https://doi.org/10.1016/j.trb.2015.12.006

Multi-Period Yard Template Planning in Container Terminals

Lu Zhen[1*], Zhou Xu[2], Kai Wang[3]

[1] *School of Management, Shanghai University, Shanghai, China*

[2] *Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University, Hung Hom, Hong Kong*

[3] *Department of Industrial and Systems Engineering, National University of Singapore, Singapore*

Abstract: This paper is about yard management in container ports. As a tactical level decision-making tool in a port, a yard template determines the assignment of spaces (subblocks) in a yard for arriving vessels, which visit the port periodically. The objective of yard template planning is to minimize the transportation cost of moving containers around the yard. To handle yard template planning, a mixed integer programming model is proposed that also takes into account traffic congestion in the yard. A further complication is that the cycle time of the vessels' periodicities is not uniform and varies among them, perhaps being one week, ten days, or two weeks, etc. However, this multiple cycle time of the periodicities of vessel arrival patterns, which complicates the yard template decision, is also considered in the model. Moreover, a local branching based solution method and a Particle Swarm Optimization based solution method are developed for solving the model. Numerical experiments are also conducted to validate the effectiveness of the proposed model, which can save around 24% of the transportation costs of yard trucks when compared with the commonly used First-Come-First-Served decision rule. Moreover, the proposed solution methods can not only solve the proposed model within a reasonable time, but also obtain near-optimal results with about 0.1~2% relative gap.

Keywords: Maritime logistics; Port operation; Yard template; Container terminals; Congestion.

1. Introduction

Since the 1990s, world container traffic has been growing at almost three times the world's GDP growth, due to the offshoring of manufacturing operations in Asia, in particular China (Meng et al., 2014). Port throughputs have increased even faster, because an increasing number of containers are transshipped (Fransoo and Lee, 2012). Efficient port operations that maximize the throughput (with ports being paid by a handling charge per container) are essential for port operators' profits.

* Corresponding author. Email: lzhen@shu.edu.cn. Tel: +86-21-66137925. Fax: +86-21-66134284.

With the advancement of quay side equipment and technologies (e.g., twin 40-ft quay cranes, indented berths), the bottleneck in port operations has moved from quay side to yard side (Stahlbock and Voß, 2008). The yard management of a port plays a significant part in its competitiveness within the global shipping network. For some large container transshipment ports such as the Port of Singapore, the yard management is significantly important because of its land scarcity, which results in a highly concentrated storage situation within the storage yard (Jin et al., 2014). The yard template is a concept applied in container ports, especially transshipment hubs, that utilizes consignment strategy. This strategy stores export and transshipped containers, which will be loaded onto the same departing vessel, at the same assigned storage locations. Yard template planning is concerned with the assignment to vessels of storage locations (subblocks) in the yard, with certain dedicated subblocks being reserved for each vessel. Yard template planning aims to minimize the transportation cost for moving containers from their incoming berths to the storage subblocks in the yard and then to their outgoing berths. Besides the yard template based consignment strategy, housekeeping strategy is also a commonly used yard management strategy in ports (Giallombardo et al., 2010). The housekeeping strategy is to retrieve the containers that will be loaded onto a vessel, move and store them in some areas near to the berthing position of the vessel before it arrives at the port. This strategy can also reduce the turnaround time of vessels but may need reshuffling when retrieving containers as well as additional movement of containers before the vessel arrives at the port. However, the yard template based consignment strategy can avoid the container reshuffling activities. This study will be based on this yard management strategy.

Traffic congestion is the most significant issue that constrains the efficiency of yard side processes (Lee et al., 2006; Han et al., 2008), and is a phenomenon that prevents yard trucks from traveling freely (Zhang et al., 2009). Typical events observed in this phenomenon are that a lot of yard trucks may crowd around certain small areas for loading (or unloading) containers from (or to) yard storage locations (i.e., subblocks), or a lot of yard trucks may travel along a particular passing lane simultaneously, which makes them have to slow down during travel. Without taking into account these traffic congestion issues, a single-period yard template planning decision is just a generalized assignment problem that is only concerned with decisions on a *spatial* dimension. However, the traffic issue brings about constraints on a *temporal* dimension, such that neighboring subblocks should not have heavy loading and unloading activities occurring at the same time, and that each passing lane should not have significant traffic flows at the same time. These two phenomena are affected by the yard template planning decisions.

Moreover, vessels (shipping liners) usually visit the port periodically, and have different cycle times. For example, some vessels have a weekly arrival pattern, but some may have ten-day or biweekly patterns (Moorthy and Teo, 2006). Nowadays, Daily Maersk even has a daily arrival pattern at each port of call, and many feeder services have twice-weekly (APL, 2015a) or thrice-weekly services (APL, 2015b). Moreover, even if a service is weekly, the ships that visit the port each week may be of different size. For the example of Columbus Loop in CMA-CGM shipping company, the ship size varies from 8197 TEUs to 9034 TEUs (CMA-CGM, 2015). In reality, the weekly pattern is the most common in actual practice. If all the vessels had a uniform cycle time, there would be no multi-period decision problem for the yard template planning. However, this paper studies the multi-period yard template planning problem by also considering the heterogeneous periodicities of vessels. It should be noted that the subblock assignment for each vessel can also vary in its periods, so a subblock needs to not be assigned to one vessel for the whole planning horizon. This dynamic nature of the subblock assignment within each vessel's multiple periods further complicates the aforementioned decisions problem involved with yard template planning. Moreover, an even more common situation is that only about half of container vessels arrive at ports on their planned day (in other words, about half of container vessels are delayed at least one day). Consequently, even if all ships are weekly pattern, port operators still have to adjust the yard templates every day, which further validates the necessity of considering the heterogeneous periodicities of vessels in the yard template decision.

This paper makes an explorative study of yard template planning in container transshipment hubs, considering both the yard traffic congestion and the multiple cycle time of the periodicities of vessel arrival patterns. This study proposes a model for multi-period yard template planning with traffic congestion constraints. A local branching based solution method and a Particle Swarm Optimization (PSO) based solution method are developed for solving the proposed model. Numerical experiments using real world like instances are conducted to validate the effectiveness of the proposed model and the efficiency of the proposed solution method.

The remainder of this paper is organized as follows: Section 2 reviews the related works. Section 3 elaborates on the background to the problem; a mathematical model is formulated in

Section 4; and two solution methods are developed in Section 5. Results of the numerical experiments are addressed in Section 6, and closing remarks are then outlined in the last section.

2. Literature review

For an introduction to general port operations, we refer readers to the review works given by Vis and de Koster (2003), Steenken et al. (2004), and Stahlbock and Voß (2008). This paper is related to strategies used in allocating storage space in a yard to arriving containers. Plenty of studies have been performed on related problems. For example, Kim and Kim (1999) proposed a segregation strategy to allocate storage space for import containers. Kim et al. (2000) designed a methodology to determine the storage location of arriving export containers by considering their weight. Zhang et al. (2003) studied a storage allocation problem using a rolling horizon approach. Kozan and Preston (2006) proposed an iterative search algorithm for an integrated container transfer and allocation model to determine the optimal storage strategy. Caserta et al. (2011) defined a two-dimensional 'corridor', and developed a corridor method inspired algorithm for a blocks relocation problem in yard stacking systems. Jin et al. (2014) proposed the concept of 'yard crane profile', and used it in an integrated optimization model on yard storage management and yard crane deployment decisions. Different from the above studies, the yard management in this paper is based on a consignment strategy that was studied by Chen et al. (1995) and Dekker et al. (2006). This strategy attempts to store containers for the same destination vessel together in a particular dedicated storage area.

Transshipment is becoming more and more popular and important in ports around the world. This trend will continue to exist, because ocean liners are using larger vessels and visiting fewer ports (Fransoo and Lee, 2012). The papers mentioned in the previous paragraph do not sufficiently address the particular needs of transshipment hubs, in which most activities involve transshipping containers among visiting vessels. In recent years, more and more studies on managing transshipment activities have emerged. Cordeau et al. (2007) studied a service allocation problem that is a tactical problem arising in the yard management of a transshipment terminal. The objective is the minimization of container rehandling operations inside the yard. Moccia et al. (2009) studied a generalized assignment problem for allocating container groups within transshipment hubs, and a column generation heuristic was designed for solving this problem. Nishimura et al. (2009) proposed a decision model on storage allocation in transshipment hubs. The objective of the model is to minimize both the time for transshipping containers and the feeders' waiting time. Bell et al. (2013) and Wang et al. (2015) studied cost-based and profit-based container assignment problems under maritime liner shipping networks, respectively.

The yard template is related to the above mentioned transshipment activities, but is mainly concerned with the assignment to arriving vessels of storage areas in the yard of a transshipment hub. The concept of a yard template was first mentioned in a paper by Moorthy and Teo (2006), but their study focused mainly on berth allocation planning. Zhen et al. (2011) extended the above work, and developed an integrated model for berth allocation planning and yard template planning. They also developed a solution method for solving the problem. Lee et al. (2006) investigated a storage allocation problem in the yards of transshipment hubs that use the yard template. For mitigating the traffic congestion of yard trucks, the study by Lee et al. (2006) considered a storage policy known as the 'high-low workload balancing protocol'. An extended work was also conducted by Han et al. (2008) for simultaneously optimizing both the yard storage allocation decision and the yard template decision. They then also proposed a two-space sharing method to improve the space utilization of a yard template when facing uncertainty (Jiang et al., 2012). Zhen (2013) designed a stochastic programming model for robust yard template planning under uncertain berthing positions and times of vessels.

Although some studies have been conducted on yard template planning, few of them have considered the heterogeneous periodical patterns of arriving vessels by realistically taking into account their actual requirements, which is an important feature of making tactical level decisions in port operations. In addition, a lot of yard management related studies ignore the yard traffic congestion issues, which have become the bottleneck that limits a port's efficiency. Without considering these traffic congestion issues, the yard template planning decision is just a generalized assignment problem. This paper therefore takes account of both the heterogeneous periodicities of vessels and the yard traffic congestion issues in yard template planning. To the best of our knowledge, few scholars have proposed a decision model for yard template planning that in reality considers the above two important factors. This paper therefore presents an exploratory study conducted for this new problem.

3. Problem description

3.1 Yard template in container terminals

Before formulating a mathematical model, the background of the yard template planning problem will first be elaborated on. The ‘yard template’ is a concept used mainly in the transshipment hubs of container ports, in which the yard management is based on a consignment strategy. This strategy stores the export and transshipped containers that will be loaded onto the same departing vessel in the same assigned subblocks. Therefore, the planning decision of the yard template is about how to assign subblocks to arriving vessels. For each vessel, one or more subblocks in the yard are reserved. According to the consignment strategy, the incoming containers, which are first discharged from vessels and then loaded onto Vessel V_i in the future, will be stored in the subblocks reserved for Vessel V_i. When Vessel V_i arrives at the terminal, all the containers placed in these dedicated subblocks will be loaded onto Vessel V_i. By using this strategy, both the number of reshuffles and the vessels’ turnaround times can be significantly reduced .

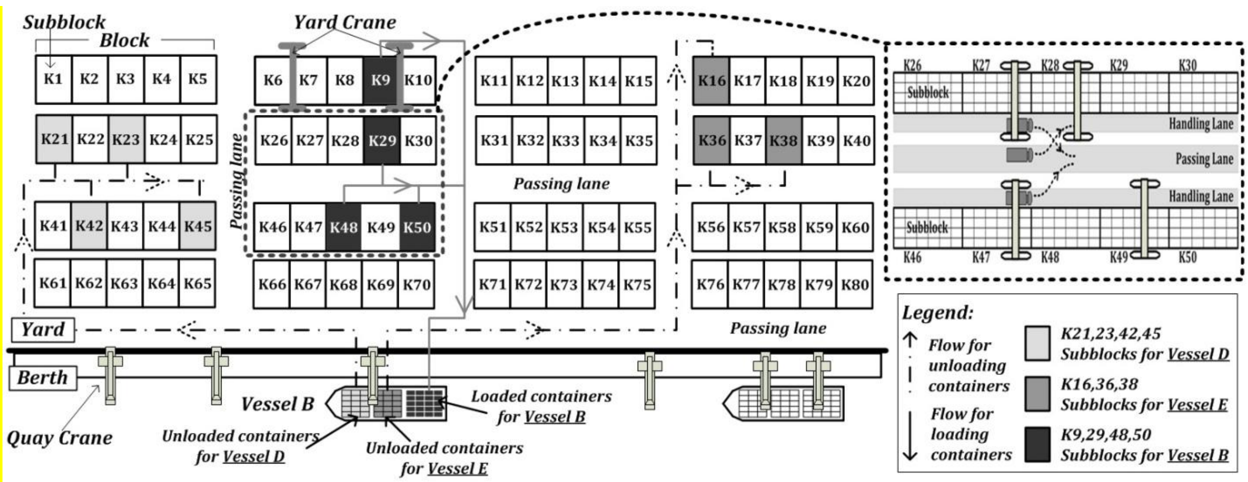

Figure 1: Typical configuration of a transshipment terminal

Figure 1 shows an example of the application of such a consignment strategy. The subblocks, shown in three different colors, are reserved for three different arriving vessels. Figure 1 demonstrates the moment when Vessel B arrives at the port. The dashed lines in Figure 1 denote the unloading routes, along which containers are discharged from Vessel B and are then transported to the subblocks reserved for Vessel D or Vessel E. The solid lines demonstrate the loading process, during which all containers stored in the subblocks reserved for Vessel B, namely *K9*, *K29*, *K48* and *K50*, are loaded onto Vessel *B* directly without any reshuffling operations.

