STORAGE YARD MANAGEMENT FOR CONTAINER TRANSSHIPMENT TERMINALS

JIN JIANGANG

NATIONAL UNIVERSITY OF SINGAPORE

2012

STORAGE YARD MANAGEMENT FOR CONTAINER TRANSSHIPMENT TERMINALS

JIN JIANGANG

(B. Eng. Tsinghua University)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF CIVIL AND ENVIRONMENTAL ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2012

Declaration

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Jin Jiangang

Jin Jiangang

22 Oct 2012

Acknowledgements

First I would like to thank my thesis committee members, Prof Lee Der-Horng, Prof Meng Qiang and Prof Tan Kok Choon for their time, suggestions and valuable comments. I also would like to thank the National University of Singapore for the President's Graduate Fellowship.

My enduring appreciation goes to Professor Lee Der-Horng who served as an "advisor" instead of "supervisor" throughout my PhD study. I have been greatly enjoying the freedom that he gave me in research and also benefited a lot from his sharp views and encouragements whenever I come across difficulties both in research and personal matters. His unique eloquence and great sense of humor have also become my treasure that will company me in the future. I am also thankful to Professor Meng Qiang who is undoubtedly a great educator. His enlightening lectures and unique insights on mathematical knowledge guided me intellectually into the research field of optimization and operations research. Prof Meng's kindness in sharing his experience and vision also left me with deep impression.

I am grateful to Dr Chen Jianghang whom I consider as my co-supervisor. I really benefited a lot from his sharing of knowledge, experience and passion in research. I also would like to thank all my colleagues and friends in NUS for the support and companionship, Huang Sixuan, Zhang Yang, Fu Yingfei, Wu Xian, Zheng Yanding, Liu Zhiyuan, Wang Shuaian, Zhang Jian, Sun Lijun, Li Siyu, Qin Han, He Nanxi, Lu Zhaoyang, Sun Leilei.

Last but not the least, I would like to take this opportunity to express wholehearted gratitude to my parents and girlfriend, Qian Junni, for their endless love and support all the way along.

Table of contents

Executive Summary

Container terminals are crucial nodes of the world's freight transportation network where intermodal services are provided including ship-to-shore services and vice versa. Since the emergence of containerized transportation, the volume of container throughput has been increasing steadily and is expected to continue growing in the future. The growing trend places port operators into a challenging situation: to achieve higher operational efficiency given limited resources. This provides a great opportunity of applying optimization techniques into various decision problems in container terminals to improve the overall performance.

This thesis is dedicated to the storage yard management for container transshipment terminals by following the promising research trend, integrated optimization approach, to develop new optimization models and solution approaches. Focusing on the *storage yard allocation problem* (SAP), two directions of integrated optimization are explored: Part I-Integration of SAP and *berth allocation problem* (BAP), and Part II-Integration of SAP and *yard crane deployment/scheduling problem* (YCDP/YCSP). The first part of the thesis deals with the integration of BAP and SAP in two transshipment terminal systems (single-terminal system and multi-terminal system). Inter-dependent decisions at the quayside (berth allocation and feeder vessel calling schedule) are modelled together with storage allocation decisions. Mathematical models and heuristic methods are developed accordingly in order to obtain an integrated berth, feeder schedule and storage template which supports the tactical planning for the two terminal systems. In the second part, the integration of SAP and YCDP/YCSP are studied. Focusing at the planning and operations within the storage yard, this part models yard crane operations simultaneously with storage allocation at two planning levels: tactical level with the operation area of the entire storage yard, and operational level with the operation area of a single yard block. Models and heuristics are proposed accordingly in order to enhance yard crane efficiency and storage effectiveness in the storage yard.

In summary, this thesis provides a comprehensive planning framework for storage yard management at container transshipment terminals. It supports storage yard allocation decisions and other interdependent decisions for terminal operators with various planning areas: single

yard block, single-terminal system and multi-terminal system, and also with various planning levels: strategic design, tactical planning and operational scheduling.

List of tables

List of figures

Chapter 1

Introduction

1.1 Background

Maritime freight transport is an important part of the global logistics system. With over 80 percent of world merchandized trade by volume being carried by sea, maritime transport remains the backbone supporting international trade and globalization (UNCTAD, 2008). Benefiting from the globalization, maritime freight transport has been growing steadily in the past several decades and the trend continues to remain strong as shown in Figure 1.1. Containerization is an evolutionary change within the development of maritime freight transport. It has increased the efficiency of freight shipping to a large extent, and has also enhanced the connectivity of maritime transport with other modes, such as train and truck transportation. Thanks to the standardized intermodal containers, modern container terminals and container shipping facilities (containerships and trucks) form a system of international freight transport.

Since the introduction of containerization and mega-vessels (Emma Maersk with a capacity of over 15,000 TEUs), benefits from the economies of scale have further boosted the liner shipping industry. Meanwhile, the hub-and-spoke system began to be implemented in which large container vessels (mother vessels) visit a limited number of transshipment terminals (hubs) while small vessels (feeders) connect the hubs with other ports (spokes). Serving as an interface between maritime transport and land transport, container terminals are important nodes of the whole shipping network providing intermodal services. The productivity of a container terminal

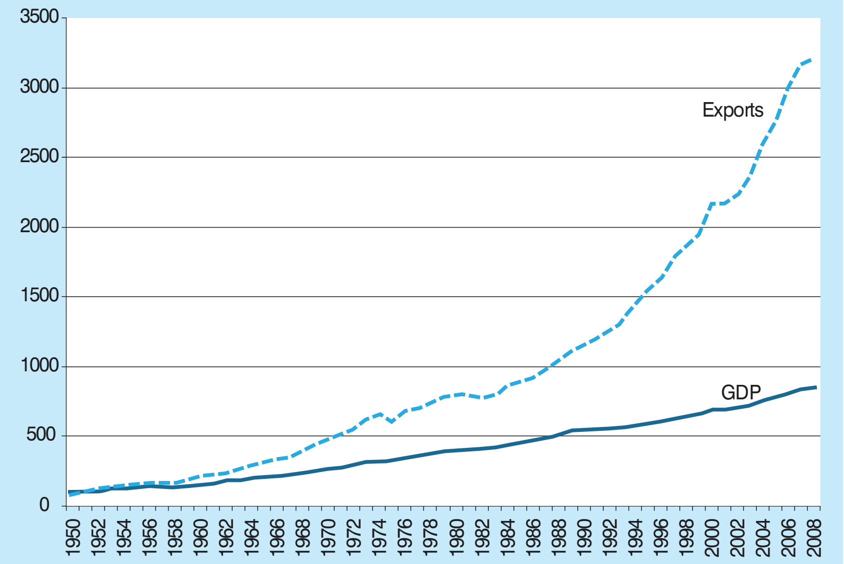


Figure 1.1: Indices for world economic growth (GDP) and world merchandise exports (volume). (1950 = 100) (UNCTAD, 2008)

not only determines its competitiveness and attractiveness, but may even affect the efficiency of the whole container shipping network, especially large transshipment hubs. Thus, efficient management of the logistic activities at container terminals should be well ensured.

As a key node of the maritime shipping network, a container terminal is a complex system involving many operations: berth allocation for incoming vessels, quay/yard crane assignment and scheduling for container loading and discharging, yard storage space allocation for container temporary storage, and truck scheduling for container movement between quayside and storage yard. In order to enhance the port competitiveness, container terminal operators, especially those operating large transshipment hubs, are always seeking to improve their services by employing modern handling equipment, and adopting advanced information and management systems. However, each of the terminal operations is a complex and highly dynamic subsystem which makes it intractable to operate the whole terminal system involving various operational areas and handling equipment. As the container throughput increases along with the booming of the shipping industry, more challenges emerge for container terminal operators, such as the management of new handling equipment and efficient storage management with land scarcity issues.

Both practitioners and researchers have devoted efforts to tackling the challenges arising at container terminals. For container terminal operators, optimization of these terminal operations mentioned above is vital in that a small improvement in efficiency could lead to significant cost reduction. For academic researchers, such a complex system makes it an ideal research area for applying advanced optimization techniques. Research on container terminal operations has been an active research topic in the past few decades and has been receiving increasing attention and interest.

1.2 Container Terminal Operations

A container terminal is a complex system which can be classified into three operation areas in terms of functionality: quayside, yard and landside. Figure 1.2 shows these three areas of a seaport container terminal and the flow of container movements. Quayside area is dedicated to loading and unloading for containerships while the yard functions as a temporary storage area. The landside area is used for container exchange between the terminal and inland customers. Container moves are commonly conducted among different areas in daily terminal operations. The standardization of containers allows terminal operators to design and employ respective handling equipment (yard truck, quay crane and yard crane) as depicted in Figure 1.3. Yard trucks are employed for moving containers while quay cranes and yard cranes are in charge of container pickup and delivery at the quayside and yard area, respectively.

In practice, the whole container terminal system is decomposed into several smaller sub-systems by terminal operators for the sake of easy management. The typical decision problems related to these sub-systems are:

- **Berth Allocation Problem (BAP)**: The main task of BAP is to decide where (along the quay) and when each container vessel should moor in the planning horizon by considering the characteristics of incoming vessels and resource constraints.

- **Quay Crane Allocation and Scheduling Problem (QCAP, QCSP)**: The QCAP determines the assignment of quay cranes to currently berthing vessels while the QCSP

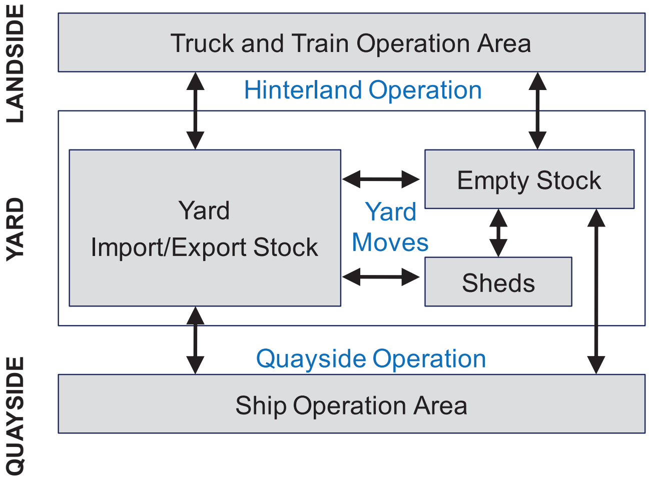


Figure 1.2: Operation areas of a seaport container terminal and flow of container movements. (Steenken et al., 2004)

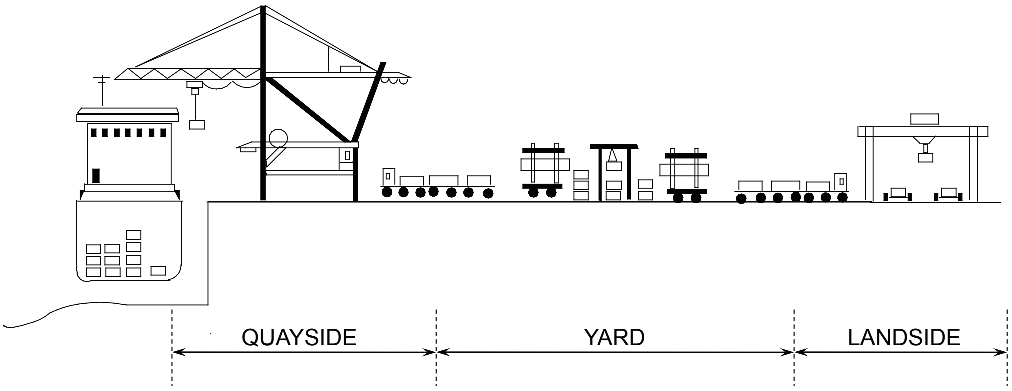


Figure 1.3: A typical container terminal system (Monaco et al., 2009)

focuses on the detailed scheduling of container loading and discharging operations for the allocated quay cranes associated with a certain vessel.

- **Yard Truck Scheduling Problem (YTSP)**: The YTSP is mainly to deal with the scheduling of yard trucks for container movement between quayside and yardside in order to synchronize the quay crane and yard crane operations.
- **Yard Crane Deployment and Scheduling Problem (YCDP, YCSP)**: The YCDP is to deploy yard cranes according to the workload in the whole yard area. Unlike the YCDP dealing with the yard cranes' movement in the whole yard, YCSP looks into a certain yard block and schedules the detailed pickup and delivery operations for yard cranes.
- **Storage Allocation Problem (SAP)**: The SAP deals with the assignment of yard storage space to containers for temporary storage, and possible container relocation decisions in their duration-of-stay.

1.3 Research Scope and Objective

Traditionally the above decision problems are solved hierarchically by terminal operators from those at the higher planning levels to others at the lower planning levels. Figure 1.4 shows such a hierarchical planning structure. At the high planning level, BAP is solved firstly and the decisions are passed onto SAP as input information. With determined berth allocation and yard allocation plans, the decision problems belonging to the medium planning level are solved including QCAP, QCSP, YCDP and YCSP. At the low planning level, the YTSP is solved after the schedules of quay cranes and yard cranes are known. The arrows in the figure indicate the information passing direction and also the solving sequence of these decision problems. The advantage of the hierarchical planning method is that the whole complex system is broken into smaller sequential problems which are much simplified and relatively easy to tackle.

In the research field, most previous literature also employs the hierarchical planning method. However, a vital weakness of the hierarchical approach is the failure of considering the inter-

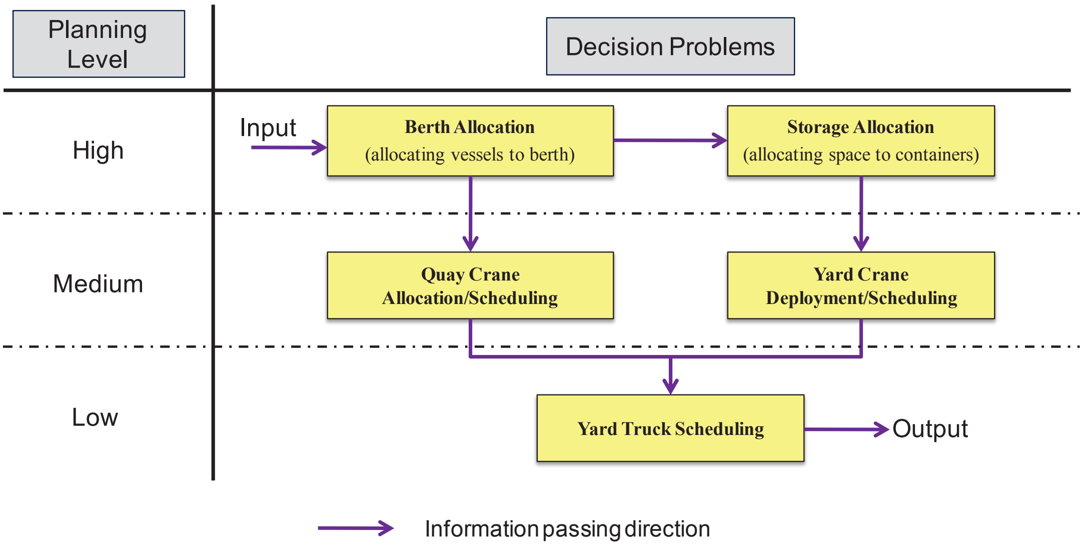

Planning Level
Decision Problems
High
Medium
Low
Input
Berth Allocation (allocating vessels to berth)
Storage Allocation (allocating space to containers)
Quay Crane Allocation/Scheduling
Yard Crane Deployment/Scheduling
Yard Truck Scheduling
Output
Information passing direction

Figure 1.4: Hierarchical planning structure of operational decisions in port container terminals.

action between different decision problems which are highly interdependent. Take BAP as an example, aiming at minimizing vessel turnaround time berth allocation decisions are made with no consideration of the container storage layout in the yard. However, storage decisions will also affect the turnaround time. The same problem exists between other inter-related decision problems. Comprehensive review of existing literature is provided in Chapter 2 and the literature review sections in Chapters 3∼7.

A promising trend on the research of container terminal operations is the integrated optimization for highly interdependent decision problems. The integrated optimization approach can act as an overall decision making center and yield better overall performance. In other words, this approach provides a feedback from the downstream decision problem to the upstream one. Figure 1.5 shows two examples of such a feedback which allows the integration of two decision problems.

Although the necessity of integrating inter-dependent decision problems for various container port operations is well recognized in the literature, the hierarchical planning approach is still prevalent due to its simplicity. Efforts should be devoted to applying the integrated planning approach for further improvement on the overall performance of terminals operations. A few recent studies have been directed to the integrated optimization approach and have yielded

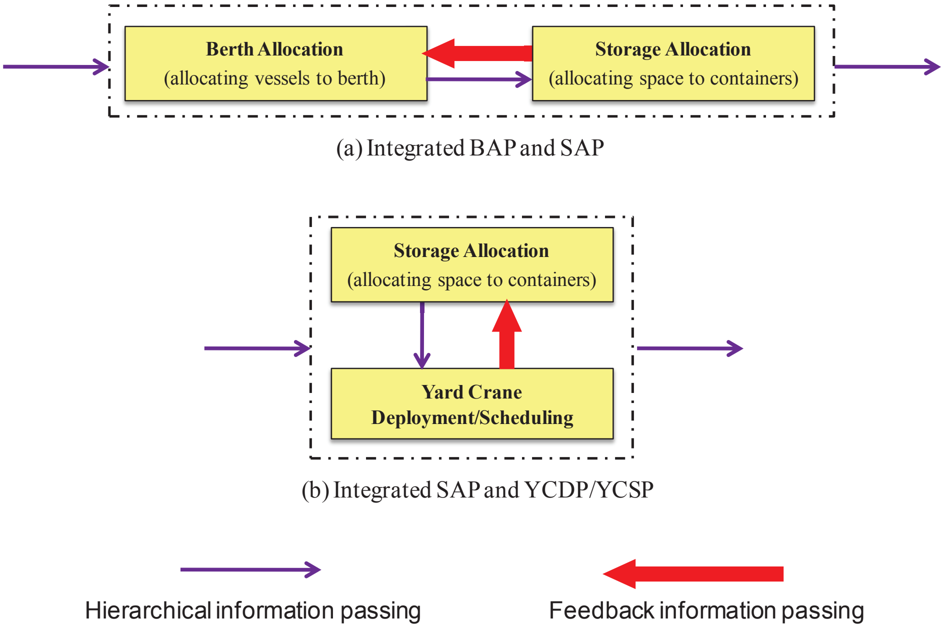

Berth Allocation
(allocating vessels to berth)
Storage Allocation
(allocating space to containers)
(a) Integrated BAP and SAP
Storage Allocation
(allocating space to containers)
Yard Crane
Deployment/Scheduling
(b) Integrated SAP and YCDP/YCSP
Hierarchical information passing
Feedback information passing

Figure 1.5: Two examples of integrated optimization for interdependent decision problems.

benefits. However, very few focus on the storage yard allocation problem which is a key challenge for large container terminals with land scarcity issues. It is necessary to explore efficient storage yard management and improve the utilization of storage space by the integrated optimization approach.

This thesis is dedicated to the storage yard management for container terminals and follows the promising research trend, integrated optimization approach, to develop new optimization models and algorithms. The organization of this thesis is illustrated by Figure 1.6. The whole thesis includes two parts listed as follows:

Part 1: Integrated Berth Allocation Problem & Storage Allocation Problem

The integration of BAP and SAP intends to include the decisions of quayside operations into the storage allocation problem, as the container flow inside the yard is highly relevant to berth allocation decision especially for transshipment containers. We apply this motivation into two types of container terminal systems: single-terminal and multi-terminal system. For a multi-terminal system operated by the same port operator, there is an important planning issue called

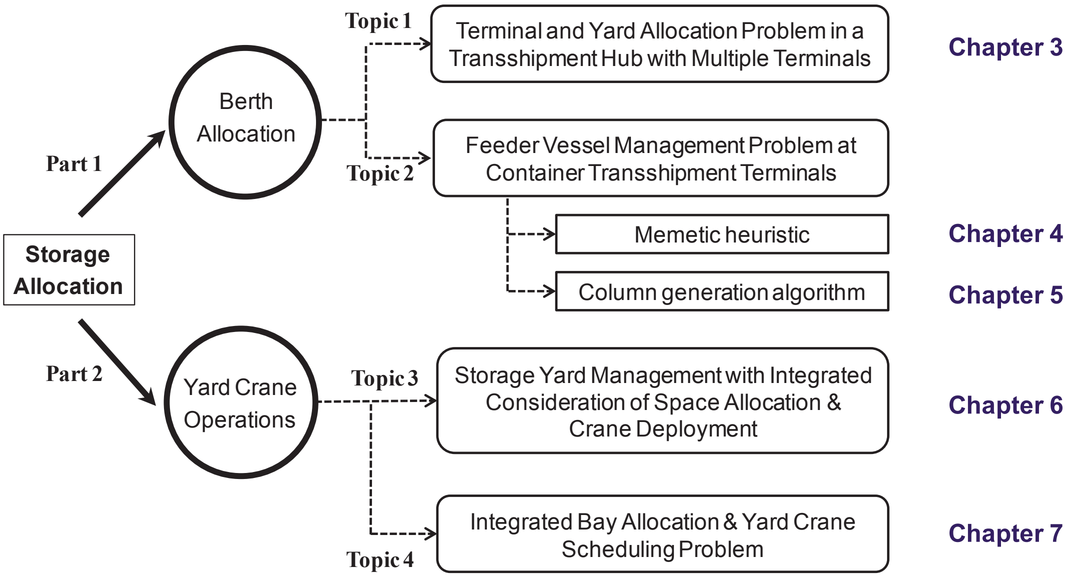

Storage Allocation
Part 1
Part 2
Berth Allocation
Yard Crane Operations
Topic 1
Topic 2
Topic 3
Topic 4
Terminal and Yard Allocation Problem in a Transshipment Hub with Multiple Terminals
Feeder Vessel Management Problem at Container Transshipment Terminals
Memetic heuristic
Column generation algorithm
Storage Yard Management with Integrated Consideration of Space Allocation & Crane Deployment
Integrated Bay Allocation & Yard Crane Scheduling Problem
Chapter 3
Chapter 4
Chapter 5
Chapter 6
Chapter 7

Figure 1.6: A schematic diagram of the thesis organization.

inter-terminal container movement which is not covered in open literature. Integrating berth allocation and yard allocation is a promising approach to reduce inter-terminal container movement volume. Besides berth allocation decision, the yard storage status is also affected by vessel calling schedules. Instead of satisfying feeder calling schedules passively, terminal operators can adopt a proactive management strategy and adjust feeder calling schedules by negotiation with shipping companies. Thus, we consider the impact of berth allocation and feeder calling schedule on the storage efficiency in a single terminal system. Mathematical models, meta-heuristic methods and exact algorithms will be developed accordingly. The application of the integrated optimization to a multi-terminal system will be studied in Chapter 3 while the application to a single terminal system will be covered in Chapter 4 & 5.

Part 2: Integrated Storage Allocation Problem & Yard Crane Deployment/Scheduling Problem

The integration of SAP and YCDP/YCSP can be analyzed at two planning levels: tactical and operational level. The planning on the tactical level looks at the entire storage yard. On one hand, it determines the storage locations for incoming container groups. On the other hand, yard cranes are deployed over the entire yard accordingly to carry out the receiving and retrieving tasks. With the determined tactical decisions, the operational level focuses on the YC scheduling within a single yard block. YC scheduling and bay allocation decisions are considered simultaneously in order to enhance the YC's efficiency of receiving and retrieving operations. The two integrated problems will be studied by mathematical modeling techniques and heuristic algorithms in Chapter 6 & 7 respectively.

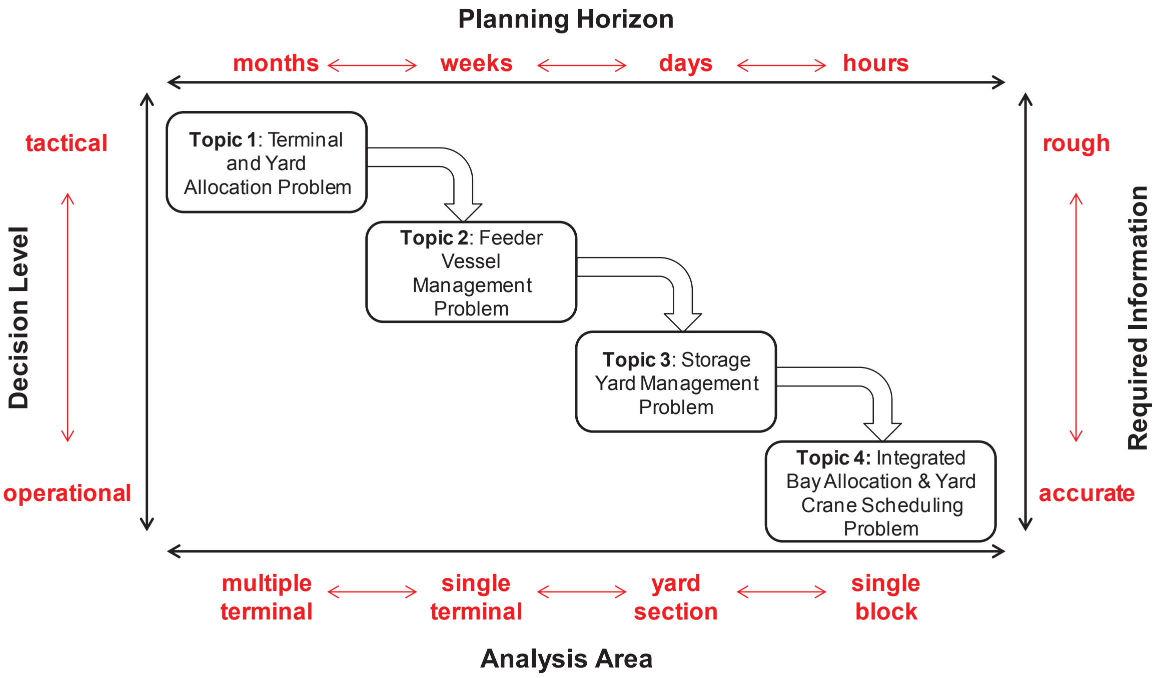

Planning Horizon
months
weeks
days
hours
tactical
operational
Decision Level
rough
accurate
Required Information
Topic 1: Terminal and Yard Allocation Problem
Topic 2: Feeder Vessel Management Problem
Topic 3: Storage Yard Management Problem
Topic 4: Integrated Bay Allocation & Yard Crane Scheduling Problem
multiple terminal
single terminal
yard section
single block
Analysis Area

Figure 1.7: Representation of the proposed comprehensive planning framework for storage yard management.

This research provides a comprehensive planning framework for the storage yard management at container transshipment terminals as shown in Figure 1.7. Storage yard allocation is tackled at various planning levels (tactical and operational) with various planning areas (multi-terminal, single-terminal, yard section and single yard block). The proposed individual research topics covers a wide range of planning horizon, from months to hours, and integrates storage allocation decision with other highly inter-dependent decision problems (terminal allocation, berth allocation, schedule design, yard crane assignment and scheduling). We remark that the proposed planning framework could support terminal operators for storage yard planning and operations at container transshipment terminals. It should be noted that this research focuses on the application of integrated planning approach to the storage yard allocation problem, and thus some other important decision problems of terminal operations like quay crane scheduling are not considered.

1.4 Thesis Organization

The thesis is organized as follows:

Chapter 1 introduces the general research background, research purpose and organization of the thesis.

Chapter 2 briefly reviews the existing literature on container terminal operations.

Chapter 3 studies the terminal allocation and yard allocation problem arising at a major transshipment hub with multiple terminals.

Chapter 4 studies the feeder vessel management problem with integrated design of berth template, schedule template and yard template at container transshipment terminals.

Chapter 5 presents a column generation based approach to the feeder vessel management problem.

Chapter 6 studies the daily storage yard manage problem with integrated consideration of space allocation and yard crane deployment as well as container traffic congestion.

Chapter 7 addresses the integrated problem for bay allocation and yard crane scheduling in transshipment container terminals.

Chapter 8 draws the concluding remarks and presents directions for future research.

Chapter 2

Literature Review

This chapter provides a literature review on two prevailing planning approaches applied for the optimization of container terminal operations: hierarchical planning approach and integrated planning approach. It reviews the existing literature in a general way while the following chapters on individual research topics also give a short literature overview of respective problems but in a much detailed manner.

2.1 Hierarchical planning approach

Traditionally, the decision problems associated with container terminal operations are solved hierarchically from those at the higher planning levels to others at the lower planning levels. The hierarchical planning approach decomposes the whole container terminal system into several subsystems with associated decision problems as shown in Figure 1.4 (berth allocation problem, quay crane scheduling problem, yard truck scheduling problem, yard crane scheduling problem and yard storage space allocation problem), and focuses on the optimization of individual decision problems. This approach is dominant in practice as well as the early stage of the literature.

At the high planning level, the berth allocation problem is solved firstly to manage the vessel traffic at the quayside by deciding the utilization of berth resource in time and space (Imai et al., 1997; Lim, 1998; Kim and Moon, 2003; Guan and Cheung, 2004; Cordeau et al., 2005; Monaco and Sammarra, 2007). The output of the berth allocation problem provides the arrival and departure times and positions of inbound and outbound containers for the subsequent decision problem, the storage space allocation problem. Yard storage space are to be allocated to containers for temporary storage with certain specific objectives (Kim and Kim, 1998; Kim et al., 2000; Zhang et al., 2003; Kim and Park, 2003a; Kim and Hong, 2006; Lee et al., 2006; Moccia and Astorino, 2007).

With the determined berth allocation and yard storage allocation plans, the decision problems belonging to the medium planning level are solved. These include quay crane assignment and scheduling problem, and yard crane deployment and scheduling problem. The quay crane assignment problem concerns how to allocate available quay cranes to currently berthing vessels, while the quay crane scheduling problem analyzes for a certain vessel the detailed job scheduling of the allocated quay cranes to conduct discharging and loading operations (Daganzo, 1989; Kim and Park, 2004; Moccia et al., 2006; Zhu and Lim, 2006; Lee et al., 2008; Chen et al., 2011). Similarly, yard cranes are to be deployed over the storage yard according to the workload distribution, and further to be scheduled in detailed to conduct the pickup and delivery jobs for individual containers (Kim and Kim, 1997; Narasimhan and Palekar, 2002; Zhang et al., 2002; Kim et al., 2003; Linn and Zhang, 2003; Lee et al., 2007; Cao et al., 2008).

At the low planning level, the yard truck scheduling problem is solved with known schedules of quay cranes and yard cranes. It is mainly to conduct the container movement between the quayside and the yardside with a focus of the synchronization between quay cranes and yard cranes (Kim and Bae, 1999, 2004; Bish et al., 2005; Nishimura et al., 2005; Ng et al., 2007; Cao et al., 2010; Lee et al., 2010b).

In research, much of the previous literature employs the hierarchical planning method. However, a vital weakness of the hierarchical approach is the failure to consider the interaction between highly interdependent decision problems. Taking the berth allocation problem as an example, aiming at minimizing vessel turnaround time berth allocation decisions are made with no consideration of the container storage layout in the yard. However, storage decisions may also affect the vessel turnaround time. The same concern exists for other inter-related decision problems.

2.1.1 Yard storage operations

Storage operation has received much interest from researchers over the decades and the storage allocation planning can be divided into two levels: macroscopic and microscopic.

Macroscopic level:

Macroscopic or aggregate level of storage space allocation problem aims to distribute import and export containers evenly among all blocks from an entire storage yard point of view. The concept of workload is used as a measurement of the amount of operation time needed for a block and the minimum unit for analysis is block.

In Kim and Kim (1998), the authors considered how to allocate storage space for import containers under the segregation strategy. Relationship between stack height and number of re-handles was analyzed in order to minimize the expected total number of re-handles for outside trucks to pick up containers in the yard. Cases were considered where the arrival rate of import containers is constant, cyclic and dynamic. For each case, the authors formulated the problem of allocating space for import containers and optimal solution was obtained based on the Lagrangian relaxation technique.

In Kim and Kim (2002), two cost models were proposed to decide the optimal amount of storage space and the optimal number of transfer cranes for handling import containers under different circumstances. The first model is from the point view of terminal operator where only space cost and transfer cranes cost were considered. The objective is to minimize the cost of terminal operator by finding the optimal combination of number of transfer cranes and stacking height. The second model introduced the cost of outside trucks trying to minimize the integrated total cost of terminal operator and customers. Numerical examples were provided to illustrate solution procedures as well as sensitivity analysis.

In Kim and Park (2003b), a mixed-integer linear programming model of pre-allocating storage space for arriving outbound containers was proposed in order to utilize space efficiently and to achieve maximum efficiency of load operations. Objective functions and constraints of both the direct and indirect transfer systems were described and formulated. Two heuristic algorithms were provided based on the duration-of-stay of containers and sub-gradient optimization technique respectively. Numerical examples showed that DOS (duration-of-stay) based decision rule results in almost the same level of objective values while it can save much computational time than the sub-gradient optimization method.

In Zhang et al. (2003), the authors studied the storage space allocation problem with a hierarchical approach in container terminals. The problem was decomposed into two levels and each level was formulated as a mathematical optimization model under a complex situation where inbound, outbound and transit containers are mixed in the storage blocks in the yard. The first level was to balance the workload among all the blocks such that the berthing time can be minimized. The second level was to minimize the total distance between storage blocks and vessel berthing locations by allocating containers corresponding to each vessel to the storage space determined in the first level. Rolling-horizon approach was employed to solve the 2-level problem. Numerical experiments showed that workload imbalance in the yard can be significantly reduced with short computational time.

In Lee et al. (2006), the authors focused on the storage space allocation problem in a transshipment hub under a given yard template. With sub-blocks assigned to certain departing vessels in advance, a workload balancing protocol was proposed trying to minimize reshuffling and traffic congestion. A mixed integer-programming model was formulated in sub-block level in order to improve the YC's handling efficiency. Two heuristics including sequential method and column generation method were developed and tested. Han et al. (2008) is an extension of the above paper where two strategies including consignment strategy and high-low balancing protocol were employed in space allocation problem. A Tabu search based heuristic algorithm was used to generate an initial yard template followed by an iterative improvement method to get a satisfactory solution.

In Moccia and Astorino (2007), the authors presented a new problem called *Group Allocation Problem* related with a direct transfer system and a transshipment terminal. A mathematical model was formulated considering all the costs of handling work occurring at discharging, loading and reallocation of container groups. A novel feature of this study is the container relocation from one yard block to another which is called reallocation. Reallocation is conducted when two connected vessels are berthed far way along the quay. The advantage of reallocation is that the discharging and loading operations at the quayside can be conducted efficiently as transshipment containers are moved to appropriate locations by reallocation. This study assumed the output of BAP is available in order to get the group data like the arrival and departure times and arrival and departure positions along the quay.

Microscopic level:

The microscopic level of space allocation focuses on how to allocate individual containers with in a bay in a way such that yard-crane cost during receiving and retrieving operations can be minimized. The cost is usually measured by number of re-handles.

In Kim (1997), the author proposed a methodology to estimate the expected number of re-handles to pick up an arbitrary container and the total number of re-handles to pick up all the containers in a bay for a given initial stacking configuration. The analysis of re-handles was restricted to a single bay and random picking up assumption. Exact evaluation of re-handles was derived by a dynamic programming model. Useful tables and equations were provided based on regression analysis to estimate the number of re-handles in a simple way.

In Kim et al. (2000), the authors formulated a dynamic programming model to determine the storage location of an arriving export container. The objective was to minimize the number of relocation movements that occur during the loading operations of a container ship. Container weight was considered in the model as the formulation was based upon the assumption that heavier containers are always loaded before lighter ones. The relocation movements occur when lighter containers are stacked on top of heavier ones in the yard. A decision tree was also proposed in order to reduce the computation time of the dynamic programming model.

In Kang et al. (2006), the authors presented a method for deriving a strategy for stacking containers with uncertain weight information to reduce the number of container re-handlings at the time of loading. Simulation experiments were conducted to evaluate different stacking strategies by calculating the lower bound on the expected number of re-handlings. Simulated annealing algorithm was employed to find a good stacking strategy. It was also found that the number of re-handlings can be further reduced by improving the accuracy of weight grouping through use of a learned classifier.

In Kim and Hong (2006), this study addressed the problem of relocating block in block-stacking warehouses. A branch-and-bound (B&B) algorithm and a heuristic rule based on probability theory were proposed. A procedure for estimating the expected additional number of relocation was suggested for various configurations of stacks. Results from numerical experiments indicated the relocations calculated by the proposed heuristic algorithm exceed that by B&B algorithm by 7.3% and 4.7% for different precedence structures while its computation time is much less than that of the latter one.

In Wan et al. (2009), an integer programming model was proposed for the problem of emptying a stack (bay) from any given configuration with the objective of minimizing the total number of reshuffles. By introducing arrival containers in retrieval process, this problem was extended to a dynamic problem. The authors also provided three index-based heuristics LS, RI and ENAR to solve the problems.

2.1.2 Berth allocation operations

The decision problem of berth allocation is to decide where (berthing position) and when (berthing time) to allocate berth resource to a container vessel. The main principle for berth allocation is to finish loading and discharging as fast as possible so that the turnaround time can be minimized. There are two approaches for dealing with berth allocation: discrete BAP and continuous BAP.

The discrete BAP divides the quay into several sections and each section can only berth one vessel at a time. Imai et al. (1997) studied the discrete BAP with the objective of minimizing total port staying time of ships and dissatisfaction of ships in terms of the berthing order. This study is static BAP as all the ships are assumed to be already arrived and waiting for berthing. Imai et al. (2001) extended it to a dynamic case where ships arrive in dynamically in the planning horizon. The objective of the problem is to minimize the sum of waiting and handling times for every ship. A Lagrangian relaxation based heuristic was developed to handle real world scale problems. The service priority was incorporated into the dynamic BAP in Imai et al. (2003). As the problem is NP-hard, the authors developed a GA based heuristic algorithm. Cordeau et al. (2005) developed another formulation for the dynamic BAP which is called MDVRPTW (multiple depot *Vehicle Routing Problem* with time windows) formulation. In this model, the ships are treated as customers and the berths as depots at which one vehicle is located. Two types of Tabu search heuristic were designed.

