol	Definition
c	Index of sub area
s	Index of yard stack, $s \in \{1, 2, \cdots, \|S_c\|\}$
C_g	Set of sub area c allocated to container group g
S_c	Set of yard stack s composing sub area c
K_{cs}	Set of weight class k preferred in sub area c and stack s
S_{ck}	Set of yard stack s in which container of weight class k can be stacked within the sub area c
K_c	Set of weight class k that can be stacked in sub area c
R_{cs}	Range of cumulative ratio proportional to the number of stacks for sub area c and stack s
R_k^g	Range of cumulative mixture weight values π_k^g for weight class k
$c(i)$	Sub area already allocated for arriving container i
t_{cs}	Number of spare containers for stacking in sub area c and stack s
n_g	Number of total containers that have not yet arrived, which belong to the container group g
X_{cs}^g	List of weights of the containers stacked in sub area c and stack s, which belong to container group g

The Gaussian weight parameter is a GMM mixing coefficient, which determines the probability that each Gaussian is selected, and it serves as a parameter that represents the size of each Gaussian. Therefore, the range of the weight parameter (R_{cs}) included in each yard stack is defined by Equation (4) to be proportional to a storage capacity of each stack. For example, if a sub area consists of five yard stacks and all the stacks have the storage capacity of six containers, as shown in Fig. 5, the weight range for each stack is set to $(0.0, 0.2]$, $(0.2, 0.4]$, …, $(0.8, 1.0]$. Subsequently, the range of the weight parameter for each weight class (R_k^g) is defined by Equation (5), based on the cumulative weights. Then, the preferred weight class for each yard stack is determined by Equation (6). For example, if three weight classes are assigned to a sub area and the weight range R_k^g is equivalent to $(0.0, 0.125]$, $(0.125, 0.875]$, $(0.875, 1.0]$ for each weight class, the preferred weight classes K_{cs} are classes 1 and 2 for stack 1, class 2 for stacks 2 to 4, and classes 2 and 3 for stack 5.

$$R_{cs} = \left\{ r \middle| \frac{s-1}{|S_c|} < r \leq \frac{s}{|S_c|}, r \in \mathbb{R} \right\} \tag{4}$$

$$R_k^g = \left\{ r \middle| \sum_{\substack{k' \in K_g \\ k' < k}} \pi_{k'}^g < r \leq \sum_{\substack{k' \in K_g \\ k' \leq k}} \pi_{k'}^g, r \in \mathbb{R} \right\} \tag{5}$$

$$K_{cs} = \left\{ k \middle| R_{cs} \cap R_k^g \neq \emptyset, k \in K_g \right\} \tag{6}$$

Next, the procedure for determining the yard stack to store the

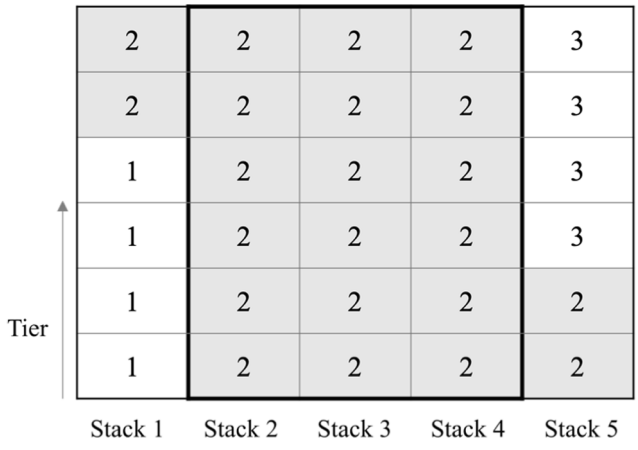


Fig. 5. Examples of yard bay configuration (three weight classes).

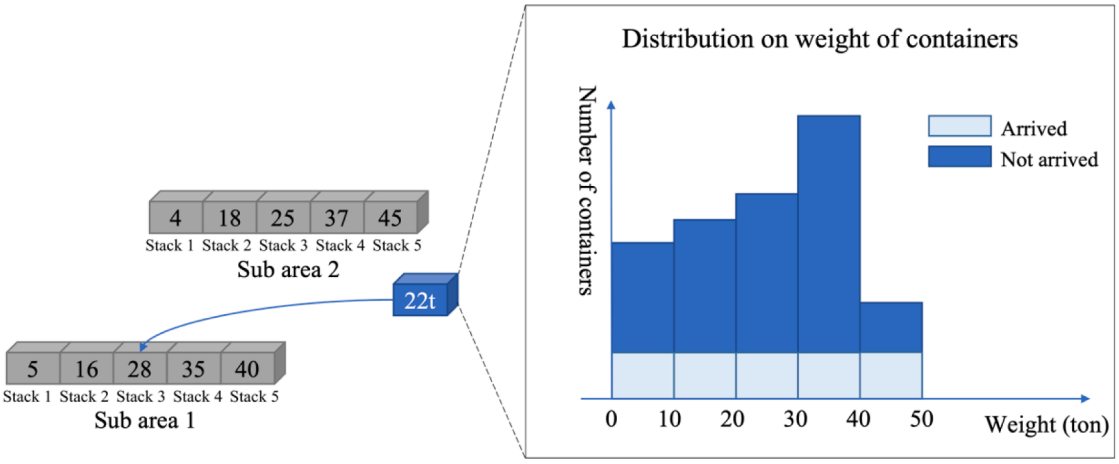
containers is described. According to the procedure described above, there may be multiple yard stacks that can be selected based on a single weight class, as shown in Fig. 5. In the stacking policy of this study, the yard stacks (bold edge section; stack 2 to 4) with a single designated weight class are prioritized. Among these yard stacks, if all are unavailable, then the (thin edge section; stack 1 and 5) jointly preferred yard stacks with other weight classes are considered in the next step. At this time, if the container is close to weight class 1, it is placed on stack 1, and if it is close to weight class 3, it is placed on stack 5. The rule for determining these priorities can be summarized as selecting a stack that maximizes $\frac{\sum_{k\in K_{cs}}\gamma(z_k^g|g)}{|K_{cs}|}$, the average responsibility of the GMM, among the available stacks in a sub area.