3.2 Multi-period yard template planning

For a terminal, arriving vessels may have different cycle times. For example, some vessels have a weekly arrival pattern, and some may have a 10-day or biweekly pattern. In practice, the weekly pattern is the most common. Figure 2 shows a planning horizon example which contains 10 periods, 7 periods, or 5 periods for vessels with weekly, 10-day, or biweekly patterns, respectively. It should be noted that the subblock assignment for each vessel can vary for different periods, as a subblock needs to not be assigned to one vessel for the whole planning horizon. The above mentioned heterogeneous periodicities of vessels, and the dynamic nature of the subblock assignment in each vessel's multiple periods, complicate the yard template planning problem in comparison with other more generalized assignment problems. For handling this issue of vessels' heterogeneous periodicities, an intuitive method is to regard a vessel in multiple periods as multiple vessels in one period. From the perspective of mathematical modeling, it does not influence the complexity of the formulated model.

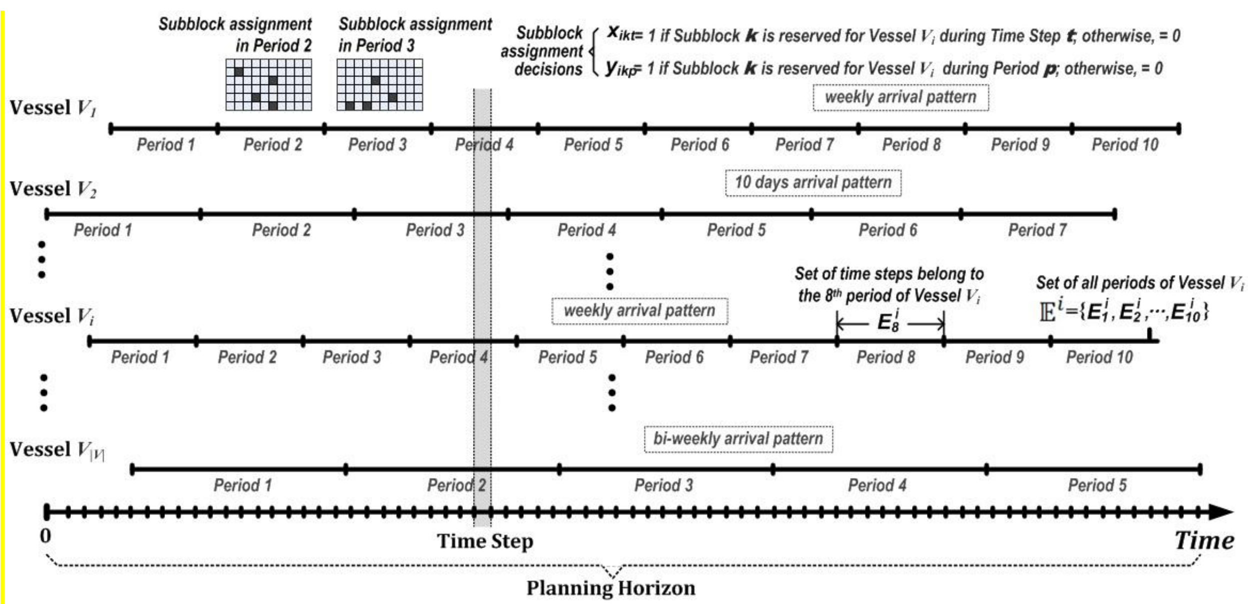

Figure 2: Multi-period yard template planning

As shown in Figure 2, the time step is the basic unit of time in this decision problem, which is not only concerned with decisions on a *spatial* dimension, (i.e., the assignment of subblocks in a yard), but also contains constraints on a *temporal* dimension for considering the yard traffic congestion issues, which are elaborated on in Section 3.3. Due to the heterogeneous periodicities of vessels, the time step acts as a uniform concept of time for all vessels, so that some limitations on activities occurring simultaneously can be formulated in constraints. The time step can be set as one day, 12 hours, 8 hours, or any other, the setting of which influences the problem scale. It should be noted that the term 'time step' is different from the term 'period'. As shown in Figure 2, a period contains several time steps.

3.3 Objective for yard template planning

The yard template planning problem is to determine which subblocks should be allocated to which vessels in different periods of the vessels, so as to minimize the total route length of all the container transshipment (discharging and loading) flows in the yard.

For calculating the length of loading routes for Vessel V_i, a parameter $d^L_{k,i}$ is defined as the length of the loading route between Subblock k and the berthing position of Vessel V_i; d^U_{ik} is defined as the length of the discharging route between the berthing position of Vessel V_i and Subblock k. Both $d^L_{k,i}$ and d^U_{ik} are known data when vessels' berthing positions are deterministic. Figure 3 illustrates the above two parameters on route lengths, i.e., $d^L_{k,i}$ and d^U_{ik}. It should be noted that the unloading route (dashed line with arrow) and loading route (solid line with arrow) between the same berth-subblock pair can be different, as shown in Figure 3.

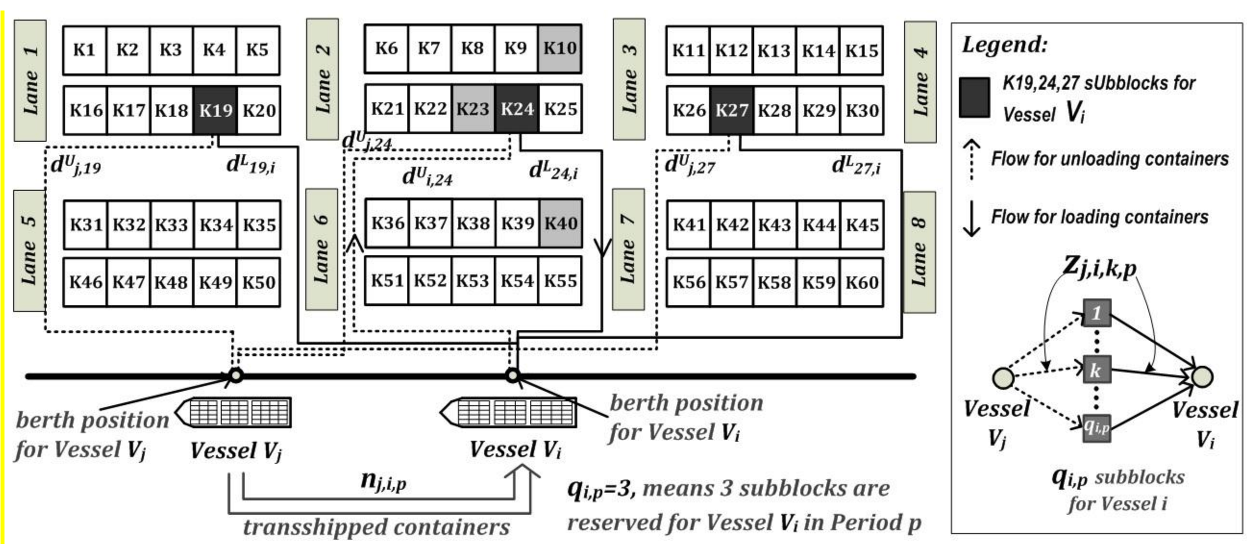

Figure 3: Loading and unloading routes of transshipped container flows

We define n_{jip} as the number of containers that are unloaded from Vessel V_j and loaded onto Vessel V_i in Period p. This parameter reflects the transshipment volume between two vessels. In addition, we define q_{ip} as the number of subblocks that should be reserved for Vessel V_i during Period p. When unloading the n_{jip} containers from Vessel V_j, the allocation of them to the q_{ip} subblocks is another important decision. As shown in the right part of Figure 3, a decision variable z_{jikp} is defined to denote the number of containers that are unloaded from Vessel V_j to Subblock k reserved for Vessel V_i during Period p. Here $\sum_{\forall k} z_{jikp} = n_{jip}$.

Suppose K denotes the set of all the subblocks; V denotes the set of all the vessels; $\mathbb{E}^i$ denotes the set of all the periods of Vessel V_i in the whole planning horizon; each element in $\mathbb{E}^i$ is denoted by E_p^i, which reflects the pth period of Vessel V_i. Then we can calculate the total route length of the transshipped container flows in the yard during the whole planning horizon according to the following formula: $\sum_{i \in V} \sum_{E_p^i \in \mathbb{E}^i} \sum_{k \in K} \left[\sum_{j \in V} \left(d_{jk}^U z_{jikp} \right) + d_{ki}^L \sum_{j \in V} z_{jikp} \right]$.

It should be mentioned that the above formula does not consider the empty trips of yard trucks but only takes account of their laden trips. This is because the routes taken for empty trips depend on the dispatching schedules of yard trucks, which belong to short-term decisions. Therefore this study uses only the route length of laden trips to evaluate the yard trucks' transportation cost. However, by assuming that the length of empty trips is approximately proportional to the length of laden trips, for the sake of simplicity the factor of empty trips can be ignored in the objective formulation.

3.4 Constraints for yard template planning

3.4.1 Basic constraints

There are some basic constraints for yard template planning. For example, there exist a minimum number of subblocks that should be reserved exclusively for each vessel. In addition, shipping liners have their favorite ranges of subblocks. For each vessel, therefore, their exclusively reserved subblocks should be selected from a given set of subblocks, as requested by the shipping liners in advance.

3.4.2 Constraints on yard crane contention in one block

For a container terminal using the consignment strategy, loading activity is very important for maintaining efficient operations, as the terminal needs to load all the containers in particular subblocks onto arriving vessels during a limited length of time.

As shown in Figure 1, five subblocks constitute a block, in which two yard cranes are usually deployed. In practice, one yard crane is usually dedicated to one subblock during the loading process. Therefore, in each block it would be best to have at most one subblock reserved for a particular vessel (or for a group of vessels being simultaneously loaded). This leaves the other spare yard crane available to be used for possible discharging activities in the same block.

As the berthing times of vessels are deterministic, the information about whether Vessel i has loading activities in time step m or not is also known in advance, which is denoted by a binary parameter $l_{i,m}$. The l parameters will be used later in the model formulation.

3.4.3 Constraints for mitigating traffic congestion

As previously mentioned, loading activity is very important for maintaining efficient operations, as the terminal needs to load all the containers in particular subblocks onto arriving vessels during a limited length of time. During the loading process occurring in a subblock, there is heavy traffic flow in the area near the subblock. Yard cranes and yard trucks are usually very busy, so traffic congestion can easily occur if a heavy workload needs to be handled within a relatively small area.

The upper right corner of Figure 1 demonstrates a scenario where subblocks *K27* and *K47* have simultaneous loading activities. In this case, a number of yard trucks will be waiting or moving nearby, which will probably lead to traffic congestion. Moreover, as shown in the upper right corner of Figure 1, high traffic flow near *K27* will also affect yard trucks going to *K28*. To ensure the smooth traffic flow of yard trucks, port operators thus need to impose some restrictions when planning the yard template. Here we define certain pairs of subblocks, such as (*K27*, *K47*) and (*K27*, *K28*). Each pair of subblocks cannot have loading activities going on at the same time. During the yard template planning process, the subblocks reserved for one vessel (or for vessels that are being loaded simultaneously during a particular period) should not be adjacent, such that two subblocks in each aforementioned pair will not have a loading process going on at the same time. Note that this study only takes account of the loading process, as the workload in a subblock when loading is usually much heavier than when discharging. During the loading activity, the port needs to load all the containers in the subblocks onto arriving vessels within a limited period of time, whereas the discharging of containers can be much more flexible, because when a container is discharged it can be distributed to any of the subblocks that are reserved for the vessel going to the container's destination. Therefore, a discharged container can be stored in a subblock where there is little yard truck traffic nearby.