The continuous berth allocation allows vessels to moor at any position along the quay for higher utilization efficiency. Lim (1998) treated the berth planning problem as a restricted version of two-dimensional packing problem. The problem was showed to be NP-Complete and transformed into a graph theoretic model. Kim and Moon (2003) formulated the continuous BAP by a mixed integer program which was solved by a simulated annealing algorithm. Guan and Cheung (2004) studied the continuous BAP with an objective of total weighted flow time. A tree-search procedure and a composite heuristic were proposed for large size problems. Lee et al. (2010a) improved the identification procedure for possible locations in the two-dimensional diagram and designed two heuristics based on the *Greedy Randomized Adaptive Search Procedure* (GRASP) for the continuous BAP.

2.1.3 Yard crane operations

Cranes are commonly used in container terminals for container lifting. Generally, they can be categorized into two types: quay cranes that are deployed at the quayside for loading/discharging containers onto/from vessels, and yard cranes that are located in the storage yard for moving containers from and into yard blocks. Here, we only review studies on yard crane operations, as this research focuses on the storage yard management and has no direct relationship with quay crane operations. The crane operations can be considered at two planning levels: a tactical level which studies the movements of all yard cranes inside the whole storage yard with a mid-term planning horizon, and an operational level which focuses the detailed scheduling of one or two yard cranes within a small area (one or two yard blocks) with a short term planning horizon.

At the tactical level, the main objective is to minimize the forecasted workload delay of all yard blocks at each time period by scheduling and routing yard cranes over the storage yard. Due to the workload imbalance over the spatial dimension as well as the temporal dimension, yard cranes should move around the storage yard to carry out workload as much as possible. A classical study was presented by Zhang et al. (2002). They proposed an optimization model to allocate the rubber tired gantry cranes (a type of yard crane with a high mobility) among yard blocks according to workload distribution in the yard. The crane deployment problem was formulated as a mixed integer program and solved by a Lagrangian relaxation method. Linn and Zhang (2003) proposed a least cost heuristic for the problem to solve real-world size instances.

When the detailed workload is determined, the operational planning level of crane operations should be conducted to schedule the pick-up and delivery operations for each yard crane. Kim and Kim (1997) proposed an optimal routing algorithm for a single yard crane to do retrieval operations within a yard block according to container loading sequence. An integer programming model and a dynamic programming model were formulated for this problem in order to minimize the total handling time including set-up time within a yard-bay and travel time between yard-bays. The bay visiting sequence and the number of containers to be pick up at each bay were determined simultaneously. Narasimhan and Palekar (2002) theoretically investigated the yard crane routing problem and prove it to be NP-Complete. An exact branch-and-bound based algorithm and a heuristic algorithm were also developed. Ng (2005) extended the single yard crane scheduling problem to the situation with multiple yard cranes. Given a set of jobs with different ready times, the multiple yard cranes scheduling problem was solved considering the interference between adjacent cranes. This study showed the problem to be NP-Complete and developed a dynamic programming based heuristic.

2.2 Integrated planning approach

A promising trend on the research of container terminal operations is the integrated optimization method for highly interdependent decision problems. The integrated optimization approach can act as an overall decision making center and yield better efficiency. In other words, this approach provides a feedback from the downstream decision problem to the upstream one.

At the quayside, the integration of berth allocation and quay crane scheduling is one of the typical topics applying the integrated planning approach. The interdependency between the two decision problems lies in the vessel processing time which is determined by the quay crane assignment plan but affects the higher level decision problem on berth allocation. The integrated problem was first studied by Park and Kim (2003) and further improved by Liu et al. (2006); Imai et al. (2008) and Meisel and Bierwirth (2009).

At the yardside, the integration of yard truck scheduling and storage allocation problems provides an opportunity for speeding up the container movement between the quayside and the storage yard, as studied by Bish et al. (2001); Bish (2003); Bish et al. (2005); Han et al. (2008) and Lee et al. (2009). Similarly, the simultaneous optimization of yard crane deployment/scheduling and storage allocation is another typical integrated planning problem which further improves the efficiency of yard operations, as studied by Kim and Kim (2002) and Lee et al. (2006).

Another stream of the integrated planning approach is the integration of the quayside operations with the yardside ones. Giallombardo et al. (2010) studied the tactical berth allocation and quay crane assignment problem with considerations of the yard housekeeping costs generated by transshipment flows between vessels. Zhen et al. (2011) formulated an integrated model for the berth template and yard template design at a transshipment terminal.

By integrating independent decision problems associated with different terminal operations, researchers have made it clear that the integrated planning approach is able to yield further more improvement in terms of efficiency and cost than the hierarchical planning approach. One potential challenge is that the integration of interdependent decision problems makes it more intricate and also computationally more difficult to solve. Nevertheless, this approach has already yielded benefits and is expected to receive more research efforts in the near future.

Chapter 3

Terminal and Yard Allocation Problem for a Container Transshipment Hub with Multiple Terminals

3.1 Introduction

In container transshipment hubs, the management of transshipment flows is an important issue to which port operators pay close attention. Transshipment containers are temporarily stored in storage yards after being discharged from inbound vessels, and wait to be loaded onto outbound vessels in the near future. This transshipment movement generates container flows between quay side and yard side. As transshipment containers do not need to move out of the terminal gates, the related operations concentrate on storage yards and along the quay. Consequently, management for transshipment flows, including berth allocation, yard allocation and so on, is required to achieve a high productivity.

The Port of Singapore is one of the world's busiest transshipment hubs and handles one-fifth of the world's total transshipment throughput. Along with the increase of containerized maritime shipping, the Port of Singapore has set up five terminals phase by phase and another one is under construction in order to meet the increasing demand. It is often the case that a large transshipment hub consists of several terminals which are close to each other. Figure 3.1 presents such a multi-terminal transshipment system with three terminals in Singapore.

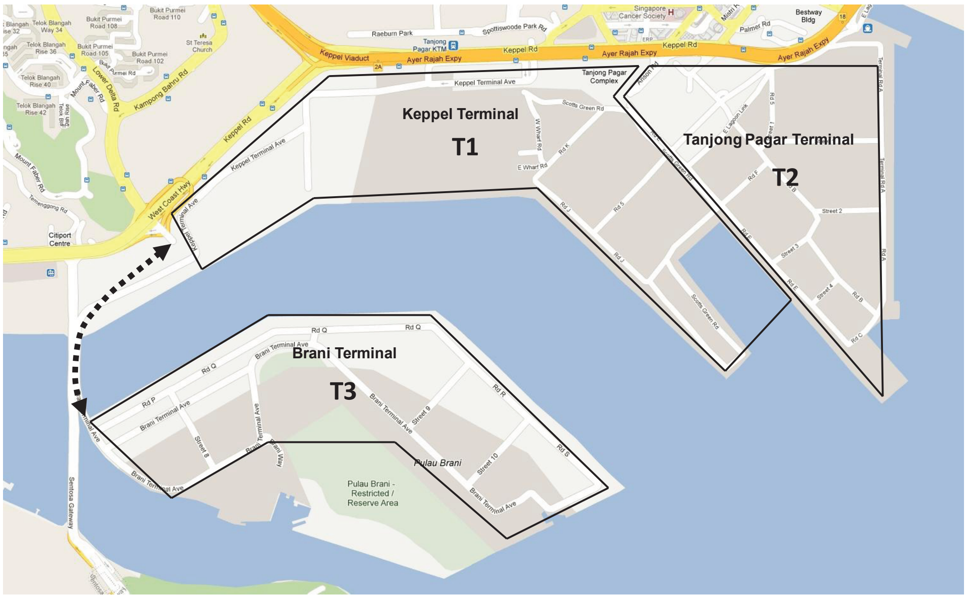


Figure 3.1: A multi-terminal system in Singapore

For such a multi-terminal system where many handling resources are involved, operations are complex and there are some unique issues calling for attention which are different from traditional ones in the management of a single terminal. One problem comes from inter-terminal traffic and it is what the port operators concern most. This is because inter-terminal traffic contributes to the whole operational cost to a large extent. In the case that two related vessels berth at two different terminals, for example, T1 and T3 in Figure 3.1, there exists an inter-terminal container movement operation which needs a lot of resources including yard cranes and yard trucks. When inter-terminal traffic volume becomes high, not only does cost increase, traffic congestion may also occur. Take Figure 3.1 as an example, there is only one traffic corridor indicated by the dotted arrows between T1 and T3 and high traffic could lead to high costs and traffic congestion. Fortunately, inter-terminal traffic could be reduced by assigning related vessels to the same terminal as long as enough berth capacity is available. Hence, terminal allocation for such a multi-terminal system deciding the visiting terminal for each vessel should be carefully planned in order to reduce inter-terminal traffic.

Another issue is to allocate yard storage space and to manage container transshipment flows within yards through their duration-of-stay. It is referred to as yard allocation problem in this chapter. Storage areas need to be allocated before containers are discharged from inbound vessels. Before loading operation, containers should be moved to a yard which is close to the berth position of the corresponding outbound vessel in order to speed up loading. Hence, a reallocation is needed to move containers between the two allocated yards, especially when the first assigned yard is far from the berthing position of the outbound vessel. Such container flows between quay side and yard side as well as between yards result in yard crane operation cost and yard truck transportation cost. In a transshipment hub where storage areas are scarce, the management of container flows plays an important role in reducing the operational costs. Yard allocation studied in this chapter concerns not only the assignment of storage resource for incoming containers but also the reallocation of yards to manage inter-terminal and intra-terminal container flows at different time periods. A reallocation conducted between yards inside a terminal and between terminals causes intra-terminal cost and inter-terminal cost, respectively. A good yard allocation plan generates low intra-terminal as well as inter-terminal costs.

As the above two problems, terminal allocation and yard allocation, could affect the operational costs significantly in a transshipment hub, we develop an integrated model for the terminal and yard allocation problem at a tactical level trying to minimize the handling cost of the transshipment flows. Our motivation in addressing this terminal and yard allocation problem from a tactical viewpoint is to help port operators improve the management of such a multi-terminal system and achieve competitive operational costs.

3.2 Literature Review

In open literature, plenty of research has studied berth allocation problem (BAP) and yard allocation problem (YAP). For BAP, the basic task is to assign berth resource to incoming vessels at certain time with specific objectives. BAP can be categorized into two types: discrete BAP and continuous BAP in terms of the management of berth resource. Imai et al. (2001) address the problem of dynamic berth allocation where berth resource is discretized. The objective of the problem is to minimize the sum of waiting and handling times for every ship. In Guan and Cheung (2004), the continuous BAP is studied with the objective of minimizing total weighted turnaround time. In the discrete case, a berth could only accommodate one vessel at a time and vessel size is not considered. However, vessels can berth at any position along the quay in the continuous case and vessel size is considered. A lot of other works extend their study and we refer readers to Bierwirth and Meisel (2010) for more information about BAP. Traditional BAP considers the situation at the operational level where the planning horizon is short and the exact calling schedule of incoming vessels is known to the port. For long term berth allocation, Giallombardo et al. (2010) develop a model which integrates berth allocation and quay crane assignment at a tactical level. By assigning berth and quay crane resources, the authors try to maximize the total value of chosen quay crane profiles (i.e. no. of quay cranes per working shift) and at the same time minimize the housekeeping costs generated by transshipment flows between vessels. Hendriks et al. (2012) study a multi-terminal container port and address the problem of spreading a set of cyclically calling vessel lines over different terminals and allocating a berthing and departure time to each vessel. The objective is to reduce the amount of inter-terminal container movement and to balance the quay crane workload over the terminals and over time. Our research resembles that of Hendriks et al. (2012) as we both consider the terminal allocation for a multi-terminal container port instead of assigning the exact berth locations within a container terminal. However, they include the consideration of quay crane workload while we consider the storage yard allocation for transshipment flows. For transshipment terminals with limited storage yards, yard allocation should be planned very carefully since the management of transshipment flows inside yards within a terminal and between terminals determines the operational costs to a large extent.

Storage yard allocation problem deals with determining the storage position in the yards and the amount of storage space to allocate for incoming containers. In Kim and Kim (2002), two cost models are presented to decide optimal amount of storage space and optimal number of transfer cranes for handling import containers under different circumstances. Kim and Park (2003b) develop a mixed integer linear programming model for pre-allocating storage space for arriving outbound containers in order to utilize space efficiently and to achieve maximum efficiency of loading operation. Zhang et al. (2003) study the storage space allocation problem with a hierarchical approach in a container terminal where import, export and transshipment containers are mixed in storage blocks. The problem is decomposed into two levels and each level is formulated as a mathematical optimization model. The first level is to balance the workload among all blocks in order to reduce berthing time. The second level is to minimize the total distance between storage blocks and vessel berthing locations by allocating containers to the storage space determined in the first level. The above literature either focuses on import and export containers or does not differentiate the types of containers. However, the different characteristics of transshipment flow make the above methods inapplicable for transshipment hubs. To the best of our knowledge, the literature about transshipment-related problems is very scarce. Moccia and Astorino (2007) present a problem called *Group Allocation Problem* considering the transshipment flow in the yards through the duration-of-stay period. A mathematical model is formulated with the objective of minimizing all the handling costs generated by discharging, loading and reallocation of container groups. However, their work applies to the single terminal operation (only intra-terminal transshipment flow cost is considered) and assumes that the berth allocation plan is given.

In this chapter, on one hand we extend the study by Moccia and Astorino (2007) to a multi-terminal circumstance. Our aim is to manage the transshipment container flow and to reduce inter-terminal and intra-terminal transportation costs. On the other hand, as terminal allocation largely affects the inter-terminal traffic, we also include the decision of terminal allocation. Hence, we study the terminal and yard allocation problem in a transshipment hub with multiple terminals from a tactical point of view. Compared with existing literature, the advantages of our study are:

- We study the container transshipment flow management problem in a multi-terminal transhipment hub to include the consideration of the inter-terminal container movement rather than only focus on the optimization of a single terminal.

- An integrated terminal allocation and yard allocation model is presented for a multi-terminal transshipment system so as to achieve a more effective management of container flows through a port.

The remainder of this chapter is organized as follows: problem description and mathematical model is presented in Section 3.3. Section 3.4 provides the heuristic approach, followed by numerical experiments in Section 3.5. At last, Section 3.6 draws the conclusion.

3.3 Mathematical Model

3.3.1 Problem description

This chapter is to study the tactical terminal and yard allocation problem (TYAP) for a container transshipment hub which consists of several terminals located close to each other. The terminal allocation problem is to allocate the visiting terminal for each calling vessel satisfying the calling schedules requested by shipping liners. The objective of the problem is to minimize the total inter-terminal transportation cost which is incurred by inappropriate terminal allocation. More specifically, if two vessels berth at different terminals and there is a group of containers exchanged between them, the containers should be moved between the two berthing terminals. This reallocation of storage yard results in inter-terminal transportation cost. Another problem is related to yard management which is to allocate and reallocate storage yards to containers. It is to decide in detail when and where to conduct inter-terminal and intra-terminal container reallocations. Figure 3.2 shows three cases of transshipment flows in a transshipment hub with three terminals. Case 1 is a transshipment flow without any reallocation as both the inbound and outbound vessels berth at Terminal 1. However, the transshipment flow in Case 3 requires an inter-terminal movement as the two connecting vessels are serviced at different terminals. The transshipment flow in Case 2 has an intra-terminal reallocation which is conducted to move the containers to a yard closer to the quayside for the sake of fast loading operation.

With a discrete planning horizon, we aim to, on one hand, assign a terminal to each vessel,

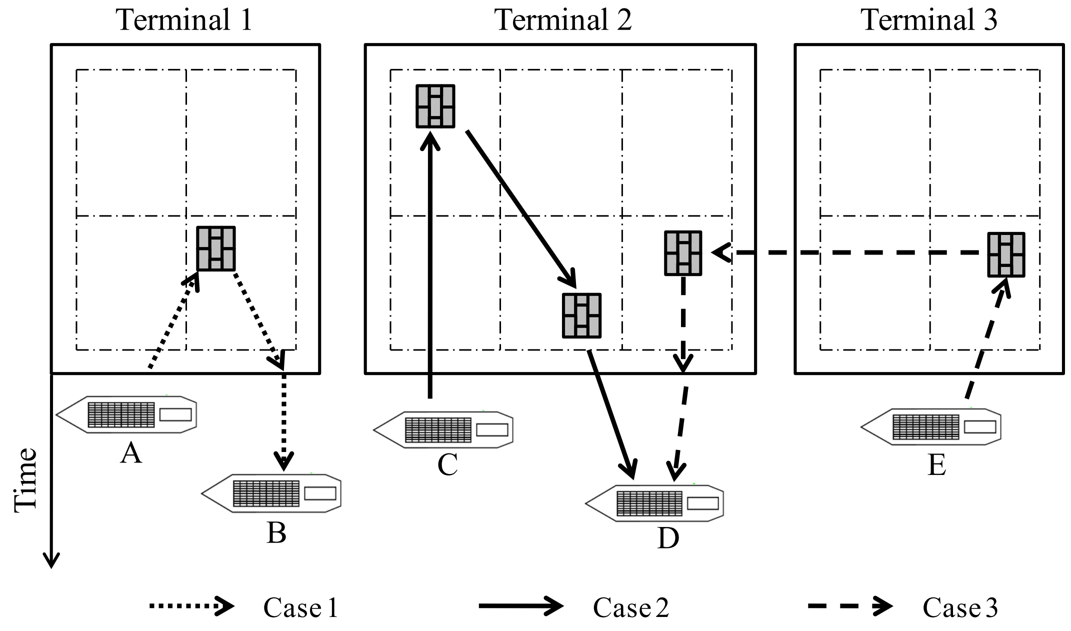


Figure 3.2: Three cases of transshipment flows in a transshipment hub with three terminals

on the other hand to determine the storage allocation plan for transshipment containers with the objective of minimizing the inter-terminal and intra-terminal transportation costs. Containers exchanged between two vessels are treated as a group. Hence, a group is a set of containers sharing the same inbound and outbound vessels as well as the schedule. In this chapter, we focus on container groups rather than individual containers. Before presenting the mathematical model, some assumptions are made as follows:

1. The calling schedule of all vessels is known to the terminal operator.
2. The exchanging container volume between vessels is assumed to be known.
3. The discharging and loading operation of one container group can be finished within one time period.

The information in Assumption 1 and 2 could be obtained from shipping liners and past data because the calling schedule is usually regular and the exchanging container volume is stable within a relatively long period. When the data varies, the TYAP should be updated.

The assumptions are used to get the data of the container groups, i.e. the arrival and departure times, the volume of groups. Assumption 3 is reasonable because a container vessel usually carries/receives multiple container groups and the service time of one container group is shorter than the turnaround time of the vessel. In case that the service of a vessel takes longer than one period, the arrival/departure time of the corresponding container groups can be assigned uniformly within the whole service time. For example, if a vessel requires two time periods for discharging and loading, we can assign half of the corresponding container groups to arrive/departure at the first time period, and the rest half to the second time period.

3.3.2 Model formulation

The TYAP can be considered as a network optimization problem with temporal and spatial dimensions. We define a graph $G_k(N, A)$ for container Group k as depicted in Figure 3.3. The source node and sink node are labeled S_k and T_k, respectively. Group k arrives in one terminal at time period a_k and leaves from one terminal at b_k. Through its duration-of-stay from a_k to b_k, there are $\overline{m}$ candidate storage yards to be considered for storing the group at each time period. A path from the source node to sink node corresponds to a terminal and yard allocation plan for the group. The dotted arrows between terminals and yards are arcs representing movements between quay side and yard side while those inside yards represent reallocation between successive time periods. The arc costs depend on the pair of linked nodes. A path indicated by the solid arrows in Figure 3.3 shows a feasible terminal and yard allocation plan for the group. As illustrated, Group k arrives at Terminal 2 and leaves from Terminal 1. Yard 2 is allocated to Group k after unloaded from its inbound vessel. Group k remains in Yard 2 until a reallocation to Yard 1 at time period b_k.

Indices:

i, j : the index for storage yards and terminals

k : the index for container groups

t : the index for time periods

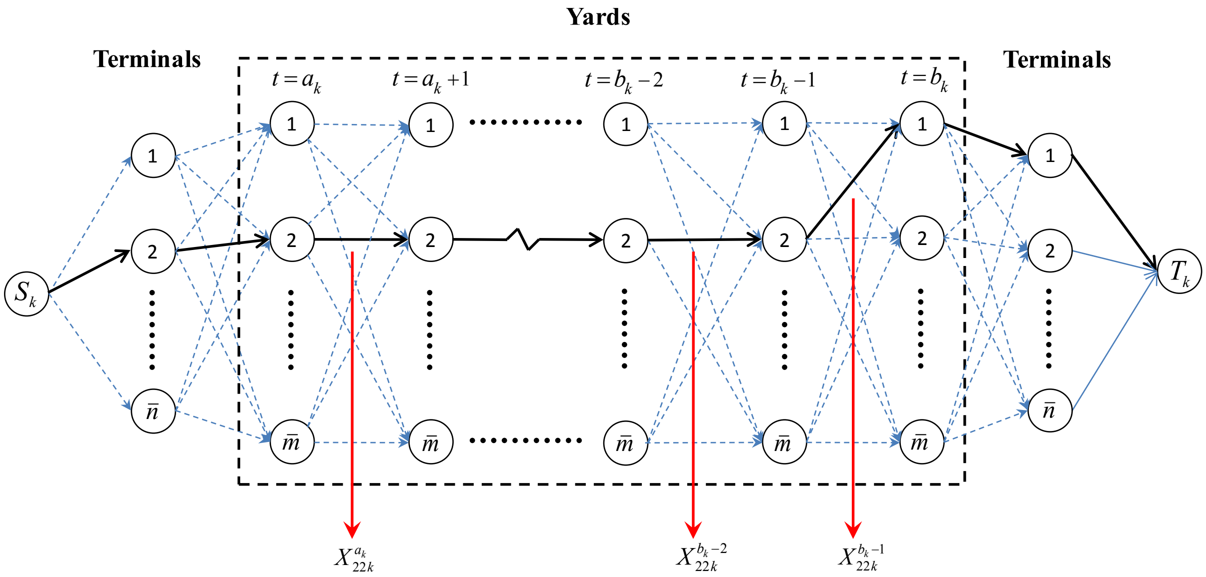


Figure 3.3: A graph representation of TYAP for Group k

Parameters:

M : set of storage yards, $M = \{1, 2, \cdots, \overline{m}\}$

N : set of terminals, $N = \{1, 2, \cdots \overline{n}\}$

V : set of vessels, $V = \{1, 2, \cdots, \overline{v}\}$

T : set of time periods

K : set of container groups

a_k : arrival time of Group k, $a_k \in T$

b_k : departure time of Group k, $b_k \in T$

q_k : storage space requirement of Group k

o_k : inbound vessel of Group k

d_k : outbound vessel of Group k

r_k : the maximum allowed number of reallocations between yards for Group k

Q_i^1 : the storage capacity of Yard i, $i \in M$

Q_i^2 : the processing capacity of Terminal i in a time period (i.e. the largest amount of containers that can be discharged or loaded at Terminal i), $i \in N$

α_{kl} : set to 1 if Group k and l have the same inbound vessel (i.e. $o_k = o_l$), and 0 otherwise, $k, l \in K$

β_{kl} : set to 1 if Group k and l have the same outbound vessel (i.e. $d_k = d_l$), and 0 otherwise, $k, l \in K$

γ_{kl} : set to 1 if the inbound vessel of Group k and the outbound vessel of Group l are the same (i.e. $o_k = d_l$), and 0 otherwise, $k, l \in K$

δ : the maximum allowed travel cost between quay side to yard side

c_{ij}^1 : travel cost between Yard i and Yard j, $i, j \in M$

c_{ij}^2 : travel cost between Terminal i and Yard j, $i \in N$, $j \in M$

The two parameters r_k and δ reflect the managerial practice about container movements. Larger r_k allows more flexibility for port operators to conduct container relocation from one yard to another, but generates more handling cost. δ is introduced to limit the distance between quay side and the container storage location as large distance would slow down the quayside operation. We remark that the cost parameters c_{ij}^1 and c_{ij}^2 actually include two parts: inter-terminal cost and intra-terminal cost. For c_{ij}^1, when Yard i and j are located within the same terminal the cost is the intra-terminal handling cost, while Yard i and j belong to different terminals the cost reflects the inter-terminal handling cost. It is similar for c_{ij}^2. Note that cost parameters can be estimated based on the geographical locations and port operators' experiences.

Decision variables:

X_{ijk}^t : set to 1 if Group k is located at Yard i at time period t and located at Yard j at time period $t+1$, and 0 otherwise, $i, j \in M$, $k \in K$, $t \in T$

U_{ijk}^t : set to 1 if Group k uses arc $i \rightarrow j$ at time period t upon arrival, and 0 otherwise, $i \in N, j \in M, k \in K, t \in T$

V_{ijk}^t : set to 1 if Group k uses arc $i \rightarrow j$ at time period t upon departure, and 0 otherwise, $i \in M, j \in N, k \in K, t \in T$

W_{ikt}^1 : set to 1 if Group k is located at Yard i at time period t, and 0 otherwise, $i \in M, k \in K, t \in T$

W_{ikt}^2 : set to 1 if Group k is processed (i.e. loaded or discharged) at Terminal i at time period t, and 0 otherwise, $i \in N, k \in K, t \in T$

Z_{ik}^1 : set to 1 if Group k uses Terminal i upon arrival, and 0 otherwise, $i \in N, k \in K$

Z_{ik}^2 : set to 1 if Group k uses Terminal i upon departure, and 0 otherwise, $i \in N, k \in K$

Objective function:

$$\min \left\{ \sum_{k \in K} \sum_{i \in M} \sum_{j \in M} \sum_{t \in T} c_{ij}^1 q_k X_{ijk}^t + \sum_{k \in K} \sum_{i \in N} \sum_{j \in M} \sum_{t \in T} c_{ij}^2 q_k \left(U_{ijk}^t + V_{jik}^t \right) \right\} \tag{3.1}$$

The objective function consists of two parts as indicated in (3.1). The first part reflects the inter-terminal and intra-terminal handling costs resulted from reallocation through the duration-of-stay in storage yards. The other part represents the transportation cost between quay side and yard side during discharging and loading operations.

Constraints:

$$\sum_{i \in N} Z_{ik}^1 = 1 \quad \forall k \in K \tag{3.2}$$

$$\sum_{i \in N} Z_{ik}^2 = 1 \quad \forall k \in K \tag{3.3}$$

$$\sum_{i \in M} X_{ijk}^{t} = \sum_{i \in M} X_{jik}^{t+1} \quad \forall k \in K, j \in M, a_k \leq t \leq b_k - 2 \tag{3.4}$$

$$Z_{ik}^{1} = \sum_{j \in M} U_{ijk}^{t} \quad \forall i \in N, k \in K, t = a_k \tag{3.5}$$

$$Z_{ik}^{2} = \sum_{j \in M} V_{jik}^{t} \quad \forall i \in N, k \in K, t = b_k \tag{3.6}$$

$$\sum_{i \in N} U_{ijk}^{t} = \sum_{i \in M} X_{jik}^{t} \quad \forall j \in M, k \in K, t = a_k \tag{3.7}$$

$$\sum_{j \in N} V_{ijk}^{t} = \sum_{j \in M} X_{jik}^{t-1} \quad \forall i \in M, k \in K, t = b_k \tag{3.8}$$

$$\sum_{i \in N} \sum_{j \in M} c_{ij}^{2} U_{ijk}^{t} \leq \delta \quad \forall k \in K, t = a_k \tag{3.9}$$

$$\sum_{i \in M} \sum_{j \in N} c_{ji}^{2} V_{ijk}^{t} \leq \delta \quad \forall k \in K, t = b_k \tag{3.10}$$

$$W_{ikt}^{1} = \sum_{j \in M} X_{ijk}^{t} \quad \forall i \in M, k \in K, a_k \leq t \leq b_k - 1 \tag{3.11}$$

$$W_{ikt}^{1} = \sum_{j \in N} V_{ijk}^{t} \quad \forall i \in M, k \in K, t = b_k \tag{3.12}$$

$$W_{ikt}^{2} = \sum_{j \in M} (U_{ijk}^{t} + V_{jik}^{t}) \quad \forall i \in N, k \in K, t \in T \tag{3.13}$$

$$\sum_{k \in K} q_k W_{ikt}^{1} \leq Q_i^{1} \quad \forall i \in M, t \in T \tag{3.14}$$

$$\sum_{k \in K} q_k W_{ikt}^2 \leq Q_i^2 \quad \forall i \in N, t \in T \tag{3.15}$$

$$\sum_{t \in T} \sum_{i \in M} \sum_{j \in M | j \neq i} X_{ijk}^t \leq r_k \quad \forall k \in K \tag{3.16}$$

$$\alpha_{kl}(Z_{ik}^1 - Z_{il}^1) = 0 \quad \forall i \in N, k \in K, l \in K, k \neq l \tag{3.17}$$

$$\beta_{kl}(Z_{ik}^2 - Z_{il}^2) = 0 \quad \forall i \in N, k \in K, l \in K, k \neq l \tag{3.18}$$

$$\gamma_{kl}(Z_{ik}^1 - Z_{il}^2) = 0 \quad \forall i \in N, k \in K, l \in K, k \neq l \tag{3.19}$$

$$X_{ijk}^t, U_{ijk}^t, V_{ijk}^t, Z_{ik}^1, Z_{ik}^2, W_{ikt}^1, W_{ikt}^2 \in \{0, 1\} \tag{3.20}$$

Constraints (2)-(8) are the flow conservation constraints. Constraints (2) show the outflow requirement at source node while Constraints (3) ensure the inflow at sink node. By the two constraints, each group has an inbound terminal and an outbound terminal and this indirectly assigns visiting terminals for calling vessels. Flow conservation inside yards is ensured by Constraints (4). The relationship between decision variable **Z** and **U**, **V** is indicated by Constraints (5) and (6) which link terminal allocation and loading, discharging operation decisions. Similarly, Constraints (7) and (8) deal with the relationship between decision variable **U**, **V** and **X**. Constraints (9) and (10) specify the travel cost requirement between quay side and yard side in order to ensure a fast loading and discharging operation. Constraints (11)-(13) define the decision variable **W** which represents container locations through duration-of-stay. At any time period, the total storage space requirement of all the groups in the same yard should not exceed the yard storage capacity, as ensured by Constraints (3.14). Similarly, terminal capacity is guaranteed by Constraints (3.15) as the total amount of loading and discharging containers should be within the processing capacity of the terminal. Constraints (3.16) guarantee the number of reallocations through duration-of-stay inside yards of each group respects the maximum allowed reallocation times. Terminal allocation constraints for groups of the same inbound/outbound vessels are indicated by (17)-(19) since such groups can only serviced at the same terminal. Finally, Constraints (3.20) specify the domain for decision variables.

3.4 Heuristic Approach

Consider the problem with a given terminal allocation plan, the TYAP is reduced to the *Group Allocation Problem* (Moccia and Astorino, 2007) which is proved to be NP-hard by the authors. Hence, the TYAP is also NP-hard and generally it is difficult and not efficient to solve the problem by a commercial solver especially for large scale problems. Hence, we propose a 2-level heuristic approach to find good solutions within a short computational time. The framework of the heuristic is introduced in Section 3.4.1 and the details are presented in the Section 3.4.2 and 3.4.3.

3.4.1 Framework of the heuristic

In this heuristic, the terminal and yard allocation problem is solved hierarchically as illustrated by the heuristic flowchart in Figure 3.4. The heuristic consists of two levels. Level 1 is designed to obtain a good terminal allocation plan for vessels. In this level, we solve the linear programming relaxation of the remaining yard allocation problem in order to evaluate the fitness of terminal allocation plans efficiently. Neighborhood search technique is employed to find better solutions in the searching process. At the end of Level 1, a good terminal allocation plan is obtained and passed onto Level 2. In Level 2, the terminal allocation plan is treated as input information and yard allocation for container groups is determined in detail. Such a sub-problem in Level 2 is actually equivalent to *Group Allocation Problem* (Moccia and Astorino, 2007). We develop a tabu search based heuristic method to find good yard allocation plans and obtain the total inter-terminal and intra-terminal handling costs.

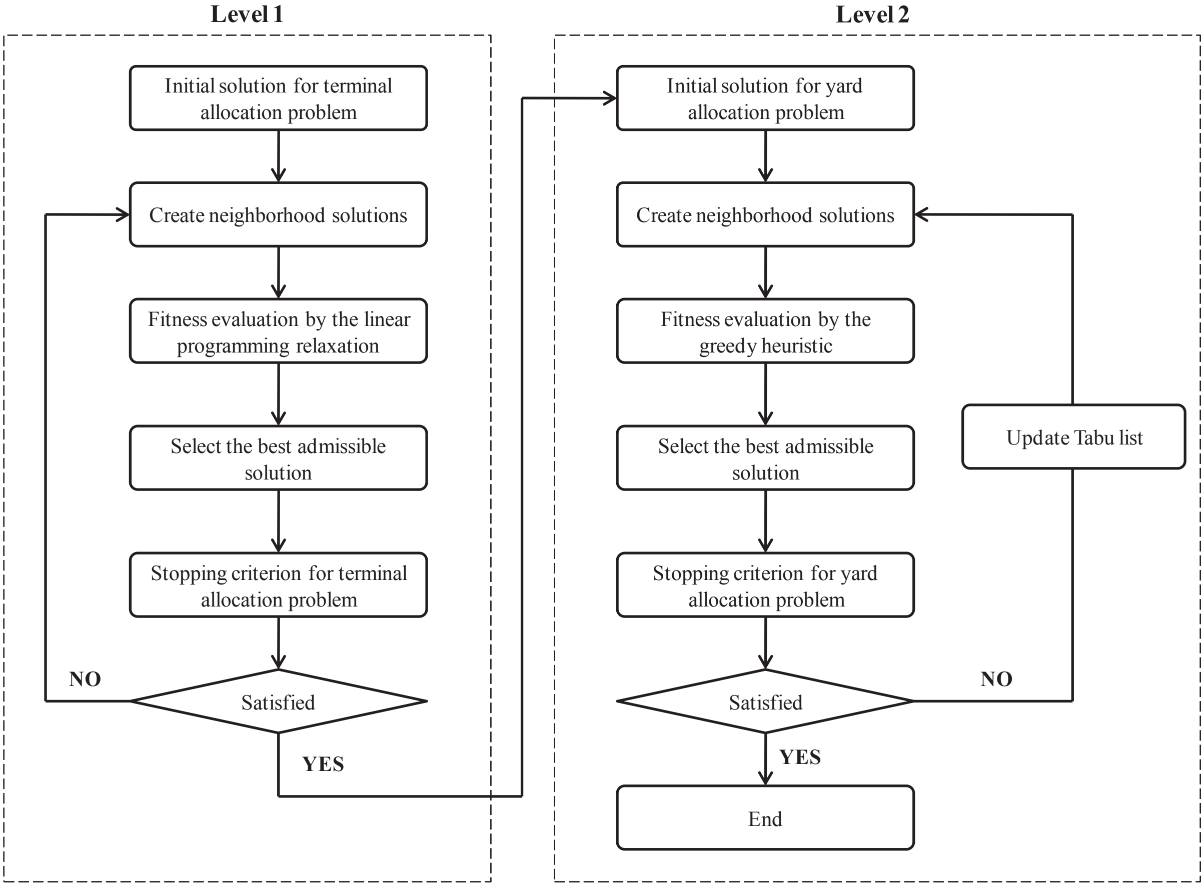

Level 1
Initial solution for terminal allocation problem
Create neighborhood solutions
Fitness evaluation by the linear programming relaxation
Select the best admissible solution
Stopping criterion for terminal allocation problem
NO
Satisfied
YES
Level 2
Initial solution for yard allocation problem
Create neighborhood solutions
Fitness evaluation by the greedy heuristic
Select the best admissible solution
Update Tabu list
Stopping criterion for yard allocation problem
NO
Satisfied
YES
End

Figure 3.4: The flowchart of the 2-level heuristic framework

3.4.2 Level 1

In Level 1, we focus on the terminal allocation problem and try to obtain a good terminal allocation plan. Firstly, an initial solution for terminal allocation problem is randomly generated. Then, neighborhood search is conducted in the neighborhood of the current solution. In each searching loop, NS_1 neighborhood solutions are generated. With a given terminal allocation plan, the related decision variables can be easily derived, i.e., $Z_{ik}^1, Z_{ik}^2, W_{ikt}^2$. To evaluate the fitness of the neighborhood solutions, the remaining yard allocation problem is solved approximately by relaxing the integer constraints of the yard allocation decision variables. At the end of each searching loop, the best admissible solution is selected to update the current solution. The stopping criterion is also checked to decide whether to continue the neighborhood search process or to end the first level. When the stopping criterion is met, the best solution in the whole searching process is selected as the terminal allocation plan and passed onto Level 2 as input information. The details are illustrated as follows:

3.4.2.1 Encoding representation

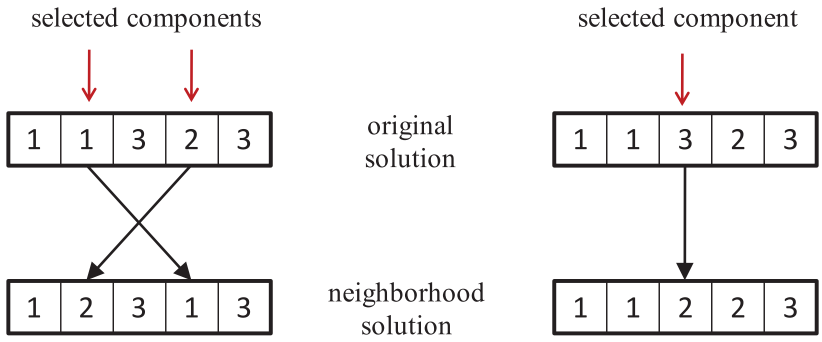
We apply a straightforward encoding representation for the terminal allocation problem. Let $S = (s_1, s_2, \cdots, s_i, \cdots, s_{\overline{v}})$ represents the terminal allocation decisions for $\overline{v}$ vessels where $s_i \in N$ indicates the calling terminal of Vessel i. For example, $S = (1, 1, 3, 2, 3)$ is a terminal allocation solution for 5 vessels. As indicated by the solution, Vessel 1 calls at Terminal 1, Vessel 4 is served at Terminal 2 and so on for the other 3 vessels.