In summary, in the container stacking step, (1) the Gaussian weight π_k^g of the GMM is used for setting the preferred weight class for each yard stack, and (2) the responsibility $\gamma(z_k^g|g)$ of the GMM is used for determining the yard stack to store the container. The GMM-based container stacking step is closely linked to the GMM creation step, which defines the original container weight class.

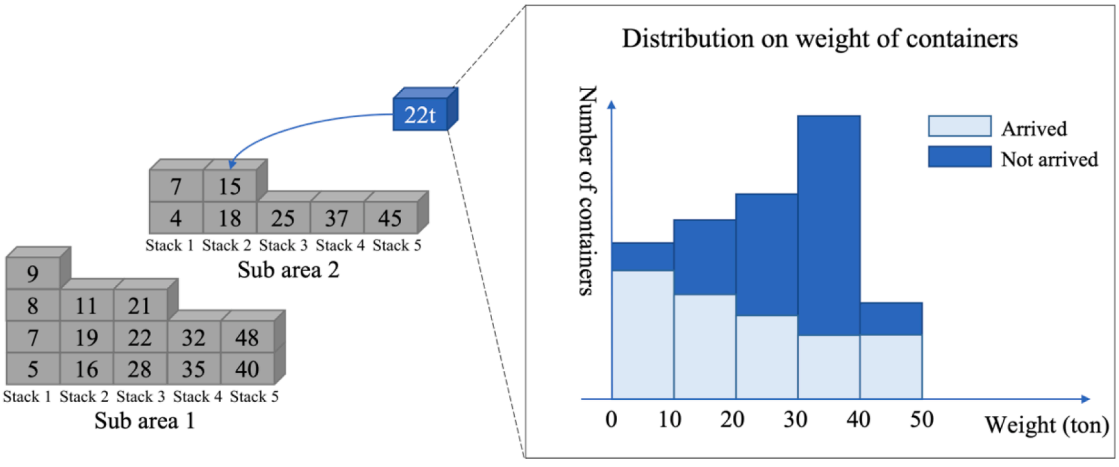
3.2.3. Step 3: Update of GMM parameters

The proposed GMM-based online optimization includes updating the parameters of the GMM every time a container is stacked one at a time. Because the parameters determine the GMM, the update of the parameters changes the responsibility derived from the GMM and affects the definition of the weight class of the newly arriving container. In contrast to using the fixed criteria of weight class in the conventional vertical stacking policy, the proposed method is novel in that it applies a dynamic weight class criterion that updates parameters based on the collected information on the weight of the containers that have already arrived.

The novelty attained by the GMM parameter update process is described with examples in Fig. 6, which shows a situation of stacking containers; although the containers have the same weight in each case, in one case (Fig. 6(a)), a container is placed on the stack 3 in the 11th arrival sequence, and in the case of the 22nd arrival sequence, shown in Fig. 6(b), it is placed on the stack 2. Compared with the conventional vertical stacking policy where containers of a particular weight are always stacked on the same stack in all sub areas, our methodology, with the accumulation of containers, handles containers with the same weight differently depending on the situation. The main idea implied in the proposed method is to utilize the weight distribution of containers that have not yet arrived. When the total number of containers to arrive and the weight distribution of containers are given by assumption (5),



(a) Stacking the container which is arrived for the 11th time (container weight = 22 tons)



(b) Stacking the container which is arrived for the 22nd time (container weight = 22 tons)

Fig. 6. Example of a stacking process with the GMM parameter update.

the weight distribution of containers to arrive can be re-estimated as the identified information of the weight of containers that have arrived. In Fig. 6(b), unlike in the case of Fig. 6(a), because the weight distribution of the containers that have arrived is somewhat skewed toward light weights, the weight distribution of containers that have not yet arrived will be skewed toward heavy weights. Therefore, based on the described property of the probability distribution, the weight class of the containers can be derived dynamically, and in this manner, containers that will arrive in the future can be stored at similar weights over the entire stacks, which is an effect that fits the purpose of this study.