3.4.4 Constraints on traffic flow in vertical passing lanes

Besides the traffic congestion caused by loading activities in neighboring subblock areas, which has been already considered by Lee et al., (2006) and Han et al., (2008), the traffic flows of yard trucks in passing lanes can result in another type of congestion in the yard. For the example shown in Figure 3, three subblocks need to be assigned to Vessel V_i. Both of the set of subblocks (*K10*, *K23*, *K40*) and the set of subblocks (*K19*, *K24*, *K27*) obey the previously defined constraints on the traffic congestion caused by loading activities in neighbor subblocks. However, the former assignment will cause all the loading truck flows pass through Lane 7, while the latter assignment distributes the loading truck flows in Lane 6, Lane 7, and Lane 8. As the loading activity for a subblock needs to transport all the containers in the subblock (up to 240 TEUs) to the quay side during a limited period of time, the loading truck flow for subblocks will cause heavy traffic. Thus we need to distribute the loading truck flows among passing lanes in a balance way. This study proposes some new constraints for limiting the number of traffic flows for loading activities in each time step.

For the mathematical model formulation in the next section, this study only considers constraints on the traffic flows of yard trucks for loading activities in the vertical passing lanes. The reason for only considering loading as opposed to unloading activities is similar to that given in the previous analysis; and the reason for only considering *vertical* passing lanes is that the traffic flows in *horizontal* passing lanes have already been limited by the constraints defined in the previous two sub-sections.

As shown in Figure 3, for each vertical passing lane u, we define a binary parameter h_{iku} to denote whether the loading flow from Subblock k to the berth where Vessel i moors passes Lane u. As the berthing positions of vessels are known, the parameters h_{iku} can be determined in advance. Based on these parameters, the total number of traffic flows that pass each lane (e.g., Lane u) can be limited by an upper bound (e.g., w_u) when making a yard template. For the example in Figure 3, $h_{i,24,7} = 1$, which means the loading flow from the subblock K24 to the berth where Vessel i moors passes Lane 7.

4. Model formulation

This section formulates two integer programming models for multi-period yard template planning in container ports. The objective of both models is to minimize the transportation cost that is related to the number of transported containers and the length of the container flows. The difference between the two models is this: The first model optimizes the tactical level yard template when taking into account the operational level yard storage allocation decisions; whereas the second model optimizes the yard template using a practical policy of equally allocating unloaded containers to the reserved subblocks.

4.1 Yard template planning model when considering storage allocation decisions

***Indices and sets*:**

i, j vessels

V set of all the vessels

k subblocks

K set of all the subblocks

u lanes

U set of all the lanes

t time steps

T set of all the time steps in the planning horizon

p periods

***Parameters*:**

q_{ip} number of subblocks that should be reserved for Vessel i during Period p

Q_i set of candidate subblocks, from which some are selected and assigned to Vessel i

n_{jip} number of containers that are unloaded from Vessel j and loaded to Vessel i in Period p

d_{ki}^{L} length of the loading route from Subblock k to the berth where Vessel i moors

d_{jk}^{U} length of the unloading route from the berth where Vessel j moors to Subblock k

l_{it} equals 1 if Vessel i has loading activities in Time Step t, and 0 otherwise

h_{iku} equals 1 if loading route from Subblock k to berth where Vessel i moors passes Lane u

w_u maximum number of routes passing Lane u simultaneously

S_r pair of subblocks that cannot be assigned to the same vessel

$\mathbb{S}$ set of all the pairs S_r, $S_r \in \mathbb{S}$

B_g group of five subblocks that belong to the same block

$\mathbb{B}$ set of all the blocks, i.e., the groups of subblocks B_g, $B_g \in \mathbb{B}$

E_p^i group of time steps that belong to the p^{th} period of Vessel i

$|E_p^i|$ cycle time of a period for Vessel i, the unit is one time step

$\mathbb{E}^i$ set of all the groups E_p^i for Vessel i, i.e., $E_p^i \in \mathbb{E}^i$

$|\mathbb{E}^i|$ number of periods for Vessel i in the planning horizon

C capacity of a subblock, i.e., the maximum TEUs that can be stored in a subblock

M a sufficiently large positive number

Variables:

x_{ikt} set to 1 if Subblock k is reserved for Vessel i during Time Step t, and 0 otherwise

y_{ikp} set to 1 if Subblock k is reserved for Vessel i during Period p, and 0 otherwise

z_{jikp} number of containers unloaded from Vessel j to Subblock k that are reserved for Vessel i during Period p

Mathematical model:

$$(\boldsymbol{M\_1}) \quad \textit{Minimize} \quad \sum_{i\in V}\sum_{E_p^i\in\mathbb{E}^i}\sum_{k\in K}\left(\sum_{j\in V} d_{jk}^U z_{jikp} + d_{ki}^L \sum_{j\in V} z_{jikp}\right) \tag{1}$$

$$s.t. \quad \sum_{i\in V} x_{ikt} \le 1 \qquad \forall k\in K, \forall t\in T \tag{2}$$

$$\sum_{k\in Q_i} y_{ikp} = q_{ip} \qquad \forall i\in V, \forall E_p^i \in \mathbb{E}^i \tag{3}$$

$$\sum_{k\in K/Q_i} y_{ikp} = 0 \qquad \forall i\in V, \forall E_p^i \in \mathbb{E}^i \tag{4}$$

$$\sum_{k\in S_r}\sum_{i\in V} l_{it} x_{ikt} \le 1 \qquad \forall S_r \in \mathbb{S}, \forall t\in T \tag{5}$$

$$\sum_{k\in B_g}\sum_{i\in V} l_{it} x_{ikt} \le 1 \qquad \forall B_g \in \mathbb{B}, \forall t\in T \tag{6}$$

$$\sum_{i\in V}\sum_{\forall k\in K} l_{it} h_{iku} x_{ikt} \le w_u \qquad \forall u\in U, \forall t\in T \tag{7}$$

$$\sum_{t\in E_p^i} x_{ikt} = |E_p^i| y_{ikp} \qquad \forall i\in V, \forall E_p^i \in \mathbb{E}^i, \forall k\in K \tag{8}$$

$$\sum_{k\in Q_i} z_{jikp} = n_{jip} \qquad \forall i,j\in V, \forall E_p^i \in \mathbb{E}^i \tag{9}$$

$$z_{jikp} \le y_{ikp} M \qquad \forall i,j\in V, \forall E_p^i \in \mathbb{E}^i, \forall k\in K \tag{10}$$

$$\sum_{j\in V} z_{jikp} \le C \qquad \forall k\in K, \forall i\in V, \forall E_p^i \in \mathbb{E}^i \tag{11}$$

$$x_{ikt} \in \{0,1\} \qquad \forall i\in V, \forall k\in K, \forall t\in T \tag{12}$$

$$y_{ikp} \in \{0,1\} \qquad \forall i\in V, \forall k\in K, \forall E_p^i \in \mathbb{E}^i \tag{13}$$

$$z_{jikp} \ge 0 \qquad \forall i,j\in V, \forall E_p^i \in \mathbb{E}^i \tag{14}$$

In the above model, Objective (1) minimizes the total route length (loading/unloading) of all the transshipped containers for all the arriving vessels during the whole planning horizon. Constraints (2) limit each subblock allocation to at most one vessel in each time step. Constraints (3) and (4) ensure a given number (q_{ip}) of subblocks are allocated to Vessel i in Period p; the allocated subblocks are selected from among a certain set of candidate subblocks, i.e., Q_i, depending on the vessels (shipping liners). Constraints (5) guarantee that two subblocks in each pair S_r cannot carry out loading activities simultaneously. Constraints (6) limit loading activities in each time step to at most one subblock in each block. Constraints (7) ensure that the number of container loading flows that pass along a lane cannot exceed the lane's traffic limitation. Constraints (8) connect the two decision variables, i.e., x_{ikt} and y_{ikp}. Constraints (9) and (10) are about storage allocation decisions based on the yard template. More specifically, Constraints (9) stipulate the allocation of the n_{jip} transshipped containers to the q_{ip} subblocks reserved for Vessel i. Constraints (10) ensure that containers transshipped to Vessel i cannot be allocated to subblocks that are not reserved for Vessel i. Constraints (11) guarantee that the number of containers allocated to each subblock cannot exceed the capacity of the subblock (i.e., 240 TEUs in reality). Constraints (12), (13) and (14) define the decision variables.

Proposition 1: Finding an optimal solution for the model M_1 is strongly NP-hard

Proof: See Appendix A ■

A lower bound for the model can be obtained by relaxing the constraints of traffic congestion, i.e., Constraints (5), (6), and (7). We use M_1^{LB} to denote the lower bound for model M_1.

($\boldsymbol{M_1^{LB}}$) Objective (1)

s.t. Constraints (2-4; 8-14)

As the constraints on traffic congestion are relaxed, the lower bound model M_1^{LB} can be solved in a much faster way than the original model. The lower bound can also be used in the section on numerical experiments to evaluate the quality of solutions obtained by our methods.

4.2 Yard template planning model under a policy of equal storage allocation

In practice, containers unloaded from the original vessel are often evenly distributed to the subblocks reserved for their destination vessels. We refer to this policy as the equal storage allocation policy. It equally allocates the n_{jip} containers, which are unloaded from Vessel j, to the q_{ip} subblocks reserved for Vessel i. More specifically, this means that there are 'n_{jip}/q_{ip}' containers transported from the berthing position of Vessel j to each one of the q_{ip} subblocks, and which are then transported to the berthing position of Vessel i. This policy, using even distribution in the unloading process, obeys real-world, pragmatic rules, which helps avoid potential traffic congestion by yard trucks in some subblocks, and minimizes the completion time of the unloading process.

Based on the above policy, the objective, i.e., minimizing the total route length of the transshipped container flows in the yard during the whole planning horizon, is formulated as: $\sum_{i\in V}\sum_{E_p^i\in\mathbb{E}^i}\sum_{k\in K} y_{ikp}\left[\sum_{j\in V}\left(d_{jk}^U n_{jip}/q_{ip}\right)+d_{ki}^L\sum_{j\in V} n_{jip}/q_{ip}\right]$. From this we then formulate the model of the yard template planning under a policy of equal storage allocation. The model is denoted by 'M_2'. This model contains only the decision variables on yard template planning, i.e., binary variables x_{ikt} and y_{ikp}.

$$(\boldsymbol{M\_2}) \quad Minimize \quad \sum_{i\in V}\sum_{E_p^i\in\mathbb{E}^i}\sum_{k\in K} y_{ikp} D_{ikp} \tag{15}$$

s.t. Constrains (2-8; 12-13)

$$\text{Here } D_{ikp}=\sum_{j\in V}\left(d_{jk}^U n_{jip}/q_{ip}\right)+d_{ki}^L\sum_{j\in V} n_{jip}/q_{ip} \tag{16}$$

As the optimal solution of model M_2 is surely a feasible solution for that of model M_1, the optimal objective value of model M_1 is no greater than the optimal objective value of model M_2. The solution obtained by solving model M_2 can act as an initial solution in searching for a good solution for model M_1, especially when solving some large scale problem instances of model M_1.

Proposition 2: If there exist feasible solutions for model M_2, there is thus more than one optimal solution (optimal yard template).

Proof: See Appendix B ■

Proposition 3: An optimal yard template for model M_2 has the following properties: (1) At least one subblock in the block that is nearest to the berthing position of Vessel i_{max} is assigned to Vessel i_{max} for its Period p_{max}. (2) At least one subblock in the block section being used, and which is the furthest from the berthing position of Vessel i_{min}, is assigned to Vessel i_{min} for its Period p_{min}. Here $(i_{max}, p_{max}) = \arg\max_{\forall(i,p),i\in V,p\in\mathbb{E}^i}\{(\sum_{j\in V} n_{j,i,p})/q_{i,p}\}$, $(i_{min}, p_{min}) = \arg\min_{\forall(i,p),i\in V,p\in\mathbb{E}^i}\{(\sum_{j\in V} n_{j,i,p})/q_{i,p}\}$.