3.4.2.2 Neighborhood structure

For the neighborhood search phase, two patterns of neighborhood structure as shown in Figure 3.5 are employed: pair-wise interchange and flipping patterns. In Pattern 1, two components of the solution are randomly selected and interchanged with each other. Pattern 2 only conducts operation on one component. The position of the component is randomly generated and the component s_i randomly flips to $s_j \in N\backslash\{s_i\}$. As illustrated by Figure 3.5, the terminal positions of Vessel 2 and 4 are interchanged with each other in Pattern 1. The terminal position of Vessel 3 is switched from 3 to 2 in Pattern 2. The two patterns of neighborhood search are conducted with an equal probability.

3.4.2.3 Fitness evaluation

With a candidate terminal allocation solution, the decision variables Z^1_{ik}, Z^2_{ik} and W^2_{ikt} can be easily derived. However, it is difficult to solve the remaining problem and obtain the exact total inter-terminal and intra-terminal costs since the yard allocation problem is still an NP-hard integer programming problem. In order to evaluate terminal allocation solutions, the linear programming relaxation technique is applied by replacing the integer constraint that the decision variables must be 0 or 1 by a weaker constraint that they belong to the interval $[0, 1]$. The relaxed linear program of the remaining yard allocation problem is as follows:



Pattern 1: pair-wise interchange

Pattern 2: flipping

Figure 3.5: Two patterns of neighborhood structure

$$\begin{aligned} \min \quad & (3.1) \\ s.t. \quad & (4) \text{ - } (16) \\ & X_{ijk}^t, U_{ijk}^t, V_{ijk}^t, W_{ikt}^1 \in [0, 1] \end{aligned}$$

It is worth noting that the linear programming relaxation has a physical meaning: container groups can be further divided into smaller sub-groups. Thus, a container group can be stored separately in different yards. Since the storage capacity of a yard or a terminal is relatively larger than the amount of storage space needed by a container group, the solution gap between the integer program and its relaxation is small. Besides, from the viewpoint of computational complexity it is much easier to solve a linear program than an integer program. That's why we apply linear programming relaxation to evaluate the fitness of terminal allocation solutions.

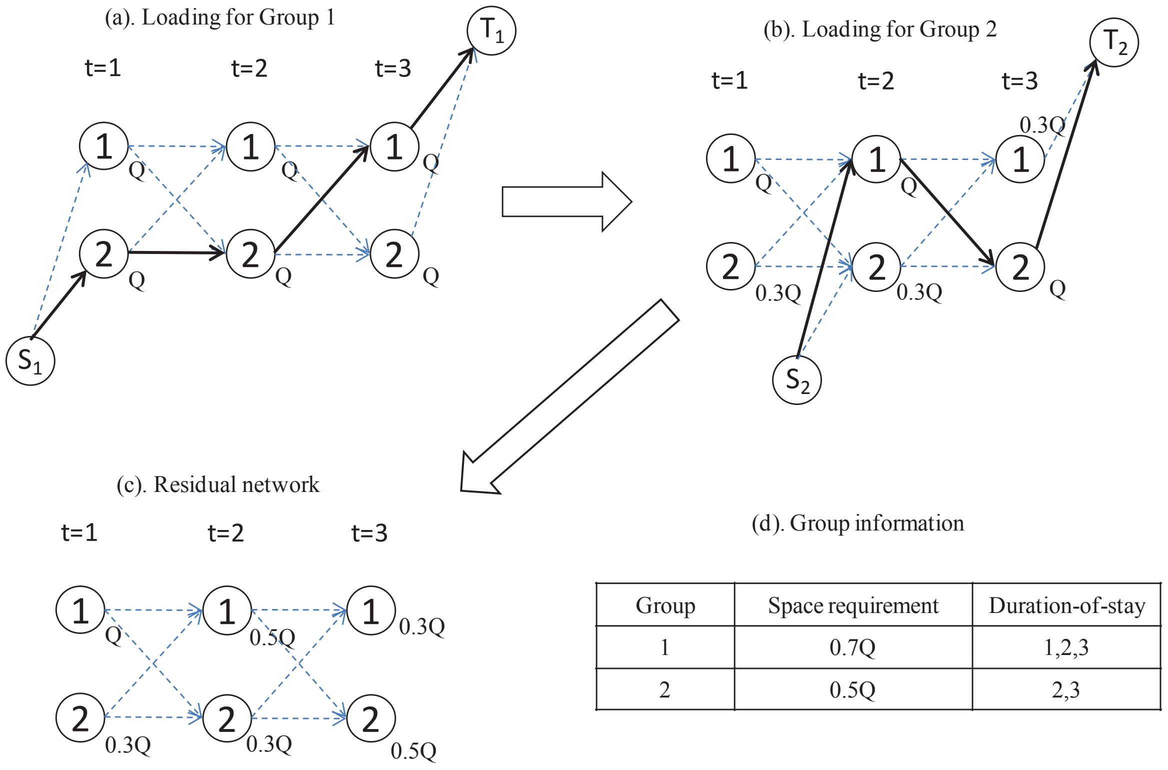
3.4.2.4 Stopping criterion

The neighborhood search for the terminal allocation problem in Level 1 terminates when the following condition is met: best solution does not change for SC_1 consecutive iterations.

3.4.3 Level 2

With a given terminal allocation plan for vessels, the arrival and departure positions of groups are known and the remaining problem is to determine the container flows in storage yards within their duration-of-stay. A tabu search based greedy heuristic is developed for Level 2.

The main idea of the heuristic is as follow: the storage area is considered as a two dimensional network with limited capacity, i.e. spatial and temporal dimensions. The groups are loaded onto the network sequentially and each group chooses its shortest path (i.e. storage plan with a least cost) in the network. The loading of each group corresponds to a loading stage. At each stage, the capacity of the network is updated after the loading of the corresponding group. With the updated network, the next group finds the shortest path available in the network. When all the groups are loaded onto the network, the complete yard allocation plan is determined and the objective function value can be easily obtained by adding the path costs of all the groups together.


(a). Loading for Group 1

(b). Loading for Group 2

(c). Residual network

(d). Group information

Group	Space requirement	Duration-of-stay
1	0.7Q	1,2,3
2	0.5Q	2,3

Figure 3.6: An illustrative example of the proposed heuristic for the problem in Level 2

Figure 3.6 presents an illustrative example with two yards, three time periods and two groups. For notational convenience, the storage capacity of all the yards is assumed to be the same denoted by Q. Group 1 is firstly loaded onto the network with full capacity and the path with least cost is $S_1 \rightarrow (2,1) \rightarrow (2,2) \rightarrow (1,3) \rightarrow T_1$ which is indicated by solid arrow in Figure 3.6(a). Then the capacity of the network can be updated and Group 2 is loaded onto the new network as depicted in Figure 3.6(b). Due to the capacity constraint, there is only one feasible path for Group 2 which is $S_2 \rightarrow (1,2) \rightarrow (2,3) \rightarrow T_2$. Finally, the residual network is shown in Figure 3.6(c) and the objective function value can be easily obtained by summing up the costs of the two paths. It should be noted that different loading sequences for groups result in different solutions and objective values.

The challenge is to determine a good sequence for loading groups onto the network by which near optimal solutions can be obtained. The tabu search technique is employed in this level to find a good loading sequence. A loading sequence determines the order of groups to be loaded onto the network. The groups loaded earlier have a network with larger capacity than latter ones. This implies earlier groups have a higher priority than the latter ones. In each neighborhood search loop, NS_2 neighborhood solutions are generated and evaluated. The details of the heuristic are introduced as follows:

3.4.3.1 Encoding representation

Let $P = (p_1, p_2, \cdots, p_{|K|})$ be the loading sequence for all the groups where p_i represents the group that are loaded onto the space-time network at the i^{th} order. For example, $P = (2,3,1)$ is an encoding representation for a case with three groups. Group 2 is firstly loaded onto the network with full capacity followed by Group 3. Group 1 is the last one to load onto the network.

3.4.3.2 Neighborhood structure

To generate neighborhood solutions, only the pair-wise interchange pattern in Figure 3.5 is applied.

3.4.3.3 Fitness evaluation

Given the sequence of groups for loading onto the network, the fitness of the solution could be evaluated by the algorithm in Table 3.1.

Table 3.1: The pseudo code for the fitness evaluation

```
$P = \left(p_1, p_2, \cdots, p_{|K|}\right)$ ←some candidate solution
Objective_value← 0        ►to record the handling cost
Network← $Q$        ►to record the network capacity
for $k = 1$ to $|K|$ do
    $PT$ ←Path($p_k$)        ► find the path with least cost for Group $p_k$
    Objective_value←Objective_value+Cost($PT$)
    Network←Network−Space($PT$)        ►update network capacity
end
return Objective_value
```

In the fitness evaluation algorithm, a very important step is to find the path with least handling cost for the current group given a network with limited capacity. To find such a path is similar to the classical shortest path problem. However, the unique features of this step are: the number of nodes between source node and sink node is fixed which is determined by the arrival and departure time periods; nodes have limited capacity and the route choice is restricted by the maximum number of reallocations allowed. Hence, in order to solve the specific step, we employ the dynamic programming algorithm presented as follows:

- Stage variable: time period $t \in T$
- State variable: node (i, t)
- Optimal value function: $L(n, i, t)$ represents the minimum cost from source node S_k to the current node (i, t) with n times of reallocation prior to the node
- Recursive function:

$$L(n, i, t+1) = \min \left\{ \begin{array}{l} L(n, i, t) \\ L(n-1, j, t) + Cost(j, i) \quad \forall j \in N \cup M, j \neq i \end{array} \right\} \tag{3.21}$$

$$\forall i \in N \cup M, t \in T, a_k \leq t < b_k, 0 \leq n \leq r_k$$

- Boundary condition: $L(n, o_k, a_k) = 0 \ \forall n = 0$ to r_k
- Optimal solution: $\min\{L(n, d_k, b_k) \ \forall n = 0 \text{ to } r_k\}$

The recursive function shows that: if no reallocation is conducted, the cost of node (i, t) is the same as that of node $(i, t-1)$ since no handling operation is conducted. However, when a reallocation is needed, the cost of the latter node is the summation of the cost of the previous node and the arc cost linking the two nodes. We remark that before calculating the cost for the current node the node the capacity constraint should be checked. Only if the capacity requirement holds, the cost can be calculated by the recursive function. Otherwise, the cost of the current node should be infinite since the storage plan is infeasible. It should be noted that we do not provide the details of how to record the least cost path for the sake of brevity. However, it is simple to accomplish the recording by introducing a variable to record the preceding node on the least cost path for each node. With the least cost path information, the network capacity could be easily updated. In case that no feasible path could be found for certain group, such a solution should be discarded.

3.4.3.4 Tabu list

In the tabu list, the positions of pair-wise interchange are recorded. First-in-first-out rule is applied to update the tabu list which means the oldest information would be removed out of the list to accommodate new one.

3.4.3.5 Stopping criterion

The heuristic with local search process in Level 2 will be terminated when the following condition is met: the best solution does not change for SC_2 consecutive iterations.

3.5 Numerical Experiment

In this section, we report the design and results of a comprehensive numerical experiment. The 2-level heuristic is coded in C++ and calls a commercial solver CPLEX 12.1 to solve the relaxed linear program in Level 1. The mathematical model presented in Section 3 is also solved by CPLEX 12.1 to obtain optimal results. A comparison between the results of the heuristic and CPLEX is presented. All the numerical tests are conducted on a PC with 3GHz CPU and 4GB

RAM. Some parameters for the heuristic are selected by trials and listed as follows:

- $NS_1 = 2log(\overline{v})$ iterations
- SC_1 : best solution does not change for 10 consecutive iterations
- NS_2=25 iterations
- SC_2: best solution does not change for 30 consecutive iterations
- Length of tabu list: 60.

3.5.1 Test instances

We use 8 hours as the length of a time period and set the storage capacity of a yard equal to 1 unit and berth capacity of a terminal to 10 unit for one time period. For experimental purpose, the maximum number of reallocations allowed r_k is set to 1 for all the groups as we intend to obtain good storage plans without too many reallocations. The maximum inflows and outflows of each vessel is set to 5. Then, the test instances are generated as follows:

Step 1: Generate the set of V for vessels whose calling schedules are uniformly distributed in the planning horizon. Go to Step 2.

Step 2: Container group set $K = \varnothing$. Go to Step 3.

Step 3: Generate a candidate container group k with three attributes: inbound vessel o_k, outbound vessel d_k and storage space requirement q_k. q_k is uniformly distributed in (0, 0.5). Go to Step 4.

Step 4: If k respects the maximum flow number requirement of each vessel, accept the candidate group and set $K \leftarrow K \cup \{k\}$, go to Step 3. Otherwise, discard the candidate group and go to Step 5.

Step 5: If 1×10^4 consecutive candidate groups are discarded, go to Step 6. Otherwise, go to Step 3.

Table 3.2: Parameters of the test instances

Instance	No. of Vessels	No. of Times	No. of Groups	Port Layout
I1	15	6	34	3×12
I2	15	6	32	3×12
I3	15	6	34	3×12
I4	20	6	49	3×12
I5	20	6	53	3×12
I6	20	6	57	3×12
I7	25	9	70	3×12
I8	25	9	58	3×12
I9	25	9	69	3×12
I10	30	21	89	4×20
I11	30	21	90	4×20
I12	30	21	92	4×20
I13	40	21	123	4×20
I14	40	21	127	4×20
I15	40	21	125	4×20

Step 6: End the instance generation process.

15 test instances are generated and 2 port layouts are selected: a smaller one with 3 terminals and 12 yards (each terminal has 4 yards) and a larger one with 4 terminals and 20 yards (each terminal has 5 yards). Table 3.2 shows the parameters of the 15 test instances. Instances I1-I9 have a shorter planning horizon with 6 or 9 time periods (2 or 3 days) and deal with less container groups. For instances I10-I15, the planning horizon is 21 (7 days) and more container groups are included.

3.5.2 Computational results

We compare the optimal results and near-optimal ones obtained from CPLEX and the heuristic, respectively. Due to the randomness in the heuristic searching process, the heuristic may return different results for each run. Hence, we run the heuristic 10 times and report the mean value and standard deviation of the results. For CPLEX, it is terminated when the computational time reaches 12 hours or it runs out of memory. Table 3.3 shows the computational results of CPLEX and the 2-level heuristic for all instances.

As can be seen from Table 3.3, CPLEX can only handle smaller instances from I1 to I9 and

Table 3.3: Computational results of CPLEX and the 2-level heuristic

Instance	CPLEX		2-level heuristic			Gap(%)
	Optimal (1)	CPU(sec)	Mean (2)	Stdev	CPU(sec)	$\frac{(2)-(1)}{(1)} \times 100$
I1	40.01	169	41.31	0.45	16.3	3.25
I2	50.34	209	51.66	0.54	18.1	2.61
I3	55.07	371	57.40	0.51	14.5	4.22
I4	78.61	5882	81.77	0.64	25.9	4.02
I5	85.53	4942	88.68	0.40	29.4	3.69
I6	87.42	1485	90.75	0.83	33.9	3.80
I7	94.43	18694	99.33	1.29	71.1	5.19
I8	96.07	28383	100.54	0.81	63.7	4.65
I9	98.37	30073	103.44	1.79	76.2	5.16
I10	-	-	157.28	2.99	1532	-
I11	-	-	153.30	2.07	1365	-
I12	-	-	155.25	3.31	1409	-
I13	-	-	213.60	1.25	4118	-
I14	-	-	220.25	1.83	3763	-
I15	-	-	234.30	2.75	4994	-

the computational time needed is highly sensitive to the instance scale. However, the 2-level heuristic is able to find high quality solutions within a much shorter computational time. The average gap between the solutions and optimal ones for instance I1-I9 is less than 5% which is acceptable from our point of view. For the larger instances I10-I15, CPLEX cannot even find feasible solutions before it terminates due to out-of-memory while the 2-level heuristic can still handle.

3.5.3 Optimization improvement

In order to assess the effectiveness of the terminal and yard allocation problem, we compare the result of the 2-level heuristic and that of a simple planning method. This method reflects a simple way to allocate terminals and yards without any optimization consideration: the visiting terminals for vessels are randomly assigned. Upon arrival, each container groups is stored in the yard which is closest to the arrival position of the inbound vessel. Similarly, upon departure each container group is moved to the yard which is closest to the departure position of the outbound vessel. The total inter-terminal and intra-terminal costs of the two planning methods are compared in Table 3.4.

Table 3.4: Comparison of the optimization model and a simple planning method

Instance	Heuristic (1)	Without optimization (2)	Ratio (1)/(2)
I1	41.31	52.13	0.79
I2	51.66	64.51	0.80
I3	57.40	71.77	0.80
I4	81.77	97.46	0.84
I5	88.68	105.56	0.84
I6	90.75	107.18	0.85
I7	99.33	119.83	0.83
I8	100.54	121.21	0.83
I9	103.44	129.66	0.80
I10	157.28	200.62	0.78
I11	153.30	200.22	0.77
I12	155.25	201.68	0.77
I13	213.60	269.53	0.79
I14	220.25	284.96	0.77
I15	234.30	299.94	0.78
		Average	0.80

As can be seen, the TYAP proposed in this chapter has a significant improvement (about 20%) over the simple planning method without any optimization consideration. This demonstrates the effectiveness of our optimization model. For the larger instances I10-I15, the improvements are even larger than those of smaller instances which implies the 2-level heuristic also performs well for larger instances.

3.5.4 Sensitivity analysis

We analyze the performance of the heuristic in Level 2 with different maximum allowed reallocation times r_k as the fitness evaluation step in Level 2 is dependent on r_k. Figure 3.7 reports the results and computational times of instances I10-I15 with r_k varying between 1 and 5. For each instance given a certain r_k, firstly ten terminal allocation plans are randomly generated. Then the heuristic in Level 2 is applied to the generated terminal allocation plans. We report the mean value of the objective function values and computational times. The results show a linear relationship between the computational time and the value of r_k. This observation is in line with the computational complexity of the heuristic in Level 2 $O(r_k|M||T||K|)$. Another observation is that increased r_k does not provide significant improvement of the results. Theoretically, the objective value with a larger r_k should be no greater than the one with a smaller r_k. However, the results of the experiment do not conform to this due to the randomness of the heuristic. Overall, $r_k = 1$ is a good compromise between computational effort and solution quality.

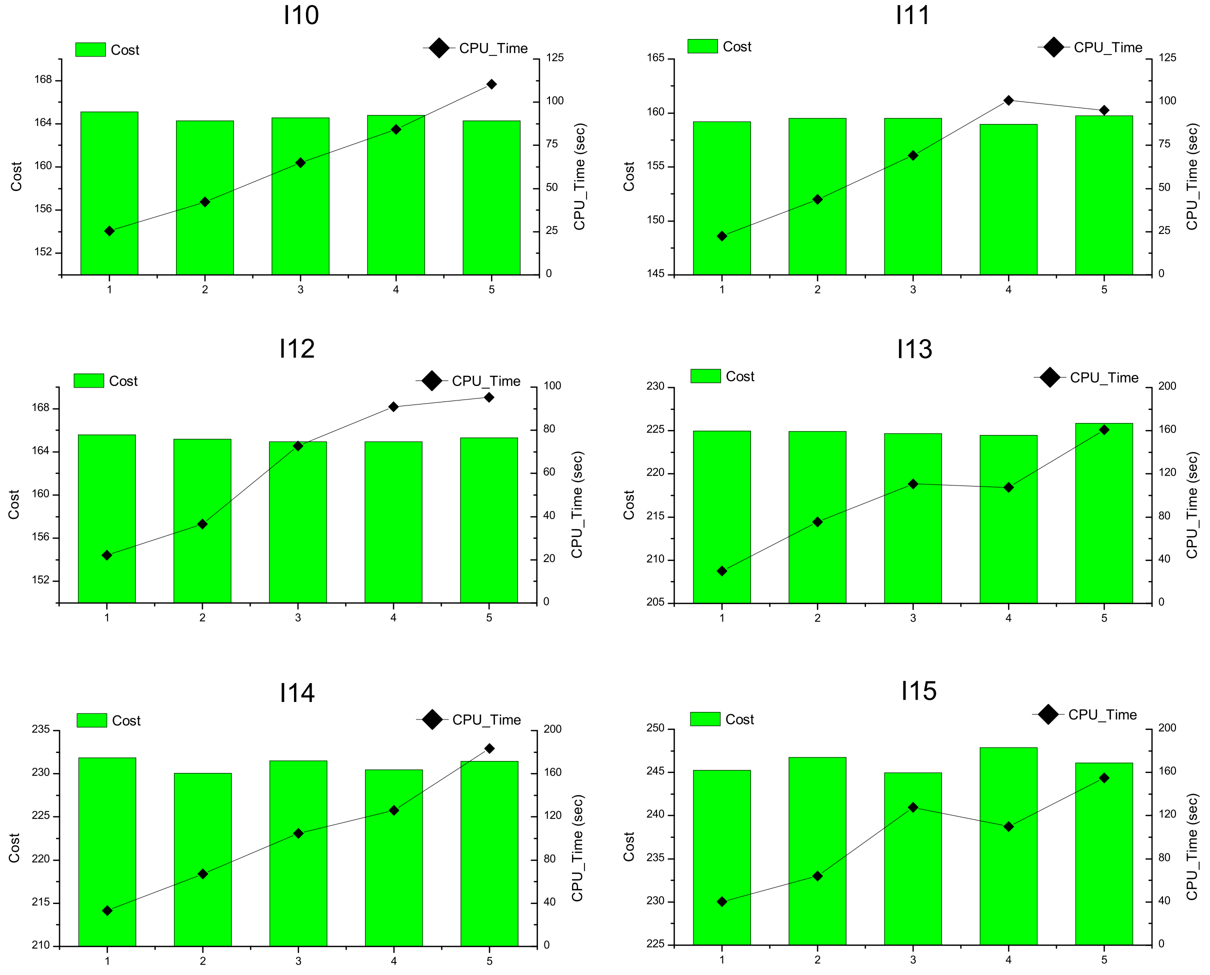


Figure 3.7: Sensitivity analysis of the maximum allowed reallocation times r_k

3.6 Summary

To conclude, the contributions of the chapter to the literature are as follows. First, this chapter has extended research on container port operations from a single terminal to a multi-terminal transshipment hub. We have studied from a tactical point of view two practical problems: terminal allocation problem for vessels and yard allocation problem for transshipment container movements. An integer programming model is developed that integrates the two decision problems and aims to minimize total inter-terminal and intra-terminal handling costs generated by transshipment flows. Our study addresses this terminal and yard allocation problem and provides decision support to port operators for the management of transshipment flows in a multi-terminal transshipment hub. Additionally, a 2-level heuristic approach is designed to tackle the integrated problem in an efficient way. Numerical experiments show that the integrated problem can gain significant improvement over a simple management method without any optimization consideration.

Chapter 4

Feeder Vessel Management at Container Transshipment Terminals

4.1 Introduction

Maritime freight transport has been supporting the international trade and globalization with over 80 percent of the world mechanized trade being carried by sea (UNCTAD 2008). Since the introduction of containerization and mega-vessels (*Emma Maersk* with a capacity of over 15,000 TEUs), the liner shipping companies have been benefiting from the economies of scale. Meanwhile, the *hub-and-spoke* system began to be implemented in which large container vessels (mother vessels) visit a limited number of transshipment terminals (hubs) while small vessels (feeders) connect the hubs with other ports (spokes). In this system transshipment hubs are the key centers where efficient management of the logistic activities should be well ensured.

As a key node of the maritime shipping network, a container transshipment hub is a complex system involving various operations: berth allocation for incoming vessels, crane assignment/scheduling for container loading and discharging, yard space allocation for the transshipment containers exchanged between mother vessels and feeders, and truck scheduling for moving containers between quayside and the storage yard. In order to remain competitive, port operators, especially those operating large transshipment hubs, always seek to improve the services by ensuring a smooth berthing process and dedicating storage areas for shipping companies. This chapter focuses on the management of feeder vessels at transshipment terminals with the aim of improving the efficiency of exchanging transshipment containers with mother vessels.

One particular motivation of the feeder management problem is to set up a tactical schedule template specifying the allocated service time for feeders. For a large transshipment terminal with heavy vessel traffic, pre-berth waiting for vessels often occurs due to the workload congestion at the quayside. Reducing the workload congestion is the key purpose of designing the feeder schedule template. Vacca et al. (2007) mentioned that the efficiency of a transshipment hub can be improved by taking into account the peculiarities of transshipment flows when the arrival time of feeders are not known in advance but can be decided by the terminal. This idea adjusts the calling schedule of feeders in such a way that the temporal distribution of quayside workload (loading and discharging) varies as little as possible. However, the general practice is that container ports accept the visiting requests of feeders passively and only optimize the berth allocation at the operational level. Such a practice often results in temporal imbalance of workload and thus congestion occurs at the periods with heavy workload. Designing a good schedule template aims to balance the temporal distribution of quayside workload and such a kind of tactical level decision could support port operators in negotiating with shipping companies on their vessel arrival time.

Another important focus of the feeder management problem is to design a berth template (i.e., favorite berth positions) for feeders and yard template (i.e., assigned yard storage locations) for the transshipment flows. In a transshipment terminal, container batches are exchanged between mother vessels and feeders. Such transshipment flows are firstly discharged from the berths of their inbound vessels, placed in certain yard areas temporarily and finally loaded to their corresponding outbound vessels. The berth template determines the arrival and departure berth positions of the transshipment flows, while the yard template directs the flows in and out of the storage yard. Thus, the two decisions affect the container moving distance through the terminal. Long moving distance not only results in higher travel cost of yard trucks but also causes an unfavorable circumstance to the desired swift loading and discharging operations at the quayside. Thus, it is essential for the port operators to design berth and yard templates in an integrated manner with the consideration of the spatial moving costs of all transshipment flows.

This chapter aims to integrate the above three tactical decisions related with feeder vessels at a transshipment terminal: 1) designing feeder schedule template, 2) designing feeder berth template, and 3) planning storage yard template for transshipment flows. We intend to reduce the quayside workload congestion by proactively adjusting feeders' calling schedule from the perspective of container terminals, and to optimize the container movements through the terminal by assigning berth positions and storage yard space. The motivation of this study lies in the following two aspects:

- To adopt a proactive management strategy by designing the calling schedule of the feeders from the perspective of container terminals;
- To integrate the quayside operations and yardside operations at a transshipment terminal by simultaneously designing the berth, schedule and yard templates.

This chapter is organized as follows. Section 4.2 reviews relevant papers in the literature and Section 4.3 presents the detailed problem description as well as mathematical formulations. A memetic heuristic is developed in Section 4.4 followed by computational experiments in Section 4.5. Finally, Section 4.6 draws the conclusion.

4.2 Literature Review

The management of container terminal operations involves many interesting optimization problems and has attracted plenty of research efforts. In this section we review some recent studies which are most relevant to the feeder vessel management problem studied in this chapter. For a comprehensive review, readers may refer to Steenken et al. (2004), Stahlbock and Voss (2008) and Monaco et al. (2009).

The *Berth Allocation Problem* (BAP) is a well studied decision problem arising at the quayside of container terminals. The basic task is to assign berth resource to incoming vessels at certain time with specific objectives. The operational BAP aims at minimizing the vessels' turnaround time with a relatively short planning horizon (hours to days), e.g., Lim (1998); Imai et al. (2001); Kim and Moon (2003); Guan and Cheung (2004) and Cordeau et al. (2005). Particular attention is paid to the minimization of pre-berth waiting time and turnaround time for vessels given the available information of vessel arrival time. Moorthy and Teo (2006) first studied the tactical BAP named as berth template problem. Covering a longer planning horizon (weeks to months), the berth template problem assigns home berth locations to cyclically visiting vessels. The output is subsequently used as a key input for the operational BAP which re-plans the berth allocation due to the deviation of vessel arrival time. Another purpose of the berth template is to plan for the yardside operations, e.g., determining storage locations of transshipment containers within the terminal. Giallombardo et al. (2010) studied the tactical BAP with considerations of quay crane assignment and the housekeeping costs generated by transshipment flows between vessels. Similar to the concept of berth template, we motivate to further optimize the quayside traffic by controlling the vessel arrival time, i.e., schedule template, in addition to the berth template. We remark the schedule template design is a new tactical planning problem that allows terminal operators to control the quayside congestion in order to balance the temporal distribution of workload. Such a schedule template design problem resembles the berth template problem as both of them manage the vessel traffic at the quayside. However, the former focuses on the temporal control while the latter deals with spatial management.

The yard storage space allocation is a typical decision problem dealing with the container storage operations at the yardside. It is a common practice that the storage policies are container type dependent and the yard is partitioned into different areas according to their operational needs (Ng et al., 2010). Kim and Kim (1999a) considered how to allocate storage space for import containers under the segregation strategy. Three arrival patterns were considered: constant, cyclic and dynamic arrival rate. Ng et al. (2010) studied the export yard template design problem for vessel services with a cyclical calling pattern. Given the daily arrival information of the export containers associated with each vessel service, the authors tried to determine the storage locations for export container clusters. Zhang et al. (2003) applied a hierarchical approach to the storage allocation problem where import, export and transshipment containers are mixed together. Regarding the management of transshipment containers, Moccia and Astorino (2007) proposed a *Group Allocation Problem* considering the container movement between quayside and storage yard as well as container relocation within the storage yard. One of the major concerns for the yard storage space allocation is the minimization of spatial workload imbalance as congestion would occur due to large workload imbalance. For the management of transshipment containers, there is a particular objective which is to minimize the container moving distance between the quayside and yardside with the available information of the arrival and departure quay position and time of the transshipment containers. Shipping companies usually ask container terminals to reserve dedicated yard storage areas and this tactical decision problem is referred to as yard template design problem in the literature. Cordeau et al. (2007) studied this problem of a European container terminal adopting *Direct Transfer System* in which the container movement cost between the quayside and yardside is represented by the distance between the arrival and departure berth positions. However, for most Asian ports with *Indirect Transfer System* the movement of transshipment container should be modeled in detail by considering the exact distance between the quayside and the yard position where the containers are stored. Besides, dedicated storage policy is not efficient in terms of space utilization in that the reserved storage spaces are occupied only in the duration-of-stay period. In order to improve the efficiency of the yard utilization, it is essential to explore shared storage policy for the yard template design as studied in this chapter.

A recent research trend is to integrate decision problems that are highly dependent yet usually solved hierarchically by terminal's planners (Vacca et al., 2010). One of the streams is the integration of the quayside operations and the yardside ones. Zhen et al. (2011) formulated an integrated model for the berth template and yard template design at a transshipment terminal. For solution approach, the authors solved the two design problems independently but applied an iterative process of local refinement to capture the interdependency between them. Centered on the management of a large transshipment hub with multiple terminals, Lee et al. (2012) integrated the terminal allocation problem (assigning home terminal for vessels) and yard storage allocation problem (determining the transshipment flows between terminals as well as within terminals). The integration of quayside operations and yardside ones provides an opportunity of further improving the overall efficiency of container terminals. In this chapter, we follow the integrated optimization trend and extend the integration of the berth template and yard template problem by including the schedule template design for feeder vessels.

4.3 Mathematical Model

In this section we first provide a detailed description of the feeder vessel management problem. Then, a mixed integer quadratic program is developed and further linearized.

4.3.1 Problem description

The studied problem is to integrate three tactical decision problems at a container transshipment terminal: berth template, schedule template and yard template design. It concerns the management of cyclically visiting feeders and the transshipment flows between mother vessels and feeders. In this chapter, we intend to tackle this integrated problem from two aspects:

(a) Spatial planning:

Spatial planning is to assign berthing positions for feeders (i.e., berth template) and to determine the storage locations for transshipment flows (i.e., storage template) with an objective of reducing the total distance of the transshipment flows from the quayside to the storage yard and vice versa. Transshipment flows are exchanged between mother vessels and feeders through the storage yard in the duration-of-stay. Determined by the berth template and storage template, the container moving distance has an impact on the operational efficiency since large distance is not in favor of fast loading and discharging. Figure 4.1 is a schematic representation of a container terminal with 16 yard sections and transshipment flows exchanged between 2 mother vessels and 3 feeders. A transshipment flow is firstly discharged from its inbound vessel and then moved to a yard section for temporary storage. Finally, when the corresponding outbound vessel arrives, it is moved back to the quayside by yard trucks for loading. The berthing positions of feeders and the storage positions of transshipment flows should be determined in such a way that the total distance of container movements is minimized. Note that in this study we focus on deciding the yard sections for container storage, and simplify the detailed considerations of container stacking and re-handles within a yard section. Besides, we consider the movements of container batches instead of individual containers. This is because the feeder vessel management problem is at the tactical planning level, and the detailed stacking and re-handling decisions for individual containers should be determined at the operational level.

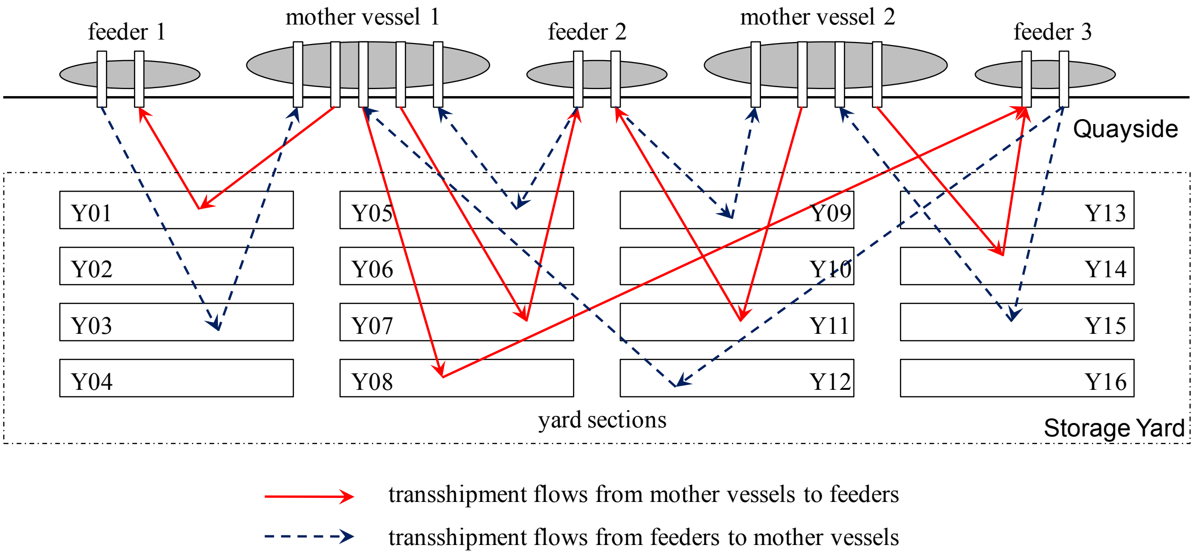


Figure 4.1: Transshipment flows between mother vessels and feeders.

(b) Temporal planning:

Temporal planning is to assign a service time slot to each container vessel (i.e., schedule template) in a proactive way from the terminal's perspective, unlike the convention that shipping liners establish the port staying time window and terminal operators have to provide services reactively according to the time window. In this chapter, we assume that the service requests from mother vessels are always satisfied while the terminal operator has the authority in deciding the service time of feeders with satisfying the preferred visiting time windows from feeder operators. We remark that this assumption is reasonable when there is an alliance between the terminal and shipping liners. With assigned service time by terminal operators, feeders follow the schedule and maintain a weekly arrival pattern. Such a proactive operational strategy provides terminal operators an opportunity of reducing the quayside workload (container loading and discharging) congestion by adjusting feeder calling schedules.

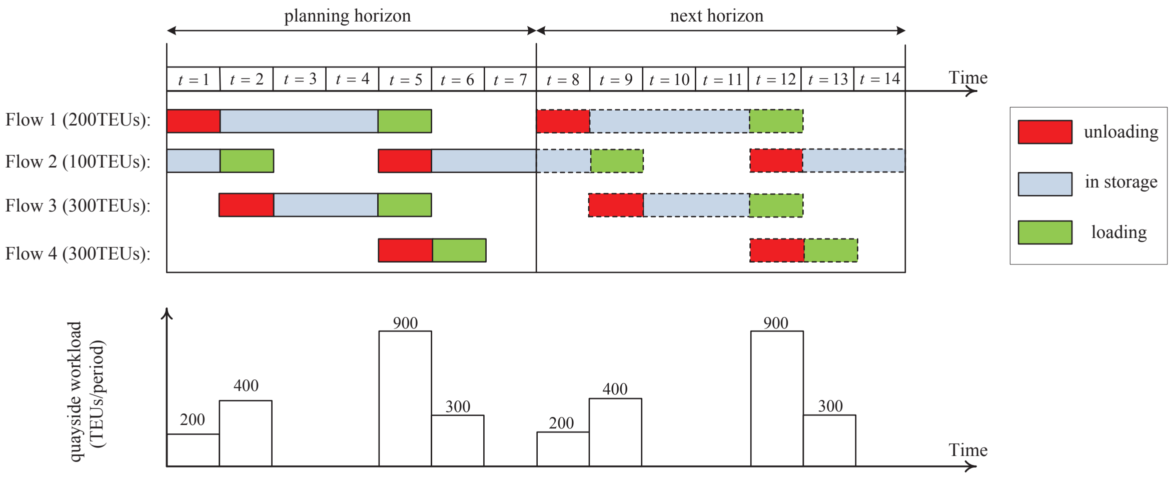


Figure 4.2: An illustrative example of workload distribution.