The target of the GMM parameter update is Gaussian weights. As mentioned above, the Gaussian weight is a parameter representing the size of each Gaussian of the GMM. Through Equation (7), the update is performed as follows: When the weight class of the currently stacked container i is $k(i)$, the size of $k(i)$ is decreased by reflecting the number of the stacked container, and the sizes of the Gaussian weights of the other weight classes are increased by reflecting a similar change. For this reason, the updated Gaussian weight still satisfies $\sum_{k\in K_g}\pi_k^g = 1$. To prevent the Gaussian weight corresponding to weight class $k(i)$ from being updated to a value of 0 or to negative values, only when the condition $\pi_{k(i)}^g \le \frac{1}{n_g}$ is satisfied is Equation (8) applied, instead of Equation (7), to replace the value to be updated with a sufficiently small positive real value ϵ.

$$\pi_k^g = \begin{cases} \dfrac{\pi_k^g \times n_g - 1}{n_g - 1}, & for\ k = k(i) \\ \dfrac{\pi_k^g \times n_g}{n_g - 1}, & for\ k \in K_g, k \neq k(i) \end{cases} \tag{7}$$

$$\pi_k^g = \begin{cases} \epsilon, & for\ k = k(i) \\ \dfrac{\pi_k^g \times n_g}{n_g - 1} - \dfrac{\pi_k^g}{\sum_{k' \in K_g,\ k' \neq k(i)} \pi_{k'}^g}\left(\epsilon - \dfrac{\pi_{k(i)}^g \times n_g - 1}{n_g - 1}\right), & for\ k \in K_g, k \neq k(i) \end{cases} \tag{8}$$

3.2.4. Pseudocode for the GMM-based vertical stacking policy

The flow of the GMM-based vertical stacking policy, which encompasses steps 1 to 3 as described above, is shown in the pseudo-code below.

Algorithm 1 GMM-based vertical stacking policy

Input: Container i

Let $g = g(i), x = x(i), c = c(i)$
Determine weight class $k(i)$ by Equation (3)
Determine the closest available weight class k'; $k' = \underset{k\in K_c}{\operatorname{argmax}} \gamma(z_k^g|g)$
Select element s from $S_{ck'}$ with the largest value of $\frac{\sum_{k\in K_{cs}}\gamma(z_k^g|g)}{|K_{cs}|}$
if more than one element leads to the largest value **then**
  Randomly choose element s from the set with the largest value
end if
if $n_g \neq 1$ **and** $\pi_{k(i)}^g > \frac{1}{n_g}$ **then**
  Update GMM parameter π_k^g by Equation (7)
else if $n_g \neq 1$ **and** $\pi_{k(i)}^g \le \frac{1}{n_g}$ **then**
  Update GMM parameter π_k^g by Equation (8)
end if
Update as $t_{cs} = t_{cs} - 1$, $n_g = n_g - 1$ and add x to X_{cs}^g
if $t_{cs} \le 0$ **then**
  Remove s from S_{ck} and update as $K_c = \{k | k \in K_g, S_{ck} \neq \varnothing\}$
end if

4. Numerical experiment

We present the results of the numerical experiment conducted to verify the performance of the proposed container stacking method. In the following sections, the experimental environment and datasets are described, results of the GMM creation for the experimental datasets are presented, and finally, the proposed stacking method is evaluated in comparison with three polices already existing in practice and/or in literature.

4.1. Experimental environment and dataset

To analyze the effect of the proposed methodology in the same setting as the actual operating environment of the container terminal in Busan New Port, one bay comprised 10 stacks and five tiers, and for the outbound and transshipment containers, the size of the sub area was set to five and 10 stacks where the size of one bay is equivalent to 10 stacks. Thus, the container storage capacity in a sub area is equivalent to 25 TEU and 50 TEU, respectively. Additionally, using practical container data collected for 10 months at the container terminal in Busan New Port, container groups were made based on the same container-specific attributes (e.g., the vessel to be loaded, destination, size, and type of containers), constructing a total of nine experimental datasets. As shown in Fig. 7, the experimental datasets have similar container-weight distribution for each voyage. Numerical experiments were performed on an Intel Core i7-7700HQ 2.80 GHz, 16.0 GB RAM, and Windows 10 OS, and the algorithm was programmed in Python language. The results of the proposed algorithm were derived within a short time of less than 1 s, making it reasonable as a real-time online algorithm.

4.2. Results of the GMM

The weight class of containers was derived by applying the GMM clustering technique to the experimental datasets. The incoming container data for all voyages, except for one randomly selected voyage, was used as the historical data for GMM creation, and the incoming container data for the remaining voyage was used as the test data for verification of the container stacking policy. In each experimental dataset, the number of weight classes of containers (i.e., the number of GMM clusters) was selected as the number with a minimum Bayesian information criterion (BIC) score. Fig. 8 shows the change in BIC score according to the number of GMM clusters, and Fig. 9 shows the distribution curve of the GMM with five clusters in the experimental dataset based on the BIC. The data histogram in Fig. 9 shows that the distribution of container weight has a shape with multiple peaks which is far from a shape of a single normal distribution. From the results of the GMM, the validity of using GMM clustering for the definition of weight class in this study is also verified.

4.3. Results of comparative analysis

The performance of the proposed container stacking policy is evaluated in comparison with the current practice, hybrid sequence stacking policy and random stacking policy for the experimental datasets. In the current practice following the vertical stacking policy, the weight class of the containers is classified into five equally divided ranges, and one preferred weight class is allocated for each yard stack. The hybrid sequence stacking policy was designed to balance the effectiveness of the vertical stacking policy and horizontal stacking strategy, so it induces containers with heavier weight classes to be stacked in the left upper locations relative to the diagonal (Chen & Lu, 2012). In the random stacking policy, the containers are placed on the stacks according to the arrival sequence without specific criteria. Although the hybrid sequence stacking policy and random policy are not complete vertical stacking approaches, they can provide a baseline on how much the proposed policy and the current practice conform to the vertical stacking policy.