Proof: See Appendix C ■

The above Proposition 3 shows that the "$(\sum_{j \in V} n_{j,i,p})/q_{i,p}$" is an important indicator in the yard template planning decision. This indicator will be used for developing solution methods for solving the model.

5. Solution approaches

For some small-scale problem instances, the proposed model M_1 can be solved by the CPLEX solver. However, for other large-scale problem instances, the model becomes too intractable for the CPLEX to solve directly. This study designs two solution approaches for solving the proposed model M_1. One is the local branching based method; the other is the Particle Swarm Optimization (PSO) based method, which can be used for solving extremely large-scale instances. Both methods require a good initial solution as a starting point. Section 5.1 addresses the method for obtaining the initial solution; and then Section 5.2 and Section 5.3 elaborate on the local branching based method and the PSO based method, respectively.

5.1 Initial solution provided by solving model M_2

An initial solution for model M_1 can be obtained by solving its special case (i.e., model M_2). However, the process by which the CPLEX solves model M_2 directly may also be very time-consuming if the problem scale is large. Here, a sequence based heuristic is proposed for obtaining a good solution to the M_2. The M_2 model is all about deciding the assignment of subblocks to vessels. As its name suggests, the *sequence based heuristic* assigns subblocks to the vessels one by one according to a given sequence of 'vessel-period' pairs. This sequence is determined on the basis of the main idea contained in Proposition 3. More specifically, the sequence is generated by the decreasing order of the number of containers loaded onto each vessel during each period, divided by the number of subblocks that are assigned to the vessel, i.e., $(\sum_{j \in V} n_{j,i,p})/q_{i,p}$. This indicator is determined according to Proposition 3.

Given a sequence of vessels, a model $M_2(n)$ is solved for the vessels one by one. Let the sequence of 'vessel-period' pairs be $\{1, \cdots, n, \cdots, \sum_{i \in V} |\mathbb{E}^i|\}$. Here $\sum_{i \in V} |\mathbb{E}^i|$ denotes the total number of all the vessels' periods, which is also equal to the number of all the possible 'vessel-period' pairs. This sequential method solves '$\sum_{i \in V} |\mathbb{E}^i|$' 'vessel-period' pairs sequentially in iterations. Suppose the value '$(\sum_{j\in V} n_{j,i,p})/q_{i,p}$' for Vessel i and its Period p is in the n^{th} position of the sequence. During the n^{th} iteration, we solve the model $M_2(n)$ for obtaining the decision of subblock assignment for the 'Vessel i - Period p' pair.

A parameter DS (i.e., the depth of search) is defined. During the n^{th} iteration for solving the subblock assignment decision of Vessel i during its Period p, the variables related to the 'vessel-period' pairs in the sub-sequence $\{1,\cdots,n-1\}$ are input data, whereas the variables related to the 'vessel-period' pairs in the sub-sequence $\{n,\cdots,\sum_{i\in V}|\mathbb{E}^i|\}$ are still decision variables. After solving model $M_2(n)$, the obtained decision variables related to the n^{th} item of the sequence (i.e., the 'Vessel i – Period p' pair) are fixed as the input data for the next iteration. The whole procedure has '$\sum_{i\in V}|\mathbb{E}^i| - DS + 1$' iterations. For understanding the parameter DS, an example is given as follow. There are 100 'vessel-period' pairs, which form a sequence. And DS is set as 20. Then there are 81 iterations (i.e., $\sum_{i\in V}|\mathbb{E}^i| - DS + 1 = 100 - 20 + 1 = 81$). The first iteration is to solve a model $M_2(1)$, in which the subblock assignments for the batch of 20 'vessel-period' pairs (i.e., the 1st, 2nd, …, 20th pairs in the sequence) are decision variables, and the obtained subblock assignment for the 1st pair is determined. The second iteration is to solve a model $M_2(2)$, in which the subblock assignments for the batch of 20 'vessel-period' pairs (i.e., the 2nd, 3rd, …, 21th pairs in the sequence) are decision variables, and the obtained subblock assignment for the 2nd pair is determined. The following iterations are performed as the similar way. The last iteration (i.e., the 81st iteration) is to solve a model $M_2(2)$, in which the subblock assignments for the batch of 20 'vessel-period' pairs (i.e., the 81st, 82nd …, 100th pairs in the sequence) are decision variables, and the obtained subblock assignments for these 20 pairs are determined.

From the above, we can see that the DS parameter setting is important for the procedure's performance. The setting of DS as a large value will result in a time-consuming solution process. The proper setting of DS is related to the computational capacity of the computer. If the DS equals zero, the method is degenerated to the normal greedy search, which means it will sequentially make the best decision over choosing subblocks for each vessel and period within the remaining time-space of subblocks. This setting results in a fast solution process but may result in much loss of optimality. In addition, if the DS is set too small, such as at zero, it will probably occur that no feasible subblocks can be assigned to the last few vessel-period pairs.

Therefore, a proper setting of the DS value can effectively avoid any infeasibility for rear vessel-period pairs.

5.2 Local branching based solution method

Model M_1 is a MIP (mixed integer programming) model and contains integer variables, i.e., $x_{ikt}, y_{ikp} \in \{0,1\}$. Other than for x_{ikt} and y_{ikp}, the variables z_{jikp} in model M_1 are continuous. The integer variables x_{ikt} depend only on the variables y_{ikp}; thus there is actually only one set of integer variables y_{ikp} in the nature of this problem. Therefore, the solution speed for model M_1 is mainly limited by the branching process of the binary variable y_{ikp} within its huge solution space. In this study, a local branching strategy is utilized to solve model M_1. The local branching strategy is exact in nature, though it can improve the heuristic behavior of the solver at hand. It alternates high-level strategic branching in order to define the solution neighborhoods, and low-level tactical branching to explore them (Fischetti and Lodi, 2003).

The proposed solution method in this study is based on the local branching strategy, the core idea of which is to use the CPLEX solver as a black-box 'tactical' tool for exploring suitable solution subspaces, which are defined and controlled at a 'strategic' level by a simple external branching framework. Our proposed solution method is in the spirit of widely used local search meta-heuristics, but the neighborhoods are formulated by introducing local branching cuts (linear inequalities). This proposed method can be regarded as a two-level branching strategy aimed at favoring early updating of the incumbent solution, hence producing improved solutions at early stages of the computation. The main procedure of the local branching is illustrated in Figure 4.

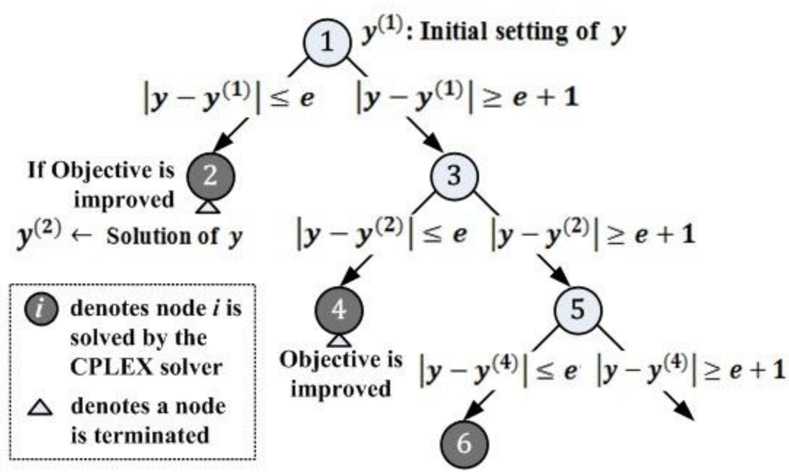

Figure 4: The main procedure of local branching

In Figure 4, node 1 is the starting point of the local branching procedure. At node 1, the binary variables y_{ikp} can be initialized according to the method proposed in Section 5.1. The initial setting on variables y_{ikp} $(\forall i \in V, k \in K, E_p^i \in \mathbb{E}^i)$ is denoted by $\boldsymbol{y}^{(1)}$ in Figure 4. Based on node 1, we derive node 2 and node 3, which denote model M_1 with further constraints $|\boldsymbol{y} - \boldsymbol{y}^{(1)}| \leq e$, $|\boldsymbol{y} - \boldsymbol{y}^{(1)}| \geq e + 1$, respectively. Here $|\boldsymbol{y} - \boldsymbol{y}^{(1)}|$ reflects the radius of $\boldsymbol{y}^{(1)}$'s neighborhood in the solution space of the $\boldsymbol{y}$ variables. More specifically, $|\boldsymbol{y} - \boldsymbol{y}^{(1)}|$ is calculated as: $|\boldsymbol{y} - \boldsymbol{y}^{(1)}| = \sum_{i \in V} \sum_{E_p^i \in \mathbb{E}^i} \sum_{k \in K} \left| y_{ikp} - y_{ikp}^{(1)} \right|$. Its value reflects the number of variables y_{ikp} that are different from their corresponding fixed values $y_{ikp}^{(1)}$. The parameter e controls the neighborhood size when searching for solutions around the current solution. If the e is small, the solution process in node 2 will be fast; otherwise it may be time-consuming.

When node 2 is solved, the binary variables y_{ikp} are denoted by $\boldsymbol{y}^{(2)}$ in Figure 4. Then node 4 denotes the model M_1 with constraints $|\boldsymbol{y} - \boldsymbol{y}^{(2)}| \leq e$, $|\boldsymbol{y} - \boldsymbol{y}^{(1)}| \geq e + 1$. Because the neighborhood size is limited by the constraint $|\boldsymbol{y} - \boldsymbol{y}^{(2)}| \leq e$, node 4 can be solved in a much faster way than would be the case without this limitation on its neighborhood size.

Figure 4 shows the main flow of the branching process, in which all the nodes marked by the dark color will be solved by the CPLEX solver. The whole solution procedure may be trapped (or deferred) by the time-consuming solution processes in some nodes. So we impose an upper limit for the solution time in each node. If the solution time exceeds this limit, the CPLEX solver will stop solving the model at this node, and will either return a feasible but non-optimal solution; otherwise, it means the model at this node has been solved optimally by the CPLEX solver. It should be noted that there is none possibility that a node cannot return a feasible solution within the time limit. For example of node 2 in Figure 4, there is at least one feasible solution $\boldsymbol{y}^{(1)}$; for node 4, there is at least one feasible solution $\boldsymbol{y}^{(2)}$.

For each node marked by the dark color, the CPLEX result can be any of four cases:

Case 1: The objective value of model M_1 with some $\boldsymbol{y}$ related constraints is improved, and the solution time limit for the CPLEX solver is not reached. This means the node is solved optimally by the CPLEX solver, and the solution is better than the incumbent best solution so far.

Case 2: The solution time limit for the CPLEX solver is reached, but the objective value is improved. This means the CPLEX solver obtains a non-optimal solution that is also better than the incumbent best solution so far.

Case 3: The solution time limit for the CPLEX solver is not reached, but the objective value is not improved. This means that the CPLEX solver obtains an optimal solution that is worse than the incumbent best solution so far.

Case 4: The solution time limit for the CPLEX solver is reached, but the objective value is not improved. This means that the CPLEX solver obtains a non-optimal solution that is worse than the incumbent best solution so far.

For the above four cases, the handling strategies are shown in Figure 5.