The calling schedule of container vessels determines the arrival and departure time of transshipment flows, as well as the temporal distribution of workload for handling equipment. Figure 4.2 shows an illustrative example of the workload distribution over two planning horizons each of which has seven time periods. There are four transshipment flows with statuses of unloading, in storage and loading. During loading and unloading time periods handling equipment (e.g., quay cranes) has to be allocated to conduct the operations, while allocated storage spaces are occupied when flows are in storage and during loading and unloading. In this example, Flow 1 arrives at the first time period and is loaded onto its outbound vessels at time period 5. The allocated storage space should be reserved from period 1 to 5. Regarding Flow 2, as its outbound vessel arrives before its inbound vessel in the current planning horizon, the containers stay in the terminal until the arrival of its outbound vessel in the next planning horizon. As can be seen from Figure 4.2, the transshipment flows lead to an imbalanced temporal distribution of workload. From the operational point of view, an evenly distributed workload circumstance is preferred in that large workload imbalance would make handling equipment sometimes busy and sometimes idle. As the schedule of unloading and loading operations is determined by the vessel calling time, the workload imbalance can be reduced by adjusting the calling schedule of vessels.

4.3.2 A mixed integer quadratic program

It is a common practice that container shipping services are maintained in a weekly manner, i.e., container vessels visit the terminals each week at the same time. The weekly container service convention has become the industry standard for liner service (Slack et al., 2002). Take Maersk Line as an example, all of its 33 Asia-Europe and Intra-Asia liner services and 22 out of 23 Intra-Asia feeder services follow the weekly service convention (Maersk, 2012). In the research area, most of the recent efforts on liner shipping generally adopt the weekly service frequency in light of the practice, e.g., Dong and Song (2009) and Wang and Meng (2012). As a consequence, container terminal operators need to allocate resources including berth and yard storage space on a weekly basis. We remark that the weekly service convention facilitates regular services and thus easy management for both liner shipping companies and container terminals.

The cyclically planning horizon of this study is set to one week which is further discretized into a series of working shifts. Given a set of mother vessels and feeders and a storage yard divided into yard sections, the feeder vessel management problem aims (i) to assign a working shift for feeders' calling, (ii) to determine the berthing position for feeders, (iii) and to decide the yard sections for the storage of transshipment container. The above three decisions are considered simultaneously in order to achieve an overall better efficiency, and are referred to as schedule template, berth template and storage template design sub-problems, respectively. One underlying assumption is that the service time of feeder vessels is within one working shift. We remark that it is in line with the practice that the port-of-stay time of most feeders is less than one working shift. Since the feeder vessel management is a tactical planning problem, we are only interested in assigning a rough working shift for vessels instead of accurate time windows. The output could serve as a template for operational planning which takes the responsibility of determining detailed service time windows for feeders. With this assumption, the yard storage space can be reserved on the basis of working shifts for transshipment flows. The objective is to balance the container loading and discharging workload at the quayside over the planning horizon as well as to minimize the container moving distance between the quayside and the storage yard. Note that the assigned service time for feeders has to respect the feasible time windows, and the yard storage capacity and the berth processing capacity must be hold. The notations of the mathematical formulation are defined as follows:

Sets:

M : set of mother vessels

N : set of feeders

K : set of yard sections

T : set of working shifts, $T = \{1, 2, \cdots, \overline{t}\}$

I_1 : set of transshipment flows from mother vessels to feeders

I_2 : set of transshipment flows from feeders to mother vessels

I : set of all transshipment flows, $I = I_1 \cup I_2$

S : set of berth sections

Parameters:

θ^N_{ij} : binary coefficient taking value 1 if the corresponding feeder of flow $i \in I$ is feeder $j \in N$, and 0 otherwise

d_i : $\in S$, the allocated berth section for the mother vessel of flow $i \in I$

r_i : $\in T$, the arrival or departure time of flow $i \in I$ corresponding to the mother vessel

q_i : the amount of containers of flow $i \in I$ in Twenty-foot Equivalent Units (TEUs)

$[\underline{T}_j, \overline{T}_j]$: feasible visiting time window of feeder $j \in N$

ℓ_{ks} : the travel distance between yard section $k \in K$ and quay position $s \in S$

Q^1_{kt} : the storage capacity of yard section $k \in K$ during working shift $t \in T$

Q^2_{st} : the processing capacity (maximum number of loading and unloading containers) of berth section $s \in S$ during working shift $t \in T$

Decision variables:

x_{ik} : $\in \{0,1\}$, 1 if flow $i \in I$ is put in yard section $k \in K$ for temporary storage, and 0 otherwise

y_{jt} : $\in \{0,1\}$, 1 if feeder $j \in N$ is serviced at working shift $t \in T$, and 0 otherwise

z_{js} : $\in \{0,1\}$, 1 if feeder $j \in N$ is serviced at berth $s \in S$, and 0 otherwise

u_{it} : $\in \{0,1\}$, 1 if flow $i \in I$ is within the terminal (unloading, in storage and loading statuses) at working shift $t \in T$, and 0 otherwise

v_i : $\in \{0,1\}$, 1 if the departure working shift of flow $i \in I$ is later than its arrival working shift in one planning horizon, and 0 otherwise

w_h : the highest workload per working shift

w_l : the lowest workload per working shift

Regarding the feasible visiting time window of feeders, $\overline{T}_j$ is allowed to be smaller than $\underline{T}_j$ since the vessels visit the terminal cyclically. With the notation defined above, the problem is formulated as follows:

Objective function:

$$\min \quad \lambda \sum_{i\in I}\sum_{j\in N}\sum_{k\in K}\left(\sum_{s\in S}\ell_{ks}\theta^N_{ij}x_{ik}z_{js} + \sum_{s\in S|s=d_i} l_{ks}\theta^N_{ij}x_{ik}z_{js}\right) + (1-\lambda)(w_h - w_l) \tag{4.1}$$

The objective function consists of two parts: spatial objective of minimizing the total moving distance of all flows between the quayside and storage yard, and temporal objective of minimizing the gap between the highest and lowest workload over the planning horizon. The spatial objective reflects the transportation cost of yard trucks for carrying out all the container movements during the loading and unloading operations while the temporal objective indicates the quay crane workload imbalance. The two objectives are weighted by parameter $\lambda \in [0,1]$. Note that λ is predetermined by port operators which takes into account their relative preferences and monetary costs.

Constraints:

$$\sum_{s \in S} z_{js} = 1 \quad \forall j \in N \tag{4.2}$$

$$\sum_{t \in T} y_{jt} = 1 \quad \forall j \in N \tag{4.3}$$

$$\sum_{t \in T | \underline{T}_j \leq t \leq \overline{T}_j} y_{jt} = 1 \quad \forall j \in N | \underline{T}_j \leq \overline{T}_j \tag{4.4}$$

$$\sum_{t \in T | t \leq \overline{T}_j} y_{jt} + \sum_{t \in T | t \geq \underline{T}_j} y_{jt} = 1 \quad \forall j \in N | \underline{T}_j > \overline{T}_j \tag{4.5}$$

$$\sum_{k \in K} x_{ik} = 1 \quad \forall i \in I \tag{4.6}$$

$$0 \leq v_i + \left(r_i - \sum_{t \in T} t y_{jt} \right) / \bar{t} < 1 \quad \forall i \in I_1, \forall j \in N | \theta_{ij}^N = 1 \tag{4.7}$$

$$1 - r_i \leq t \left(u_{it} + v_i - 1 \right) \leq \sum_{t' \in T} t' y_{jt'} \quad \forall i \in I_1, \forall j \in N, \forall t \in T | \theta_{ij}^N = 1 \tag{4.8}$$

$$\sum_{t' \in T} t' y_{jt'} - \bar{t} \leq (\bar{t} + 1 - t)(u_{it} + v_i - 1) \leq \bar{t} + 1 - r_i \quad \forall i \in I_1, \forall j \in N, \forall t \in T | \theta_{ij}^N = 1 \tag{4.9}$$

$$\sum_{t \in T} (u_{it} + v_i - 1) = \sum_{t \in T} t y_{jt} - r_i + 1 \quad \forall i \in I_1, \forall j \in N | \theta_{ij}^N = 1 \tag{4.10}$$

$$0 \leq v_i + \left(\sum_{t \in T} t y_{jt} - r_i \right) / \bar{t} < 1 \quad \forall i \in I_2, \forall j \in N | \theta_{ij}^N = 1 \tag{4.11}$$

$$1 - \sum_{t' \in T} t' y_{jt'} \leq t(u_{it} + v_i - 1) \leq r_i \quad \forall i \in I_2, \forall j \in N, \forall t \in T | \theta_{ij}^N = 1 \tag{4.12}$$

$$r_i - \bar{t} \leq (\bar{t} + 1 - t)(u_{it} + v_i - 1) \leq \bar{t} + 1 - \sum_{t' \in T} t' y_{jt'} \quad \forall i \in I_2, \forall j \in N, \forall t \in T | \theta_{ij}^N = 1 \tag{4.13}$$

$$\sum_{t \in T} (u_{it} + v_i - 1) = r_i - \sum_{t \in T} t y_{jt} + 1 \quad \forall i \in I_2, \forall j \in N | \theta_{ij}^N = 1 \tag{4.14}$$

$$w_h \geq \sum_{i \in I | r_i = t} q_i + \sum_{i \in I} \sum_{j \in N} \theta_{ij}^N q_i y_{jt} \quad \forall t \in T \tag{4.15}$$

$$w_l \leq \sum_{i \in I | r_i = t} q_i + \sum_{i \in I} \sum_{j \in N} \theta_{ij}^N q_i y_{jt} \quad \forall t \in T \tag{4.16}$$

$$\sum_{i \in I} q_i x_{ik} u_{it} \leq Q_{kt}^1 \quad \forall k \in K, \forall t \in T \tag{4.17}$$

$$\sum_{i \in I | r_i = t, d_i = s} q_i + \sum_{i \in I} \sum_{j \in N} \theta_{ij}^N q_i y_{jt} z_{js} \leq Q_{st}^2 \quad \forall s \in S, \forall t \in T \tag{4.18}$$

$$x_{ik}, y_{jt}, z_{js}, u_{it}, v_i \in \{0, 1\} \quad \forall i \in I, \forall j \in N, \forall k \in K, \forall s \in S, \forall t \in T \tag{4.19}$$

$$w_h, w_l \geq 0 \tag{4.20}$$

Constraints (4.2) define the berth template for vessels by assigning each feeder one berthing position, while Constraints (4.3)-(4.5) imposes the restrictions related with schedule template. The assigned service time for each feeder is guaranteed to respect the feasible time window. The storage template constraints (4.6) determine the storage locations for the transshipment flows. Constraints (4.7)-(4.13) introduce the relationship between auxiliary decision variables u_{it},v_i and the feeders' service time variable y_{jt}. Imposed by Constraints (4.7), the binary variable v_i of transshipment flow $i \in I_1$ is set to 1 if and only if the service working shift of its corresponding feeder $\sum_{t \in T} t y_{jt}$ is later than its arrival time r_i. Similarly, Constraints (4.11) assign values to v_i for transshipment flows I_2. Constraints (4.8)-(4.10) and (4.12)-(4.14) define the variable u_{it} by enforcing its relationship with variable z_i and y_{jt}. Figure 4.3 shows two scenarios of different arrival and departure schedules of transshipment flow $i \in I_1$ and the values of the corresponding variables. Take Figure 4.3(a) as an example, Constraints (4.8) set u_{i6} and u_{i7} to 0. Similarly, u_{i1} takes 0 as ensured by Constraints (4.9). Constraints (4.10), along with (4.8) and (4.9), assign 1 to the rest of u_{it}. Constraints (4.15)-(4.16) assign the highest and lowest workload per working shift to variables w_h and w_l, respectively. The total amount of containers within a yard section should not exceed the storage capacity at each working shift, as guaranteed by Constraints (4.17). Similarly, the quayside workload of container loading and discharging at any berth should respect the berth processing capacity as ensured by Constraints (4.18). Note that the berth processing capacity actually depends on number of vessels, vessel types, number of quay crane assigned, etc. As this study focuses on tactical planning, we simplify the modeling of quayside operations (quay crane assignment and scheduling) by just imposing the restriction on the maximum number of containers processed at each berth. The quayside operations would be tackled in detail at the operational planning level. Finally, the domain of decision variables is defined by Constraints (4.19)-(4.20).

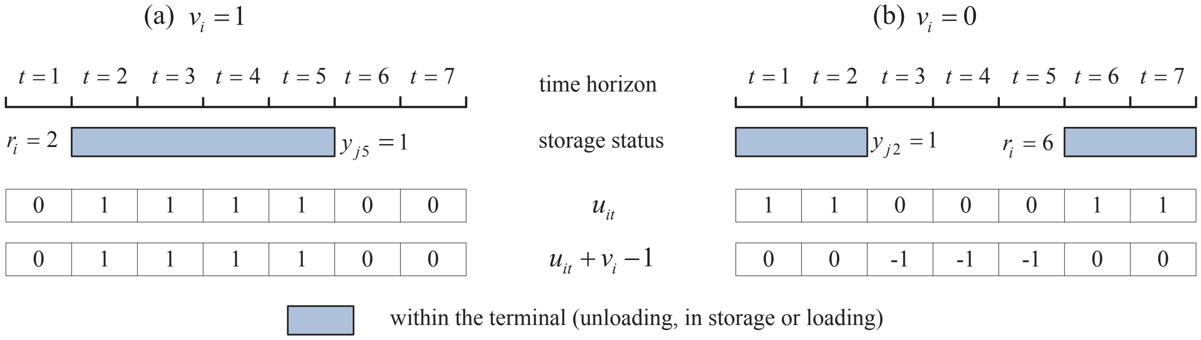


Figure 4.3: Two scenarios with different arrival and departure schedules for transshipment flow $i \in I_1$

4.3.3 Model linearization

Note that quadratic terms are involved in the objective function (4.1) and constraints (4.17) and (4.18). Additional auxiliary decision variables are introduced to linearize the model:

- $\delta_{ijks} \in \{0,1\} \quad \forall i \in I, \forall j \in N, \forall k \in K, \forall s \in S$, 1 if both of the decision variables x_{ik} and z_{js} take the value of 1, and 0 otherwise;
- $\varphi_{ikt} \in \{0,1\} \quad \forall i \in I, \forall k \in K, \forall t \in T$, 1 if both of the decision variables x_{ik} and u_{it} take the value of 1, and 0 otherwise;
- $\omega_{jts} \in \{0,1\} \quad \forall j \in N, \forall t \in T, \forall s \in S$, 1 if both of the decision variables y_{jt} and z_{js} take the value of 1, and 0 otherwise.

The related additional constraints are defined as follows:

$$\delta_{ijks} \geq x_{ik} + z_{js} - 1 \quad \forall i \in I, \forall j \in N, \forall k \in K, \forall s \in S \tag{4.21}$$

$$\delta_{ijks} \leq x_{ik} \quad \forall i \in I, \forall j \in N, \forall k \in K, \forall s \in S \tag{4.22}$$

$$\delta_{ijks} \leq z_{js} \quad \forall i \in I, \forall j \in N, \forall k \in K, \forall s \in S \tag{4.23}$$

$$\varphi_{ikt} \geq x_{ik} + u_{it} - 1 \quad \forall i \in I, \forall k \in K, \forall t \in T \tag{4.24}$$

$$\varphi_{ikt} \leq x_{ik} \quad \forall i \in I, \forall k \in K, \forall t \in T \tag{4.25}$$

$$\varphi_{ikt} \le u_{it} \quad \forall i \in I, \forall k \in K, \forall t \in T \tag{4.26}$$

$$\omega_{jts} \ge y_{jt} + z_{js} - 1 \quad \forall j \in N, \forall t \in T, \forall s \in S \tag{4.27}$$

$$\omega_{jts} \le y_{jt} \quad \forall j \in N, \forall t \in T, \forall s \in S \tag{4.28}$$

$$\omega_{jts} \le z_{js} \quad \forall j \in N, \forall t \in T, \forall s \in S \tag{4.29}$$

$$\delta_{ijks}, \varphi_{ikt}, \omega_{jts} \in \{0,1\} \quad \forall i \in I, \forall j \in N, \forall k \in K, \forall s \in S, \forall t \in T \tag{4.30}$$

With the additional defined decision variables, the objective function (4.1) and Constraints (4.17)-(4.18) can be updated as follows:

$$\lambda \sum_{i\in I}\sum_{j\in N}\sum_{k\in K}\left(\sum_{s\in S}\ell_{ks}\theta_{ij}^{N}\delta_{ijks} + \sum_{s\in S|s=d_i} l_{ks}\theta_{ij}^{N}\delta_{ijks}\right) + (1-\lambda)(w_h - w_l) \tag{4.31}$$

$$\sum_{i\in I} q_i\varphi_{ikt} \le Q_{kt}^{1} \quad \forall k \in K, \forall t \in T \tag{4.32}$$

$$\sum_{i\in I|r_i=t,d_i=s} q_i + \sum_{i\in I}\sum_{j\in N}\theta_{ij}^{N} q_i\omega_{jts} \le Q_{st}^{2} \quad \forall s \in S, \forall t \in T \tag{4.33}$$

Therefore, the feeder vessel management problem can be formulated as a mixed integer linear program as follows:

$$[\mathbf{P}] \quad \min \quad \lambda \sum_{i\in I}\sum_{j\in N}\sum_{k\in K}\left(\sum_{s\in S}\ell_{ks}\theta_{ij}^{N}\delta_{ijks} + \sum_{s\in S|s=d_i} l_{ks}\theta_{ij}^{N}\delta_{ijks}\right) + (1-\lambda)(w_h - w_l)$$

$$\text{s.t.} \quad (4.2)-(4.16), (4.19)-(4.30) \text{ and } (4.32)-(4.33)$$

4.3.4 Computational complexity

Consider a particular case of the feeder vessel management problem where the planning horizon only covers one time period (i.e., $|T| = 1$) and the berth template for feeders is pre-determined. In this case, all vessels arrive at the same time period and the workload imbalance in the objective function becomes a constant. Since the berth allocation for feeders is known, the resulting problem is to decide the assignment of container flows to yard sections with the objective of minimizing the cost of container spatial movement. The remaining constraint only includes the storage capacity requirement for all the yard sections. If we consider container flows and yard sections as items and bins, respectively, the problem is equivalent to the *Generalized Assignment Problem* (Martello and Toth, 1992): the storage allocation decision corresponds to the bin-item assignment; the cost of spatial movement of container flows corresponds to the assignment cost of items; the storage capacity constraint is identical to the bins' budget restriction. The above transformation has shown the reduction of the *Generalized Assignment Problem* to a particular case of the proposed feeder vessel management problem. Since the *Generalized Assignment Problem* is known to be NP-hard, the feeder vessel management problem is also NP-hard.

4.4 Heuristic Approach

As the complexity of the problem precludes solving formulation **[P]** exactly for real-world instances, we develop a memetic heuristic to obtain near-optimal solutions in an efficient way. The memetic heuristic is a hybrid meta-heuristic of a population-based approach with a local improvement procedure for individuals. In this chapter, we combine genetic algorithm and tabu search. With a set of randomly generated initial population, the genetic search procedure applies genetic operations on the population to generate offspring, each individual of which is post-optimized by a tabu search procedure. With the original population and the post-optimized offspring, a selection procedure is applied to pick the solutions with good performance and discard the rest. Afterwards, the genetic search procedure reruns the above steps iteratively with the updated population until any stopping condition is met. The key difference between the memetic heuristic and genetic algorithm lies in the behavior of memes which itself is capable of evolving by local refinement. The details of the memetic heuristic are illustrated as follows.

4.4.1 Solution representation

A solution is encoded by $|N|$ memes associated with $|N|$ feeders. Each meme comprises several elements indicating the assigned berth position and service time for the corresponding feeder, and the allocated storage locations for associated transshipment flows. Figure 4.4 shows an illustrative example of the solution representation with three feeder vessels and nine transshipment flows. The feeder-flow relationship is shown on the left and the solution encoding is presented on the right. As shown, Feeder 1 connects two flows (Flow 1, Flow 3) and the corresponding storage locations are indicated by the last two elements of Meme 1 (Y5, Y3). Another two elements are also attached in Meme 1 representing the assigned berth position (B2) and service time (T1) for Feeder 1. It is similar for the other two memes. The three types of decisions for the berth template, schedule template and yard template design are represented by the first, second and the rest elements of all memes.

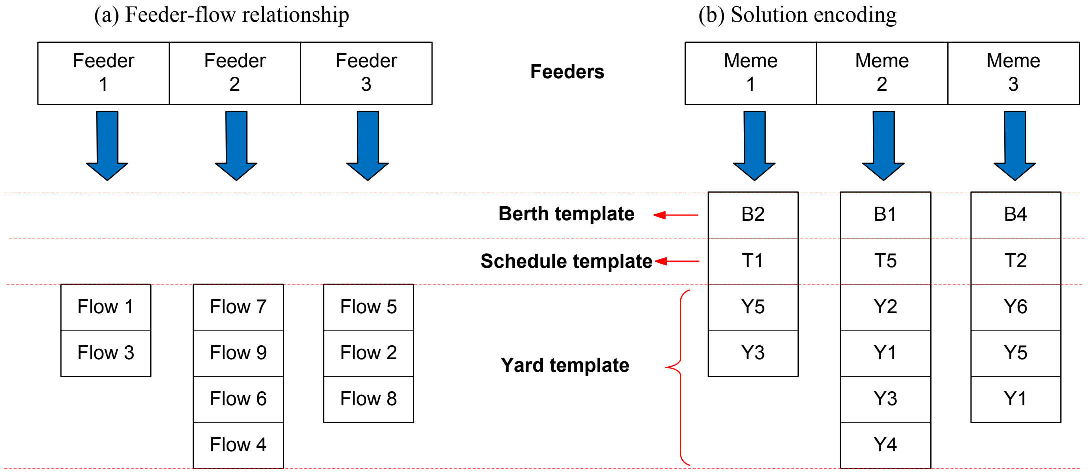


Figure 4.4: An illustrative example of the solution representation.

4.4.2 Initial population and fitness evaluation

With the above designed solution representation, a set of initial population with a size of $NbPop$ is randomly generated in the following way. Each candidate solution assigns the feeders uniformly to the berth positions and to the feasible service time windows without consideration of the berth processing capacity constraints (4.33). Similarly, the transshipment flows are also uniformly distributed to the storage yard sections disregarding the block capacity constraints (4.32). For a candidate solution σ, let $c(\sigma)$ denote the value of the objective function (4.31) and let $p(\sigma)$ be the total violation of the capacity constraints (4.32)-(4.33) as defined by Equation (4.34) where φ^{σ}_{ikt} and ω^{σ}_{jts} are the values of the decision variables φ_{ikt} and ω_{jts} associated with solution σ. Then, the fitness of solution σ is calculated by the function $f(\sigma) = c(\sigma) + \alpha p(\sigma)$, where the constraint violation term is penalized by a positive parameter α. Initially, α is set to be a small number to facilitate exploring the search space. After each genetic search iteration, α is multiplied by $\tau > 1$. Thus, as the genetic search procedure progresses, the focus of the search process moves from space exploration to solution feasibility.

$$\begin{aligned} p(\sigma) = \sum_{k \in K} \sum_{t \in T} \max \left\{ 0, \left(\sum_{i \in I} q_i \varphi_{ikt}^{\sigma} - Q_{kt}^{1} \right) \right\} \\ + \sum_{s \in S} \sum_{t \in T} \max \left\{ 0, \left(\sum_{i \in I | r_i = t, d_i = s} q_i + \sum_{i \in I} \sum_{j \in N} \theta_{ij}^{N} q_i \omega_{jts}^{\sigma} - Q_{st}^{2} \right) \right\} \end{aligned} \tag{4.34}$$

4.4.3 Genetic search procedure

The genetic search procedure mimics the process of natural evolution to generate new solutions based on the original population. For each genetic iteration, the population is paired randomly and a uniform crossover operation is applied for each pair. Given a pre-determined crossover parameter β, each meme pair of the two parents is exchanged only if a number chosen randomly from 0.0 to 1.0 is smaller than β. After the crossover, a pair of offspring is produced. Figure 4.5 illustrates an example of the crossover operation. The randomly generated numbers suggest that only the memes at the second and last positions should be swapped while the rest are kept still. The new generated offspring are further optimized by a tabu search procedure introduced in Section 4.4.4. Note that the traditional mutation operator of genetic algorithm is not applied since the tabu search procedure has taken the responsibility of introducing diversity for the generated solutions. For our particular solution encoding method which has two levels (the meme level and the level of individual elements within a meme as shown in Figure 4.4), the structure of memes is preserved during the crossover operation while the individual memes are still a string instead of single elements. This makes the traditional mutation operator is no longer suitable, and thus we apply tabu search to post-optimized the generated offspring.

The post-optimized offspring and the parents are evaluated according to the fitness function. Since the number of parents and offspring are the same, we rank them in order according to their

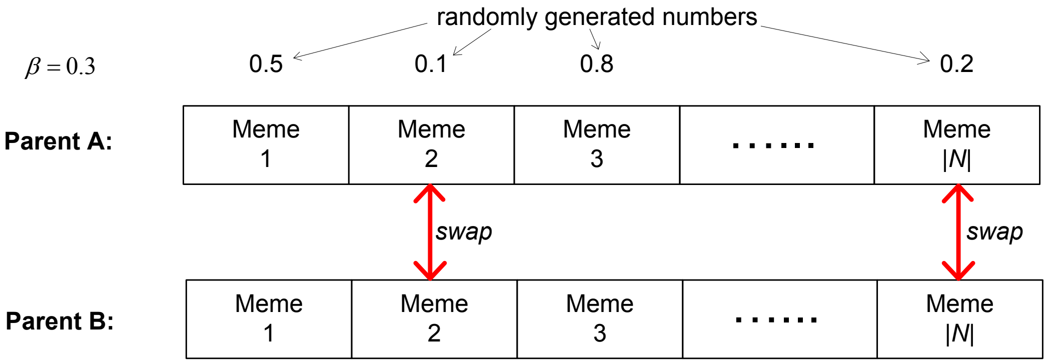


Figure 4.5: An illustrative example of the parameterized uniform crossover operation.

fitness and select half of the solutions with best fitness and treat them as the new population without considering whether they belong to parents or offspring. The rest of the solutions are discarded. With the updated population, we continue with the search procedure until certain termination criteria are met. In order to keep track of the best solution found during the search procedure, we define $Best$ which is initially set to empty and is updated by the best solution found among the population during each iteration if its fitness is superior to $Best$ and it is feasible. The genetic search procedure terminates when the following condition is met: the overall fitness of the population does not get improved for GC_1 consecutive iterations or the maximum number of genetic search iterations GC_2 is reached.

4.4.4 Tabu search procedure

After the crossover operation, a tabu search procedure is employed to improve the individuals of the offspring. In this procedure, the elements of memes are allowed to evolve by searching the neighborhood space in order to improve the fitness of the individual solutions. The neighborhood solution is generated by randomly picking an element of a solution and then assigning it with an alternative value. Figure 4.6 shows an example of the neighborhood solution generation method. In this example, the third element of Meme 2 is changed from Y2 to Y4. Note that the randomly picked element is assigned with an alternative value without any consideration of solution feasibility which is only reflected by the violation penalty in the fitness evaluation function. For each tabu search iteration, TC_1 neighborhood solutions are generated to sample the gradient (i.e., the best of the TC_1 neighborhood solutions is used to update the original solution), and the best solution is updated by the gradient solution if its fitness is superior to that of the current best solution. In the tabu list, we record the position of the changed element in the new best solution compared with the previous one during each iteration. The elements' positions stored in the tabu list are prohibited to change when generating neighborhood solutions. The length of the tabu list is set to TC_2 and the information are stored in the list in a first-in-first-out rule. The tabu search procedure terminates when the following condition is met: the best solution does not get improved for TC_3 consecutive iterations or the maximum number of tabu search procedure TC_4 is reached.

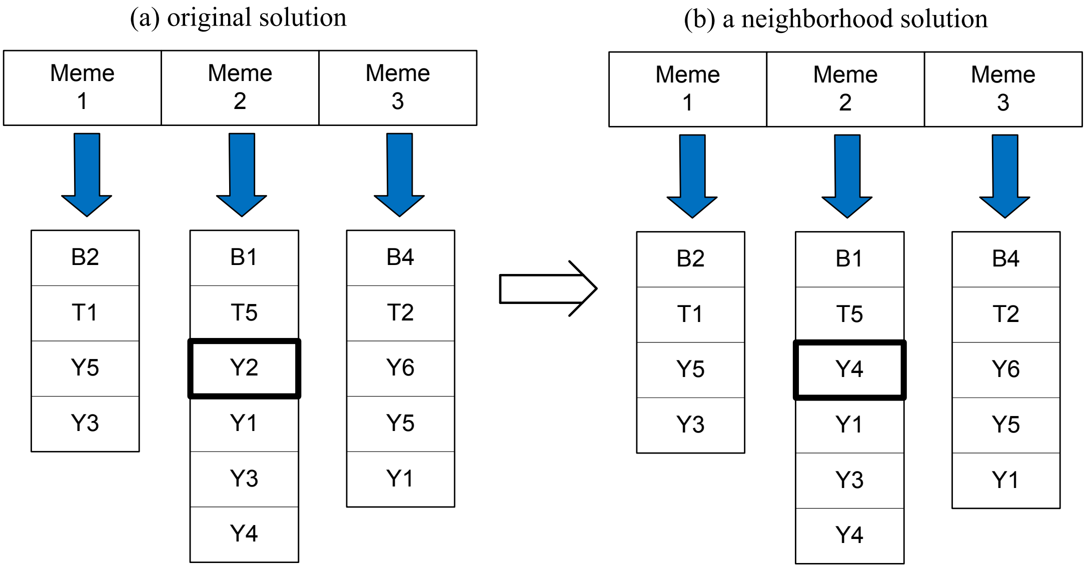


Figure 4.6: Neighborhood solution generation method.

The entire memetic heuristic procedure is summarized as follows:

Memetic heuristic procedure:

```
input: an instance, heuristic parameters
output: the best feasible solution $Best$

generate a set of initial population $Pop$;
$Best \leftarrow \emptyset$;
evaluate the fitness of all individuals in $Pop$;
repeat genetic search procedure
  randomly pair the population;
  for each pair
    do crossover operation to generate a pair of offspring;
    for each individual offspring
      post-optimize the offspring by tabu search procedure;
    end for
  end for
  evaluate the fitness of all individuals in the offspring;
  update $Pop$;
  update $Best$ according to the fitness and feasibility of $Pop$;
  update $\alpha \leftarrow \alpha \times \tau$;
until the stopping condition is met
Return $Best$;
```

4.5 Numerical Experiment

In this section we firstly illustrate the generation of test instances and the parameter settings of the memetic heuristic. Then, we assess the efficiency and effectiveness of the developed heuristic approach by comparing its results with the optimal ones obtained by solving the MIP model [**P**]. The performance of the feeder vessel management is evaluated by some scenario analysis. Finally, sensitivity analysis is conducted to see the effect of the weight parameter λ in the objective function. The optimization model and the memetic heuristic are coded in C++ and CPLEX 12.1 is used as the MIP solver. All the numerical experiments are conducted on a PC with 3 GHz CPU and 4 GB RAM.

4.5.1 Instance generation and algorithm settings

Four sets of test instances are randomly generated based on the Brani terminal in Singapore as shown in Figure 4.7. Each set has ten instances with the same parameters as listed in Table 4.1: number of mother vessels, number of feeders, number of transshipment flows, number of

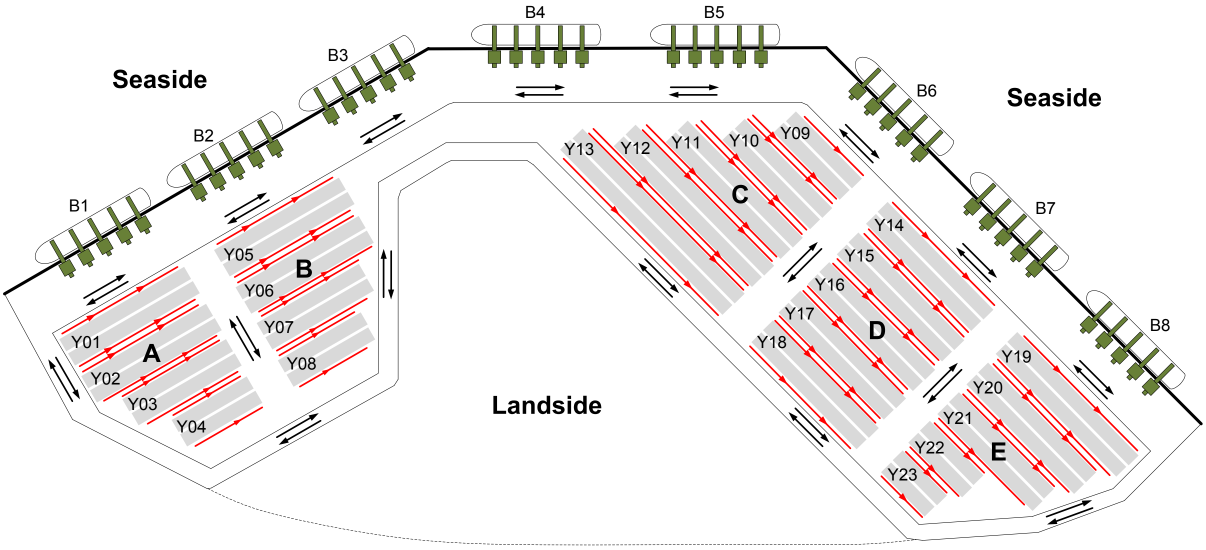


Figure 4.7: A schematic view of the Brani terminal in Singapore.

working shifts within one horizon, available storage yard sections and available berth sections. Take Set 4 as an example, the terminal is visited by 10 mother vessels and 30 feeders weekly (21 working shifts) and there are 60 transshipment flows. All the berth sections and the yard sections are available for serving the vessels and flows. The amount of containers in each flow q_i is uniformly distributed within [100, 500] TEUs. The service time of mother vessels and the feasible service time windows of feeders are randomly distributed within the planning horizon. The length of the feasible service time windows is set to be six. The relationship between flows and vessels is generated in the following manner: For each transshipment flow, we randomly assign a feeder and a mother vessel and the flow direction (containers are transshipped from mother vessel to feeder or in the reverse direction) is also randomly determined. The following restriction is imposed when generating the flows: a mother vessel and a feeder can be associated with at most six and three flows, respectively.

The weight parameter λ is set to 0.1 for the computational tests in Section 4.5.2 and 4.5.3. The parameters of the memetic heuristic are selected by trials and listed as follows:

- Genetic algorithm search: $NbPop = 6\lceil \ln(|I| \cdot |K|) \rceil$, $\beta = 0.45$, $GC_1 = 10$, $GC_2 = 200$;
- Tabu search: $TC_1 = \lceil \sqrt{|I| \cdot |K|} \rceil$, TC_2=30, $TC_3 = 10$, $TC_4 = 100$;

Table 4.1: Instance parameters.

	$\lvert M\rvert$	$\lvert N\rvert$	$\lvert I\rvert$	$\lvert T\rvert$	K	S
Set1	5	10	20	12	Y01-Y08	B1-B3
Set2	6	15	30	15	Y09-Y23	B4-B8
Set3	8	20	40	18	Y01-Y23	B1-B8
Set4	10	30	60	21	Y01-Y23	B1-B8

- Length of tabu list: 30;

- Fitness evaluation: $\alpha = 1.0$, $\tau = 1.1$.

4.5.2 Results of memetic heuristic

In order to assess the efficiency and effectiveness of the developed memetic heuristic, the results are compared with the best solutions found by CPLEX. Tables 4.2-4.5 list the computational results of all instances. The information returned by CPLEX is reported in the second and third columns. CPLEX is given a computational budget of 360 minutes and returns the best solution values once reaching the time limit. Due to the randomness of the memetic heuristic, we run it ten times and report the best and worst solution values as well as the average computational time in columns (4)-(6). The last two columns compare the solution values of the memetic heuristic and CPLEX. As can be seen from Table 4.2, memetic heuristic is able to find near-optimal solutions with less than 0.1% gaps on average, and the computational time is comparable to that of CPLEX. As the instance scale increases, CPLEX is no longer able to solve the MIP model [P] to optimum by the time limit. However, memetic heuristic can still find near-optimal or even better solutions than truncated CPLEX. For the largest data Set 4, memetic heuristic not only finds better solutions than CPLEX, but also requires less computational efforts.

4.5.3 Scenario analysis

Computational experiments are also conducted to investigate the effectiveness of integrating the berth template, schedule template and yard template design for the feeder vessels in a container transshipment terminal. The following four scenarios are compared:

- Scenario 1: all the three templates are allowed to be optimized;

Table 4.2: Computational results of data Set 1.