Because the vertical stacking policy aims to stack containers of similar weight in one yard stack, it is evaluated in terms of the weight

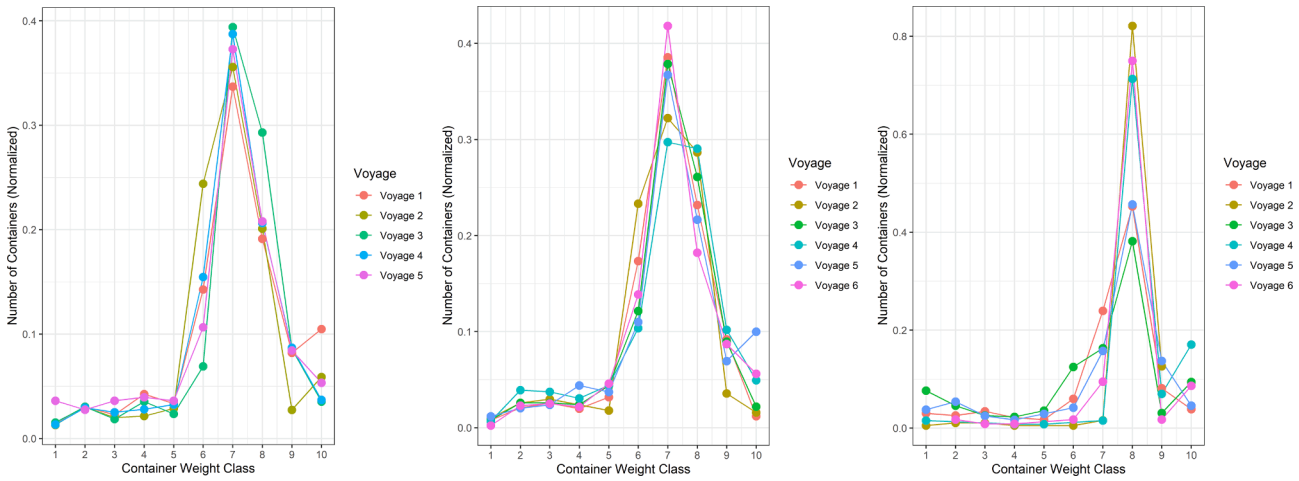


Fig. 7. Example of the distribution of container weight by voyage (instance 2, 6 and 8).

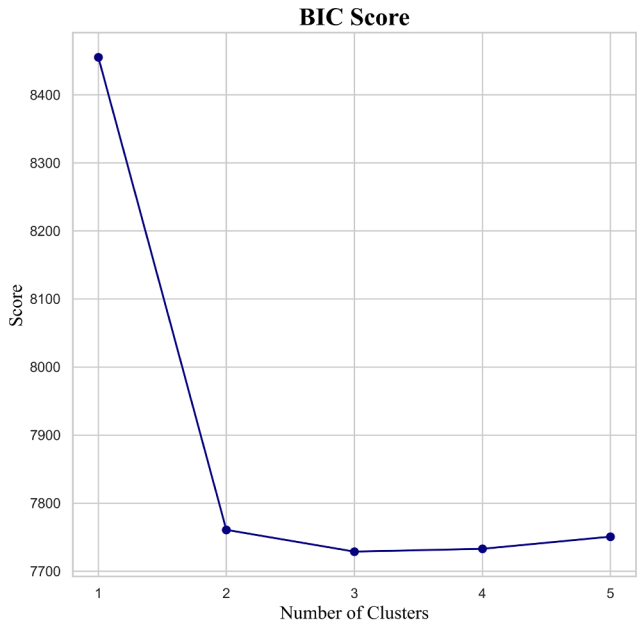


Fig. 8. BIC score with respect to number of GMM clusters (instance 5).

variance of the stacked containers for all stacks. Therefore, the following evaluation function was used to evaluate the stacking policy:

$$E(g) = \frac{\sum_{c \in C_g} \sum_{s \in S_c} Var\left(X_{cs}^{g}\right)}{\sum_{c \in C_g} |S_c|} \tag{9}$$

For comparison of the four different stacking policies, each policy was tested 100 times in the same environment by random shuffling of the incoming sequence for the container data subsets of the one voyage that was not used for GMM generation in the experimental datasets. The sub area was predetermined, and the sub areas were arbitrarily allocated to each randomly shuffled container. The total number of sub areas was set to the minimum number required to match the number of incoming containers in the experimental datasets (the same container group), and the random allocation of sub areas was limited by the condition of not exceeding the storage capacity.

Table 4 is the result of $E(g)$ for the average of 100 tests for the entire experimental datasets for the case of five stacks. The dataset covers containers loaded on a voyage for different vessels. Considering that the shipment size (number of containers) and the original container weight distribution are all different for each vessel, the average $E(g)$ is also yielded differently for each dataset. Still, the results show that the proposed stacking policy showed smaller results of $E(g)$ for all the experimental datasets compared with the other three policies and was approximately an average of 37 %, 41 % and 43 % better than the current practice, hybrid sequence stacking policy and random stacking policy, respectively. This result indicates that the proposed stacking policy worked to best fit the aim of the vertical stacking policy.