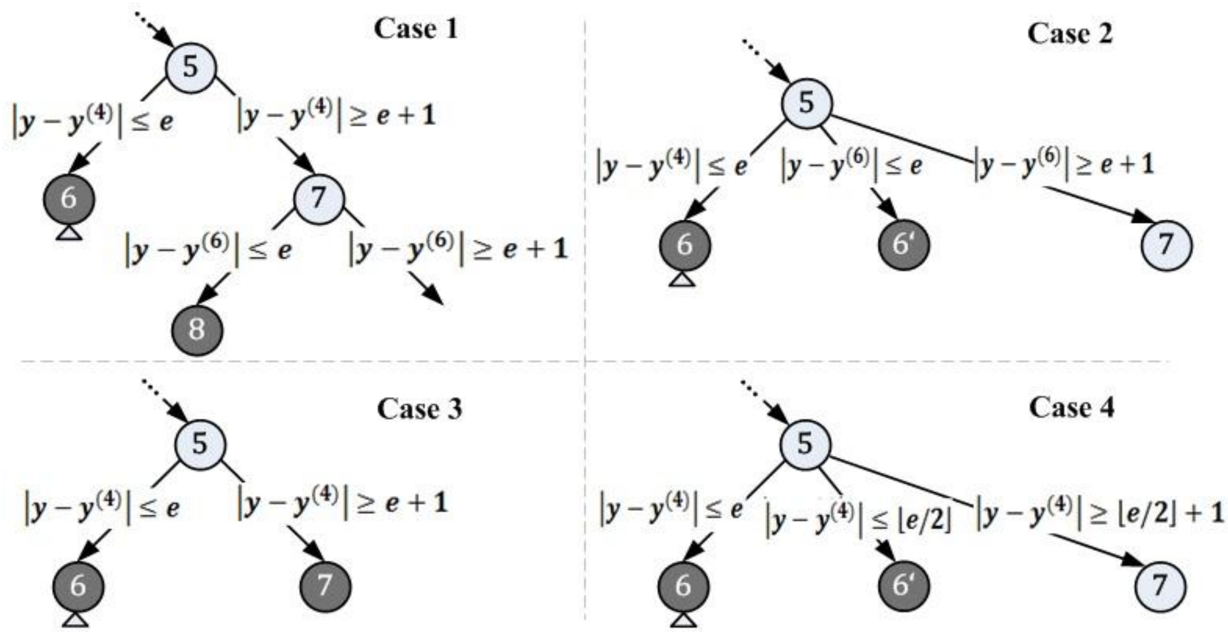

Figure 5: Four different cases obtained during the branching process

Case 1 is the most common case in the local branching procedure. It denotes the standard flow of the solution process, as shown in Figure 4.

Case 2 results from updating of the neighborhood related constraint, which is changed from $|\boldsymbol{y} - \boldsymbol{y}^{(4)}| \leq e$ to $|\boldsymbol{y} - \boldsymbol{y}^{(6)}| \leq e$, because the solution of node 6 replaces that of node 4 and becomes the incumbent best one so far.

Case 3 means that node 6 is completely pruned, because the optimally solved solution at this node is worse than the incumbent best solution so far. Therefore, the solution of the original model equals the solution of node 7.

Case 4 means that the neighborhood size needs reducing so as to either solve a solution in the node optimally or obtain a better solution than the incumbent best one so far.

The above four cases cover all possible situations throughout the procedure, which is in line with well-known local search meta-heuristics. In this study, the neighborhoods used in the local branching procedure are realized by defining certain linear inequalities (or branching cuts) with respect to binary variables (y_{ikp}). The whole solution procedure terminates if the incumbent best objective value has not been improved over a given number of consecutive iterations.

5.3 PSO based solution method

When solving some extremely large-scale instances of model M_1, the above local branching based method may be somewhat time consuming, so we have to consider heuristic solution approaches. PSO was first proposed by Eberhart and Kennedy (1995), and they used it to optimize continuous nonlinear functions. In recent years, the PSO has been used for solving certain port operation problems, e.g., berth allocation problem (Ting et al., 2014), quay crane scheduling problems (Tang et al., 2014). This paper designs a PSO based solution approach for solving model M_1.

5.3.1 Solution representation and velocity updating strategy

Like the well known genetic algorithm, PSO is also a population based heuristic method. In the population of PSO, each solution is denoted by a particle whose status contains its position and velocity at each iteration. A particle's position denotes the quality of the solution that corresponds with the particle; while the particle's velocity denotes the direction along which it will move in the next iteration.

In the proposed model M_1, the decision variables x_{ikt} and y_{ikp} are actually related to each other. Once one type of variable is determined, the other type of variable is also known. In this study, we use the y_{ikp} variable to define particles in the PSO. More specifically, for a particle m in iteration n, its position is defined by $\mathbb{Y}_m^n = \{Y_{mikp}^n\}$, and its velocity is defined by $\mathbb{v}_m^n = \{v_{mikp}^n\}$, $\forall i \in V, \forall k \in K, \forall E_p^i \in \mathbb{E}^i$. We define $YpBest_{mikp}^n$ as the best position of the particle m on dimensions i, k, p until iteration n, and $YgBest_{ikp}^n$ as the best position of the whole swarm on dimensions i, k, p until iteration n. Then the updating formulae of velocity and position are:

$$v_{mikp}^{n+1} = w^n v_{mikp}^n + c_1 r_1 \left(YpBest_{mikp}^n - Y_{mikp}^n\right) + c_2 r_2 \left(YgBest_{ikp}^n - Y_{mikp}^n\right) \tag{17}$$

$$Y_{mikp}^{n+1} = Y_{mikp}^n + v_{mikp}^n \tag{18}$$

Here w^n is the inertia weight, and is calculated by $w^n = \frac{N-n}{N}(w_{ini} - w_{end}) + w_{end}$, in which N is the number of iterations; w_{ini} and w_{end} are the initial inertia weight and the final inertia weight, respectively. The inertia weight affects the PSO procedure's convergence towards the optimal solution. At the beginning of the procedure, a large inertia weight causes the procedure to achieve good convergence towards the global optimal solution. As the procedure continues, the inertia weight becomes smaller and smaller, which causes the final stage of the procedure to achieve good convergence towards the local optimal solution.

In addition, c_1 and c_2 are acceleration weights; r_1 and r_2 are random numbers generated between zero and one.

In order to avoid the PSO procedure falling only into local optima, the above updating formulae of velocity, i.e., Formula (18), is modified as follows:

$$v_{mikp}^{n+1} = w^n\, v_{mikp}^n + c_1 r_1\left(YpBest_{mikp}^n - Y_{mikp}^n\right) + c_2 r_2\left(YgBest_{ikp}^n - Y_{mikp}^n\right) + c_3 r_3\left(Ya_{ikp}^n - Yb_{ikp}^n\right) \quad (19)$$

Here Ya_{ikp}^n and Yb_{ikp}^n are the positions of two randomly chosen particles on dimensions i, k, p in iteration n. Then, a particle not only moves in the direction of the best positions for both the whole swarm and the particle, but may also fly to two randomly chosen particles. In this case, the particle will be more likely to leave the local optima to search for better solutions.

Because the decision variables in this study are binary variables, we cannot calculate a particle's position according to the above Formula (18). It is thus modified as follows:

$$Y_{mikp}^{n+1} = \begin{cases} 1, & when\ r_4 < S(v_{mikp}^{n+1}) \\ 0, & when\ r_4 \geq S(v_{mikp}^{n+1}) \end{cases} \quad (20)$$

Here r_4 is a random number generated between zero and one; $S(v_{mikp}^{n+1})$ is a sigmoid function, $S(v_{mikp}^{n+1}) = 1/(1 - e^{-v_{mikp}^{n+1}})$.

5.3.2 Main framework of the PSO procedure

Based on the above definition of the solution representation and velocity updating strategy, the main framework of the PSO procedure is as follows.

Step 1: Set the iteration number $n = 1$. Initialize a swarm containing M particles whose positions determine which subblocks are assigned to which vessels during each period. The assignment plan is randomly generated.

Step 2: For certain particles (e.g., particle m) whose 'vessel-subblock' assignment plan disobeys some constraints of the model, a sub-procedure $\boldsymbol{Adjust}(m)$ is performed to revise the assignment so as to make particle m's assignment plan feasible. Details of the sub-procedure $\boldsymbol{Adjust}(m)$ are addressed later.

Step 3: For all the particles, calculate their fitness value by solving model M_1, in which the $\boldsymbol{x}$ and $\boldsymbol{y}$ variables are fixed according to the particles. As model M_1 with the fixed $\boldsymbol{x}$ and $\boldsymbol{y}$ variables is a linear programming model, it can be solved by the CPLEX solver directly in affordable running time.

Step 4: Update the best position of each particle, i.e., $\{YpBest_{mikp}^{n}\}$, and the best position of the swarm, i.e., $\{YgBest_{ikp}^{n}\}$.

Step 5: Update the velocity according to Formula (19), and update the position according to Formula (20).

Step 6: If the iteration number reaches the preset maximum value, stop; otherwise, $n = n + 1$ and then go to Step 2.

In the above steps, the sub-procedure $\boldsymbol{Adjust}(m)$ is important for the whole PSO solution method. Its target is to make the infeasible particles become feasible by means of adjusting the 'vessel-subblock' assignment plan so as to satisfy all the constraints of the model. A detailed description of the sub-procedure $\boldsymbol{Adjust}(m)$ is addressed in Appendix D.

6. Computational experiments

Several experiments are performed on a PC (Intel Core i5, 1.6G Hz; Memory, 4G) to validate the efficiency of the solution method and the effectiveness of the proposed model M_TP. The PSO based solution method is implemented by C# (VS2008). For solving the model optimally, it is implemented by CPLEX12.1 with concert technology of C# (VS2008).

6.1 Generation of test cases

We generate test cases as follow. The planning horizon is about 70 days, which means 10 periods for vessels with a weekly arrival pattern, 5 periods for vessels with a biweekly arrival pattern, and 7 periods for vessels with a 10-day arrival pattern. For each instance, we assume that about 2/3, 1/6, 1/6 of all the vessels belong to the weekly pattern, bi-weekly pattern and 10-day pattern, respectively. The time step is set to contain eight hours. We randomly generate the starting time step of the periods for each vessel. Then we can determine in advance the groups of time steps that belong to each period of each vessel (i.e., E_p^i). In addition, we can also determine the time steps during which each vessel has loading activities (i.e., l_{it}). Each block contains six containers (TEUs) along the depth dimension and 40 containers along the length dimension. Each block contains five subblocks. Each subblock contains eight containers along the length dimension, which means a subblock's length is about 50 meters. The stacking height is five tiers (containers). Thus the capacity of a subblock is about 240 (= 6×8×5) TEUs. This study uses the concept of 'subblock' as the basic unit for the decisions on yard storage allocation and yard template planning. The average number of subblocks that should be reserved for a vessel (i.e, q_{ip}) is about five. The number of containers that are transshipped between pairs of vessels in a period (i.e., n_{jip}) are randomly generated such that the average number of containers for loading onto a vessel is about 1000 TEUs. The width of the passing lanes is set as 30 meters. The maximum number of routes passing along each lane simultaneously (i.e., w_u) is set as one. The vessels' berthing positions along the quay are randomly generated. The configuration of subblocks in a yard is arranged similar to the example in Figure 1. The pairs of subblocks that are neighbors (i.e., S_r and $\mathbb{S}$) and the groups of five subblocks that belong to the same block (i.e., B_g and $\mathbb{B}$) are determined accordingly. Based on the configuration of subblocks and the berthing positions of vessels, the loading/unloading route length (i.e., d_{jk}^U and d_{ki}^L) can be calculated. According to the vessels' berthing positions, whether a loading route from a subblock to a vessel's berth passes a Lane or not can also be determined (i.e., h_{iku}).

6.2 Performance of the proposed solution methods

Experiments on small-scale cases are conducted first to validate the efficiency of the local branching based method. The optimal results are obtained using the CPLEX solver. From Table 1 we can see that the local branching based solution algorithm can obtain near-optimal results. The average gap between the objective values of the local branching based solution algorithm and the optimal results is just 0.09%. The CPU time for the local branching based solution algorithm is longer than the CPLEX solver's CPU time when solving some very small-scale problem cases, but is much shorter than the CPLEX solver for some middle-scale problem cases. Generally, the local branching based solution algorithm's CPU time is acceptable, but displays its relative merit as the problem scale increases. More importantly, it is seen that the CPLEX cannot solve the model under large-scale problem cases, whereas the local branching solution algorithm can solve such problems within a reasonable time period.