Instance	CPLEX		Memetic			Gap	
	Result	Time(m)	Best	Worst	Time(m)	Gap1(%)	Gap2(%)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
Set1-I01	1562.03	0.5	1562.03	1564.29	0.3	0.00	0.14
Set1-I02	2246.71	2.9	2246.71	2247.22	0.4	0.00	0.02
Set1-I03	2038.63	0.3	2038.63	2040.53	0.3	0.00	0.09
Set1-I04	1936.26	0.5	1938.63	1939.38	0.4	0.12	0.16
Set1-I05	1832.49	0.5	1832.49	1832.92	0.3	0.00	0.02
Set1-I06	2276.48	0.2	2276.48	2277.04	0.3	0.00	0.02
Set1-I07	1898.62	0.2	1898.62	1900.20	0.3	0.00	0.08
Set1-I08	2083.18	0.2	2083.18	2083.18	0.3	0.00	0.00
Set1-I09	1198.51	1.0	1198.51	1201.08	0.3	0.00	0.21
Set1-I10	1857.50	0.5	1857.50	1861.88	0.4	0.00	0.24
Average						0.01	0.10

Gap1=[(4)-(2)]/(2)×100%, Gap2=[(5)-(2)]/(2)×100%

Table 4.3: Computational results of data Set 2.

Instance	CPLEX		Memetic			Gap	
	Result	Time(m)	Best	Worst	Time(m)	Gap1(%)	Gap2(%)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
Set2-I01	2086.61	360.0	2082.80	2163.61	1.8	-0.18	3.69
Set2-I02	2345.79	360.0	2332.60	2343.23	1.7	-0.56	-0.11
Set2-I03	2145.94	360.0	2146.26	2164.95	2.9	0.02	0.89
Set2-I04	2179.41	360.0	2174.63	2187.26	3.4	-0.22	0.36
Set2-I05	2679.55	360.0	2697.98	2714.67	1.4	0.69	1.31
Set2-I06	2030.78	360.0	2021.74	2031.52	1.9	-0.45	0.04
Set2-I07	2767.35	360.0	2767.35	2777.71	2.0	0.00	0.37
Set2-I08	2646.93	360.0	2645.41	2658.45	2.0	-0.06	0.44
Set2-I09	1799.49	360.0	1781.09	1798.80	2.4	-1.02	-0.04
Set2-I10	2906.67	360.0	2900.36	2910.49	2.0	-0.22	0.13
Average						-0.20	0.71

Gap1=[(4)-(2)]/(2)×100%, Gap2=[(5)-(2)]/(2)×100%

Table 4.4: Computational results of data Set 3.

Instance	CPLEX		Memetic			Gap	
	Result	Time(m)	Best	Worst	Time(m)	Gap1(%)	Gap2(%)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
Set3-I01	4949.51	360.0	4944.31	4947.92	4.2	-0.11	-0.03
Set3-I02	4301.03	360.0	4259.96	4270.24	4.7	-0.95	-0.72
Set3-I03	2929.27	360.0	2874.96	2896.29	6.6	-1.85	-1.13
Set3-I04	3887.86	360.0	3824.06	3836.93	6.2	-1.64	-1.31
Set3-I05	3646.37	360.0	3600.13	3603.90	5.3	-1.27	-1.16
Set3-I06	4155.08	360.0	4149.12	4157.02	5.4	-0.14	0.05
Set3-I07	3300.60	360.0	3215.31	3240.62	7.0	-2.58	-1.82
Set3-I08	3915.65	360.0	3899.01	3910.01	5.2	-0.42	-0.14
Set3-I09	4155.83	360.0	4116.50	4142.60	8.5	-0.95	-0.32
Set3-I10	3170.56	360.0	3142.70	3165.45	6.0	-0.88	-0.16
Average						-1.08	-0.67

Gap1=[(4)-(2)]/(2)×100%, Gap2=[(5)-(2)]/(2)×100%

Table 4.5: Computational results of data Set 4.

Instance	CPLEX		Memetic			Gap	
	Result	Time(m)	Best	Worst	Time(m)	Gap1(%)	Gap2(%)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
Set4-I01	4652.00	360.0	4603.62	4620.64	32.8	-1.04	-0.67
Set4-I02	6602.45	360.0	6189.31	6212.89	34.4	-6.26	-5.90
Set4-I03	4301.50	360.0	3781.59	3865.89	40.9	-12.09	-10.13
Set4-I04	6058.58	360.0	5467.73	5514.80	33.4	-9.75	-8.98
Set4-I05	4858.25	360.0	4432.23	4532.03	35.0	-8.77	-6.71
Set4-I06	4429.21	360.0	3881.79	3910.72	32.5	-12.36	-11.71
Set4-I07	6100.52	360.0	5831.86	5869.89	46.0	-4.40	-3.78
Set4-I08	6234.11	360.0	5710.98	5770.31	42.9	-8.39	-7.44
Set4-I09	6044.69	360.0	5958.94	6008.63	26.6	-1.42	-0.60
Set4-I10	6052.09	360.0	5276.94	5317.23	37.3	-12.81	-12.14
Average						-7.73	-6.81

Gap1=[(4)-(2)]/(2)×100%, Gap2=[(5)-(2)]/(2)×100%

- Scenario 2: fix the berth template and optimize the other two;
- Scenario 3: fix the schedule template and optimize the other two;
- Scenario 4: fix both of the berth and schedule template, and optimize the yard template.

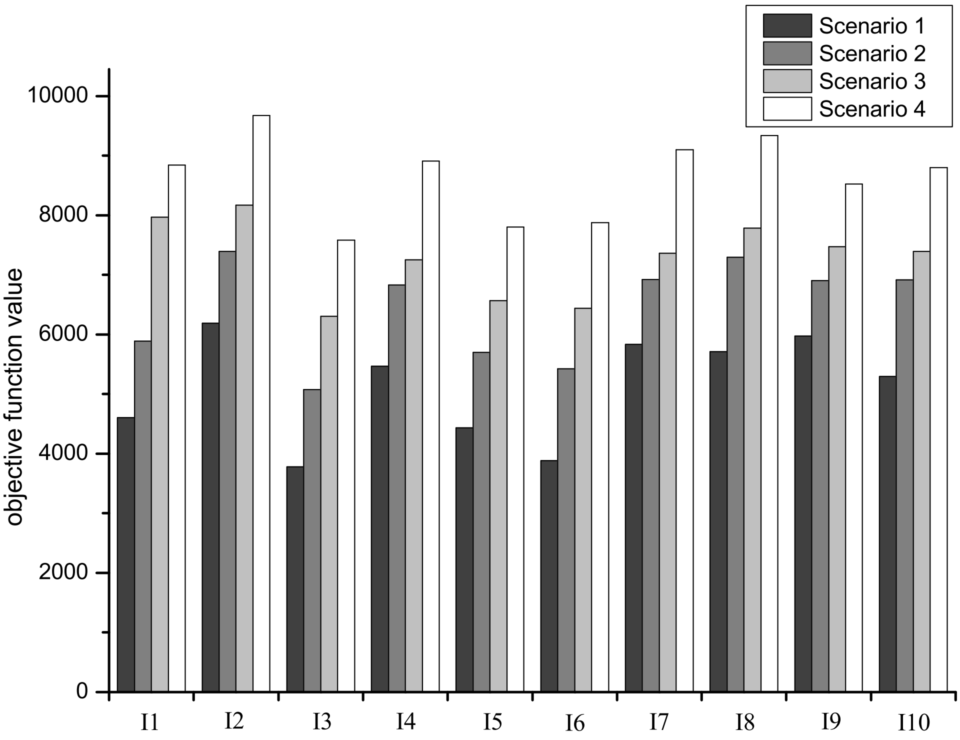


Figure 4.8: Scenario comparison for Set 4

Scenario 1 corresponds to the problem studied in this chapter. Scenario 2 integrates two decisions, schedule template and yard template, with determined berth template. Similarly, Scenario 3 only optimizes the berth template and yard template while the feeders' schedule information is given. Scenario 4 focuses on the storage yard management for transshipment flows without integration with the quayside operations. For the last three scenarios, the fixed berth/schedule templates are generated by randomly assigning the berth positions/service time of feeder vessels. With the fixed templates, part of the decision variables of **[P]** is known and the resulting problem can be solved by CPLEX. The last three scenarios are run ten times with ten randomly determined templates, and the mean solution values are reported. Note that

Scenario 3 corresponds to the current practice of terminal operations as container terminals just accept the visiting requests of vessels passively and there is no rule-of-thumb for determining or optimizing feeder schedule template. Thus, the randomly determined schedule templates are representative of the real-world cases. Figure 4.8 compares the solution values of the above four scenarios for data Set 4. As can be seen, the best and worst cases correspond to Scenarios 1 and 4, respectively. By integrating the yard template design problem with two quayside operations, berth template and schedule template design, the objective function value can be reduced by $30 \sim 50\%$. For the other two scenarios, the results suggest that integrating yard and schedule template design yields lower objective function value than that of integrating yard and berth template design. Note that the gap between Scenario 1 and 3 indicates the improvement of the proposed feeder vessel management problem upon the current practice of terminal operations. As can be seen, integrating the schedule template design could yield $20 \sim 40\%$ reduction of the objective function value. Overall speaking, there is a significant improvement if the berth, schedule and yard templates are flexible and receptive to be adjusted.

4.5.4 Sensitivity analysis

The two components of the objective function, container moving distance measured in kilometers and workload imbalance measured in number of containers, are weighted by the parameter λ. Port operators have to pre-determine the parameter with consideration of the monetary costs of the two components as well as their relative preferences. The former could be easily transformed into monetary cost by multiplying the container moving distance by the truck operating cost per kilometers. However, the latter component cannot be directly transformed into monetary cost. Port operators could adjust λ to achieve a trade-off between the two conflicting objectives so that the container moving cost is as low as possible while the workload imbalance is within an acceptable range. Figure 4.9 shows the results of spatial cost and temporal cost obtained with regard to diverging values of λ for two test instances. The larger λ is, the more the spatial cost is weighted in the objective function while the less the temporal cost is weighted. Based on the computational results, we can see that both of the two costs are not sensitive to λ when it falls in $[0.1, 0.5]$. Within this range, we can achieve a status where the spatial cost and the temporal cost can be minimized at the same time.

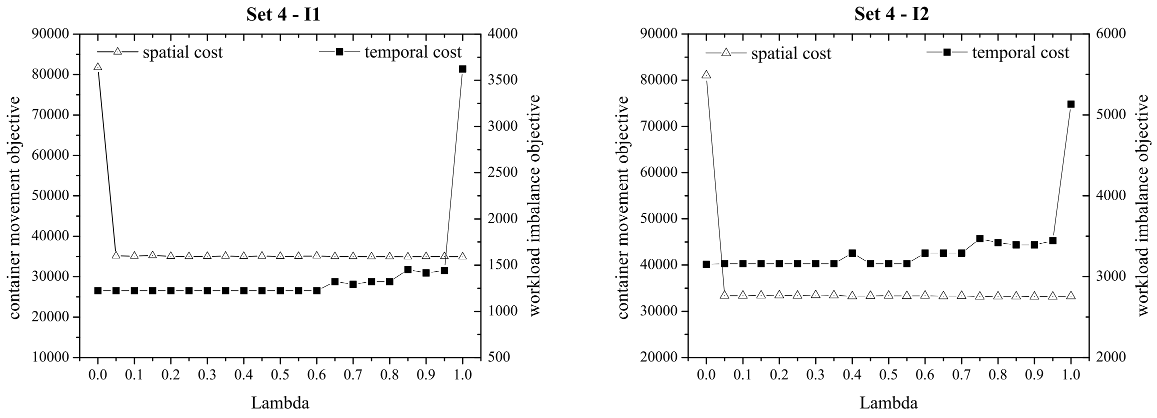


Figure 4.9: Effect of the weight parameter λ.

4.6 Summary

In this chapter we have studied the management of feeder vessels in a container transshipment terminal including three tactical decision problems: berth template, schedule template and yard template design. Unlike previous efforts found in the literature, we adjust the feeders' calling schedules from the perspective of container terminals so as to optimize the temporal distribution of quayside workload. Feeder arrival time is permitted to be adjusted within the feasible time windows by the terminal operator while the calling requests of mother vessels must be satisfied. Meanwhile, the transshipment flows between mother vessels and feeders are organized in an optimal manner with spatial considerations by designing the berth template for feeders and yard template for the flows. A mixed integer program is developed for the integrated problem, and a memetic heuristic is designed to obtain near-optimal solutions efficiently. Computational experiments have validated the performance of the proposed heuristic and the effectiveness of integrating the schedule template design with other terminal operations.

The contribution of this chapter to the literature includes the following:

(1) We adopt a proactive management strategy from container terminals' perspective and

optimize the quayside traffic by adjusting the feeders' calling schedule. This schedule template design is integrated with another two tactical decision problems, berth and yard template design. A mixed integer quadratic programming model is presented and further linearized;

(2) A memetic heuristic is developed to tackle the difficulty of solving the model exactly, which combines genetic algorithm search and tabu search. Numerical tests have shown that the proposed feeder vessel management problem could be solved by the memetic heuristic efficiently, and yields significant improvement upon current practice of terminal operations.

For future research, we are interested in extending the developed model to account for some irregular cases of feeder visiting patterns, e.g., multiple visits during each week, since such cases occasionally happen at some container terminals. Another promising direction for future research is to exploit the formulation structure of the feeder vessel management problem and apply decomposition methods in order to devise exact solution algorithms.

Chapter 5

A Column Generation based Heuristic to Feeder Vessel Management Problem

5.1 Introduction

In this chapter, we develop a column generation based heuristic to the Feeder Vessel Management problem introduced in Chapter 4.

Column generation is a efficient algorithm for solving large-scale linear programs where the number of decision variables are much more than that of constraints. The basic idea of tackling linear programs with such a special structure of constraint matrix is to iteratively solve a restricted problem only involving a subset of decision variables (*restricted master problem*) and generate beneficial decision variables by a sub-problem (*pricing sub-problem*) dynamically. The iterative procedure of solving the restricted master problem and generating decision variables terminates when the objective function value of the restricted master problem cannot get improved with introduction of any more decision variables. Therefore, only a small subset of decision variables is generated and included in the restricted master problem. The benefit of computational efficiency comes from the restricted master problem and the pricing sub-problem which are much easier to tackle than the original problem.

The special structure of the original formulation **[P]** developed for the feeder vessel management problem in Chapter 4 motivates us to design an efficient solution method based on column generation. Observe that the Constraints (4.2)-(4.5) and (4.7)-(4.14) are defined with respect to each feeder and thus the constraints of different feeders are independent. Therefore, we are motivated to treat the decisions related with an individual feeder as a column (decision variable) and reformulate **[P]** via Dantzig-Wolfe decomposition. Note that each column is modeled with extensive information related with a certain feeder including the berth position and the service time assigned for the feeder and the storage locations for its associated transshipment flows.

Column generation has been successfully applied to many problems in the literature such as vehicle routing and crew scheduling. In the field of container terminal operations, Moccia et al. (2009) studied a container location and relocation problem (referred to as dynamic generalized assignment problem in the paper) and developed efficient column generation based methods. Vacca et al. (2012) proposed an exact algorithm for the integrated planning of berth allocation and quay crane assignment problem in which column generation is embedded in a branch-and-bound procedure.

The remainder of this chapter is organized as follows. Section 5.2 reformulates the feeder vessel management problem as a set covering model via Dantzig-Wolfe decomposition. Section 5.3 develops the column generation based heuristic for solving the reformulation. Computational tests are conducted in Section 5.4 and conclusion is made in Section 5.5.

5.2 A Set Covering Model

The feeder vessel management problem could be modeled as a set covering model. Let P_j denote the set of service plans associated with feeder $j \in N$. Each service plan $p \in P_j$ defines the dedicated resources assigned to vessel $j \in N$, including the assigned berth position, service time, and the allocated storage locations for associated transshipment flows. We remark that the definition of service plans is similar to that of the memes in Chapter 4. We now define a binary decision variable δ_{jp}, where $\delta_{jp} = 1$ if and only if the service plan $p \in P_j$ is assigned to feeder $j \in N$. We further define decision variables w_h and w_l to represent the highest and lowest workload per working shift, respectively. The following coefficients are also introduced to represent the information associated with each service plan.

- c_{jp}: the total distance of container flow movements associated with service plan $p \in P_j$ of feeder $j \in N$
- α_{jpkt}: the storage space requirement at yard section $k \in K$ at time period $t \in T$ of service plan $p \in P_j$ (TEUs)
- β_{jpst}: the berth processing requirement at berth $s \in S$ at time period $t \in T$ of service plan $p \in P_j$ (TEUs)

With the above notations, the set covering model for the feeder vessel management problem is presented as follows:

$$
\begin{aligned}
\textbf{[MP]} \quad & \text{minimize} \quad && \lambda \sum_{j \in N} \sum_{p \in P_j} c_{jp}\delta_{jp} + (1-\lambda)(w_h - w_l) && (5.1)\\
& \text{subject to} \quad && \sum_{p \in P_j} \delta_{jp} = 1 \quad \forall j \in N && (5.2)\\
& && \sum_{j \in N} \sum_{p \in P_j} \alpha_{jpkt}\delta_{jp} \le Q^1_{kt} \quad \forall k \in K, \forall t \in T && (5.3)\\
& && \sum_{j \in N} \sum_{p \in P_j} \beta_{jpst}\delta_{jp} \le Q^2_{st} \quad \forall s \in S, \forall t \in T && (5.4)\\
& && w_h - \sum_{s \in S} \sum_{j \in N} \sum_{p \in P_j} \beta_{jpst}\delta_{jp} \ge 0 \quad \forall t \in T && (5.5)\\
& && w_l - \sum_{s \in S} \sum_{j \in N} \sum_{p \in P_j} \beta_{jpst}\delta_{jp} \le 0 \quad \forall t \in T && (5.6)\\
& && \delta_{jp} \in \{0,1\} \quad \forall i \in N, \forall p \in P_j && (5.7)\\
& && w_h, w_l \ge 0 && (5.8)
\end{aligned}
$$

Objective function (5.1) minimizes a convex combination of the spatial cost of container flow movements and temporal cost of the quayside workload imbalance. Pre-determined by terminal operators, the weight parameter λ should reflect the monetary costs of the two conflicting objectives as well as their relative preferences. Constraints (5.2) ensure that one service plan should be chosen exactly for each feeder vessel. The total amount of storage space requested by all the chosen service plans should observe the storage capacity of yard sections, as guaranteed by Constraints (5.3). Similarly, Constraints (5.4) ensure that the total amount of the quayside workload of all the chosen service plans respect the berth processing capacity. Constraints (5.5) and (5.6) impose the relationship between decision variables w_h, w_l with δ_{jp}, respectively. Finally, Constraints (5.7)-(5.8) define the domain of decision variables.

Compared with the formulation **[P]** developed in Section 4.3.3, **[MP]** reduces the size of the constraints at the cost of increasing the number of decision variables. The cardinality of the service plan set $|P| = \mathcal{O}(|N| \cdot |S| \cdot |T| \cdot |K|^n)$, where n is the maximum number of container flows associated with a feeder vessel. Given the real-world terminal layout shown in Figure 4.7 and 30 feeders with $n = 3$, the size of P would be larger than 60 million. Hence, it is hardly possible to solve the above set covering formulation **[MP]** by a commercial solver. It is observed that only N service plans would be selected out of the large pool of service plan candidates. We motivate to develop a column generation based heuristic to find near-optimal solutions in an efficient way.

5.3 A Column Generation based Heuristic

The column generation heuristic starts solving the *restricted master problem* (RMP) with only a small subset of columns (i.e., decision variables) of the original linear program. According to the status of the RMP , *Pricing sub-problems* (PSP) are solved dynamically to find new columns with negative reduced cost. Then the columns found are added into the RMP which could be re-solved. Such a procedure continues until no columns with negative reduced cost can be found by the PSP.

5.3.1 Restricted master problem

We apply the column generation algorithm to solve the linear relaxation of [**MP**]. The [**RMP**] is as follows:

$$
\begin{aligned}
\textbf{[RMP]} \quad \text{minimize} \quad & \lambda \sum_{j \in N} \sum_{p \in \bar{P}_j} c_{jp}\delta_{jp} + (1-\lambda)(w_h - w_l) + \sum_{j \in N} M\mu_j && (5.9)\\
\text{subject to} \quad & \sum_{p \in \bar{P}_j} \delta_{jp} + \mu_j = 1 \quad \forall j \in N && (5.10)\\
& \sum_{j \in N} \sum_{p \in \bar{P}_j} \alpha_{jpkt}\delta_{jp} \leq Q^1_{kt} \quad \forall k \in K, \forall t \in T && (5.11)\\
& \sum_{j \in N} \sum_{p \in \bar{P}_j} \beta_{jpst}\delta_{jp} \leq Q^2_{st} \quad \forall s \in S, \forall t \in T && (5.12)\\
& w_h - \sum_{s \in S} \sum_{j \in N} \sum_{p \in \bar{P}_j} \beta_{jpst}\delta_{jp} \geq 0 \quad \forall t \in T && (5.13)\\
& w_l - \sum_{s \in S} \sum_{j \in N} \sum_{p \in \bar{P}_j} \beta_{jpst}\delta_{jp} \leq 0 \quad \forall t \in T && (5.14)\\
& 0 \leq \delta_{jp} \leq 1 \quad \forall j \in N, \forall p \in \bar{P}_j && (5.15)\\
& w_h, w_l \geq 0 && (5.16)
\end{aligned}
$$

The [**RMP**] is similar to the original linear program [**MP**]. However, the sets of service plans are set to be empty initially, and are represented as $\bar{P}_j, \forall j \in N$ $(\bar{P}_j \in P_j)$. In order to make the [**RMP**] feasible, we further introduce artificial decision variables $\mu_j, \forall j \in N$ which are penalized by a large constant M in the objective function. Besides, the decision variables δ_{jp} are relaxed to be continuous.

5.3.2 Pricing sub-problem

Let $\pi_j^1, \pi_{kt}^2, \pi_{st}^3, \pi_t^4, \pi_t^5$ be the dual variables associated with Constraints (5.10)-(5.14) in **[RMP]**, respectively. Then, the reduced cost corresponding to feeder $j \in N$ is:

$$\tilde{c}_{jp} = \lambda c_{jp} - \pi_j^1 - \sum_{k \in K} \sum_{t \in T} \alpha_{jpkt} \pi_{kt}^2 - \sum_{s \in S} \sum_{t \in T} \beta_{jpst} \pi_{st}^3 + \sum_{s \in S} \sum_{t \in T} \beta_{jpst} \pi_t^4 + \sum_{s \in S} \sum_{t \in T} \beta_{jpst} \pi_t^5 \tag{5.17}$$

N pricing sub-problems are then activated each of which is responsible for identifying service plans with negative $\tilde{c}_{ip}$ for each feeder. We further introduce the following decision variables for pricing sub-problem $j \in N$:

- x_{ik}: $\in \{0, 1\}$, 1 if flow $i \in I$ is put in yard section $k \in K$ for temporary storage, and 0 otherwise
- y_t : $\in \{0, 1\}$, 1 if feeder j is serviced at working shift $t \in T$, and 0 otherwise
- z_s: $\in \{0, 1\}$, 1 if feeder j is serviced at berth $s \in S$, and 0 otherwise
- u_{it}: $\in \{0, 1\}$, 1 if flow $i \in I$ is within the terminal (unloading, in storage and loading statuses) at working shift $t \in T$, and 0 otherwise
- v_i: $\in \{0, 1\}$, 1 if the departure working shift of flow $i \in I$ is later than its arrival working shift in one planning horizon, and 0 otherwise

Hence, the parameters $c_{jp}, \alpha_{jpkt}, \beta_{jpst}$ can be expressed as:

$$c_{jp} = \sum_{i \in I} \sum_{k \in K} \left(\sum_{s \in S} l_{ks} \theta_{ij}^N x_{ik} z_s + \sum_{s \in S | s = d_i} l_{ks} \theta_{ij}^N x_{ik} z_s \right) \tag{5.18}$$

$$\alpha_{jpkt} = \sum_{i \in I} \theta_{ij}^N q_i x_{ik} u_t \tag{5.19}$$

$$\beta_{jpst} = \sum_{i \in I | \theta_{ij}^N = 1, r_i = t, d_i = s} q_i + \sum_{i \in I} \theta_{ij}^N q_i y_t z_s \tag{5.20}$$

And the reduced cost of feeder vessel $j \in N$ can be updated as:

$$\begin{aligned}
\tilde{c}_{jp} = \quad & \lambda c_{jp} - \pi^1_j - \sum_{k\in K}\sum_{t\in T}\alpha_{jpkt}\pi^2_{kt} + \sum_{s\in S}\sum_{t\in T}\beta_{jpst}(-\pi^3_{st} + \pi^4_t + \pi^5_t) \\
= \quad & \lambda \sum_{i\in I}\sum_{k\in K}\left(\sum_{s\in S} l_{ks}\theta^N_{ij}x_{ik}z_s + \sum_{s\in S|s=d_i} l_{ks}\theta^N_{ij}x_{ik}z_s\right) - \pi^1_j - \sum_{k\in K}\sum_{t\in T}\pi^2_{kt}\sum_{i\in I}\theta^N_{ij}q_i x_{ik}u_t \\
& + \sum_{s\in S}\sum_{t\in T}(-\pi^3_{st} + \pi^4_t + \pi^5_t)\left(\sum_{i\in I|\theta^N_{ij}=1,r_i=t,d_i=s} q_i + \sum_{i\in I}\theta^N_{ij}q_i y_t z_s\right) \qquad (5.21)
\end{aligned}$$

The pricing sub-problem for feeder $i \in N$ **[PSP(i)]**could be formulated as follows:

$$\textbf{[PSP(i)]} \quad \text{minimize} \quad \tilde{c}_{jp} \tag{5.22}$$

$$\text{subject to} \quad \sum_{s\in S} z_s = 1 \tag{5.23}$$

$$\sum_{t\in T} y_t = 1 \tag{5.24}$$

$$\sum_{t\in T|\underline{T}_j \leq t \leq \overline{T}_j} y_t = 1 \quad \text{if } \underline{T}_j \leq \overline{T}_j \tag{5.25}$$

$$\sum_{t\in T|t\leq \overline{T}_j} y_t + \sum_{t\in T|t\geq \underline{T}_j} y_t = 1 \quad \text{if } \underline{T}_j > \overline{T}_j \tag{5.26}$$

$$\sum_{k\in K} x_{ik} = 1 \quad \forall i \in I \tag{5.27}$$

$$0 \leq v_i + \left(r_i - \sum_{t\in T} t y_t\right)/\bar{t} < 1 \quad \forall i \in I_1|\theta^N_{ij} = 1 \tag{5.28}$$

$$1 - r_i \leq t\left(u_{it} + v_i - 1\right) \leq \sum_{t'\in T} t' y_{t'} \quad \forall i \in I_1, \forall t \in T|\theta^N_{ij} = 1 \tag{5.29}$$

$$\sum_{t'\in T} t' y_{t'} - \bar{t} \leq (\bar{t} + 1 - t)(u_{it} + v_i - 1) \leq \bar{t} + 1 - r_i \quad \forall i \in I_1, \forall t \in T|\theta^N_{ij} = 1 \tag{5.30}$$

$$\sum_{t\in T}(u_{it} + v_i - 1) = \sum_{t\in T} t y_t - r_i + 1 \quad \forall i \in I_1|\theta^N_{ij} = 1 \tag{5.31}$$

$$0 \leq v_i + \left(\sum_{t\in T} t y_t - r_i\right)/\bar{t} < 1 \quad \forall i \in I_2|\theta^N_{ij} = 1 \tag{5.32}$$

$$1 - \sum_{t' \in T} t' y_{t'} \leq t(u_{it} + v_i - 1) \leq r_i \quad \forall i \in I_2, \forall t \in T | \theta_{ij}^N = 1 \tag{5.33}$$

$$r_i - \bar{t} \leq (\bar{t} + 1 - t)(u_{it} + v_i - 1) \leq \bar{t} + 1 - \sum_{t' \in T} t' y_{t'} \quad \forall i \in I_2, \forall t \in T | \theta_{ij}^N = 1 \tag{5.34}$$

$$\sum_{t \in T} (u_{it} + v_i - 1) = r_i - \sum_{t \in T} t y_t + 1 \quad \forall i \in I_2 | \theta_{ij}^N = 1 \tag{5.35}$$

$$x_{ik}, y_t, z_s, u_{it}, v_i \in \{0, 1\} \quad \forall i \in I, \forall k \in K, \forall s \in S, \forall t \in T \tag{5.36}$$

The objective function (5.22) minimizes the reduced cost $\tilde{c}_{jp}$ of feeder $j \in N$. If the optimal objective function value is found to be negative, the optimal decision variables translate into a new service plan for feeder $j \in N$ and then added into $\bar{P}_j$. Otherwise, $\bar{P}_j$ is not updated. Constraints (5.24)-(5.36) correspond to (4.2)-(4.14) of the compact formulation **[P]** in Chapter 4, except that the decision variable **y** and **z** are no longer defined with respect to a specific feeder since all pricing sub-problems share the same constraint set.

5.3.3 Obtaining Integer solution

The column generation algorithm only solves the linear relaxation of the original formulation **[MP]** and does not guarantee to find integer solutions. Therefore, we devise the following two approaches to find good integer solutions. Note that the following two column generation based approaches are motivated by those heuristics designed by Moccia et al. (2009).

5.3.3.1 Column generation heuristic 1: COL1

This approach firstly performs the column generation procedure to generate a set of columns. The restricted master problem and pricing sub-problems are solved to optimum by a commercial solver. Then, it conducts a post-branch & bound procedure based on the set of columns generated.

5.3.3.2 Column generation heuristic 2: COL2

Similar to COL1, COL2 also performs the column generation firstly and apply the post-branch & bound procedure. But we design an extended column generation procedure to expand the set of columns in the hope of finding better integer solutions by providing more columns. Besides, the pricing sub-problems are solved by a local search heuristic instead of solving it to optimum. We remark that it is not necessary to return optimal solutions of the pricing sub-problems, any solutions with negative reduced cost will do. Therefore, we employ a local search heuristic method to find near-optimal solutions in a more efficient way.

The extended column generation procedure is illustrated as follows: At the end of the original column generation, the extended procedure firstly finds those saturated berths and yard sections by identifying those constraints with negative dual values π^2_{kt}, π^3_{st}. The right-hand-side of those capacity constraints are reduced by $Q^1_{kt}/4$ and $Q^2_{st}/4$ accordingly. With the updated constraints, the column generation is activated again to generate more columns. By reducing the capacity of those saturated berths and yard sections, we are forcing the generation of new service plans that avoid these berths and yard sections. Such an extended column generation procedure can be applied for more than once if necessary. In this study, it is activated twice.

The local search heuristic for finding near-optimal solutions for the pricing sub-problem is illustrated as follows: The solution is coded as a string with the information of assigned berth and service time for the feeder as well as the assigned yard sections for associated transshipment flows. Figure 5.1 presents an illustrative example for a feeder vessel which has four transshipment flows. The initial solution is generated by randomly assigning the feeder over the berths and the planning horizon, and also randomly assigning the associated transshipment flows over the entire storage yard. The neighborhood solutions are generated by randomly picking an element of the current solution and then assigning it with an alternative value. With a given solution coding, the decision variables of the pricing sub-problem can be easily derived and then we are able to obtain the fitness of the solution by calculating (5.21). For each local search iteration, LC_1 neighborhood solutions are generated to sample the gradient (i.e., the best of the LC_1 neighborhood solutions is used to updated the current solution), and the best solution is updated by the gradient solution if its fitness is superior to that of the best solution. The local search procedure terminates when the following condition is met: the best solution does not get improved for TC_2 consecutive iterations or the maximum number of the local search procedure LC_3 is reached. In case the fitness of the best solution is not negative, the local search procedure is employed once again unless the total times of local search reaches the upper limit LC_4.

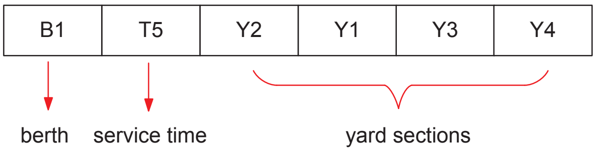


Figure 5.1: An illustrative example of the solution representation

5.4 Computational Experiments

In this section, we conduct computational experiments to calibrate the parameters of COL2, and also to compare the performance of the developed column generation based heuristics COL1 and COL2 with that of the memetic heuristic in Chapter 4. The test instances are the same as described in Chapter 4. COL1 and COL2 are coded in C++ and CPLEX 12.1 is used as the MIP solver. All the computational experiments are conducted on a PC with 3 GHz CPU and 4 GB RAM.

5.4.1 Parameter setting

Due to the randomness of the local search procedure, COL2 runs five times and the average objective function values are reported. The post-branch & bound procedure employed by COL1 and COL2 is given a time budget of 600 seconds for the sake of efficiency. The truncated solutions are returned once the computational time reaches the time limit. The parameters of COL2 are set as follows:

- Gradient sample size: $LC_1 = 10$;
- Maximum number of non-improving iterations: $LC_2 = 10$;
- Maximum number of iterations of a local search procedure: $LC_3 = 50$;
- Maximum times of local search: $LC_4 = 5$.

Figure 5.2 shows the sensitivity of the objective function value and computational time to parameter TC_1 for test instances Set3-I1 and Set4-I1. TC_1 varies within the set {4, 6, 8, 10, 12, 14, 16, 18} while other parameters are fixed as listed above. As can be seen, increasing the sampling size for the neighborhood search from 4 to 10 lowers the objective function value at the cost of longer computational efforts. Larger values than 10 only yields marginal benefit. Note that the trend of the computational time of Set4-I1 is not so obvious. This is mainly because for larger scale test instances, the computational time of the post-branch &bound procedure is not so stable. However, the computational time of the column generation procedure shows a similar trend to that of Set3-I1 in Figure 5.2.

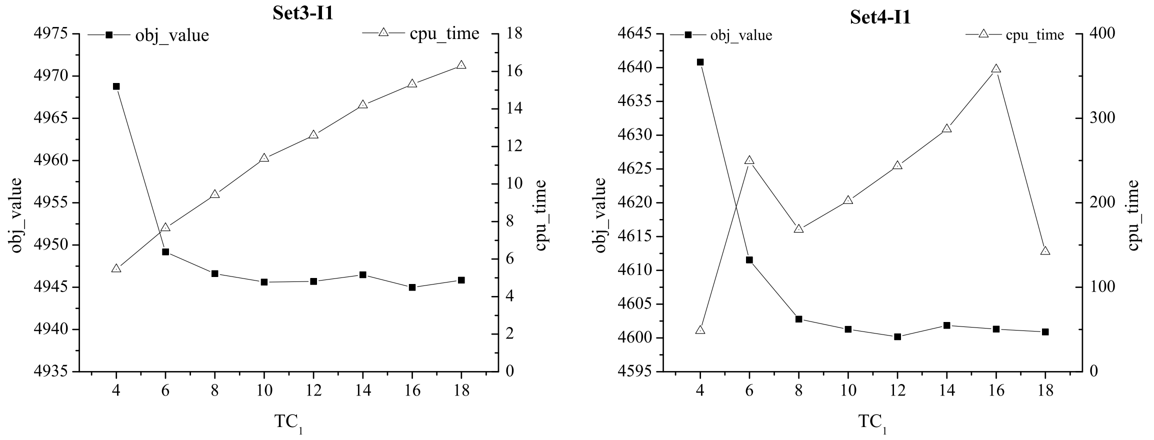


Figure 5.2: Sensitivity analysis of parameter TC_1.

Figure 5.3 shows the sensitivity of the objective function value and computational time to parameter TC_2 for test instances Set3-I1 and Set4-I1. TC_2 varies within the set {4, 6, 8, 10, 12, 14, 16, 18} while other parameters are fixed as listed above. A tradeoff value balancing the solution quality and computational time is obtained at $TC_2 = 10$.

Figure 5.4 shows the sensitivity of the objective function value and computational time to parameter TC_3 for test instances Set3-I1 and Set4-I1. TC_3 varies within the set {20, 30, 40, 50, 60, 70, 80, 90} while other parameters are fixed as listed above. A tradeoff value balancing the solution quality and computational time is obtained at $TC_3 = 50$.

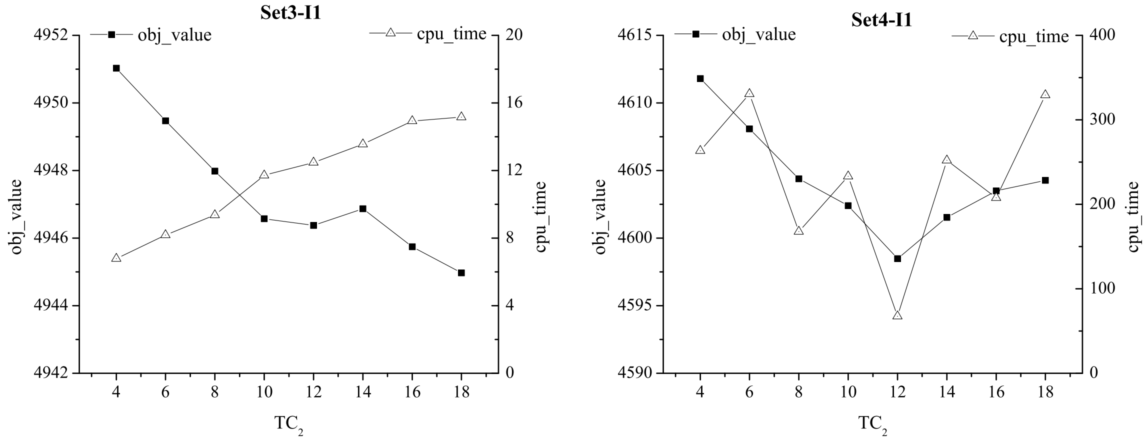


Figure 5.3: Sensitivity analysis of parameter TC_2.