For the purpose of verifying the overall performance of the proposed stacking policy, Table 5 report the result of $E(g)$ for the average of 100 tests for the entire experimental datasets for case 10 stacks. Fig. 10 is a box plot for the 100 tests for both cases. Consistent with previous results in Table 4, the proposed stacking policy showed smaller results of $E(g)$ on average compared with the other three policies and was approximately 44 %, 77 % and 85 % better than the current practice, hybrid sequence stacking policy and random stacking policy, respectively. As shown in Fig. 10, it is observed that there is room for a further reduction of the average $E(g)$ through the vertical stacking policy when the number of stacks is 10 compared to five, and the variance of $E(g)$ can also be reduced where the reduction in variance means the robustness of performance.

Fig. 11 shows yard bay configuration results for the proposed stacking policy and current practice, and is the case where the proposed policy worked effectively. The numbers in the square indicate the arriving sequence in the yard bay and the container weight. When following the proposed stacking policy, $E(g)$ for the example sub area is reduced by 91 % against one for the current practice. In particular, it can be observed that $E(g)$ is drastically reduced by allocating containers to efficient locations in consideration of uncertainties. This is because even if the weight class is properly predetermined, the ratio of the weight class of the container allocated to one bay is changed. Therefore, the current practice has limitations in responding to uncertainty due to container weight and arrival order. Fig. 12 reports the changes in $E(g)$ as containers are stacked one by one in a specific sub area of instance 3. For all stacking policies, $E(g)$ tends to increase as the number of containers approaches the capacity of the sub area. However, the proposed stacking policy outperformed the other stacking policies with keeping stable variation in $E(g)$. This is because, unlike other policies, the proposed stacking policy responded efficiently to uncertainties while updating policies according to the situation. Therefore, we believe that robust performance of the proposed stacking policy can be expected in the weight uncertainty of containers for the practical yard operation in port container terminals.

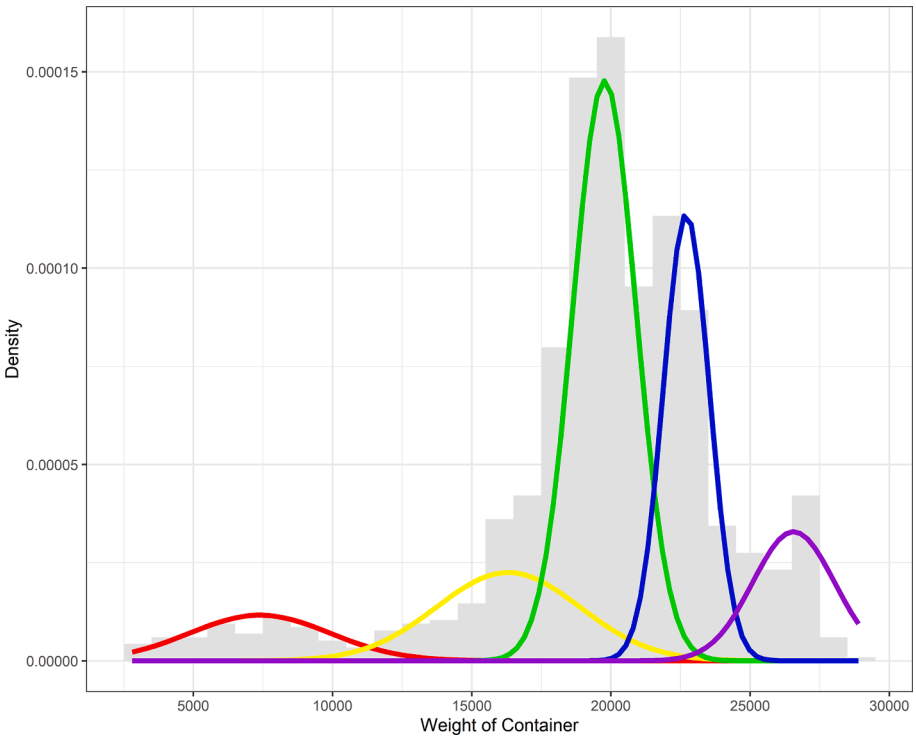


Fig. 9. Distribution curve of the GMM (instance 6).

Table 4
Results of comparison between four types of stacking policies for the case of five stacks.

Instance no.	Number of containers	Average $E(g)$			
		Proposed stacking policy	Current stacking policy in practice	Hybrid sequence stacking policy	Random stacking policy
1	855	21.6	21.4	36.4	37.4
2	582	12.2	25.6	29.6	31.2
3	116	9.0	10.1	10.5	10.6
4	580	15.1	27.6	29.3	29.6
5	236	8.3	12.0	11.7	11.5
6	873	10.2	19.3	21.3	21.6
7	642	11.7	22.4	25.9	27.0
8	116	8.4	9.9	10.4	10.6
9	578	12.0	23.3	27.2	28.4

All average $E(g)$ are scaled by a factor of 10^{-6}.

Table 5
Results of comparison between four types of stacking policies for the case of 10 stacks.

Instance no.	Number of containers	Average $E(g)$			
		Proposed stacking policy	Current stacking policy in practice	Hybrid sequence stacking policy	Random stacking policy
1	855	8.2	8.3	17.0	29.5
2	582	2.2	2.8	10.1	24.2
3	116	1.6	3.0	6.7	8.1
4	580	6.4	8.5	15.5	23.1
5	236	1.0	4.1	7.3	8.8
6	873	1.1	2.1	9.0	17.2
7	642	1.6	3.2	10.2	21.1
8	116	1.6	4.9	8.6	8.1
9	578	1.3	3.4	9.8	22.3

All average $E(g)$ are scaled by a factor of 10^{-6}.