Table 1: Comparison between the local branching algorithm and the CPLEX solver

Instances		**Optimal results**		**Local branching based method**				**Lower Bound**	
Scale	**ID**	Z_C	T_C	Z_L	T_L	GAP_L	T_L/T_C	LB	GAP_{LB}
6	6_40_1	18,780,078	12	18,787,078	77	0.04%	6.42	18,743,789	0.19%
Vessels	6_40_2	19,819,045	24	19,819,045	127	0.00%	5.29	19,786,350	0.16%
&	6_40_3	18,410,123	19	18,414,139	132	0.02%	6.95	18,396,923	0.07%
40	6_40_4	18,815,707	23	18,816,859	110	0.01%	4.78	18,800,079	0.08%
Subblocks	6_40_5	18,673,848	32	18,673,848	188	0.00%	5.88	18,667,432	0.03%
9	9_60_1	41,165,233	341	41,192,521	365	0.07%	1.07	41,140,417	0.06%
Vessels	9_60_2	40,191,729	413	40,213,264	317	0.05%	0.77	40,174,089	0.04%
&	9_60_3	39,468,476	264	39,484,267	262	0.04%	0.99	39,454,668	0.03%
60	9_60_4	41,094,872	547	41,107,157	438	0.03%	0.80	41,079,610	0.04%
Subblocks	9_60_5	39,829,041	574	39,853,650	352	0.06%	0.61	39,724,793	0.26%
12	12_80_1	53,207,203	1,625	53,253,455	590	0.09%	0.36	53,158,412	0.09%
Vessels	12_80_2	57,201,700	1,547	57,286,412	632	0.15%	0.41	57,126,320	0.13%
&	12_80_3	59,538,137	2,090	59,594,184	675	0.09%	0.32	59,283,614	0.43%
80	12_80_4	55,473,855	1,874	55,548,822	714	0.14%	0.38	55,347,313	0.23%
Subblocks	12_80_5	59,974,347	1,718	60,046,988	774	0.12%	0.45	59,790,448	0.31%
15	15_100_1	75,806,885	4,503	75,940,828	1,092	0.18%	0.24	75,779,699	0.04%
Vessels	15_100_2	72,027,118	5,838	72,165,303	1,011	0.19%	0.17	71,808,740	0.30%
&	15_100_3	74,929,292	5,476	75,114,549	1,212	0.25%	0.22	74,826,055	0.14%
100	15_100_4	82,736,577	5,543	82,866,984	1,083	0.16%	0.20	82,561,398	0.21%
Subblocks	15_100_5	83,281,560	6,381	83,387,140	1,163	0.13%	0.18	83,225,790	0.07%
				Average:		**0.09%**			**0.15%**

Notes: (1) The optimal results are obtained by using the CPLEX solver. The optimal objective values and the CPU time are denoted by Z_C and T_C, respectively. (2) The objective values and the CPU time of the local branching algorithm are denoted by Z_L and T_L, respectively. $GAP_L = (Z_L - Z_C) / Z_C$. (3) The lower bounds are obtained by solving the model M_1^{LB} proposed in Section 4.1. $GAP_{LB} = (Z_C - LB) / Z_C$.

Table 1 also demonstrates the quality of the lower bound (i.e., M_1^{LB}) proposed in Section 4.1. The gap between the proposed lower bound and the optimal result is about 0.15% on average. As the lower bound can be obtained in an easier way than by solving the original model, it will be used as a criterion for evaluating the performance of the proposed solution methods when facing some large-scale problem instances.

Besides the local branching algorithm, another method based on the PSO is also proposed for solving large-scale instances. The performance of the proposed two types of solution methods is also investigated using comparative experiments. The results are shown in Table 2 as follows:

Table 2: Comparison between the local branching method and the PSO method

Instances		***LB***	**Local branching method**			**PSO method**			$\frac{T_P}{T_L}$
Scale	**ID**		Z_L	T_L	GAP_L	Z_P	T_P	GAP_P	
15	15_100_1	74,397,548	74,663,131	1,088	0.36%	75,575,951	386	1.58%	0.35
Vessels	15_100_2	79,647,014	79,890,584	1,078	0.31%	81,103,171	495	1.83%	0.46
&	15_100_3	82,205,374	82,642,541	1,245	0.53%	83,243,513	514	1.26%	0.41
100	15_100_4	71,661,053	71,917,954	969	0.36%	72,852,153	430	1.66%	0.44
Subblocks	15_100_5	83,172,711	83,357,904	877	0.22%	84,322,887	380	1.38%	0.43
18	18_120_1	105,028,710	105,269,475	2753	0.23%	106,672,604	790	1.57%	0.29
Vessels	18_120_2	95,105,536	95,420,672	1,849	0.33%	96,876,440	726	1.86%	0.39
&	18_120_3	105,073,034	105,373,151	2,255	0.29%	107,074,052	819	1.90%	0.36
120	18_120_4	95,124,070	95,619,624	2,653	0.52%	96,588,320	858	1.54%	0.32
Subblocks	18_120_5	102,635,574	103,060,198	2,347	0.41%	104,913,811	790	2.22%	0.34
21	21_140_1	120,552,274	121,099,057	4,608	0.45%	122,587,291	1,009	1.69%	0.22
Vessels	21_140_2	114,179,590	114,493,737	3,990	0.28%	115,871,475	1,130	1.48%	0.28
&	21_140_3	118,823,449	119,300,962	4,198	0.40%	120,575,135	918	1.47%	0.22
140	21_140_4	121,006,723	121,274,173	4,293	0.22%	123,646,146	1,119	2.18%	0.26
Subblocks	21_140_5	118,284,955	118,612,993	3,890	0.28%	120,894,503	997	2.21%	0.26
24	24_160_1	132,203,697	132,584,614	5,910	0.29%	133,960,772	1,208	1.33%	0.20
Vessels	24_160_2	128,544,588	129,004,866	6,138	0.36%	130,166,114	1,449	1.26%	0.24
&	24_160_3	131,601,171	132,114,939	5,807	0.39%	134,616,907	1,433	2.29%	0.25
160	24_160_4	133,102,076	133,453,741	6,865	0.26%	135,777,889	1,224	2.01%	0.18
Subblocks	24_160_5	128,220,615	128,842,951	5,779	0.49%	130,518,957	1,150	1.79%	0.20
			Average:		**0.35%**			**1.73%**	**0.31**

Notes: (1) $GAP_L = (Z_L - LB) / LB$, $GAP_P = (Z_P - LB) / LB$. (2) T_L and T_P denote the CPU time of the local branching method and the PSO method, respectively.

For the middle-scale problem instances in Table 2, both the local branching based method and the PSO based method can obtain near-optimal solutions according to the comparison with the lower bounds. The results show that the average gap between the result of the local branching based method (and the PSO based method) and the optimal result is less than 0.35% (and 1.73%). Although, according to the criterion of the objective value, the local branching based method outperforms the PSO based method, the CPU time of the PSO based method is just thirty percent of the CPU time of the local branching based method. This indicates that the PSO based method can obtain satisfying solutions in a much faster way than the local branching based method.

We also compare the performance of the proposed PSO based solution method with certain other widely used metaheuristics, e.g., the genetic algorithm (GA). The results are listed in Table 3, which indicates that the PSO based method outperforms the GA based method in most problem cases. According to the comparison with the lower bound (LB), the average gap between the results of the PSO based method and that of the LB is 1.25%, which is much less than the value of 4.33%, which is the average gap between the results of the GA based method and the LB. As for the CPU time of the two methods, the average ratio of them both is 1.03, which indicates that their computation time is similar to each other.

Table 3: Comparison between the PSO based method and the GA based method

Instances		***LB***	**PSO method**			**GA method**			$\frac{T_G}{T_P}$
Scale	**ID**		Z_P	T_P	GAP_P	Z_G	T_G	GAP_G	
24	24_160_1	130,392,711	132,131,488	1,243	1.33%	135,416,707	1,439	3.85%	1.16
Vessels	24_160_2	127,978,164	129,327,582	1,260	1.05%	130,074,677	1,407	1.64%	1.12
&	24_160_3	127,961,458	130,008,702	1,429	1.60%	138,850,728	1,193	8.51%	0.83
160	24_160_4	129,932,112	131,464,208	1,211	1.18%	137,380,458	1,334	5.73%	1.10
Subblocks	24_160_5	126,414,100	127,587,246	1,355	0.93%	137,705,593	1,556	8.93%	1.15
27	27_180_1	152,499,587	154,200,375	2,132	1.12%	157,850,908	1,987	3.51%	0.93
Vessels	27_180_2	153,262,257	154,919,313	2,664	1.08%	153,303,772	1,755	0.03%	0.66
&	27_180_3	150,199,293	152,177,436	1,878	1.32%	154,663,467	1,590	2.97%	0.85
180	27_180_4	150,989,468	153,336,676	1,911	1.55%	154,679,956	2,724	2.44%	1.43
Subblocks	27_180_5	154,337,434	156,849,526	2,295	1.63%	165,823,329	2,442	7.44%	1.06
30	30_200_1	168,538,444	170,673,470	3,433	1.27%	183,034,096	3,125	8.60%	0.91
Vessels	30_200_2	179,693,092	182,213,882	3,180	1.40%	182,846,052	4,072	1.75%	1.28
&	30_200_3	173,609,696	175,407,547	3,594	1.04%	186,285,321	3,021	7.30%	0.84
200	30_200_4	179,819,015	182,753,841	3,302	1.63%	185,033,276	3,570	2.90%	1.08
Subblocks	30_200_5	176,038,420	178,853,009	2,963	1.60%	178,253,690	4,183	1.26%	1.41
33	33_220_1	202,302,331	205,272,133	4,007	1.47%	214,394,704	3,907	5.98%	0.98
Vessels	33_220_2	201,973,990	204,037,346	5,381	1.02%	205,246,590	4,621	1.62%	0.86
&	33_220_3	199,919,406	201,764,614	4,970	0.92%	208,616,768	4,774	4.35%	0.96
220	33_220_4	203,331,107	205,142,659	5,162	0.89%	207,662,326	4,046	2.13%	0.78
Subblocks	33_220_5	211,257,043	213,501,119	4,343	1.06%	223,257,371	5,319	5.68%	1.22
			Average:		**1.25%**			**4.33%**	**1.03**

Notes: (1) $GAP_P = (Z_P - LB) / LB$, $GAP_G = (Z_G - LB) / LB$. (2) T_P and T_G denote the CPU time of the PSO based method and the GA based method, respectively.

6.3 Evaluating the effectiveness of the proposed models

For evaluating the effectiveness of the proposed models, we compare the proposed decision model M_1 with a common decision rule (FCFS) that allocates subblocks according to the arrival order of vessels. When assigning subblocks to a vessel, we give priority to those subblocks that are both available and near the vessel. As the problem cases are large-scale, the PSO-based solution method is employed to solve the proposed model. The results of the comparative experiments between the proposed model and the decision rule are listed as follows:

Table 4: Comparison between the proposed model M_1 and the FCFS decision rule

Instances		**Model M_1**	**FCFS rule**	**GAP**
Scale	**ID**	Z_{M_1}	Z_{FCFS}	$(Z_{FCFS} - Z_{M\_1}) / Z_{M\_1}$
15	15_100_1	75,575,951	98,201,306	29.94%
vessels	15_100_2	81,103,171	97,044,844	19.66%
&	15_100_3	83,243,513	98,902,482	18.81%
100	15_100_4	72,852,153	96,515,460	32.48%
subblocks	15_100_5	84,322,887	102,404,228	21.44%
21	21_140_1	122,587,291	162,678,507	32.70%
vessels	21_140_2	115,871,475	148,840,186	28.45%
&	21_140_3	120,575,135	147,657,164	22.46%
140	21_140_4	123,646,146	146,046,294	18.12%
subblocks	21_140_5	120,894,503	146,576,991	21.24%
27	27_180_1	154,200,375	188,739,412	22.40%
vessels	27_180_2	154,919,313	187,488,266	21.02%
&	27_180_3	152,177,436	179,102,540	17.69%
180	27_180_4	153,336,676	189,397,220	23.52%
subblocks	27_180_5	156,849,526	211,623,674	34.92%
33	33_220_1	205,272,133	258,978,901	26.16%
vessels	33_220_2	204,037,346	239,366,589	17.32%
&	33_220_3	201,764,614	244,030,955	20.95%
220	33_220_4	205,142,659	256,327,165	24.95%
subblocks	33_220_5	213,501,119	278,434,553	30.41%
			Average:	**24.23%**

Table 4 shows that the proposed model M_1 outperforms the FCFS decision rule by 24% on average with respect to their objective values, i.e., the total route length of transporting containers within the yard. In other words, the proposed model can help the port operator save about 24% of his yard truck transportation costs in comparison to the commonly used FCFS decision rule. This result validates the effectiveness of the proposed model.

Besides model M_1, another decision model M_2 is also proposed in Section 4. Model M_2 is formulated under a policy of equal storage allocation, which follows real-world, pragmatic rules, because this method can help avoid the potential traffic congestion of yard trucks in some subblocks, and can minimize the completion time of the unloading process. We therefore need to investigate how this simplified model M_2 (i.e., the policy of equal storage allocation) will affect the total route length of transporting containers within the yard. A experiment is conducted by comparing the objective values of the solutions obtained by solving the two models.