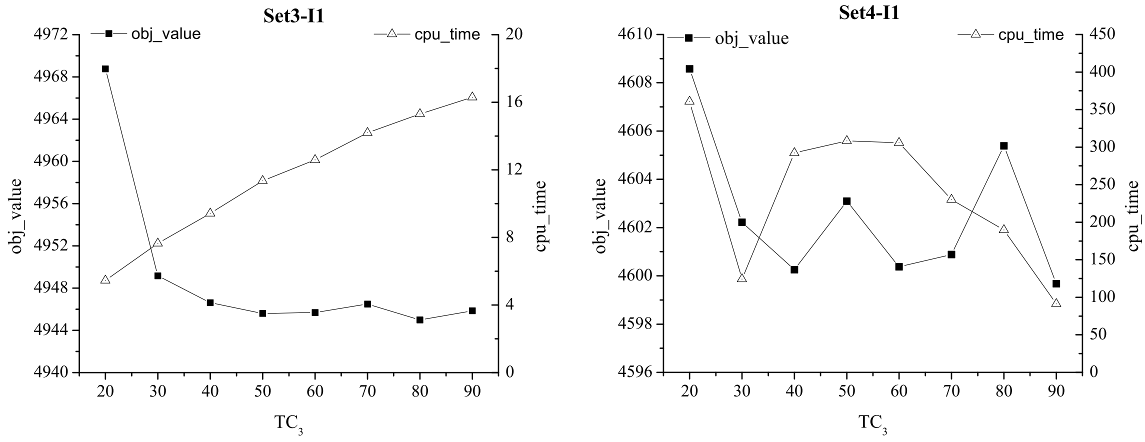


Figure 5.4: Sensitivity analysis of parameter TC_3.

Figure 5.5 shows the sensitivity of the objective function value and computational time to parameter TC_4 for test instances Set3-I1 and Set4-I1. TC_4 varies within the set $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ while other parameters are fixed as listed above. A tradeoff value balancing the solution quality and computational time is obtained at $TC_4 = 5$.

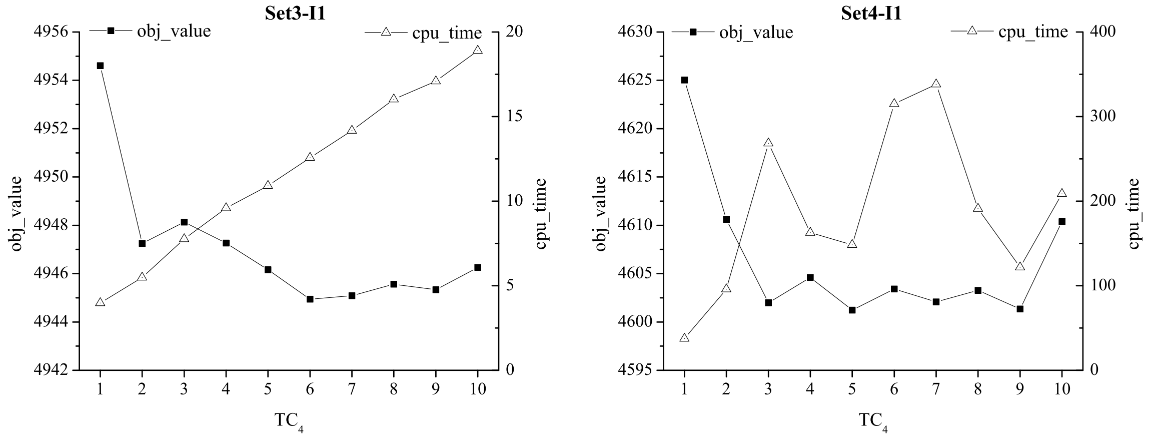


Figure 5.5: Sensitivity analysis of parameter TC_4.

5.4.2 Results

We compare the results of the column generation based heuristics COL1 and COL2 with those of CPLEX and memetic heuristic reported in Chapter 4. Tables 5.1-5.4 present the results of the four methods for instance Sets 1-4. CPLEX is treated as a benchmark and the Memetic heuristic, COL1 and COL2 are assessed accordingly. The gaps of the three methods with respect to CPLEX are provided in the last three columns. As can be seen, the instances of Set 1 can be easily solved by CPLEX. However instances of Sets 2-4 cannot be solved to optimum within 360 minutes by CPLEX and hence the truncated solutions are reported. The other three methods are able to produce near-optimal solutions more efficiently. Regarding solution quality, COL2 is comparable to the Memetic heuristic while COL1 does not produce as good solutions as COL2. This confirms the necessity of the extended column generation procedure introduced in COL2. Regarding efficiency, COL2 reduces computational time significantly thanks to the application of local search heuristic instead of solving the pricing sub-problems exactly by CPLEX. Besides, COL2 is roughly one order of magnitude more efficient than the Memetic heuristic. In summary, the developed column generation based heuristic COL2 achieves the same solution quality as Memetic heuristic while significantly reducing the computational efforts.

Table 5.1: Computational results of data Set 1.

Instance	CPLEX		Memetic		COL1		COL2			Gap		
	Result (1)	Time (min)	Result (2)	Time (min)	Result (3)	Time (min)	Result (4)	Stdev	Time (min)	Gap1 (%)	Gap2 (%)	Gap3 (%)
Set1-I1	1562.0	0.5	1562.6	0.3	1567.6	0.20	1567.5	1.3	0.03	0.04	0.35	0.35
Set1-I2	2246.7	2.9	2246.9	0.4	2290.6	0.05	2254.4	3.1	0.02	0.01	1.95	0.34
Set1-I3	2038.6	0.3	2040.0	0.3	2285.3	0.07	2046.9	5.5	0.02	0.07	12.10	0.41
Set1-I4	1936.3	0.5	1939.0	0.4	1986.3	0.19	1975.0	19.5	0.03	0.14	2.58	2.00
Set1-I5	1832.5	0.5	1832.6	0.3	1935.0	0.05	1837.5	5.0	0.02	0.01	5.59	0.27
Set1-I6	2276.5	0.2	2276.6	0.3	2435.4	0.09	2300.3	19.9	0.02	0.01	6.98	1.05
Set1-I7	1898.6	0.2	1898.9	0.3	1907.0	0.06	1907.1	6.2	0.02	0.02	0.44	0.45
Set1-I8	2083.2	0.2	2083.2	0.3	2119.2	0.07	2085.3	2.6	0.02	0.00	1.73	0.10
Set1-I9	1198.5	1.0	1199.6	0.3	1201.6	0.09	1206.3	7.0	0.03	0.09	0.26	0.65
Set1-I10	1857.5	0.5	1859.0	0.4	1862.2	0.09	1865.8	12.1	0.03	0.08	0.25	0.44
Average										0.04	3.23	0.61

Gap1 = [(2)-(1)]/(1)×100%
Gap2 = [(3)-(1)]/(1)×100%
Gap3 = [(4)-(1)]/(1)×100%

Table 5.2: Computational results of data Set 2.

Instance	CPLEX		Memetic		COL1		COL2			Gap		
	Result (1)	Time (min)	Result (2)	Time (min)	Result (3)	Time (min)	Result (4)	Stdev	Time (min)	Gap1 (%)	Gap2 (%)	Gap3 (%)
Set2-I1	2086.6	360.0	2115.8	1.8	2210.1	2.3	2131.4	31.5	0.1	1.40	5.92	2.15
Set2-I2	2345.8	360.0	2337.0	1.7	2344.8	0.5	2345.7	11.4	0.1	-0.37	-0.04	0.00
Set2-I3	2145.9	360.0	2156.2	2.9	2181.6	3.7	2157.4	8.8	2.4	0.48	1.66	0.54
Set2-I4	2179.4	360.0	2180.2	3.4	2254.8	0.9	2203.0	19.4	0.1	0.03	3.46	1.08
Set2-I5	2679.6	360.0	2708.2	1.4	2715.7	1.2	2693.6	15.4	0.1	1.07	1.35	0.53
Set2-I6	2030.8	360.0	2024.4	1.9	2105.4	0.9	2025.9	9.4	0.1	-0.31	3.68	-0.24
Set2-I7	2767.3	360.0	2771.5	2.0	2923.6	1.4	2793.3	24.2	0.1	0.15	5.65	0.94
Set2-I8	2646.9	360.0	2651.0	2.0	3349.0	1.0	2657.7	39.5	0.1	0.15	26.52	0.41
Set2-I9	1799.5	360.0	1787.7	2.4	1815.2	2.7	1803.1	8.9	0.1	-0.65	0.87	0.20
Set2-I10	2906.7	360.0	2904.7	2.0	3457.8	2.3	2912.4	9.9	0.2	-0.07	18.96	0.20
Average										0.19	6.80	0.58

Gap1 = [(2)-(1)]/(1)×100%
Gap2 = [(3)-(1)]/(1)×100%
Gap3 = [(4)-(1)]/(1)×100%

Table 5.3: Computational results of data Set 3.

Instance	CPLEX		Memetic		COL1		COL2			Gap		
	Result (1)	Time (min)	Result (2)	Time (min)	Result (3)	Time (min)	Result (4)	Stdev	Time (min)	Gap1 (%)	Gap2 (%)	Gap3 (%)
Set3-I1	4949.5	360.0	4946.0	4.2	4949.1	12.3	4945.7	1.1	0.2	-0.07	-0.01	-0.08
Set3-I2	4301.0	360.0	4264.0	4.7	4953.3	7.9	4254.5	6.4	0.3	-0.86	15.16	-1.08
Set3-I3	2929.3	360.0	2885.5	6.6	3120.1	24.1	2945.0	44.7	0.4	-1.49	6.52	0.54
Set3-I4	3887.9	360.0	3829.0	6.2	3891.8	10.7	3838.0	24.4	0.2	-1.51	0.10	-1.28
Set3-I5	3646.4	360.0	3601.8	5.3	3769.8	7.8	3618.2	16.9	0.2	-1.22	3.38	-0.77
Set3-I6	4155.1	360.0	4153.8	5.4	4373.0	20.4	4186.9	35.4	0.2	-0.03	5.25	0.77
Set3-I7	3300.6	360.0	3226.3	7.0	3706.0	15.0	3234.5	8.5	1.3	-2.25	12.28	-2.00
Set3-I8	3915.6	360.0	3903.5	5.2	3904.6	13.1	3936.7	28.9	0.2	-0.31	-0.28	0.54
Set3-I9	4155.8	360.0	4130.4	8.5	4129.7	55.6	4119.0	8.6	9.8	-0.61	-0.63	-0.89
Set3-I10	3170.6	360.0	3148.2	6.0	3137.7	32.8	3154.6	25.7	0.3	-0.71	-1.04	-0.50
Average										-0.91	4.07	-0.48

Gap1 = [(2)-(1)]/(1)×100%
Gap2 = [(3)-(1)]/(1)×100%
Gap3 = [(4)-(1)]/(1)×100%

Table 5.4: Computational results of data Set 4.

Instance	CPLEX		Memetic		COL1		COL2			Gap		
	Result (1)	Time (min)	Result (2)	Time (min)	Result (3)	Time (min)	Result (4)	Stdev	Time (min)	Gap1 (%)	Gap2 (%)	Gap3 (%)
Set4-I1	4652.0	360.0	4610.14	32.8	4662.0	26.8	4602.2	4.2	3.7	-0.90	0.21	-1.07
Set4-I2	6602.4	360.0	6197.29	34.4	6304.4	29.5	6176.6	10.9	10.4	-6.14	-4.51	-6.45
Set4-I3	4301.5	360.0	3817.80	40.9	3870.8	47.0	3819.2	33.1	10.4	-11.24	-10.01	-11.21
Set4-I4	6058.6	360.0	5483.31	33.4	5640.3	22.2	5476.5	30.5	8.7	-9.50	-6.90	-9.61
Set4-I5	4858.3	360.0	4467.34	35.0	4541.8	35.1	4474.4	41.3	3.2	-8.05	-6.51	-7.90
Set4-I6	4429.2	360.0	3894.04	32.5	4126.5	16.5	3882.2	10.3	10.4	-12.08	-6.83	-12.35
Set4-I7	6100.5	360.0	5855.01	46.0	6139.0	30.0	5842.9	8.2	10.4	-4.02	0.63	-4.22
Set4-I8	6234.1	360.0	5738.82	42.9	5720.4	40.5	5713.5	5.8	10.0	-7.94	-8.24	-8.35
Set4-I9	6044.7	360.0	5977.07	26.6	6282.1	14.6	5965.4	8.9	5.7	-1.12	3.93	-1.31
Set4-I10	6052.1	360.0	5297.55	37.3	5303.2	40.0	5287.5	12.4	10.4	-12.47	-12.37	-12.63
Average										-7.35	-5.06	-7.51

Gap1 = [(2)-(1)]/(1)×100%
Gap2 = [(3)-(1)]/(1)×100%
Gap3 = [(4)-(1)]/(1)×100%

5.5 Summary

In this chapter, we have developed a column generation based heuristic for the Feeder Vessel Management Problem. The problem is reformulated as a set covering model and solved by column generation. Computational tests have been run to compare the performance of the column generation based method with Memetic heuristic developed in Chapter 4. The results have shown that the developed column generation based method achieves similar solution quality to Memetic heuristic, but reduces the computational time by an order of magnitude. For future research, we are interested in embedding the column generation method in a branch-and-bound procedure in order to obtain optimal solutions.

Chapter 6

Storage Yard Management with Integrated Consideration of Space Allocation and Crane Deployment

6.1 Introduction

With the rapid growth of containerized maritime shipping industry over the past decades, major seaport container terminals have been striving to maintain and improve their competitiveness. As a vital node of the global multi-modal transportation network, a container terminal provides an interface between different transportation modes where handling productivity and efficiency should be well ensured. Meanwhile, container terminals also function as a warehouse for container temporary storage. A container terminal can be in general divided into two parts: quayside and landside. The quayside is open to container vessels directly and requires fast container loading and discharging operations, while the landside offers a temporary storage area for containers and collects/delivers containers from/to inland customers. In order to achieve a seamless container flow through the terminal, various handling equipment (e.g., quay cranes, yard cranes, trucks) and decision support systems are employed for daily planning and scheduling of terminal operations.

For container transshipment port such as the Port of Singapore, a common issue that occurs during the growing period is the problem of land scarcity which makes a highly concentrated storage situation within the storage yard. The storage yard management is complex in practice and involves two inter-related decision problems: (1) storage space allocation problem which is to determine the storage locations for incoming containers, and (2) yard crane (YC) deployment problem which is to decide the number of YCs working in each block and their movements between blocks. Yard planners usually solve the two decision problems sequentially in such a way that space allocation is determined firstly and the resulting workload is used to deploy and route YC accordingly. However, this planning method fails in considering the impact of storage allocation decision on the performance of YC operation, since the storage allocation plan determines the distribution of YC workload over the entire yard and affects YC deployment decisions. In some cases, the YC workload concentrates in certain blocks due to inappropriate space allocation, and this may incur workload delay since there is a limit of YCs that can work simultaneously within a block. Therefore, it is necessary to integrate the two decision problems as a system so that the space allocation and YC deployment can be properly coordinated.

Another emerging issue is the traffic congestion problem in container terminals due to the increasing container volume (Vacca et al., 2010). Traffic congestion often lies at bottlenecks of container terminals such as the quayside where heavy loading and discharging operations take place, storage yard where YC activities (container grounding and retrieval operations) are concentrated, and terminal gates. As one of the congestion bottlenecks, the storage yard calls for well planned YC activity distribution so that the traffic congestion can be avoided. Therefore, the container traffic concern should be taken into account when conducting storage yard management.

This chapter aims to improve the storage yard management by integrating the two inter-related decision problems, space allocation and YC deployment. The traffic congestion issue at the storage yard is also taken into account so that the decisions of the yard planning are made with full awareness of the container traffic. We focus on the planning of storage yard operations on the daily basis. The contribution of this chapter includes the following:

- The development of the daily storage yard management problem with integrated consideration of space allocation and YC deployment, together with the container traffic congestion consideration;
- An optimization model that is developed for the daily storage yard management;
- A divide-and-conquer strategy based heuristic approach that is able to find near-optimal solutions in a very efficient manner.

The remainder of the chapter is organized as follows. Section 6.2 reviews relevant papers in the literature and Section 6.3 presents the detailed problem description as well as mathematical formulation. The heuristic approach is developed in Section 6.4 followed by computational experiments in Section 6.5. Finally, Section 6.6 draws the conclusion.

6.2 Literature Review

The management of container terminal operations is rich in terms of the application of operations research and optimization techniques. Plenty of efforts have been made by researchers in the literature. In this section, we review some of them related with the storage yard management including space allocation and YC deployment. For a comprehensive review, readers may refer to Steenken et al. (2004), Stahlbock and Voss (2008), and Monaco et al. (2009).

Space allocation can be analyzed at various levels according to the storage space unit considered: yard section (Lee et al., 2012; Lee and Jin, 2013), yard block (Zhang et al., 2003; Moccia et al., 2009), yard sub-block (Han et al., 2008; Zhen et al., 2011), yard bay (Ng et al., 2010; Lee et al., 2011) and individual slot (Kim et al., 2000; Kang et al., 2006). Lee et al. (2012) and Lee and Jin (2013) studied the tactical space allocation problem for transshipment containers, and the storage locations are identified to yard sections each of which consists of a few neighboring yard blocks. The objective is to minimize the total travel distance/cost between the quayside and the storage yard. Zhang et al. (2003) developed a hierarchical approach for the space allocation problem at the block level with the workload balancing consideration. Moccia et al. (2009) treated the space allocation problem at the block level as a *Dynamic Generalized*

Assignment Problem in which containers are allowed to be relocated from one block to another in the during-of-stay. The sub-block space allocation is studied in Han et al. (2008) and Zhen et al. (2011), and is referred as yard template design. Ng et al. (2010) tackled the problem of assigning yard bay resources to vessel services with a cyclically calling pattern, while Lee et al. (2011) studied the integrated problem of bay allocation and yard crane scheduling. Kim et al. (2000) and Kang et al. (2006) focused on the slot assignment within a yard stack for individual export containers with the objective of minimizing container re-handling.

With a determined storage allocation plan, the information of grounding and retrieval activities in all blocks can be available for planning the YC deployment. In the previous literature, two strategies of YC deployment can be identified. One strategy focuses on the minimization of YC workload delay at the end of each time period (Zhang et al., 2002; Cheung et al., 2002). This strategy is applicable to those container terminals where YC is a scarce resource and workload delay is allowed. Zhang et al. (2002) proposed a mixed integer programming model to allocate *Rubber Tired Gantry Cranes* over the yard based on the forecasted workload of each block. Times and routes of crane movement among blocks are determined in such a way that the total delayed workload is minimized. Cheung et al. (2002) also studied the inter-block crane deployment problem with the objective of minimizing the total unfinished workload at the end of each time period. However, they used a much shorter time interval which is a sub-multiple of the time it takes to conduct an inter-block movement. Guo and Huang (2012) proposed a similar strategy that dynamically partitions space and time for YCs with the objective of minimizing average vehicle job waiting time. Another strategy of YC deployment can be found in Lee et al. (2006) and Han et al. (2008). Instead of minimizing the workload delay, the two studies focused on the minimization of total number of deployed active YCs with the requirement of finishing all the workload. This YC deployment strategy is applicable to container terminals where YC is an excess resource and workload delay is strictly prohibited. In such kind of terminals, yard planners need to determine which YCs should be activated to conduct activities while the rest should hibernate so that the total operating cost can be minimized.

The need for integration of the space allocation and YC deployment is well regarded, and some researchers have made efforts in tackling the integrated problem (Kim and Kim, 2002;

Han et al., 2008; Won et al., 2012). Kim and Kim (2002) proposed two cost models to decide the optimal amount of storage space and number of YCs for handling import containers under various circumstances. The space allocation problem considered by the authors concerns the amount of storage space and stacking height. However, the decision of storage location is not included. Han et al. (2008) proposed a storage strategy called high-low workload balancing protocol for a transshipment hub so as to reduce potential traffic congestion of yard trucks. The objective is to minimize the YC operating cost by designing a sub-block allocation plan (referred as yard template in the chapter) and determining the number of containers that are allocated to sub-blocks at each time period. However, the YC inter-block movement is not considered. Recently, Won et al. (2012) addressed a yard-planning system considering the assignment of yard spaces and the YCs' workload distribution. Two YC operating systems were modeled and compared, one allowing YCs move between blocks within the same row and the other without YC movement.

This storage yard management problem studied in this chapter is a follow-up work of Lee et al. (2012) and Lee and Jin (2013). With the pre-determined assignment decision of yard sections from the tactical planning level, this chapter deals with more detailed daily sub-block allocation problem for incoming containers at the operational level. Meanwhile, in order to achieve a better efficiency for the storage yard management the space allocation decision is integrated with YC deployment. Similar to Han et al. (2008), this study simultaneously consider the space allocation and YC operation in the storage yard. However, we try to further improve the efficiency of space utilization by dynamically assigning sub-blocks to incoming containers, while the yard template is static in Han et al. (2008) (sub-blocks are reserved exclusively and are not shared by different container groups). Besides, we specifically include the YC inter-block movement decision in addition to the YC deployment decision in each block in order to comprehensively reflect the overall YC related cost including inter-block movement cost and operating cost. The workload-based yard-planning problem introduced in Won et al. (2012) is close to ours as both of us deal with the assignment of yard spaces with the consideration of YC workload. However, we model the YC operations in a more detailed level by introducing YC deployment profiles and inter-block movement.

6.3 Mathematical Model

In this section, we provide a detailed description of the storage yard management problem and formulate it as an integer linear programming model.

6.3.1 Problem description

This chapter tackles the daily storage yard management problem in container terminals with particular integrated optimization of space allocation and YC deployment decisions. More specifically, with the information of incoming and outgoing containers in the near future, this study on the one hand assigns available storage space to incoming containers, and on the other hand deploys and routes YCs over the storage yard to perform container grounding and retrieval activities. The storage space allocation and YC deployment are two typical decision problems involved in the daily storage yard management, and are closely inter-related since the space allocation decision would determine the workload distribution over the storage yard for YCs to handle. It is our particular motivation to model the integration of the two decision problems to achieve a better efficiency for storage yard management. The modeling concepts of the two decision problems are illustrated as follows.

6.3.1.1 Storage space allocation problem

For a storage yard, all containers can be categorized into two types: *incoming container groups* that arrive in the terminal in the near future, and *outgoing container groups* that are already stored in the yard waiting to be retrieved. A container group is a collection of containers sharing the same attributes, such as arrival vessel, departure vessel and inland customer. It is a common practice that containers belonging to the same groups are stored in the yard in an aggregated manner. In this study, we partition the entire storage yard into a set of yard sections each of which consists of four yard blocks and two traffic accessing lanes as shown in Figure 6.1. Each yard block is further divided into five sub-blocks as is the case in the Port of Singapore. The storage space allocation problem is studied in the sub-block level, i.e., the unit for space allocation is one sub-block. It is assumed that the storage yard section for incoming container groups has been determined in the upper planning level, and thus the remaining task is to determine the detailed sub-blocks within the given yard section for storage. For example, for an incoming container group that has been assigned yard Section 1 and requires three sub-blocks for storage, we may select any three available sub-blocks from M01-M20.

Constraints that should be considered when allocating storage space include:

- Each sub-block should at most be assigned to one container group for any time period;
- The traffic volume of the two accessing lanes respects the capacity for any time period;
- The grounding and retrieval activities do not exceed YC handling capacity.

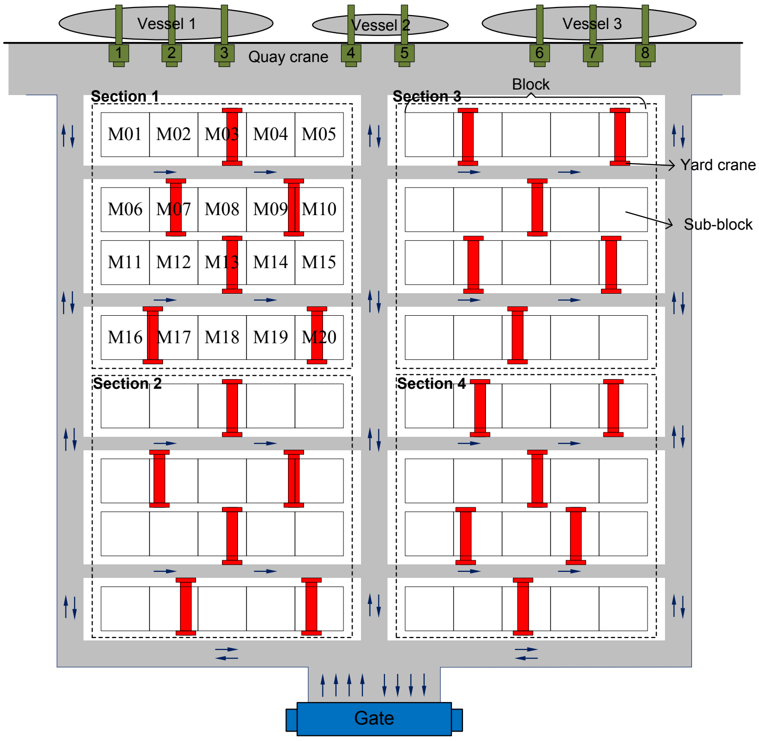


Figure 6.1: An illustrative layout of a seaport container terminal.

6.3.1.2 Yard crane deployment problem

The YC deployment problem is to deploy and route YCs over the entire storage yard according to the workload distribution. In this study, we assume sufficient YCs are available to perform the container grounding and retrieval activities and there is no workload delayed at the end of each time period. Under this assumption, we determine the YCs that should be activated for operations at each time period while the rest should hibernate, and at the same time route the YC inter-block movement in such a way that the overall operating cost is minimized. In order to model the YC activities, we introduce the concept of YC deployment profiles to define the service area (sub-blocks) covered by one YC. Figure 6.2 shows the types of YC deployment profiles and their associated handling capacities. As can be seen, for each yard block five types of deployment profiles are defined with respect to the size of service area (number of sub-blocks). The handling capacity (measured in the maximum number of moves that can be fulfilled within a time period) of the YC deployment profiles decreases as the size of service areas increases, since large service area requires more frequent YC movement. Note that the exact YC capacity may vary for different container terminals and should be calibrated from historical data. As one yard block consists of five sub-blocks, there are 15 YC deployment profiles in total defined for a block (5 Type A profiles, 4 Type B profiles, 3 Type C profiles, 2 Type D profiles and 1 Type E profile). We remark the YC deployment profile concept defined in this chapter is similar to the quay crane (QC) assignment profile introduced in Giallombardo et al. (2010). The QC assignment profile specifies the allocated number of QCs over the berthing period for every vessel while our YC deployment profile corresponds to the service area for a certain YC during a time period.

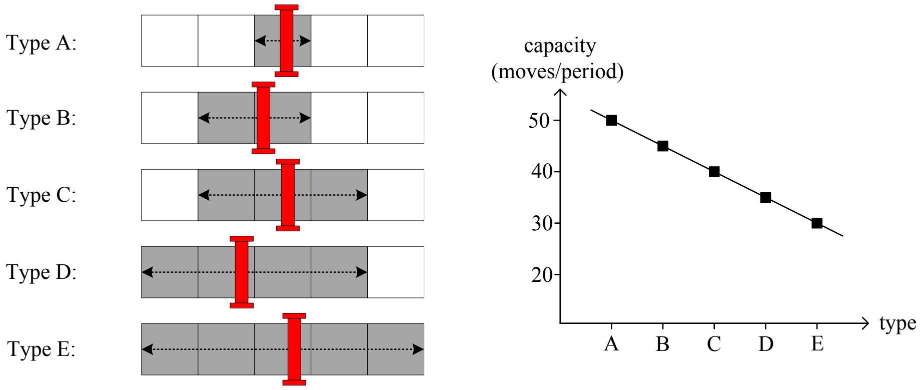


Figure 6.2: Five types of YC deployment profile.

The storage yard management problem with integration of the space allocation and crane deployment is challenging in the sense that the space allocation decisions for incoming container groups has an impact on the workload distribution among blocks. As a result, in order to minimize the YC operating cost, not only do YCs need to be deployed according to the workload distribution, but also sub-blocks should be assigned appropriately to incoming container groups to make full use of the active YCs. Nevertheless, inclusion of space allocation decisions provides an opportunity of adjusting the workload distribution and controlling the space allocation and YC deployment more efficiently.

6.3.2 Assumptions

The following assumptions are made in this chapter:

(1) The information of incoming and outgoing container groups within the planning horizon is assumed to be available, such as grounding time window and rate, retrieval time window and rate, and storage space requirement. Such information can be obtained from container vessels and inland customers who usually inform terminal operators the detailed information of their loading or discharging containers.

(2) Each incoming container group is associated with a favorite yard section where they can be only stored. The assignment of yard sections for container groups are pre-determined in the tactical storage yard allocation stage (Lee et al., 2012; Lee and Jin, 2013) which aims at minimizing the total transportation distance between the quayside area and storage locations.

(3) It is assumed that sufficient YCs are available and workload delay is not allowed. The reason behind this assumption is that we focus on the storage yard management for leading container terminals (e.g., the Port of Singapore) which invest excess resource in order to maintain high efficiency and competitiveness.

(4) Real-time decisions, such as yard truck dispatching and job sequencing for YCs, are not considered in this study.

6.3.3 An integer linear programming model

The planning horizon is set to be one day which is further discretized into twelve two-hour time periods. Given a set of incoming container groups and a set of outgoing groups, the daily storage yard management problem aims (i) to assign available sub-blocks to incoming container groups for temporary storage satisfying the space requirement; (ii) to determine which YC deployment profiles should be selected for each block and in turn to activate YCs accordingly; (iii) to route the YCs over the entire storage yard to balance the YC shortage and surplus in certain blocks. The above three decisions are considered simultaneously to achieve an overall better efficiency for space allocation and YC deployment. The objective is to minimize the operating cost for the activated YCs as well as the inter-block YC movement penalty. Note that the traffic congestion issue in the yard is also considered by ensuring that the traffic of the accessing lanes should observe the capacity constraint. Before presenting the mathematical formulation, we first introduce the notations as follows:

Sets:

T : set of time periods, $T = \{1, 2, \cdots, 12\}$

K_1 : set of incoming container groups that arrive in the planning horizon

K_2 : set of containers groups that are already stored in the yard

K : $K = K_1 \cup K_2$

I : set of yard blocks

M : set of sub-blocks

Ω : set of YC deployment profiles

S : set of yard sections

N : set of accessing traffic lanes

Parameters:

q_k : storage space requirement of group $k \in K$ in terms of number of sub-blocks

f_k : grounding/retrieval rate in each sub-block of container group $k \in K$ (moves/period)

l_k : $\in S$, the assigned storage section for group $k \in K_1$

α_{kt} : 1, if group $k \in K$ is within the storage yard at time period t; 0, otherwise

β_{kt} : 1, if group $k \in K$ is in grounding/retrieval status at time period t; 0, otherwise

$\tilde{x}_{mk}$: 1, if sub-block $m \in M$ is assigned to group $k \in K_2$; 0, otherwise

c_0 : the operating cost of an YC per time period

c_{ij} : the YC moving cost from block $i \in I$ to $j \in I$

d_i : the initial number of YCs deployed in block $i \in I$

d_{max} : maximum number of active YCs that are allowed to be deployed within a block

Q^1_ω : the handling capacity associated with YC deployment profile $\omega \in \Omega$ (moves/period)

Q^2_s : the traffic capacity of accessing lane $s \in S$ (moves/peroid)

γ_{is} : 1, if block $i \in I$ belongs to yard section $s \in S$; 0, otherwise

$\theta_{i\omega}$: 1, if YC deployment profile $\omega \in \Omega$ is associated with yard block $i \in I$; 0, otherwise

$\delta_{m\omega}$: 1, if sub-block $m \in M$ is covered by YC deployment profile $\omega \in \Omega$; 0, otherwise

σ_{mn} : 1, if lane $n \in N$ is the accessing lane to sub-block $m \in M$; 0, otherwise

λ_{ms} : 1, if sub-block $m \in M$ belongs to yard section $s \in S$; 0, otherwise

$\mu_{s\omega}$: 1, if the associated block of YC deployment profile $\omega \in \Omega$ belongs to yard section $s \in S$; 0, otherwise

φ_{sn} : 1, if accessing lane traffic lane $n \in N$ is within yard section $s \in S$; 0, otherwise

M : a large constant

Note that the set of YC deployment profile Ω consists of all the profiles defined for each yard block and for each time period. Thus, $|\Omega| = 15|I||T|$. The parameter list includes the basic input data $q_k, f_k, l_k, \alpha_{kt}, \beta_{kt}$ for each container group $k \in K$; the storage status $\tilde{x}_{mk}$ for each outgoing container group $k \in K_2$; the YC operational costs c_0, c_{ij}; the YC deployment data d_i, d_{max}; the handling capacity of YC profiles Q^1_ω; and the capacity of accessing traffic lanes Q^2_s. The two capacity related parameters are defined in terms of container moves per period. The rest of the parameters are binary coefficients indicating the relationship between two objects (section, block, sub-block, profile and lane).

Decision variables:

x_{mk} : $\in \{0,1\}$. 1, if sub-block $m \in M$ is assigned to group $k \in K_1$ for storage; 0, otherwise

y_ω^t : $\in \{0,1\}$. 1, if YC deployment profile $\omega \in \Omega$ is employed at time period $t \in T$; 0, otherwise

z_{ij}^t : $\in \mathbb{Z}^+$. the number of YCs moving from block $i \in I$ to $j \in I$ between time period $t-1$ and $t \in T$

With the notations defined above, the problem is formulated as follows:

Objective function:

$$[\mathbf{P0}] \quad \min \quad c_0 \sum_{t\in T} \sum_{\omega\in\Omega} y_\omega^t + \sum_{i\in I} \sum_{j\in I} \sum_{t\in T} c_{ij} z_{ij}^t \tag{6.1}$$

The objective function is the summation of YC operating cost and YC inter-block movement cost over the planning horizon. The YC operating cost is determined by the number of YC deployment profiles employed, i.e., the total number of active YCs of all time periods. The YC inter-block movement also contributes to the objective function, as it takes efforts and sometimes even affects the container traffic in the local area. It is worthwhile to note that the total transportation distance between quayside area and the storage locations is not considered in the objective function. This is because the transportation distance is determined by the assignment of storage sections from the upper planning level and the sub-block allocation decision only affects the YC operations.

Constraints:

$$\sum_{m\in M|\lambda_{ml_k}=1} x_{mk} = q_k \quad \forall k \in K_1 \tag{6.2}$$

$$\sum_{k\in K_1} \alpha_{kt} x_{mk} + \sum_{k\in K_2} \alpha_{kt} \tilde{x}_{mk} \leq 1 \quad \forall m \in M, \forall t \in T \tag{6.3}$$

$$\sum_{k\in K_1}\beta_{kt}x_{mk}+\sum_{k\in K_2}\beta_{kt}\tilde{x}_{mk}\leq\sum_{\omega\in\Omega}\delta_{m\omega}y_{\omega}^{t}\quad\forall m\in M,\forall t\in T\tag{6.4}$$

$$\sum_{m\in M|\delta_{m\omega}=1}\left(\sum_{k\in K_1}f_k\beta_{kt}x_{mk}+\sum_{k\in K_2}f_k\beta_{kt}\tilde{x}_{mk}\right)\leq Q_{\omega}^{1}+(1-y_{\omega}^{t})M\quad\forall\omega\in\Omega,\forall t\in T\tag{6.5}$$

$$\sum_{j\in I}z_{ji}^{t}=\sum_{j\in I}z_{ij}^{t+1}\quad\forall i\in I,\forall t\in T\backslash\{|T|\}\tag{6.6}$$

$$\sum_{j\in I}z_{ij}^{1}=d_i\quad\forall i\in I\tag{6.7}$$

$$\sum_{j\in I}z_{ji}^{t}\geq\sum_{\omega\in\Omega}\theta_{i\omega}y_{\omega}^{t}\quad\forall i\in I,\forall t\in T\tag{6.8}$$

$$\sum_{\omega\in\Omega}\theta_{i\omega}y_{\omega}^{t}\leq d_{max}\quad\forall i\in I,\forall t\in T\tag{6.9}$$

$$\sum_{\omega\in\Omega}\delta_{i\omega}y_{\omega}^{t}\leq 1\quad\forall m\in M,\forall t\in T\tag{6.10}$$

$$\sum_{m\in M|\sigma_{mn}=1}\left(\sum_{k\in K_1}f_k\beta_{kt}x_{mk}+\sum_{k\in K_2}f_k\beta_{kt}\tilde{x}_{mk}\right)\leq Q_{s}^{2}\quad\forall n\in N,\forall t\in T\tag{6.11}$$

$$x_{mk}\in\{0,1\}\quad\forall m\in M,\forall k\in K\tag{6.12}$$

$$y_{\omega}^{t}\in\{0,1\}\quad\forall\omega\in\Omega,\forall t\in T\tag{6.13}$$

$$z_{ij}^{t}\in\mathbb{Z}^{+}\quad\forall i\in I,\forall j\in I,\forall t\in T\tag{6.14}$$

Constraints (6.2) assign sub-blocks to each incoming container group within the pre-determined yard section satisfying the requested amount of storage space. Constraints (6.3) define the storage capacity restriction by ensuring that any sub-block can at most be occupied by one group at any time period. During the grounding or retrieval status for any sub-block, certain YC deployment profile should be employed as guaranteed by Constraints (6.4). The left-hand-side of Constraints (6.4) takes the value of 1 if the sub-block is performing grounding or retrieval operations, and it forces certain YC profile to be active on the right-hand-side. Constraints (6.5) guarantee that for any employed YC deployment profile the workload conducted in its service area respects the handling capacity of the profile. Constraints (6.5) are redundant for any YC deployment profile that is not employed. The YC movement conservation is specified by Constraints (6.6)-(6.7). The number of YCs moving into a certain block before a time period should be the same as the number of YCs staying still and moving out after that period. For each block, the number of YCs deployed within the block should be larger than the YC deployment profiles employed at each time period, as ensured by Constraints (6.8). Meanwhile, the number of active YCs should observe the upper limit for each block at each time period, as guaranteed by Constraints (6.9). Constraints (6.10) state the non-overlapping requirement for YC deployment profiles by ensuring that one sub-block can be served by at most one profile at any time period. Constraints (6.11) correspond to the traffic capacity restriction for each accessing lane. Finally, the domain of decision variables is defined by Constraints (6.12)-(6.14).