5. Conclusion

With an increase in volume and development of container transportation, the efficient operation of the limited resources of container terminals has emerged as a significant issue for hub terminals. One source of inefficiency is from the operation of yard cranes. The yard cranes handle the containers located on the top tier of the yard block first before handling containers at the bottom. Therefore, an inefficient stacking policy causes container relocation movements in future loading operations, resulting in delays in the process and a bottleneck in the container flow. To resolve this problem, in this study, a GMM-based vertical stacking policy that determines the stacking positions of the outbound and transshipment containers arriving at the terminal was investigated.

The proposed GMM-based vertical stacking policy works according to the following three steps. The first step is the creation of GMM, which defines the weight class of containers; an appropriate weight class is generated for each container group based on historical data. The next step is container stacking according to the vertical stacking policy, and in this policy, a stack with the largest average value of responsibility of GMM is selected among the available stacks. The final step is the update of the GMM parameters, and the initial GMM parameters derived for each container group are updated dynamically in response to the changing sub area. To verify the proposed methodology, numerical experiments were performed to compare the results with those of the planner using the data from the Busan container terminals. When comparing the $E(g)$ of the random and current stacking policies with the proposed stacking policy, the proposed stacking policy outperformed other policies. The results show that containers can be stacked to facilitate flexible responses to various situations with uncertainties and reduce the time taken for container relocation movements.

Future research could focus on extending our stacking policy to proactive approach in the two aspects. Firstly, predictive methodologies can be applied to reduce uncertainty, which is a characteristic of the

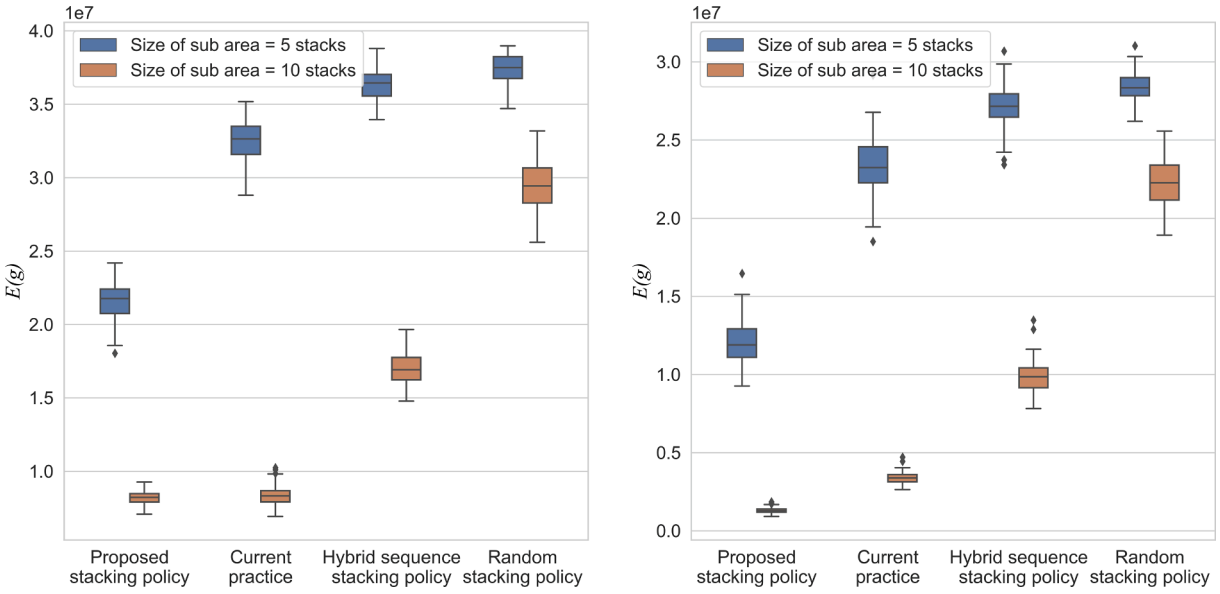


Fig. 10. Box plot for $E(g)$ for the four stacking policies (instance 1 and 9).

24th 4,950	**14th** 17,955	**22nd** 22,330	**15th** 23,125	
18th 5,903	**12th** 15,904	**20th** 22,720	**5th** 22,516	**23rd** 23,450
13th 2,772	**11th** 19,327	**19th** 22,260	**4th** 22,460	**21st** 26,272
10th 12,930	**8th** 18,564	**17th** 17,809	**2nd** 22,293	**16th** 26,860
9th 6,166	**7th** 17,738	**3rd** 21,594	**1st** 23,650	**6th** 26,260
Stack 1	Stack 2	Stack 3	Stack 4	Stack 5

Proposed stacking policy
($E(g)$ =4,576,819 for this sub area)

	21st 26,272	**17th** 17,809	**12th** 15,904	**5th** 22,516
24th 4,950	**20th** 22,720	**16th** 26,860	**11th** 19,327	**4th** 22,460
23rd 23,450	**19th** 22,260	**15th** 23,125	**8th** 18,564	**3rd** 21,594
22nd 22,330	**18th** 5,903	**14th** 17,955	**7th** 17,738	**2nd** 22,293
13th 2,772	**9th** 6,166	**10th** 12,930	**6th** 26,260	**1st** 23,650
Stack 1	Stack 2	Stack 3	Stack 4	Stack 5

Current practice
($E(g)$ =52,668,380 for this sub area)

Fig. 11. Example of test results for the proposed stacking policy and current practice (instance 7).