Table 5: Comparison between the proposed models M_1 and M_2

Solution Methods	Instances		Model M_1		Model M_2		$\frac{Z_2 - Z_1}{Z_1}$	$\frac{T_2}{T_1}$
	Scale	ID	Z_1	T_1	Z_2	T_2		
CPLEX Solver	9	9_60_1	41,165,233	341	45,725,007	53	11.08%	0.16
	vessels	9_60_2	40,191,729	413	44,388,200	42	10.44%	0.10
	&	9_60_3	39,468,476	264	43,846,808	76	11.09%	0.29
	60	9_60_4	41,094,872	547	46,282,198	41	12.62%	0.07
	subblocks	9_60_5	39,829,041	574	45,595,175	63	14.48%	0.11
	12	12_80_1	53,207,203	1,625	65,489,021	263	23.08%	0.16
	vessels	12_80_2	57,201,700	1,547	68,728,497	302	20.15%	0.20
	&	12_80_3	59,538,137	2,090	70392905	254	18.23%	0.12
	80	12_80_4	55,473,855	1,874	68995343	410	24.37%	0.22
	subblocks	12_80_5	59,974,347	1,718	69505982	323	15.89%	0.19
Local Branching	18	18_120_1	105,269,475	2,753	116,978,869	563	11.12%	0.20
	vessels	18_120_2	95,420,672	1,849	116,551,627	617	22.15%	0.33
	&	18_120_3	105,373,151	2,255	118,937,913	578	12.87%	0.26
	120	18_120_4	95,619,624	2,653	117,919,939	790	23.32%	0.30
	subblocks	18_120_5	103,060,198	2,347	119,207,301	755	15.67%	0.32
	21	21_140_1	121,099,057	4,608	141,509,416	877	16.85%	0.19
	vessels	21_140_2	114,493,737	3,990	141,924,529	863	23.96%	0.22
	&	21_140_3	119,300,962	4,198	138,915,115	1,078	16.44%	0.26
	140	21_140_4	121,274,173	4,293	141,293,348	903	16.51%	0.21
	subblocks	21_140_5	118,612,993	3,890	141,784,556	1,175	19.54%	0.30
PSO	27	27_180_1	154,200,375	2,132	192,476,566	1,566	24.82%	0.73
	vessels	27_180_2	154,919,313	2,664	186,904,526	1,480	20.65%	0.56
	&	27_180_3	152,177,436	1,878	185,918,958	1,448	22.17%	0.77
	180	27_180_4	153,336,676	1,911	186,857,614	1,529	21.86%	0.80
	subblocks	27_180_5	156,849,526	2,295	184,973,841	1,645	17.93%	0.72
	30	30_200_1	170,673,470	3,433	231,538,475	2,304	35.66%	0.67
	vessels	30_200_2	182,213,882	3,180	237,806,212	2,476	30.51%	0.78
	&	30_200_3	175,407,547	3,594	239,208,123	3,127	36.37%	0.87
	200	30_200_4	182,753,841	3,302	236,980,620	2,776	29.67%	0.84
	subblocks	30_200_5	178,853,009	2,963	234,799,959	2,543	31.28%	0.86
						Average:	**22.47%**	**0.51**

The results in Table 5 demonstrate that the average gap between their objective values is about 22%, which means that the proposed model M_1 can save about 22% of the transportation costs of yard trucks in comparison to the policy of equal storage allocation. The reason for this evident advantage may be that the berthing position of a vessel (e.g., Vessel 1) might be close to some sub-blocks (e.g., K1, K2), and the berthing position of another vessel (e.g., Vessel 2) might be close to some other sub-blocks (e.g., K3, K4). Using the equal allocation policy will lead to very long travel distance from Vessel 1 to K3, K4, and from Vessel 2 to K1, K2.

Also, as the problem scale grows in size, this relative advantage of the proposed model M_1 becomes more and more significant. These phenomena validate the effectiveness of the proposed model M_1. It should be mentioned that the advantages of model M_1 are based on a longer computation time than model M_2. The results of Table 5 show that the average CPU time for solving model M_2 is almost half of the CPU time needed for solving model M_1. Although the CPU time of solving model M_1 is relatively longer, it is still affordable. This longer computation time is not a significant issue, because yard template planning is a tactical level (long-range) decision problem for port operators.

6.4 Some discussions

At last, it should be mentioned that some existing studies (Lee et al., 2006; Han et al., 2008) proposed a sequential method for solving the problem with homogeneous periodicities. The sequential method is to solve the subblock assignment decision for vessels stage by stage (period by period). However, the sequential method may not be applied to this problem with heterogeneous periodicities after minor modification, because it is difficult to divide the planning horizon into several stages. For a case with homogeneous weekly arrival pattern, the length of the above mentioned stage should be 7 days. However, for a case with both weekly arrival pattern and 10-days arrival pattern, the length of the above mentioned stage should be 70 days (because the least common multiple of seven and ten is 70) if we use the sequential method. When we solve the subblock assignment decision for vessels every 70-day by 70-day, the solution space of the sub-models embedded in this procedure will be very large. If the number of vessels is large, the sub-models may be intractable for the CPLEX solver, or the whole solution process becomes extremely time-consuming.

The local branching based solution method that is proposed in this study does not need to divide the planning horizon into stages as the sequential method. It uses the CPLEX solver as a black-box 'tactical' tool for exploring suitable solution subspaces controlled by an external branching framework. By comparing with the previously proposed sequential method, the advantage of the local branching based method lies in its independence on the stage divisions along the time dimension. Thus the local branching based method can also apply to the problem with homogeneous periodicities.

Moreover, the local branching based method is superior to the previously proposed sequential method when solving the problem with homogeneous periodicities. The reason is as follows. We borrow the idea of the sequential method and designed a method based on the sequence of 'vessel-period' pairs (described in Section 5.1) for generating an initial solution for the local branching based method. Therefore, the solution obtained by the new method is surely not worse than the sequential method. Because this paper's topic focuses on the problem with heterogeneous periodicities, we have not showed the experimental results that compare the two methods for solving the problem with homogeneous periodicities so as to demonstrate the outperformance of the proposed method.

7. Conclusions

This paper studies the yard template planning problem for arriving vessels, which visit the port periodically, e.g., weekly, biweekly, every ten days. A mixed integer programming model is proposed for this problem. In addition, a local branching based solution method and a PSO based solution method are developed for solving the model. Numerical experiments are also conducted to validate the effectiveness of the proposed model, which can save around 24% of the transportation costs of yard trucks in comparison with the commonly used FCFS decision rule. Moreover, the numerical experiment also shows that the proposed local branching method can obtain near optimal solutions within a much shorter time than the CPLEX solver. The gap from the optimal objective value is only 0.09% on average. In addition, the proposed PSO solution method outperforms the widely used GA based solution method, and can not only solve the proposed model within a reasonable time but can also obtain near-optimal results, with only about 1.73% relative gap from the lower bound. The contribution this study makes is mainly in regard to the following three aspects:

(1) Most previous yard template related studies do not consider the multiple periods of the planning horizon, which is especially important because the cycle time of the vessels' periodicities is not uniform among them. This paper makes an explorative study into this area, and proposes a model for the multi-period yard template planning problem that takes account of the heterogeneous periodicities of vessels.

(2) Yard traffic congestion factors are usually ignored in many yard management studies. The proposed model in this study contains particular constraints that minimize or avoid this yard truck congestion phenomenon, which has become an important factor that limits the performance of a container terminal.

(3) For solving the proposed integer programming model under large-scale problem cases, this study develops two solution methods, based on local branching and the PSO. Their efficiency is validated by several numerical experiments on real world like instances.

However, there are limitations for the current models. The objective of the models (i.e., to minimize the cost of transportation between subblocks and berths) mainly considers laden trips, but does not take account of all the empty trips occurring in a real-world environment. In addition, as more and more maritime logistic related studies consider uncertain issues, the models in this paper can then be extended to consider these uncertain factors, such as stochastic arrival time, operation time, and berthing positions of vessels. These limitations will form the research directions for our future studies.

Appendices

Appendix A. Proof of Proposition 1

Proposition 1: Finding an optimal solution for model M_1 is strongly NP-hard.

Proof: We can prove the proposition by a reduction to the problem from the maximum independent set problem, which is well-known to be strongly NP-complete. Given an undirected graph $G = (V, E)$ with the vertex set $V = \{1,2,\cdots,n\}$ and the edge set E, the maximum independent set problem is to find an independent set of size greater than or equal to m, i.e., to find a subset S of V with $|S| \geq m$ such that no pairs of vertices u and v in S are joined by an edge in E.

Given any instance of the maximum independent set problem, consider the following instance of the problem, where there is only one vessel that requires m subblocks of the yard at period 1. The yard has n subblocks, and for each pair of subblocks, u and v, they cannot be assigned to the vessel during the same time period if, and only if (u, v) is in E. It can be seen that the yard planning problem has a feasible solution if and only if G has an independent set of size greater than m. Therefore, finding an optimal solution to the yard planning problem is strongly NP-hard. ■

Appendix B. Proof of Proposition 2

Proposition 2: If there exist feasible solutions for model M_2, there are more than one optimal solutions (optimal yard templates).

Proof: As shown in Figure 1, a yard template contains blocks, which can be further divided into subblocks. Above the level of block, there is another concept, namely ‘block section’ (or big block), which is also widely used in real-world yard management (Lee et al., 2006; Han et al., 2008). A block section is composed of one or two blocks that share the same horizontal lane. An illustration of the ‘block section’ concept is shown in Figure A.

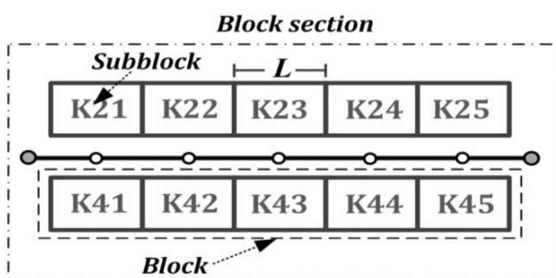

Figure A: An example of block sections in a yard template

For the subblocks in a block section, any change in their assignment to a particular vessel will not affect the final objective value. For the example in Figure A, Subblock K22 is assigned to Vessel i, and Subblock K24 is assigned to Vessel i'. If we exchange them, then K22 is assigned to i', and K24 is assigned to i. However, the objective values of both of the two vessels will not change. For Vessel i, the numbers of both the unloaded and loaded containers going through this lane are the same as in the previous case, i.e., ‘$(\sum_{j \in V} n_{j,i,p})/r_{i,p}$’. For Vessel i, the length of the *loading* route decreases by ‘$2L$’, and the length of the *unloading* route increases by ‘$2L$’, L being the length of a subblock. According to the definition of the model’s objective, i.e., the sum of the containers multiplied by their travel distance, the reassignment of subblocks to a vessel within a block section has no effect on the final objective value.

Another important issue also needs clarifying. This issue concerns the fact that reassigning subblocks to a vessel within a block section also has no influence on the constraints. The reason is that model M_2 assumes that any containers for transshipping to a vessel are equally distributed within the subblocks reserved for that vessel. This policy results in the fact that, no matter where they are stored in the subblocks reserved for a vessel, the number of containers actually totals the same. Therefore, any reassignment has no influence on the constraints.

As the reassignment of subblocks to a vessel within a block section will not affect either the objective value or the constraints, we can conclude that there exists more than one optimal solution (optimal yard template) for model M_2. ■

Appendix C. Proof of Proposition 3

Proposition 3: An optimal yard template for model M_2 has the following properties: (1) At least one subblock in the block that is nearest to the berthing position of Vessel i_{max} is assigned to Vessel i_{max} for its Period p_{max}. (2) At least one subblock in the block section being used, and which is the furthest from the berthing position of Vessel i_{min}, is assigned to Vessel i_{min} for its Period p_{min}. Here $(i_{max}, p_{max}) = \arg\max_{\forall(i,p),i\in V,p\in\mathbb{E}^i} \{(\sum_{j\in V} n_{j,i,p})/q_{i,p}\}$, $(i_{min}, p_{min}) = \arg\min_{\forall(i,p),i\in V,p\in\mathbb{E}^i} \{(\sum_{j\in V} n_{j,i,p})/q_{i,p}\}$.

Proof: We prove property (1) by two steps. Each block's position in a yard template has two dimensional coordinates; one is the vertical dimension, and the other is the horizontal dimension.