6.3.4 Computational complexity

Consider a particular case of the storage yard management problem where the planning horizon only covers one time period (i.e., $|T| = 1$), space allocation is not necessary due to no incoming container groups and the YC inter-block movements are prohibited ($c_{ij} = M$). The remaining task consists of $|I|$ sub-problems each of which is to determine the YC deployment profile selection for a yard block. For each yard block, we eliminate those YC deployment profiles violating the handling capacity constraint according to the workload distribution over the five sub-blocks. The remaining task is to find a minimum subset of YC deployment profiles covering all the sub-blocks with container grounding/retrieval activities, and this problem is equivalent to the *set packing problem*. Therefore, we have reduced the set packing problem to a particular case of the storage yard management problem. Since the set packing problem is known to be NP-hard, the storage yard management problem in this chapter is thus also NP-hard.

6.4 Heuristic Approach

In this section, we develop a heuristic approach for the storage yard management problem employing a divide-and-conquer strategy, harmony search and constraint satisfaction techniques. The heuristic framework is introduced in Section 6.4.1 and the details are presented in Sections 6.4.2-6.4.5.

6.4.1 Heuristic framework

We observe that Constraints (8) are the only constraints linking YC movement decision variable $\mathbf{z}$ with the other decision variables $\mathbf{x}$ and $\mathbf{y}$. By relaxing them, the original problem could be divided into two sub-problems: one involving space allocation and YC deployment profile selection decisions and the other only concerning about YC inter-block movement. Furthermore, the decisions for different yard sections in the former sub-problem are no longer bound together, and thus it could be decomposed into $|S|$ smaller sub-problems. Such an observation motivates us to employ the divide-and-conquer strategy and to solve the storage yard management problem in a sequential way but with a feedback loop, as illustrated by Figure 6.3.

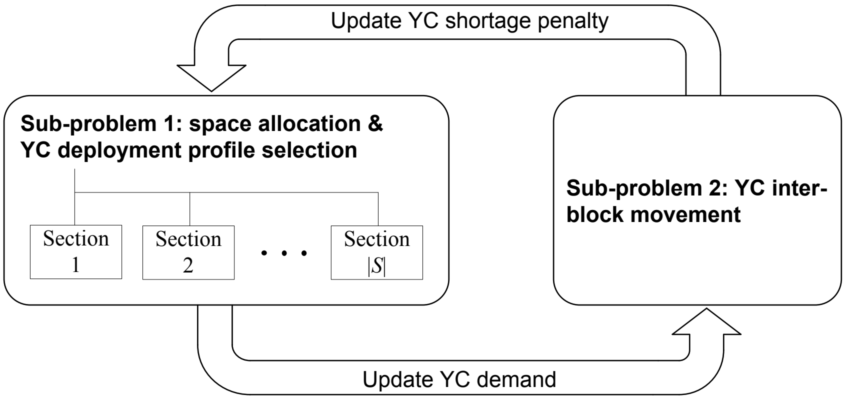

Update YC shortage penalty
Sub-problem 1: space allocation & YC deployment profile selection
Section 1
Section 2
. . .
Section |S|
Sub-problem 2: YC inter-block movement
Update YC demand

Figure 6.3: Illustration of the heuristic framework.

In the heuristic, Sub-problem 1 is firstly solved by determining the space allocation and YC deployment profile selection for each yard section. The initial YC deployment is taken into account when solving Sub-problem 1, and any violation will be penalized in the objective function. After that, the YC demand over the entire storage yard could be obtained and passed onto Sub-problem 2. The YC inter-block movement is determined subsequently according to the YC demand. Stopping criteria are checked to see whether to terminate the search process or to update the YC shortage penalty and continue further iterations.

6.4.2 Sub-problem 1: space allocation & YC deployment profile selection

Sub-problem 1 deals with space allocation and YC deployment profile selection for all yard sections each of which could be solved separately. Computational efficiency in solving each of them is a key concern as the solution approach involves many iterations. Thus, a harmony search algorithm and a constraint satisfaction method are developed. The flowchart of the algorithm designed for Sub-problem 1 is presented in Figure 6.4. The initial harmony memory of the harmony search algorithm is generated firstly, and the harmony search continues only if all the solutions in the harmony memory are feasible. In case any of the solution is infeasible, the constraint satisfaction search is invoked as an alternative method. We remark that the harmony search algorithm is more suitable for cases where the workload is low and feasible solutions can be relatively easy to generate, while the constraint satisfaction search is more effective in generating feasible solutions when the workload is high and randomly generated solutions can easily violate the YC related constraints. The generation of initial harmony memory takes the responsibility of deciding which of the two methods should be employed by checking the feasibility of the solutions in the initial harmony memory. In the following, the details of harmony search algorithm and constraint satisfaction search are introduced.

6.4.2.1 Formulation

We further introduce a decision variable u_{it} to represent the number of YC shortage for block $i \in I$ at time period $t \in T$ compared with the initial number of YCs deployed in that block. Then, the sub-problem for yard section $s \in S$ is formulated as follows:

$$[\textbf{P-Sub1}[\textbf{s}]] \quad \min \quad \sum_{t \in T} \sum_{\omega \in \Omega | \mu_{s\omega = 1}} c_0 y_\omega^t + \sum_{i \in I} \sum_{t \in T} p_{it} u_{it} \tag{6.15}$$

$$\text{s.t.} \quad \sum_{m \in M | \lambda_{ms} = 1} x_{mk} = q_k \quad \forall k \in K_1 | l_k = s \tag{6.16}$$

$$\sum_{k \in K_1 | l_k = s} \alpha_{kt} x_{mk} + \sum_{k \in K_2} \alpha_{kt} \tilde{x}_{mk} \leq 1 \quad \forall m \in M, \forall t \in T | \lambda_{ms} = 1 \tag{6.17}$$

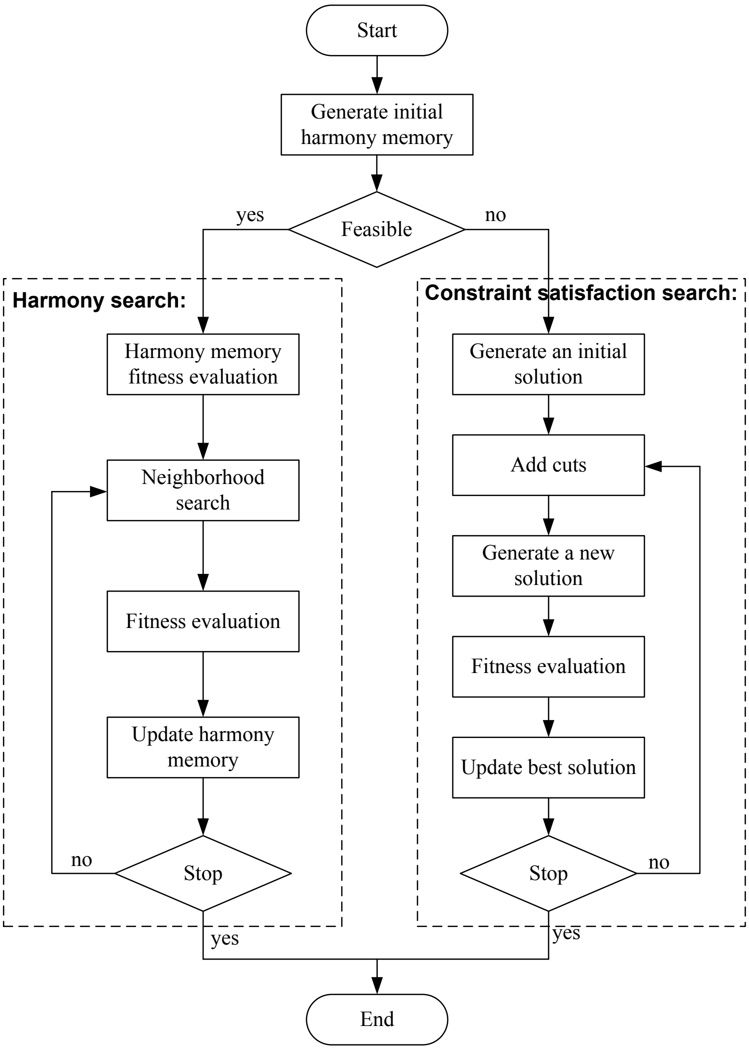


Figure 6.4: Flowchart of the algorithm designed for Sub-problem 1.

$$\sum_{k\in K_1|l_k=s}\beta_{kt}x_{mk}+\sum_{k\in K_2}\beta_{kt}\tilde{x}_{mk}\leq\sum_{\omega\in\Omega|\mu_{s\omega}=1}\delta_{m\omega}y^t_\omega\quad\forall m\in M,\forall t\in T|\lambda_{ms}=1\tag{6.18}$$

$$\sum_{m\in M|\delta_{m\omega}=1}\left(\sum_{k\in K_1|l_k=s}f_k\beta_{kt}x_{mk}+\sum_{k\in K_2}f_k\beta_{kt}\tilde{x}_{mk}\right)\leq Q^1_\omega+(1-y^t_\omega)M$$

$$\forall\omega\in\Omega,\forall t\in T|\mu_{s\omega}=1\tag{6.19}$$

$$\sum_{\omega\in\Omega|\mu_{s\omega}=1}\theta_{i\omega}y^t_\omega\leq d_{max}\quad\forall i\in I,\forall t\in T|\gamma_{is}=1\tag{6.20}$$

$$\sum_{\omega\in\Omega|\mu_{s\omega}=1} \delta_{i\omega} y_{\omega}^{t} \leq 1 \quad \forall m \in M, \forall t \in T|\lambda_{ms} = 1 \tag{6.21}$$

$$\sum_{m\in M|\sigma_{mn}=1} \left(\sum_{k\in K_1|l_k=s} f_k\beta_{kt}x_{mk} + \sum_{k\in K_2} f_k\beta_{kt}\tilde{x}_{mk} \right) \leq Q_s^2 \quad \forall n \in N, \forall t \in T|\varphi_{sn} = 1 \tag{6.22}$$

$$u_{it} \geq \sum_{\omega\in\Omega|\mu_{s\omega}=1} \theta_{i\omega} y_{\omega}^{t} - d_i \quad \forall i \in I, \forall t \in T \tag{6.23}$$

$$x_{mk} \in \{0,1\} \quad \forall m \in M, \forall k \in K|\lambda_{ms} = 1 \tag{6.24}$$

$$y_{\omega}^{t} \in \{0,1\} \quad \forall \omega \in \Omega, \forall t \in T|\mu_{s\omega} = 1 \tag{6.25}$$

$$u_{it} \geq 0 \quad \forall i \in I, \forall t \in T|\gamma_{is} = 1 \tag{6.26}$$

Objective function (6.15) minimizes the YC operating cost and the total number of YC shortage weighted by penalty coefficient **p**. Constraints (6.16)-(6.22) are the same as their corresponding ones in the original formulation [**P0**], except that they are defined with respect to a particular yard section $s \in S$. Constraints (6.23) define the decision variable **u**.

6.4.2.2 Harmony search algorithm

We now describe the components of the harmony search algorithm (Geem, 2009), one of the meta-heuristics inspired by the improvisation process of musicians, for solving problem [**PSub1[s]**]. A solution is coded as a vector $\mathbf{e} = \{\cdots, e_{ki}, \cdots\}$ indicating the information of space allocation decisions for incoming container groups. Each element e_{ki} of the vector represents the sub-block id assigned to container group k as its i^{th} sub-block, and it should observe the the assigned storage section l_k. Figure 6.5 shows an illustrative example of solution coding. Incoming group 1 requests two sub-blocks for storage ($q_k = 2$) and is assigned with sub-blocks 5 and 8. Similarly, three sub-blocks (4, 9, 12) are assigned to the second container group.

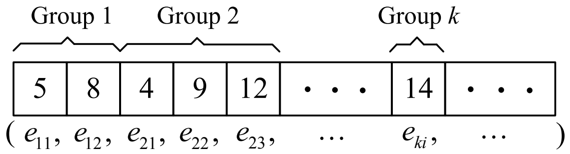


Figure 6.5: An illustrative example of the solution coding.

Initially, HMS solutions are randomly generated as the initial harmony memory. For each incoming group k, denote E_k as a set of candidate sub-blocks that are not occupied by outgoing container groups when k is within the yard. Then, an initial solution can be generated by randomly picking elements in E_k for each e_{ki} in the solution. Storage capacity constraints (6.3) and the traffic capacity (6.11) of accessing lanes should be checked to determine the feasibility of the generated solutions. In case of infeasibility, the solution should be re-generated unless reaching a maximum iteration limit $\mathrm{SC}_1^{\mathrm{HSA}}$.

Fitness of candidate solutions could be evaluated by translating solution vector $\mathbf{e}$ into the decision variable $\mathbf{x}$ and solving the resulting problem [**P-Sub1[s]**]. With given $\mathbf{x}$, problem [**P-Sub1[s]**] could be separated to four smaller sub-problems. Each of them only involves the YC deployment profile selection for one of the four yard blocks, and can be solved by standard solver efficiently. In case no feasible solution exists due to the violation of upper limit on active YC number (Constraints (6.9)) for any of the four sub-problems, the candidate solution is considered as infeasible and is assigned with a large constant as its fitness. Note that the solution fitness includes YC operating cost and the penalty for YC shortage as indicated by Equation(6.15).

The neighborhood search step of the harmony search algorithm mimics the improvisation process of a group of musicians to generate a new candidate solution $\mathbf{e}'$. Each element of the new solution e'_{ki} acts like an individual musician and either selects an value randomly in its harmony memory $\{e_{ki}^1, e_{ki}^2, \cdots, e_{ki}^{HMS}\}$ or composes a new one in E_k according to a probability HMCR$\in (0,1)$, as expressed by Equation (6.27). Within the new candidate solution, any element is examined to determine whether it should be adjusted locally within $[e'_{ki} - \mathrm{HMM}, e'_{ki} + \mathrm{HMM}]$ according the PAR parameter. This operation is referred as pitch adjustment for music improvisation and expressed mathematically by Equation (6.28). Solution feasibility is also checked to ensure the new solution respects the storage capacity and lane capacity constraints.

$$e'_{ki} \leftarrow \begin{cases} \{e_{ki}^1, e_{ki}^2, \cdots, e_{ki}^{HMS}\} & \text{w.p. HMCR} \\ E_k \quad \text{w.p. (1-HMCR)} \end{cases} \tag{6.27}$$

$$e'_{ki} \leftarrow \begin{cases} [e'_{ki} - \text{HMM}, e'_{ki} + \text{HMM}] & \text{w.p. PAR} \\ e'_{ki} \quad \text{w.p. (1-PAR)} \end{cases} \tag{6.28}$$

With a new candidate solution generated by the neighborhood search step, fitness evaluation is conducted to determine whether the harmony memory should be updated or not. In case the new candidate solution is superior to the worst one in the current harmony memory in terms of fitness, the harmony memory could be updated by replacing the worst solution by the new one. The neighborhood search process terminates either it reaches the iteration limit SC_2^{HSA} or the harmony memory is not updated for SC_3^{HSA} consecutive iterations.

6.4.2.3 Constraint satisfaction search

Constraint satisfaction search is an alternative approach for solving problem [**P-Sub1[s]**]. It takes the advantage of standard solvers in finding feasible solutions, and is invoked when the harmony search algorithm fails in generating the initial harmony memory. We remark that constraint satisfaction search performs well especially in case of high yard space utilization which highly prevents the solution generation method of the harmony search algorithm producing feasible solutions.

Feasible solutions to [**P-Sub1[s]**] are generated based on a new program with constraint set (6.16)-(6.26) and objective function **0**. With each generated solution, a cut is defined according to Equation (6.29) where Φ_0 and Φ_1 are sets of (m, k) pairs whose corresponding decision variable x_{mk} of the solution takes the value of 0 and 1, respectively. Such cuts are added to the program so as to avoid producing repeated solutions.

$$\sum_{(m,k)\in\Phi_0} x_{mk} + \sum_{(m,k)\in\Phi_1} (1 - x_{mk}) \geq 1 \tag{6.29}$$

The process of constraint satisfaction search is shown in Figure 6.4. As can be seen, feasible solutions are generated continually and the best solution is updated accordingly. The search processing terminates either if it reaches the iteration limit SC_1^{CSS}, or the best solution is not updated for SC_2^{CSS} consecutive iterations.

6.4.3 Sub-problem 2: YC inter-block movement

With the output decisions of space allocation and YC deployment profile selection from Sub-problem 1, the YC demand of each block over the planning horizon can be obtained. Thus, the remaining task is to determine YC inter-block movement so as to meet the YC demand of each block. Denote $\bar{y}^t_\omega$ as the YC deployment profile selection decision from Sub-problem 1. Then, Sub-problem 2 could be formulated as follows:

$$[\textbf{P-Sub2}] \quad \min \quad \sum_{i\in I}\sum_{j\in I}\sum_{t\in T} c_{ij} z^t_{ij} \tag{6.30}$$

$$\text{s.t.} \quad \sum_{j\in I} z^t_{ji} = \sum_{j\in I} z^{t+1}_{ij} \quad \forall i \in I, \forall t \in T\backslash\{|T|\} \tag{6.31}$$

$$\sum_{j\in I} z^1_{ij} = d_i \quad \forall i \in I \tag{6.32}$$

$$\sum_{j\in I} z^t_{ji} \geq \sum_{\omega\in\Omega} \theta_{i\omega}\bar{y}^t_\omega \quad \forall i \in I, \forall t \in T \tag{6.33}$$

$$z^t_{ij} \in \mathbb{Z}^+ \quad \forall i \in I, \forall j \in I, \forall t \in T \tag{6.34}$$

The objective function (6.30) only concerns with the cost incurred by the YC inter-block movement. Constraints (6.31)-(6.32) define the YC movement conservation. The total number of YCs within a block should meet the demand at each time period, as imposed by Constraints (6.33). The integrality requirement of decision variable **z** is defined by Constraints (6.34).

It can be verified that Sub-problem 2 is equivalent to the minimum cost network flow problem, as illustrated by an example with two blocks and two time periods in Figure 6.6. The source node S has a supply of $d_1 + d_2$ while the sink node T has a demand with the same amount. Links are associated with two parameters: link cost and lower bound of link flow. One block is represented by two nodes (in-node and out-node) and a link in between at each time period. The in-nodes are connected with the out-nodes of the previous time period by certain links indicating the YC inter-block movement. The other links defined between the in-nodes and out-nodes of the same time period are imposed with a lower bound for the link flow. These links are introduced to represent the YC demand of the corresponding blocks. Since all the input data of the minimum cost network flow problem are integers, the optimal solutions consist of only integers. This observation motivates us to relax the integrality constraints (6.34) to be $z_{ij}^t \geq 0$. Therefore, Sub-problem 2 could be solved efficiently by commercial solvers.

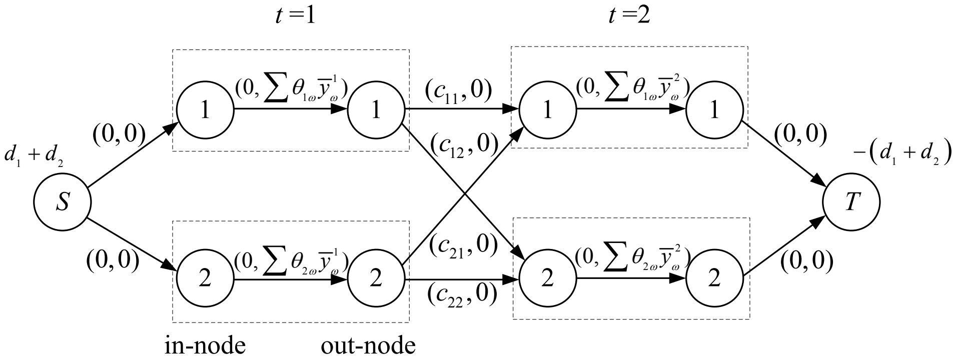

$t=1$
$t=2$
$(0,\sum\theta_{1\omega}\bar{y}_\omega^1)$
$(c_{11},0)$
$(0,\sum\theta_{1\omega}\bar{y}_\omega^2)$
$(0,0)$
d_1+d_2
$(c_{12},0)$
$(0,0)$
$-(d_1+d_2)$
S
T
1
1
1
1
$(0,0)$
$(c_{21},0)$
$(0,0)$
$(0,\sum\theta_{2\omega}\bar{y}_\omega^1)$
$(0,\sum\theta_{2\omega}\bar{y}_\omega^2)$
2
2
2
2
$(c_{22},0)$
in-node
out-node

Figure 6.6: Minimum cost network flow representation of Sub-problem 2.

6.4.4 Penalty updating scheme

The YC shortage penalty $\mathbf{p}$ is initialized as $\mathbf{p}^0$. After solving both of the two sub-problems at each iteration, the penalty coefficient is updated by Equation (6.35) where parameter $\zeta > 1$ and $\bar{z}_{ji}^t$ is the YC movement decision obtained from Sub-problem 2. The idea behind the penalty updating scheme is to increase the penalty weight for blocks if their YC demand violates the YC deployment at the previous time period, and to decrease the penalty weight otherwise. In doing so, the decisions of space allocation and YC deployment selection in Sub-problem 1 are forced to follow the YC deployment at previous time periods to a more extend as the search process continues. Initially, the penalty is set to be a small value in order to allow less restricted YC movement. Therefore, by scanning the penalty from less to more restricted values, we are able to find a certain point that minimizes the overall objective function (6.1).

$$p_{it} \leftarrow \begin{cases} p_{it}/\zeta & \text{if } \sum_{\omega\in\Omega}\theta_{i\omega}\bar{y}_\omega^t \leq d_i(t=1) \text{ or } \sum_{\omega\in\Omega}\theta_{i\omega}\bar{y}_\omega^t \leq \sum_{j\in I}\bar{z}_{ji}^{t-1}(t\in T\backslash\{1\}) \\ p_{it}\times\zeta & \text{otherwise} \end{cases} \tag{6.35}$$

6.4.5 Stopping criteria

Upon solving both of the two sub-problems at each iteration, the total cost of the storage yard management problem, including YC operating cost and YC inter-block movement cost, can be obtained. The search process in Figure 6.3 terminates either if it reaches the iteration limit $\mathrm{SC}_1^{\mathrm{HEU}}$ or the total cost does not get improved for $\mathrm{SC}_2^{\mathrm{HEU}}$ consecutive iterations.

6.5 Computational Experiment

In this section we firstly introduce a simple lower bound of the storage yard management problem. Then, we illustrate the generation of test instances and the settings for the heuristic parameters. The numerical results of the integer program [**P0**] and the developed heuristic approach are presented and compared. Finally, the integration improvement of the two decision problems, space allocation and crane deployment, is evaluated. The integer program [**P0**] and the heuristic approach are coded in C++ and use CPLEX 12.1 as the MIP solver. All the computational experiments are conducted on a PC with 3 GHz CPU and 4 GB RAM.

6.5.1 Lower bound

A simple lower bound can be obtained by excluding the YC inter-block movement decision and its contribution to objective function from the integer program [**P0**], as stated by the following program [**P-LB**]. The objective function (6.36) only consists of the YC operating cost of all blocks over the planning horizon. The constraints are the same as those in [**P0**] excluding those related with YC inter-block movement decision. Note that problem [**P-LB**] could be decomposed into $|S|$ sub-problems as the decisions for different yard sections are not bound together. This makes solving the lower bound program much easier than the original one.

$$[\mathbf{P\text{-}LB}] \quad \min \quad c_0 \sum_{t \in T} \sum_{\omega \in \Omega} y_\omega^t \tag{6.36}$$

$$\text{s.t.} \quad (6.2) - (6.5), (6.9) - (6.13)$$

6.5.2 Instance generation and algorithm settings

Four sets of test instances are randomly generated based on the terminal layout shown in Figure 6.7. Each set has five instances with the same parameters as listed in Table 6.1: storage area, number of yard blocks, number of incoming and outgoing container groups. The storage space requirement parameter q_k and the grounding/retrieval rate parameter f_k are uniformly distributed within $[1, 3]$ sub-blocks and $[10, 40]$ moves/period, respectively. For each container group, parameters α_{kt}, β_{kt} can be obtained by randomly assigning the grounding/retrieval time windows whose length is distributed within [1,3] time periods. The storage locations of outgoing container groups are randomly assigned over the storage yard, and thus parameter $\tilde{x}_{mk}$ is known. The YC operating cost c_0 is set to 30 per period while the YC movement between neighboring blocks contributes 5 to the objective function. The YC inter-block movement cost is set to be proportional to the distance between blocks. Note that YCs are only allowed to move between neighboring block columns (e.g., from C1 to C2), and an upper limit of 15 is imposed. In case more than 15 is needed to conduct YC movement, the cost parameter c_{ij} is set to a large enough constant so as to avoid such YC movements. Regarding the initial YC distribution, two cases are considered: Case A with more YCs as shown by the block columns C1-C3 in Figure 6.7 and Case B with less YCs as indicated by the rest block columns C4-C6. The handling capacity of all types of YC deployment profiles is shown in Figure 6.2, ranging from 30 to 50 moves/period. The traffic capacity of all accessing lanes is set to 200 moves/period.

Table 6.1: Instance parameters.

	Storage area	$\|I\|$	$\|K_1\|$	$\|K_2\|$
Set 1	R1, C1-C2	8	4	6
Set 2	R1-R2, C1-C3	24	15	30
Set 3	R1-R3, C1-C4	48	30	50
Set 4	R1-R3, C2-C6	72	50	80

The parameters of the developed heuristic approach are selected by trials and listed as follows:

- Searching process of the heuristic approach: $\text{SC}_1^{\text{HEU}} = 50$, $\text{SC}_2^{\text{HEU}} = 10$;

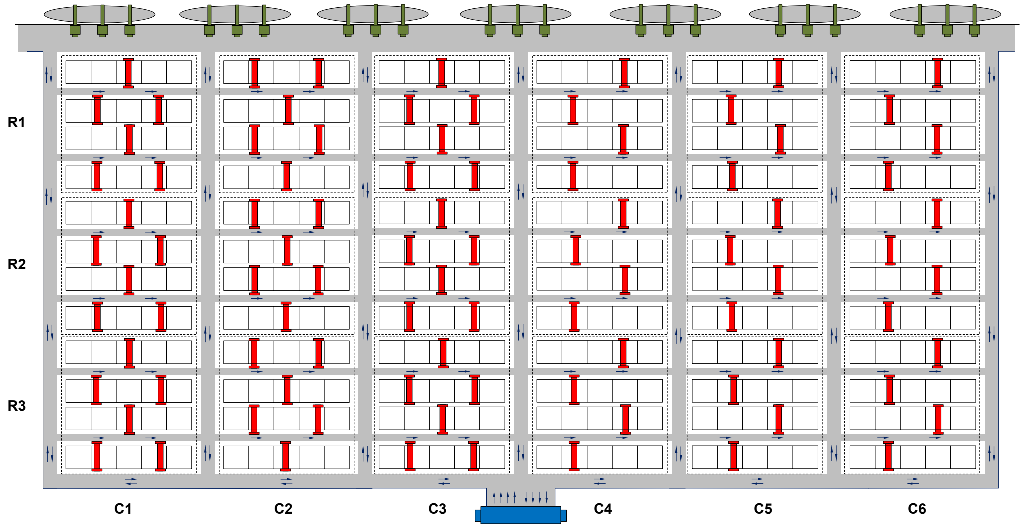


Figure 6.7: Terminal layout for the computational experiment.

- Harmony search algorithm for Sub-problem 1: HMS=5, $\text{SC}_1^{\text{HSA}} = 100$, HMCR=0.9, PAR=0.1, HMM=2, $\text{SC}_2^{\text{HSA}} = 50$, $\text{SC}_3^{\text{HSA}} = 15$;
- Constraint satisfaction search for Sub-problem 1: $\text{SC}_1^{\text{CSS}} = 100$, SC_2^{CSS}=15;
- Penalty updating scheme: $p_{it}^0 = 10\ \forall i \in I\ \forall t \in T$, $\zeta = 1.2$.

6.5.3 Experiment results

Tables 6.2-6.5 present the experimental results for all the test instances under the two cases of YC deployment. The second column reports the lower bound (LB) for each test instance. CPLEX is given a time budget of one hour and returns the truncated solutions upon reaching the time limit. Due to the randomness involved in the heuristic approach, we run it ten times for each test instance and report the mean objective function value and standard deviation in columns (6) and (7). The computational time consumed by CPLEX and the heuristic approach are reported in columns (5) and (8). The last two columns evaluate the solution quality of CPLEX and the heuristic approach by comparing their results with lower bounds. As can be seen, the average gap between CPLEX and LB is smaller than 1% which indicates that the YC inter-block movement cost does not contribute much to the objective function when the storage allocation and YC deployment is properly coordinated. One possible reason is that the storage space allocation is determined in such a way that YC inter-block movement is seldom carried out. In other words, the flexibility of space allocation is able to adjust the workload according to the initial YC deployment and thus provides an opportunity of reducing YC inter-block movements. Regarding the heuristic approach, it is much more efficient than CPLEX as it produces near-optimal solutions within few minutes. The solution quality of the heuristic approach is also acceptable from our point of view, as the average gap between the solutions and LB is less than 3%. Overall speaking, CPLEX performs better if the computational efficiency is not a concern while the heuristic approach is an alternative method which balances solution quality and computational efficiency.

Table 6.2: Computational results of data Set 1.

Instance	YC case	LB	CPLEX		Heuristic			Gap(%)	
			Result	CPU(s)	Mean	Stdev	CPU(s)	Gap1	Gap2
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
Set1-I1	A	450	450	163	450.0	0.0	9.3	0.00	0.00
Set1-I2	A	660	660	1019	660.0	0.0	6.3	0.00	0.00
Set1-I3	A	420	420	53	420.0	0.0	5.8	0.00	0.00
Set1-I4	A	540	540	4528	540.0	0.0	7.8	0.00	0.00
Set1-I5	A	300	300	2	300.0	0.0	5.8	0.00	0.00
Set1-I1	B	450	450	258	450.0	0.0	10.6	0.00	0.00
Set1-I2	B	660	670	3039	670.0	0.0	7.3	1.52	1.52
Set1-I3	B	420	420	12	420.0	0.0	6.8	0.00	0.00
Set1-I4	B	540	540	1357	540.0	0.0	8.3	0.00	0.00
Set1-I5	B	300	300	2	300.0	0.0	6.8	0.00	0.00
Average								0.15	0.15

Gap1=[(4)-(3)]/(3)×100%, Gap2=[(6)-(3)]/(3)×100%

6.5.4 Integration improvement

Computational experiments are also conducted to investigate how much improvement can be gained from the integration of the two decision problems, space allocation and YC deployment, for the daily storage yard management. Two scenarios are compared: integration scenario and non-integration scenario. The integration scenario corresponds to the problem tackled in this chapter. For the non-integration scenario, we randomly generate 100 space allocation plans

Table 6.3: Computational results of data Set 2.

Instance	YC case	LB	CPLEX		Heuristic			Gap(%)	
			Result	CPU(s)	Mean	Stdev	CPU(s)	Gap1	Gap2
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
Set2-I1	A	2250	2260	3600	2268.0	5.4	38.1	0.44	0.80
Set2-I2	A	2130	2135	3600	2140.5	1.6	27.9	0.23	0.49
Set2-I3	A	3120	3135	3600	3140.0	0.0	30.8	0.48	0.64
Set2-I4	A	2880	2890	3600	2985.0	0.0	18.6	0.35	3.65
Set2-I5	A	2340	2355	3600	2415.0	0.0	22.5	0.64	3.21
Set2-I1	B	2250	2285	3600	2308.5	5.8	43.9	1.56	2.60
Set2-I2	B	2130	2140	3600	2157.0	13.6	30.7	0.47	1.27
Set2-I3	B	3120	3180	3600	3199.0	2.1	33.0	1.92	2.53
Set2-I4	B	2880	2905	3600	2986.5	2.4	39.0	0.87	3.70
Set2-I5	B	2340	2380	3600	2445.5	1.6	29.4	1.71	4.51
Average								0.87	2.34

Gap1=[(4)-(3)]/(3)×100%, Gap2=[(6)-(3)]/(3)×100%

Table 6.4: Computational results of data Set 3.

Instance	YC case	LB	CPLEX		Heuristic			Gap(%)	
			Result	CPU(s)	Mean	Stdev	CPU(s)	Gap1	Gap2
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
Set3-I1	A	4260	4260	3600	4321.0	25.1	74.6	0.00	1.43
Set3-I2	A	4860	4870	3600	5027.0	13.2	57.6	0.21	3.44
Set3-I3	A	4170	4175	3600	4237.5	13.4	92.8	0.12	1.62
Set3-I4	A	3690	3705	3600	3798.5	19.0	47.6	0.41	2.94
Set3-I5	A	4500	4505	3600	4527.0	20.4	71.6	0.11	0.60
Set3-I1	B	4260	4275	3600	4357.5	22.3	79.2	0.35	2.29
Set3-I2	B	4860	4900	3600	5071.5	6.7	71.6	0.82	4.35
Set3-I3	B	4170	4195	3600	4283.5	29.1	85.1	0.60	2.72
Set3-I4	B	3690	3715	3600	3828.0	27.6	69.7	0.68	3.74
Set3-I5	B	4500	4510	3600	4534.5	9.6	80.1	0.25	0.77
Average								0.35	2.39

Gap1=[(4)-(3)]/(3)×100%, Gap2=[(6)-(3)]/(3)×100%

Table 6.5: Computational results of data Set 4.

Instance	YC case	LB	CPLEX		Heuristic			Gap(%)	
			Result	CPU(s)	Mean	Stdev	CPU(s)	Gap1	Gap2
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
Set4-I1	A	6540	6555	3600	6661.5	40.8	85.1	0.23	1.86
Set4-I2	A	6840	6850	3600	6922.5	7.9	105.0	0.15	1.21
Set4-I3	A	7500	7515	3600	7700.0	21.5	127.4	0.20	2.67
Set4-I4	A	7230	7235	3600	7271.5	25.5	119.1	0.07	0.57
Set4-I5	A	6090	6100	3600	6133.0	14.2	121.4	0.16	0.71
Set4-I1	B	6540	6590	3600	6724.5	31.5	86.8	0.76	2.82
Set4-I2	B	6840	6865	3600	6969.0	13.1	108.1	0.37	1.89
Set4-I3	B	7500	7550	3600	7735.5	36.8	130.5	0.67	3.14
Set4-I4	B	7230	7260	3600	7344.5	20.2	113.5	0.41	1.58
Set4-I5	B	6090	6120	3600	6166.5	13.8	124.7	0.49	1.26
Average								0.35	1.77

Gap1=[(4)-(3)]/(3)×100%, Gap2=[(6)-(3)]/(3)×100%

satisfying the storage capacity constraints (6.3) and lane traffic capacity constraints (6.11), and subsequently solve the remaining YC deployment profile selection and YC inter-block movement problem. Note that the remaining problem could be infeasible due to the maximum active YC constraints (6.9), and thus we introduce the feasibility rate parameter defined as the ratio of number of feasible cases over total number of cases (100). Table 6.6 reports the comparison of the two scenarios for data Set 4. As can be seen, I1 and I3 are two instances with low feasibility rate. This is probably due to high container grounding and retrieval activities in which situations space allocation could easily make YC operations tough to conduct. However, the integration scenario is always able to generate feasible solutions. Moreover, about 10% improvement can be obtained from the integration of the two decision problems as shown in the last column. The results of other data sets are similar to those reported in Table 6.6 for data Set 4 in terms of integration improvement, and are not presented for the sake of brevity.

6.6 Summary

In this chapter, we have studied the integration of two inter-related decision problems, space allocation and yard crane (YC) deployment, for the daily storage yard management in mar-

Table 6.6: Comparison of integration and non-integration scenarios.