CSP. In this study, outbound and transshipment containers are handled the same in that they are loaded on the ship. However, unlike outbound containers, transshipment containers arrive at terminals through ships, so information on transshipment containers (e.g., type of container group, number of containers, arrival timing of containers etc.) can be predicted using machine learning techniques. It would be interesting to extend our method to a stack policy that considers the characteristics of transshipment containers based on various data collected from the port. Secondly, robustness can be considered to minimize the negative impacts caused by uncertainty. In this study, we assume the random bay-level allocation of outbound containers because the workload of YC fluctuates in real-time. However, in some cases, containers are concentrated in one bay, reaching the local optimum. Therefore, it is an interesting research topic to incorporate bay-level decisions considering the job queue of YCs into our method, which allows us to utilize the storage space of multiple bays to obtain a more robust and optimal solution.

CRediT authorship contribution statement

Sung Won Cho: Conceptualization, Methodology, Validation, Formal analysis, Software, Writing – original draft, Writing – review & editing, Project administration, Supervision. **Hyun Ji Park:** Conceptualization, Methodology, Formal analysis, Writing – original draft, Writing – review & editing. **Armi Kim:** Conceptualization, Methodology, Software, Data curation, Visualization, Writing – original draft. **Jin Hyoung Park:** Writing – review & editing, Project administration, Supervision, Funding acquisition.

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.

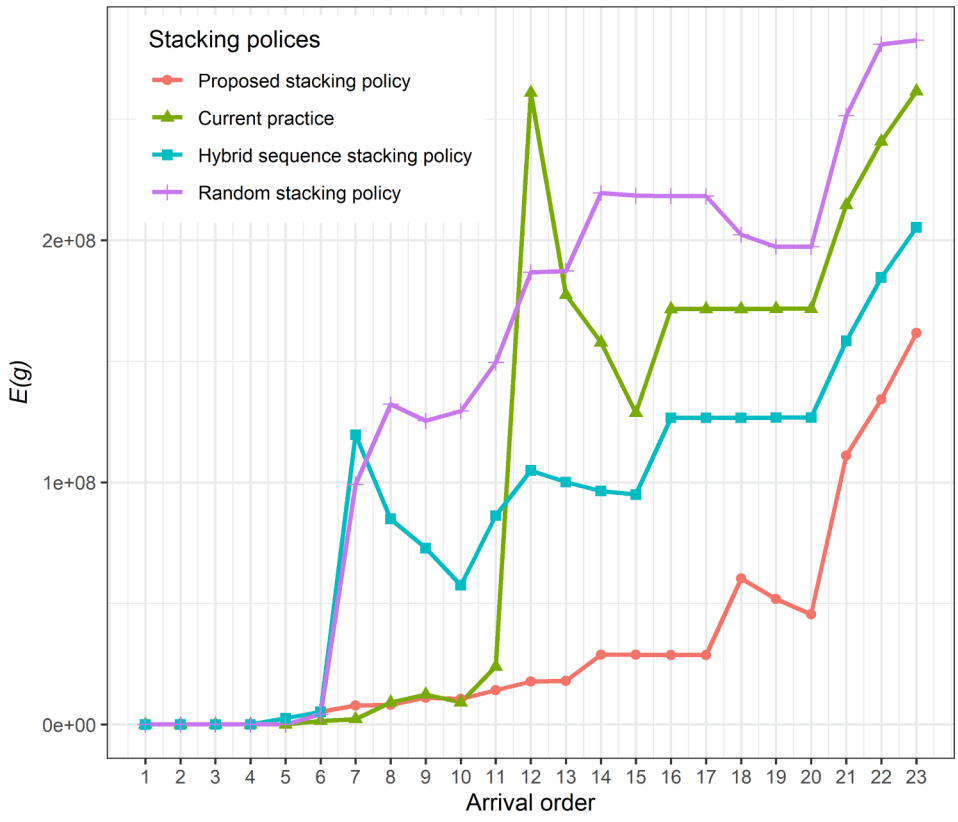


Fig. 12. Changes in $E(g)$ according to the order of container arrival (instance 3).

Data availability

The data that has been used is confidential.

Acknowledgement

This research was supported by Korea Research Institute of Ships and Ocean Engineering a grant from Endowment Project of "Development of Open Platform Technologies for Smart Maritime Safety and Industries" funded by Ministry of Oceans and Fisheries(1525012040)

References

Borgman, B., Van Asperen, E., & Dekker, R. (2010). Online rules for container stacking. *OR spectrum, 32*(3), 687–716.

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), 412–430.

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

Cho, S. W., Park, H. J., & Lee, C. (2021). An integrated method for berth allocation and quay crane assignment to allow for reassignment of vessels to other terminals. *Maritime Economics & Logistics, 23*(1), 123–153.

Corne, D., Dhaenens, C., & Jourdan, L. (2012). Synergies between operations research and data mining: The emerging use of multi-objective approaches. *European Journal of Operational Research, 221*(3), 469–479.