The first step is in the vertical dimension. We prove that at least one subblock in the block row that is nearest to the quay side is assigned to Vessel i_{max} for its Period p_{max}. We prove it by contradiction. Suppose that in an optimal yard template, none of the subblocks in the row that is nearest to the quay side is assigned to Vessel i_{max}. Using the example in Figure 1, if Subblock K25 is assigned to Vessel i_{max} and Subblock K65 is assigned to Vessel i, then we exchange their assignments: K65 is assigned to Vessel i_{max} and K25 is assigned to Vessel i. The objective value related to Vessel i increases by $2\mathrm{W}(\sum_{j\in V} n_{j,i,p})/q_{i,p}$; while the objective value related to Vessel i_{max} decreases by $2\mathrm{W}(\sum_{j\in V} n_{j,i_{max},p_{max}})/q_{i_{max},p_{max}}$. Here W is the distance between two parallel adjacent passing lanes in the horizontal dimension. As

$2W(\sum_{j\in V} n_{j,i_{max},p_{max}})/q_{i_{max},p_{max}} > 2W(\sum_{j\in V} n_{j,i,p})/q_{i,p}$, the final objective value will decrease, which contradicts the fact that the given solution is optimal.

The second step is in the horizontal dimension. In the row of subblocks that is next to the quay side, we define x as the horizontal position of the subblock assigned to Vessel i, and s_i as the berthing position favored by Vessel i. $s_{i'_1}$, ⋯, $s_{i'_m}$, ⋯, $s_{i'_M}$ are the berthing positions favored by Vessel i'_1, ⋯, i'_m, ⋯, i'_M, respectively. These vessels all tranship containers to Vessel i. The objective value related to the subblock assignment for Vessel i is proportional to $f(x) = (\sum_{m=1}^{M} n_{i'_m,i}) \cdot |x - s_i| + \sum_{m=1}^{M}\{n_{i'_m,i} \cdot |x - s_{i'_m}|\}$. It is easy to prove that when $x = s_i$, $f(x)$ reaches its minimum value. This means that the berthing position of a vessel affects the decision on its subblock assignments, from which it is easy to see that not all vessels can have their favorite subblocks allocated to them. However, Vessel i_{max} can always be allocated with its favorite subblock, as stated in property (1). Following is the proof of the above statement by contradiction.

Suppose that in an optimal yard template, none of the subblocks in the block that is nearest to the berthing position of Vessel i_{max} is assigned to Vessel i_{max} for its Period p_{max}. We use $s_{i_{max}}$ to denote Vessel i_{max}'s favourite subblock, but this subblock is already assigned to another Vessel i. A subblock in location x is assigned to Vessel i_{max}. Without loss of the generality, we assume $x > s_{i_{max}}$, which means that position x is on the right of position $s_{i_{max}}$ along the horizontal axis. We exchange the subblock assignment for the two vessels. $s_{i_{max}}$ is assigned to Vessel i_{max}, and x is assigned to Vessel i. Case 1: If x is the best position for Vessel i, obviously this exchange will surely decrease the objective value, which contradicts the fact that the given solution is optimal. Case 2: If x is the worst position for Vessel i, which implies that the berthing position of Vessel i is on the left of position $s_{i_{max}}$, the objective value of Vessel i increases by $(x - s_{i_{max}}) \cdot (\sum_{j\in V} n_{j,i,p})/q_{i,p}$, and the objective value of Vessel i_{max} decreases by $(x - s_{i_{max}}) \cdot (\sum_{j\in V} n_{j,i_{max},p_{max}})/q_{i_{max},p_{max}}$. As $(\sum_{j\in V} n_{j,i_{max},p_{max}})/q_{i_{max},p_{max}} > (\sum_{j\in V} n_{j,i,p})/q_{i,p}$ the final objective value will decrease, which also contradicts the fact that the given solution is optimal. Case 3: If x is between the worst position and the best position, this is also easy to prove.

Similar to the above proof of property (1) by contradiction, we can also prove the property (2) that there is at least one subblock in the block section used that is furthest from the berthing position of Vessel i_{min}. This subblock is assigned to Vessel i_{min} for its Period p_{min}. ■

Appendix D. The sub-procedure $Adjust(m)$ in the PSO based method

The sub-procedure $\boldsymbol{Adjust}(m)$

Initialize all the variable X_{mikp} as zero
For all the i, $i \in V$
  For all the p, $E_p^i \in \mathbb{E}^i$
    Define a set K_{ip} // This set is used to avoid assigning a subblock to several vessels in a time step
    If $i = 1$ Then
      $K_{ip} = \emptyset$
    Else
      $K_{ip} = \left\{k \middle| Y_{mi_1kp_1}^n = 1, i_1 \in \{1, \dots, i-1\}, k \in K, E_p^i \cap E_{p_1}^{i_1} \neq \emptyset, E_{p_1}^{i_1} \in \mathbb{E}^{i_1}\right\}$
    End If
    For all the k, $k \in K$
      If $Y_{mikp}^n = 1$ and $k \in K_{ip}$ Then
        $Y_{mikp}^n = 0$
      End If
      If $Y_{mikp}^n = 1$ and $k \in \{K - Q_i\}$ Then
        $Y_{mikp}^n = 0$ // According to Constraint (4)
      End If
    End For
    Sum up all the subblocks assigned to Vessel i in Period p, the total number is $Total_{ip}$
    If $Total_{ip} < q_{ip}$ Then
      For all $k \in Q_i$ and $k \notin K_{ip}$, sort them by the decreasing order of $S(v_{mikp}^n)$, and then set $Y_{mikp}^n = 1$ until $Total_{ip} = q_{ip}$ // According to Constraint (3)
    End If
    If $Total_{ip} > q_{ip}$ Then
      For all $k \in Q_i$ and $k \notin K_{ip}$, sort them by the inceasing order of $S(v_{mikp}^n)$, and then set $Y_{mikp}^n = 0$ until $Total_{ip} = q_{ip}$ // According to Constraint (3)
    End If
    Calculate X_{ikt}^m according to Y_{mikp}^n
    For all $t \in T, S_r \in \mathbb{S}, B_g \in \mathbb{B}$
      If $\sum_{\boldsymbol{k} \in S_r} \sum_{i_1 \in \{1,..i\}} l_{i_1t}\, x_{i_1kt} > 1$ or $\sum_{\boldsymbol{k} \in B_g} \sum_{i_1 \in \{1,..i\}} l_{i_1t}\, x_{i_1kt} > 1$ Then
        As to Vessel i, unassign the subblocks that disobey the constraints, and assign the subblocks that are randomly chosen from the set $\{k | k \in Q_i \text{ and } k \notin K_{ip}\}$ to the vessel. Repeat this process until Constraints (5) and (6) hold.
      End If
    End For
End For

Acknowledgements

The authors would like to thank anonymous reviewers for their valuable comments and constructive suggestions, which have greatly improved the quality of this paper. For this research, the first author is supported by the National Natural Science Foundation of China (71422007, 71101087), Program for Professor of Special Appointment (Eastern Scholar) at Shanghai Institutions of Higher Learning, Shanghai Social Science Research Program (2014BGL006); the second author is supported in part by Hong Kong Research Grants Council (RGC), General Research Fund (GRF) No. PolyU 152186/14E.

References

APL. 2015a. Service. http://aplinfo.apl.com/services/documents/sells_mkt_ia_cokcok2.pdf. Accessed 24 June, 2015.

APL. 2015b. Service. http://aplinfo.apl.com/services/documents/sells_mkt_ia_ct1ct2ct3.pdf. Accessed 24 June, 2015.

Bell, M.G.H., Liu, X., Rioult, J., Angeloudis, P., 2013. A cost-based maritime container assignment model. *Transportation Research Part B* 58, 58-70.

Caserta, M., Voß, S., Sniedovich, M. 2011. Applying the corridor method to a blocks relocation problem. *OR Spectrum* 33, 915-929.

Chen, C.S., Lee, S. M., Shen, Q. S. 1995. An analytical model for the container loading problem. *European Journal of Operational Research* 80, 68-76.

CMA-CGM. 2015. Service. http://www.cma-cgm.com/products-services/line-services/flyer/COLUMBUS. Accessed 24 June, 2015.

Cordeau, J.-F., Gaudioso, M., Laporte, G., Moccia, L. 2007. The service allocation problem at the Gioia Tauro Maritime Terminal. *European Journal of Operational Research* 176, 1167-1184.

Dekker, R., Voogd, P., Asperen, E. 2006. Advanced methods for container stacking. *OR Spectrum* 28, 563-586.

Eberhart, R.C., Kennedy, J. 1995. A new optimizer using particle swarm theory. *Proceedings of 6th international symposium on micromachine and human science Nagoya Japan*, 39-43.

Fischetti, M., Lodi, A. 2003. Local branching. *Mathematical Programming Series B* 98, 23-47.

Fransoo, J.C., Lee, C.-Y. 2012. The Critical Role of Ocean Container Transport in Global Supply Chain Performance. *Production and Operations Management* 22, 253-268.

Giallombardo, G., Moccia, L., Salani, M., and Vacca, I. 2010. Modeling and solving the Tactical Berth Allocation Problem. *Transportation Research Part B* 44, 232-245.

Han, Y., Lee, L.H., Chew, E.P., Tan, K.C. 2008. A yard storage strategy for minimizing traffic congestion in a marine container transshipment hub. *OR Spectrum* 30, 697-720.

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, 64-73.

Jin J.G., Lee, D.-H., Cao, J.X. (2014) Storage Yard Management in Maritime Container Terminals. *Transportation Science* (in press) doi: 10.1287/trsc.2014.0527.

Kim, K.H., Kim, H.B. 1999. Segregating space allocation models for container inventories in port container terminals. *International Journal of Production Economics* 59, 415-423.

Kim, K.H., Park, Y.M., Ryu, K.R. 2000. Deriving decision rules to locate export containers in storageyards. *European Journal of Operational Research* 124, 89-101.

Kozan, E., Preston, P. 2006. Mathematical modeling of container transfers and storage locations at seaport terminals. *OR Spectrum* 28, 519-537.

Lee, L.H., Chew, E.P., Tan, K.C., Han, Y. 2006. An optimization model for storage yard management in transshipment hubs. *OR Spectrum* 28, 539-561.

Meng, Q., Wang, S., Andersson, H., Thun, K., 2014. Containership routing and scheduling in liner shipping: overview and future research directions. *Transportation Science* 48, 265-280.

Moccia, L., Cordeau, J.-F., Monaco, M.F., Sammarra, M. 2009. A column generation heuristic for a dynamic generalized assignment problem. *Computer and Operations Research* 36, 2670-2681.

Moorthy, R., Teo, C.P. 2006. Berth management in container terminals: the template design problem. *OR Spectrum* 28, 495-518.

Nishimura, E., Imai, A., Janssens, G.K., Papadimitriou, S. 2009. Container storage and transshipment marine terminals. *Transportation Research Part E* 45, 771-786.

Stahlbock, R., Voß, S. 2008. Operations research at container terminals: A literature update. *OR Spectrum* 30, 1-52.

Steenken, D., Voß, S., Stahlbock, R. 2004. Container terminal operation and operations research - A classification and literature review. *OR Spectrum* 26, 3-49.

Tang, L., Zhao, J., Liu, J. 2014. Modeling and solution of the joint quay crane and truck scheduling problem. *European Journal of Operational Research* 236, 978-990.

Ting, C.-J., Wu, K.-C., Chou, H. 2014. Particle swarm optimization algorithm for the berth allocation problem. *Expert Systems with Applications* 41, 1543-1550.

Vis, I.F.A., de Koster, R. 2003. Transshipment of containers at a container terminal: An overview. *European Journal of Operational Research* 147, 1-16.

Wang, S., Liu, Z., Bell, M.G.H. 2015. Profit-based maritime container assignment models for liner shipping networks. *Transportation Research Part B* 72, 59-76.

Zhang, C., Liu, J., Wan, Y.-W., Murty, K.G., Linn, R.J. 2003. Storage space allocation in container terminals. *Transportation Research Part B* 37, 883-903.

Zhang, M., Batta, R., Nagi, R. 2009. Modeling of Workflow Congestion and Optimization of Flow Routing in a Manufacturing/Warehouse Facility. *Management Science* 55, 267-280.

Zhen, L., Chew, E.P. Lee, L.H. 2011. An Integrated Model for Berth Template and Yard Template Planning in Transshipment Hubs. *Transportation Science* 45, 483-504.

Zhen, L. 2013. Yard template planning in transshipment hubs under uncertain berthing time and position. *Journal of the Operational Research Society* 64, 1418–1428.