Covic, F. (2017). Re-marshalling in automated container yards with terminal appointment systems. *Flexible Services and Manufacturing Journal, 29*(3), 433–503.

Covic, F. (2018). In *October). A literature review on container handling in yard blocks* (pp. 139–167). Cham: Springer.

de Castillo, B., & Daganzo, C. F. (1993). Handling strategies for import containers at marine terminals. *Transportation Research Part B: Methodological, 27*(2), 151–166.

Dekker, R., Voogd, P., & Van Asperen, E. (2007). In *Advanced methods for container stacking* (pp. 131–154). Berlin, Heidelberg: Springer.

Feng, Y., Song, D. P., & Li, D. (2022). Smart stacking for import containers using customer information at automated container terminals. *European Journal of Operational Research, 301*(2), 502–522.

Filom, S., Amiri, A. M., & Razavi, S. (2022). Applications of machine learning methods in port operations–A systematic literature review. *Transportation Research Part E: Logistics and Transportation Review, 161*, Article 102722.

Gaete, M., González-Araya, M. C., González-Ramírez, R. G., & Astudillo, C. (2017, February). A dwell time-based container positioning decision support system at a port terminal. In International Conference on Operations Research and Enterprise Systems (Vol. 2, pp. 128-139). SciTePress.

Gunawardhana, J. A., Perera, H. N., & Thibbotuwawa, A. (2021). Rule-based dynamic container stacking to optimize yard operations at port terminals. *Maritime Transport Research, 2*, Article 100034.

Güven, C., & Eliiyi, D. T. (2014). Trip allocation and stacking policies at a container terminal. *Transportation Research Procedia, 3*, 565–573.

Güven, C., & Eliiyi, D. T. (2019). Modelling and optimisation of online container stacking with operational constraints. *Maritime Policy & Management, 46*(2), 201–216.

He, Y., Wang, A., & Su, H. (2020). The impact of incomplete vessel arrival information on container stacking. *International Journal of Production Research, 58*(22), 6934–6948.

Jaillet, P., & Wagner, M. R. (2010). Online Optimization—An introduction in risk and optimization in an uncertain world (pp. 142–152). INFORMS.

Kang, J., Ryu, K. R., & Kim, K. H. (2006). Deriving stacking strategies for export containers with uncertain weight information. *Journal of Intelligent Manufacturing, 17* (4), 399–410.

Kim, A., Park, H. J., Park, J. H., & Cho, S. W. (2021). Rescheduling strategy for berth planning in container terminals: An empirical study from Korea. *Journal of Marine Science and Engineering, 9*(5), 527.

Kim, K. H. (1997). Evaluation of the number of rehandles in container yards. *Computers & 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., Park, Y. M., & Ryu, K. R. (2000). Deriving decision rules to locate export containers in container yards. *European Journal of Operational Research, 124*(1), 89–101.

Maldonado, S., González-Ramírez, 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.

McLachlan, G. J., Lee, S. X., & Rathnayake, S. I. (2019). Finite mixture models. *Annual review of statistics and its application, 6*, 355–378.

Notteboom, T., Pallis, T., & Rodrigue, J. P. (2021). Disruptions and resilience in global container shipping and ports: The COVID-19 pandemic versus the 2008–2009 financial crisis. *Maritime Economics & Logistics*, 1–32.

Park, H. J., Cho, S. W., & Lee, C. (2021). Particle swarm optimization algorithm with time buffer insertion for robust berth scheduling. *Computers & Industrial Engineering, 160*, Article 107585.

Park, H. J., Cho, S. W., Nanda, A., & Park, J. H. (2022). Data-driven dynamic stacking strategy for export containers in container terminals. *Flexible Services and Manufacturing Journal*, 1–26.
Park, T., Choe, R., Kim, Y. H., & Ryu, K. R. (2011). Dynamic adjustment of container stacking policy in an automated container terminal. *International Journal of Production Economics, 133*(1), 385–392.
Reynolds, D. A. (2009). Gaussian mixture models. *Encyclopedia of biometrics, 741* (659–663).
Sauri, S., & Martin, E. (2011). Space allocating strategies for improving import yard performance at marine terminals. *Transportation Research Part E: Logistics and Transportation Review, 47*(6), 1038–1057.
Ting, S. C., Wang, J. S., Kao, S. L., & Pitty, F. M. (2010). Categorized stacking models for import containers in port container terminals. *Maritime Economics & Logistics, 12*(2), 162–177.
Zhang, C., Chen, W., Shi, L., & Zheng, L. (2010). A note on deriving decision rules to locate export containers in container yards. *European Journal of Operational Research, 205*(2), 483–485.
Zhang, C., Liu, J., Wan, Y. W., Murty, K. G., & Linn, R. J. (2003). Storage space allocation in container terminals. *Transportation Research Part B: Methodological, 37*(10), 883–903.
Zhang, C., Wu, T., Kim, K. H., & Miao, L. (2014a). Conservative allocation models for outbound containers in container terminals. *European Journal of Operational Research, 238*(1), 155–165.
Zhang, C., Wu, T., Zhong, M., Zheng, L., & Miao, L. (2014b). Location assignment for outbound containers with adjusted weight proportion. *Computers & Operations Research, 52*, 84–93.
Zhen, L., Jiang, X., Lee, L. H., & Chew, E. P. (2013). A review on yard management in container terminals. *Industrial Engineering and Management Systems, 12*(4), 289–304.
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.