YC case	Instance	Integration	Non-integration		Improvement
			Result	Feasibility rate	
(1)	(2)	(3)	(4)	(5)	(6)
A	Set4-I1	6662	7430	51%	10.3%
	Set4-I2	6923	7717	97%	10.3%
	Set4-I3	7700	8578	47%	10.2%
	Set4-I4	7272	7995	68%	9.1%
	Set4-I5	6133	6677	92%	8.1%
B	Set4-I1	6725	7509	38%	10.5%
	Set4-I2	6969	7804	98%	10.7%
	Set4-I3	7736	8600	44%	10.1%
	Set4-I4	7345	8117	70%	9.5%
	Set4-I5	6167	6754	97%	8.7%
Average					9.7%

(6)=[(4)-(3)]/(4)×100%

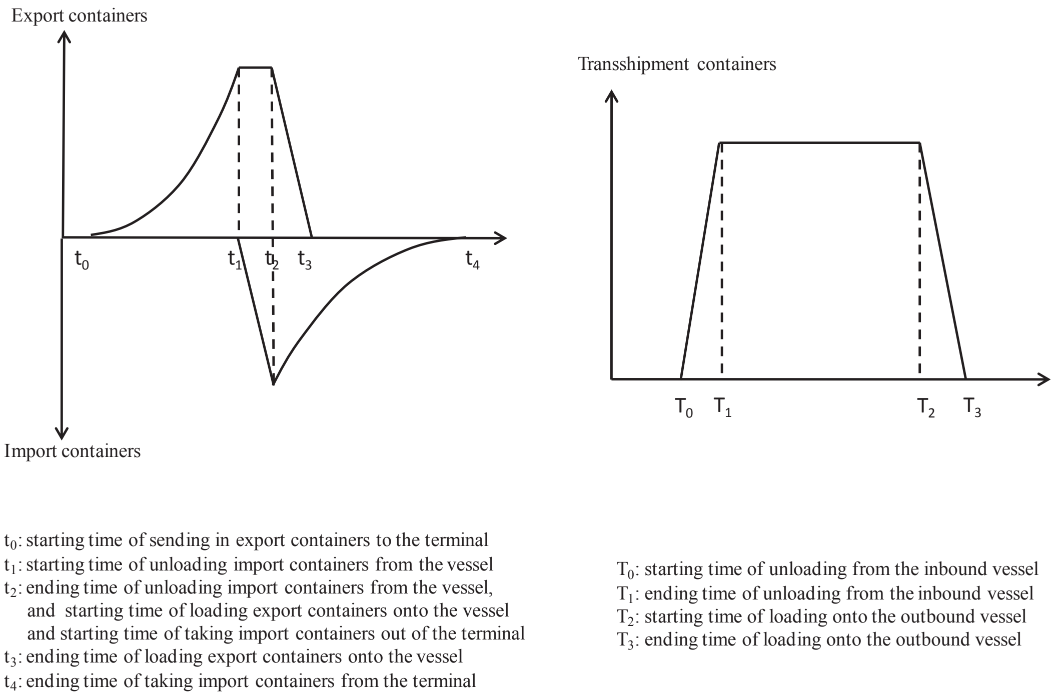
itime container terminals. The space allocation is conducted at the sub-block level and an YC deployment profile concept is introduced to model the YC activities in the storage yard. A particular attention is paid to the container traffic control to avoid potential traffic congestion in the storage yard. An integer linear programming model is developed for the integrated problem with the objective of minimizing the sum of YC operating cost and YC inter-block movement cost. We have developed a heuristic approach based on the divide-and-conquer strategy to solve the integrated problem efficiently. Harmony search algorithm and constraint satisfaction techniques have been designed and employed to solve the sub-problems in the heuristic approach. Computational experiments have shown that the developed optimization model is able to find very good solutions while the heuristic approach balances the solution quality and computational efficiency. The experiment results also have demonstrated that the integration of the two decision problems not only yields cost reduction, but also is able to find feasible solutions for hard situations where non-integrated planning would easily generate solutions violating YC operational restrictions.

Chapter 7

Integrated Bay Allocation and Yard Crane Scheduling Problem for Transshipment Containers

7.1 Introduction

From the viewpoint of container terminals, all the containers can be categorized into three groups: import, export and transshipment containers. Different groups of containers have different flow characteristics. Figure 7.1 shows the number of import, export and transshipment containers stored in the yards associated with a vessel versus time. As can be seen, the retrieving operation of import containers from storage yards to hinterland (t_2, t_4) has a much larger time span compared with retrieving operation (t_1, t_2) from the quayside to storage yards (i.e., unloading). Similarly, the receiving operation of export containers from hinterland (t_0, t_1) lasts much longer than retrieving operation (t_2, t_3) (i.e., loading). This is because the receiving of export containers and retrieving of import containers are very random and are determined by customers and consigns, not by terminal operators. For transshipment containers, the inflow and outflow are different from previous two kinds of containers. As can be seen from Figure 7.1, the receiving and retrieving (i.e., unloading and loading) operations are both intense and



t_0: starting time of sending in export containers to the terminal
t_1: starting time of unloading import containers from the vessel
t_2: ending time of unloading import containers from the vessel, and starting time of loading export containers onto the vessel and starting time of taking import containers out of the terminal
t_3: ending time of loading export containers onto the vessel
t_4: ending time of taking import containers from the terminal

T_0: starting time of unloading from the inbound vessel
T_1: ending time of unloading from the inbound vessel
T_2: starting time of loading onto the outbound vessel
T_3: ending time of loading onto the outbound vessel

Figure 7.1: Number of import, export and transshipment containers in yards associated with a vessel versus time

focused in a short time period.

Between the receiving and retrieving operations, containers are temporarily stored at certain yards and yard cranes are responsible for moving containers in and out of yards. The location of a container is determined in receiving operation which has an impact on the efficiency of retrieving operation. In other words, receiving operation and retrieving operation are highly correlated and should be considered together to ensure an efficient utilization of yard storage space as well as yard cranes.

The major difference among the three types of containers is that both receiving and retrieving operations for transshipment containers are known to the terminal operators while only one of them is known for import and export containers. As a result, for import and export containers, port operators have to utilize the information of only one operation to plan the handling work, such as space allocation. However, for transshipment containers, it is possible and necessary to consider both receiving and retrieving operations to achieve better planning since the information is perfect. In this chapter, we try to make use of the information of the receiving and retrieving operations and to achieve a better handling efficiency by integrating space allocation and yard crane scheduling for transshipment containers.

7.2 Literature Review

In this section, the recent studies related with storage space allocation and yard crane scheduling problem are introduced. We also investigate the deficiencies of them and raise our motivations.

7.2.1 Storage Space Allocation

There are some studies considering space allocation problem. In Kim and Kim (1999a), different arrival patterns of import containers are analyzed to allocate storage space under segregation strategy in a way such that the expected total number of re-handles can be minimized. For export containers, the problem of allocating storage space is also discussed with the objective of utilizing yard space and making loading operation more efficient in Kim and Park (2003b). In these studies, the impact of receiving operation on retrieving operation is considered by introducing some indexes to measure the efficiency of retrieving operation. The problem is that information about retrieving operation is not available which makes detailed retrieving operation cannot be considered. However, for transshipment containers, the receiving and retrieving schedule can be obtained from berth allocation and quay crane scheduling plans. In Lee et al. (2006), consignment strategy is applied to a transshipment hub and a mixed integer programming model is proposed for the space allocation. In their paper, the yard template is predetermined and sub-blocks are assigned to certain vessels in advance. Kang et al. (2006) present a method for deriving a strategy for stacking containers with uncertain weight information in order to reduce the number of re-handles occurring during loading operation. This kind of storage allocation decision focuses on slot position (stack and tier) within a bay.

7.2.2 Yard Crane Scheduling

There are also some studies dealing with yard crane scheduling in order to speed up loading by optimizing the retrieving process. Kim and Kim (1999b) focus on optimal routing for a single transfer crane to do retrieval operation within a block. The problem is to minimize the total handling time of a transfer crane with the constraint of loading sequence. In Zhang et al. (2002), yard crane deployment problem is analyzed in order to allocate RTGCs among all the yard blocks with respect to the workload distribution. Times and routes of crane movements among blocks can be obtained from the model based on the forecasted workload of each block. In Lee et al. (2007), the single transfer crane scheduling problem is extended to two transfer cranes scheduling among two blocks. Bay visiting sequence and number of containers picked up at each visit of the two cranes are determined simultaneously.

7.2.3 Transshipment-related Problem

Research on transshipment terminals is a new trend and the literature is not as rich as that on import/export container terminals. Cordeau et al. (2007) study a tactical problem called Service Allocation Problem (SAP) for the yard management of a transshipment container terminal. The objective is to minimize the container rehandling operations resulting from yard to yard container transfer called housekeeping. Moccia and Astorino (2007) present an operational problem called Group Allocation Problem (GAP) also dealing with transshipment containers with the objective of minimizing all the handling cost of containers. The group concept is used in order to treat the container flows through the yards in an aggregate level. Data for the container groups like the arrival and departure times and the arrival and departure positions along the quay is assumed to be known and is used to help the dynamic allocation of groups through their duration-of-stay inside the terminal.

Based on the literature, two major deficiencies can be found listed as follows. Also, we introduce our motivations of this study.

- Aggregate level and individual level of space allocation problems are to decide which block and which slot within a bay to be assigned for an incoming container. These two problems have been well analyzed in the literature which could be considered as macroscopic and microscopic problems. However, no study has been done to support bay allocation decision of which bay to be assigned within a block. In this study, the mesoscopic problem of bay-level space allocation is analyzed.

- Receiving operation and retrieving operation are analyzed separately in the literature for export/import container terminals. There is a need to incorporate both of them for yard management of container transshipment terminals in order to ensure a more efficient process in storage yard.

7.3 Problem Formulation

In this section, the integrated bay allocation and yard crane scheduling problem for transshipment containers is put forward and a mathematical model is developed.

7.3.1 Problem Description

In this paper, the consignment strategy is used for transshipment containers which is a practical method for container storage and is introduced in the literature (Lee et al., 2006; Chen et al., 1995; Davies and Bischoff, 1999). The convention is to group containers according to their inbound vessel, outbound vessel, container types, etc. All the containers within the same group are then aggregated and considered as a whole unity which is received and retrieved together. Usually groups are not mixed within the same stack and no re-handle is needed during retrieving operation. That's why consignment strategy is commonly applied in practice. Figure 7.2 shows an example of the consignment strategy for a bay where 3 groups are located. They are stored at different stacks and not mixed with each other. Note that if the amount of a group exceeds the capacity of a whole bay, the group should be further divided.

For transshipment containers, since detailed information (unloading and loading time) is usually available in advance from the berth allocation and quay crane scheduling plans, we can integrate receiving and retrieving operation for yard crane operation.

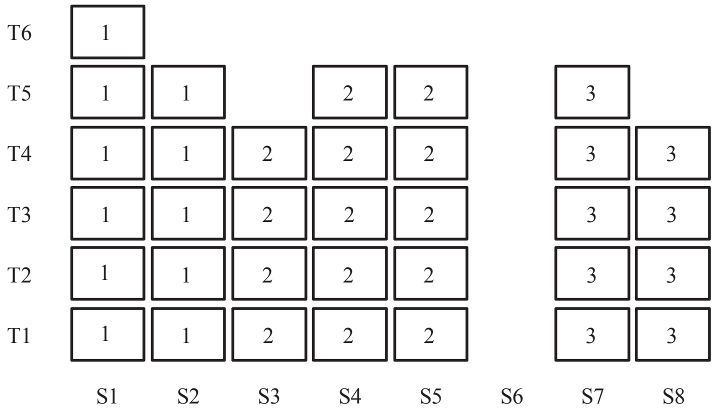


Figure 7.2: A consignment strategy for storage of a yard bay

The problem in this paper is to allocate yard-bay resource to incoming container groups in the planning horizon within a yard block taking into account both receiving and retrieving process. The objective is to minimize total cost occurring during the whole process including setup cost and travel cost of the yard crane as well as task delay. In the following the operation of one group is referred to as a task.

7.3.2 Assumption

To simplify the problem, the following assumptions are made.

(1) Retrieving operation begins after the completion of receiving operations.

(2) The yard block is served by one yard crane (YC).

(3) Re-handle operation is not considered in this paper.

(4) Detailed information of tasks is available i.e. scheduled loading/unloading time window.

Usually containers unloaded from a vessel will be stored in yards for some time before being loaded to another vessel. That's why we make assumption (1). Assumption (3) is reasonable because of the consignment strategy. The scheduled loading/unloading data for vessels can be gathered from Berth Allocation Plan (BAP) and quay crane scheduling (QCSP) (Moccia and

Astorino, 2007). As both the BAP and QCSP are determined at a higher decision level than the bay allocation plan, assumption (4) holds.

7.3.3 Formulation

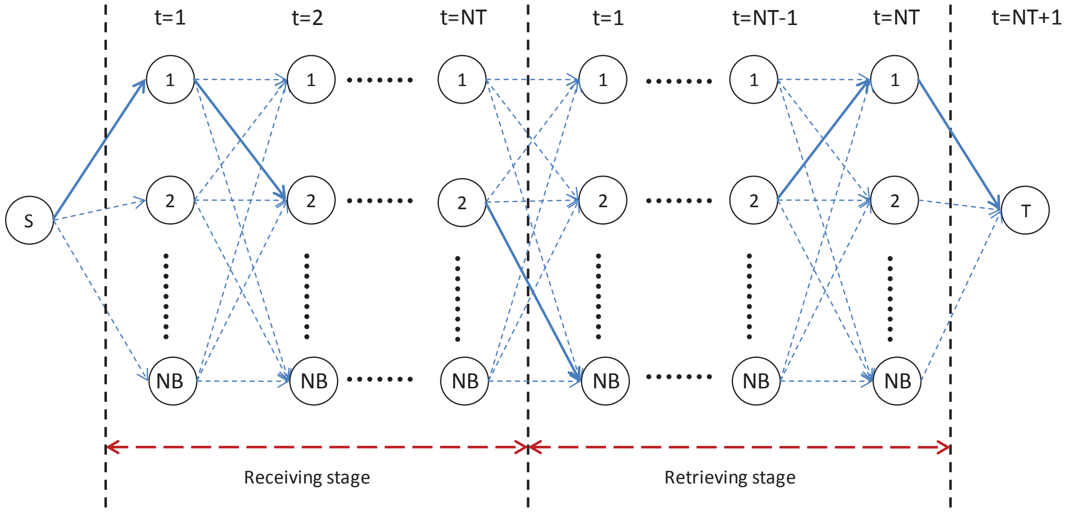


Figure 7.3: A network flow representation of the integrated problem

The integrated bay allocation and yard crane scheduling problem can be considered as a network problem with two stages. The first stage refers to the receiving operation while the second stage considers the retrieving operation. Figure 7.3 shows a network flow representation for the integrated problem. The network is represented by a set of vertices and a set of arcs. A path from source node S to sink node T represents the bay visiting decisions for YC in receiving stage and retrieving stage. Every path consists of a number of tours which represent the YC's moving between bays. At each tour of the receiving stage, all the bays are candidate positions for storage and YC goes to the allocated bay to store the task. At the retrieving stage, YC needs to go to the storage position of the current task for retrieval. So a path from S to T corresponds to a YC movement plan for receiving and retrieving the tasks. Take Figure 7.3 as an example, the path indicated by solid arcs shows that the YC moves from initial bay to Bay 1 to store the first task at receiving tour 1 and then moves to Bay 2 for the next task at receiving tour 2. Similarly, YC moves from Bay 2 to Bay 1 to retrieve the last task at retrieving tour

NT. The last tour represents the YC's moving to last position which is a dummy task.

Indices:

i, j, k : the index for bays

m : the index for tasks

t : the index for tours

Parameters:

NB : total number of bays

NT : total number of tasks as well as total number of tours in one stage

Φ : the set of bays, $|\Omega| = NB$

Ψ : the set of tasks, $|\Psi| = NT$

I_0 : dummy task which represents the task of sink node

Ψ_0 : the set of all tasks including dummy task, $|\Psi_0| = \Psi \bigcup \{I_0\}$

Ω : the set of tours, $|\Omega| = 2NT + 1$

(u_{m1}, v_{m1}) : the scheduled receiving operation time window of task $m, m \in \Psi$ while u_{m1} is the starting time and v_{m1} is the end time

(u_{m2}, v_{m2}) : the scheduled receiving operation time window of task $m, m \in \Psi$ while u_{m2} is the starting time and v_{m2} is the end time

d_{ij} : distance between bay i and j, measured in $|i - j|, i, j \in \Phi$

e_{ij} : set to 1 if $i \neq j$, and 0 otherwise

T_d : YC travel time for one bay distance

T_s : YC set-up time occurring after bay travel (set-up time is incurred since YC needs to position at the exact bay position and stop the sway of the hoist after each bay travel)

q_i : the capacity of bay i measured in number of stacks. For an empty by with 8 stacks, $q_i = 8$

p_m : the storage space requirement of task m measured in stacks, for example in Figure 7.2, $p_m = 2$

w_1 : the weightage of YC cost in the objective function

w_2 : the weightage of delay cost in the objective function

M : a relatively large number

Decision variables:

X^t_{ijm} : 1, if YC travels from bay i to j to store task m at receiving tour t; 0, otherwise

Y^t_{ijm} : 1, if YC travels from bay i to j to pick up task m at retrieving tour t; 0, otherwise

C_{m1} : completing time of receiving task m

C_{m2} : completing time of retrieving task m

Objective function:

$$min \left\{ w_1 \sum_{t\in\Omega} \sum_{i\in\Phi} \sum_{j\in\Phi} \sum_{m\in\Psi} (X^t_{ijm} + Y^t_{ijm})(T_d d_{ij} + T_s e_{ij}) + w_2 \sum_{m\in\Psi} [(C_{m1} - v_{m1}) + (C_{m2} - v_{m2})] \right\} \tag{7.1}$$

Constraints:

$$\sum_j \sum_m X^1_{Sjm} = 1 \tag{7.2}$$

$$-\sum_i \sum_{m\in\{I_0\}} Y^{NT+1}_{iTm} = -1 \tag{7.3}$$

$$\sum_i \sum_m X^t_{ijm} - \sum_k \sum_n X^{t+1}_{jkn} = 0 \quad \forall j, t \tag{7.4}$$

$$\sum_i \sum_m Y_{ijm}^t - \sum_k \sum_n Y_{jkn}^{t+1} = 0 \quad \forall j, t \tag{7.5}$$

$$\sum_i \sum_m X_{ijm}^{NT} - \sum_k \sum_n Y_{jkn}^1 = 0 \quad \forall j \tag{7.6}$$

$$\sum_t \sum_i \sum_j X_{ijm}^t = 1 \quad \forall m \tag{7.7}$$

$$\sum_t \sum_j \sum_m p_m X_{ijm}^t \leq q_i \quad \forall i \tag{7.8}$$

$$C_{n1} - (v_{n1} - u_{n1}) \geq C_{m1} + T_d d_{ij} + T_s e_{ij} + M \left(\sum_k X_{kim}^{t-1} + X_{ijn}^t - 2 \right) \quad \forall i, j, m, n, t \tag{7.9}$$

$$C_{n2} - (v_{n2} - u_{n2}) \geq C_{m2} + T_d d_{ij} + T_s e_{ij} + M \left(\sum_k Y_{kim}^{t-1} + Y_{ijn}^t - 2 \right) \quad \forall i, j, m, n, t \tag{7.10}$$

$$C_{n2} - (v_{n2} - u_{n2}) \geq C_{m1} + T_d d_{ij} + T_s e_{ij} + M \left(\sum_k X_{kim}^{NT} + Y_{ijn}^1 - 2 \right) \quad \forall i, j, m, n \tag{7.11}$$

$$C_{m1} \geq v_{m1} \quad \forall m \tag{7.12}$$

$$C_{m2} \geq v_{m2} \quad \forall m \tag{7.13}$$

$$\sum_t \sum_i X_{ijm}^t = \sum_t \sum_i Y_{ijm}^t \quad \forall j, m \tag{7.14}$$

$$X_{ijm}^t, Y_{ijm}^t \in \{0, 1\} \quad \forall i, j, m, t \tag{7.15}$$

$$C_{m1}, C_{m2} \geq 0 \quad \forall m \tag{7.16}$$

Constraints (7.2)-(7.6) are the flow conservation constraints. Constraint (7.2) deals with the outflow requirement at the source node while Constraint (7.3) ensures the inflow requirement at the sink node. Constraints (7.4) and (7.5) are to ensure the flow conservation at other nodes for receiving and retrieving stages respectively. For the linking of these two stages, Constraints (7.6) ensure the flow conservation at the starting point of retrieving stage. Constraints (7.7) are the task distribution requirements which ensure that one task could be put into one bay. Constraints (7.8) are to guarantee the total amount of tasks in one bay do not exceed the capacity. Constraints (7.9)-(7.11) are the operation time requirements for two consecutive tasks because the latter one could only begin after the yard crane finishes the former task and moves to the bay position that is assigned to the latter one. Constraints (7.9) correspond to receiving stage. For example, task m and n are received at tour $t-1$ and t and are put in bay i and j respectively, the part in the bracket will become 0. The left hand side represents the starting time of task n and the right hand side indicates the completion time of task m plus the YC travel time and set-up time. Hence, the constraints ensure that there is no time overlapping between the two tasks. Similarly, Constraints (7.10) and (7.11) correspond to retrieving stage and the linkage of the two stages. Constraints (7.12) and (7.13) guarantee tasks could only start after they are ready. Constraints (7.14) are the position requirements which ensure the retrieving position of one task is the same as the position that is assigned to the same task during receiving operation. Finally, Constraints (7.15) and (7.16) specify the domain for the decision variables.

7.4 Simulated Annealing Heuristic

It is well known that the single yard crane scheduling problem is an NP-complete problem (Narasimhan and Palekar, 2002). As the problem in this paper can be reduced to yard crane scheduling problem with determined bay allocation decision, it is more complex and exact algorithms are not suitable to solve practical size problems. Hence, heuristic algorithms are required to solve this integrated problem efficiently. In this study, a simulated annealing (SA) heuristic algorithm is developed to solve the problem.

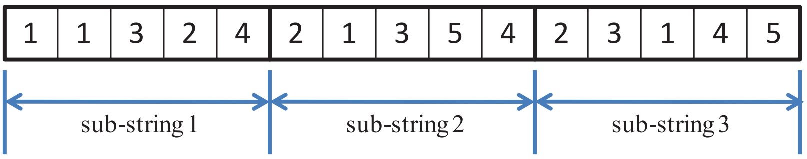


Figure 7.4: An example of solution representation

For solution encoding, Figure 7.4 shows an example of the coding of candidate solutions. The coding string constitutes of three sub-strings. The first sub-string indicates the bay allocation decision while the later 2 sub-strings represent receiving and retrieving sequences. For example, the first number in sub-string 1 which is 1 indicates that Task 1 is put into Bay 1. In sub-string

2, the first number is 2 which means Task 1 is the second task during receiving operation. It is similar for sub-string 3.

To generate the neighborhood solutions, the method shown in the Figure 7.5 is applied including two patterns called pair-wise interchange and flipping. For sub-string 1, either of the two patterns will be conducted with an equal probability. For sub-string 2 and 3, only pair-wise interchange operation will be adopted since flipping will generate two tasks with the same sequence. For Pattern 1, the pair-wise interchange positions are randomly determined firstly and then the numbers are interchanged. For Pattern 2, the flipping position is randomly generated and the number is flipped into another number randomly. At each the neighborhood solution generation process, only one of the three sub-strings will change while the other two keep unchanged, and they change with equal probability.

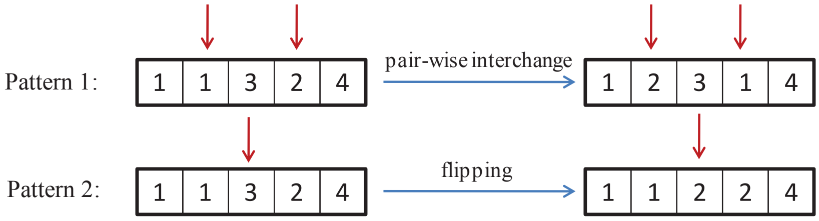


Figure 7.5: Neighborhood solution generation method

Since the encoding string represents the decisions of bay allocation and operation sequences, YC carries on the tasks as early as possible. The objective function can be calculated with respect to the candidate string easily. Then the following criterion is adopted to judge whether to accept it or not. Let, $\Delta = f(s) - f(s_0)$ where s_0 represents the current solution and s represents the neighborhood solution generated from current solution. f_s indicates the objective value of solution s. A random number r in $[0, 1)$ generated from a uniform distribution. The neighborhood solution will be accepted if $r \leq exp(-\Delta/T_i)$ where T_i represents the current temperature. If not, the current solution will be kept to the next iteration.

The temperature is updated by the following equation: $T_{i+1} = \frac{T_i}{1+\beta T_i}$ where $\beta = \frac{T_1 - T_K}{(K-1)T_1 T_K}$. The SA searching process will be terminated if either current temperature T_i falls below T_K or current iteration number exceeds K.

7.5 Numerical Experiments

7.5.1 Lower Bound

In order to get a lower bound, the integrated problem is decomposed into two independent problems namely YC cost problem (P1) and task delay problem (P2). P1 is to determine the minimum YC cost. Firstly number of bays that can accommodate all the tasks could be obtained easily by only considering the space requirement information. Then, under the assumption that all the tasks can be processed at any time, the minimum yard crane cost could be easily calculated. The objective function only involves YC costs including travel cost and set-up cost. For the problem of P2, it assumes that one bay could accommodate all the tasks which means there is no yard crane costs. Under this assumption, the problem tries to obtain the optimal operation delay for all the tasks. This problem is very similar to dynamic berth allocation problem formulated by Imai et al. (2001) while our problem can be considered as the one dimension dynamic berth allocation. So we treat bay as berth and formulate our problem according to the dynamic berth allocation problem.

YC cost problem (P1)

Decision variables:

α_{mk} : set to 1 if task m is put into bay k, and 0 otherwise

β_k : set to 1 if bay k is occupied by certain tasks, and 0 otherwise

Objective function:

$$\min \left\{ 2 \left(\sum_k \beta_k - 1 \right) (T_d + T_s) \right\} \tag{7.17}$$

Constraints:

$$\sum_k \alpha_{mk} = 1 \quad \forall m \tag{7.18}$$

$$\sum_m p_m \alpha_{mk} \leq q_i \quad \forall k \tag{7.19}$$

$$\sum_m p_m \alpha_{mk} - M\beta_k \leq 0 \quad \forall k \tag{7.20}$$

$$\alpha_{mk} \in \{0,1\} \quad \forall m,k \tag{7.21}$$

$$\beta_k \in \{0,1\} \quad \forall k \tag{7.22}$$

In the objective function (7.17), $\sum_k \beta_k$ represents the number of bays that are occupied by tasks. As a result, YC at least needs to move $2\,(sum_k beta_k - 1)$ times and each time the least YC cost is $(T_d + T_s)$. Hence, (7.17) indicates the minimum YC costs. Constraints (7.18) ensure that each task should be put into one bay and constraints (7.19) represent the bay capacity requirement for each bay. Constraints (7.20) ensure the definition of decision variables β_k. Constraints (7.21) and (7.22) specify the domain for the decision variables.

Task delay problem (P2)

Decision variables:

δ_{mn} : set to 1 if task m is served as the n^{th} task, and 0 otherwise

γ_{mn} : idle time between the completion of $(n-1)^{th}$ and n^{th} task when task m is served as the n^{th} task

Objective function:

$$\min \left\{ \sum_m \sum_n \left[(NT-n)(v_m-u_m)-u_m\right]\delta_{mn} + \sum_m \sum_n (NT-n+1)\gamma_{mn} \right\} \tag{7.23}$$

Constraints:

$$\sum_n \delta_{mn} = 1 \quad \forall m \tag{7.24}$$

$$\sum_m \delta_{mn} = 1 \quad \forall n \tag{7.25}$$

$$\sum_{h|h<n} \sum_{l} (v_l - u_l)\delta_{lh} + \gamma_{lh} \geq u_m \delta_{mn} \quad \forall m, n \tag{7.26}$$

$$\delta_{mn} \in \{0, 1\} \quad \forall m, n \tag{7.27}$$

$$\gamma_{mn} \geq 0 \quad \forall m, n \tag{7.28}$$

Objective function (7.23) represents the total tasks delay. Constraints (7.24) ensure each task should be served and Constraints (7.25) indicate only one task can be served at a time. Constraints (7.26) ensure the definition of decision variables $gamma_{mn}$. Constraints (7.27) and (7.28) specify the domain for the decision variables.

P1 and P2 are independent in that P1 only considers the YC costs and P2 only deals with task delay. The interaction of YC traveling and task operation is not considered in either of the two problems. Hence, the optimal objective value of P1 is a lower bound of the yard crane cost part in the integrated problem. Similarly, the task delay part in the integrated problem will have a higher objective value than P2. Then, the sum of optimal objective values of P1 and P2 could serve as a lower bound for the integrated problem.

We define LB_1 and LB_2 as the optimal objective values of P1 and P2 respectively and the two reduced problems are relatively simple and can be solved by a commercial solver within a reasonable time, we set the lower bound of the integrated problem by (7.29) and treat it as a benchmark for the numerical experiments.

$$LB = w_1 LB_1 + w_2 LB_2 \tag{7.29}$$

7.5.2 Small Scale Experiments

In order to calibrate the parameters of SA, a sensitivity analysis is conducted in advance and the following parameters are determined:

- Initial temperature $T_1 = 1 \times 10_4$
- Stopping temperature $T_K = 0.001$
- Iteration number $K = 1 \times 10_4$

For test instance generation, 15 instances are generated for each instance set at three different YC occupancy level. Occupancy level refers to the ratio of working time versus total time for YC and reflects how busy for the YC in conducting the receiving and retrieving operation. Why we generate instances at different YC occupancy levels is because the algorithm can be fully tested in order to see the performance at different workload circumstances. All the instances are randomly generated as follows: we set $q_i = 1$ unit $\forall i \in \Phi$ and the storage space requirement p_m is randomly distributed in $[0.3q_i, q_i]$. With the group information and the occupancy level, the whole planning horizon can be calculated. Then, the receiving and retrieving operation schedule of each group is randomly distributed within the planning horizon. We set $w_1 = 0.8$, $w_2 = 0.2$ in the objective function.

The SA algorithm is coded in Matlab and CPLEX 12.1 is used to get the optimal objective values of the integrated problem, as well as problem P1 and P2. All the experiments are running on a PC with 3 GHz CPU and 4G RAM. For each instance, we run SA 10 times and take the average objective value as the result.

Firstly small scale numerical experiments with five tasks and five bays are conducted. Table 7.1 presents the results of the small scale instances by SA algorithm and CPLEX. As can be seen, SA algorithm shows a good performance as the gap between the SA and optimal values is only 0.1% on average. Besides, the small standard deviations indicate the robustness of the SA algorithm. In terms of computational efficiency, the CPU time of SA is about 3.8 seconds while it can be hours for CPLEX to find optimal solutions even for such small scale instances. The good performance shown in the small scale experiments proves that SA is a promising algorithm for the studied problem.

In Table 7.1, the comparison of SA and lower bounds is also provided. As can be seen, the gap between SA and lower bounds is 7.4% on average although the SA's solutions are very close to optimal ones. This implies that the lower bounds are not very tight.

7.5.3 Large Scale Experiments

In order to test the performance of SA for large scale experiments, 45 instances are generated with 3 sets and 3 different YC occupancy levels. Table (7.2)-(7.4) show the performance of SA

Table 7.1: Small scale numerical experiments (5 tasks × 5 bays)

Occupancy	Instance	LB^a	CPLEX		SA		$GAP1^b$	$GAP2^c$
Level	No.		Optimal	Cpu_time	Mean	Std.	(%)	(%)
0.5	1	100.4	106.4	2935s	106.6	0.0	6.2	0.2
	2	151.2	161.8	3017s	162.0	0.5	7.1	0.1
	3	96.4	105.0	2910s	105.0	0.0	8.9	0.0
	4	95.2	99.2	1913s	99.2	0.0	4.2	0.0
	5	84.8	89.6	8649s	89.6	0.0	5.7	0.0
0.7	1	89.0	95.6	5347s	95.7	0.3	7.6	0.1
	2	96.4	104.6	4067s	104.9	0.7	8.9	0.3
	3	223.8	231.4	14435s	231.9	1.1	3.6	0.2
	4	55.0	62.2	3117s	62.2	0.0	13.1	0.0
	5	42.6	48.2	2496s	48.2	0.0	13.1	0.0
0.9	1	105.0	111.6	3518s	111.8	0.3	6.5	0.2
	2	176.6	184.4	50998s	185.5	1.0	5.1	0.6
	3	298.4	319.0	38117s	319.0	0.1	6.9	0.0
	4	85.6	92.4	3721s	92.6	0.5	8.1	0.2
	5	290.2	309.0	9042s	309.0	0.0	6.5	0.0
Average							7.4	0.1

[a] : LB : lower bound
[b] : $GAP1$=(SA-LB)/LB
[c] : $GAP2$=(SA-CPLEX)/CPLEX

for large scale numerical experiments and the comparison of SA and LB. It is noted that the standard deviation of SA becomes larger as the scale increases. This indicates SA for larger scale problems is not as stable as that for small scale problems. However, SA is very efficient as the computational time for large scale experiments is also relatively small. For LB, it is obtained from CPLEX by solving the problem P1 and P2. Even for the reduced problems, CPLEX cannot find optimal solutions when the instance scale is large. Only 2 out of 15 instances for the set of 30 × 30 can be solved optimally. For the rest 13 instances, we provide the truncated objective values when the computational time reaches 24 hours. As a result, the LB for the 13 instances is not real lower bounds and the gaps for these instances listed in Table (7.4) are smaller than the real gaps between SA and LB. That's why a negative gap appears.

From small scale experiments to large scale ones, the average gap increases. The reason for the increased gap could be either of the poor estimation of LB or the poor performance of SA

Table 7.2: Large scale numerical experiments (10 tasks × 10 bays)

Occupancy	Instance	LB	SA			GAP^a	Average
Level	No.		Mean	Std.	Time	(%)	(%)
0.5	1	109.8	135.9	1.4	12.8s	23.8	12.9
	2	235.8	260.6	1.9		10.5	
	3	220.6	240.8	3.0		9.2	
	4	511.6	563.4	5.7		10.1	
	5	133.8	148.7	2.1		11.2	
0.7	1	374.2	398.3	2.2	12.8s	6.4	7.4
	2	251.2	273.1	1.5		8.7	
	3	559.2	603.8	3.1		8.0	
	4	411.2	444.0	3.1		8.0	
	5	357.0	377.5	3.2		5.7	
0.9	1	1231.4	1257.7	11.1	12.8s	2.1	6.3
	2	743.2	803.9	3.2		8.2	
	3	586.2	608.8	3.3		3.9	
	4	378.2	412.5	2.3		9.1	
	5	719.4	779.3	3.5		8.3	

^{a} : GAP=(SA-LB)/LB

Table 7.3: Large scale numerical experiments (20 tasks × 20 bays)

Occupancy	Instance	LB	SA			GAP^a	Average
Level	No.		Mean	Std.	Time	(%)	(%)
0.5	1	584.6	697.3	8.3	149s	19.3	19.4
	2	506.0	640.4	6.7		26.6	
	3	606.6	713.4	8.6		17.6	
	4	710.0	822.4	6.6		15.8	
	5	795.8	934.9	9.3		17.5	
0.7	1	1370.0	1529.7	23.3	149s	11.7	14.5
	2	658.8	784.9	16.2		19.1	
	3	727.4	860.2	9.7		18.3	
	4	1399.0	1539.9	6.0		10.1	
	5	987.2	1120.1	15.5		13.5	
0.9	1	1586.2	1770.8	19.8	149s	11.6	14.1
	2	1825.8	2075.5	37.6		13.7	
	3	1513.2	1693.9	24.0		11.9	
	4	1235.0	1479.4	34.4		19.8	
	5	1508.4	1714.8	22.6		13.7	

^{a} : GAP=(SA-LB)/LB

Table 7.4: Large scale numerical experiments (30 tasks × 30 bays)

Occupancy	Instance	LB	SA			GAP^b	Average
Level	No.		Mean	Std.	Time	(%)	(%)
0.5	1	701.0^a	953.3	28.4	791s	36.0	14.1
	2	1211.6^a	1349.3	18.6		11.4	
	3	963.2^a	966.7	26.8		0.4	
	4	1658.2^a	1985.1	70.7		19.7	
	5	1087.4^a	1123.1	30.5		3.3	
0.7	1	1849.6^a	1821.9	17.1	791s	-1.5	15.5
	2	2055.2^a	2363.6	63.4		15.0	
	3	1991.0^a	2300.4	27.6		15.5	
	4	1084.2^a	1399.8	129.7		29.1	
	5	2121.4^a	2527.9	60.4		19.2	
0.9	1	3347.8^a	3948.6	201.7	791s	17.9	26.8
	2	2131.0	2773.5	122.2		30.1	
	3	2160.2^a	2799.0	102.9		29.6	
	4	4030.0^a	4676.5	175.5		16.0	
	5	2488.8	3494.0	205.4		40.4	

[a] : Truncated lower bound by CPLEX for 24 hrs

[b] : GAP=(SA-LB)/LB

algorithm. However, the small scale experiments show that the gap is mostly due to the loose estimation of LB. And this makes us believe that the loose LB contributes to the large gap to a large extent for these large scale experiments.

Numerical experiments are also conducted to see how well the integrated problem improves the operation efficiency of YC by comparing with the scenario that only yard crane scheduling is optimized and bays are assigned randomly in receiving operation. SA is employed to obtain the near optimal solutions for both of the integrated problem and the reference scenario. The reference scenario serves as a benchmark and the improvement from the integrated operation could be obtained as the absolute gap to the benchmark. Table (7.5) shows the improvement of the integrated problem. As can be seen, there is a significant improvement when receiving and retrieving operations are integrated. The average improvement is larger than 10%. Besides, the improvement for larger scale instances is higher than that of smaller scale instances.

Table 7.5: The improvement of the integrated operation (%)

Scale	Occupancy	Instance No.					Average
bay × task	Level	1	2	3	4	5	
10×10	0.5	17.7	14.9	13.2	9.3	23.0	12.3
	0.7	13.9	15.2	9.1	12.1	11.9	
	0.9	6.2	8.8	10.1	9.8	9.5	
20×20	0.5	19.0	18.0	20.5	15.8	13.9	16.0
	0.7	11.3	21.2	19.3	16.0	14.1	
	0.9	10.1	10.0	11.7	24.8	14.1	
30×30	0.5	27.3	23.7	31.9	11.8	22.0	19.0
	0.7	19.2	15.8	18.6	21.1	14.6	
	0.9	16.2	17.1	20.0	7.6	18.5	

7.6 Summary

In this study, the integrated bay allocation and yard crane scheduling problem for transshipment containers is proposed in order to achieve a more efficient operation for yard crane. The consignment strategy is employed to do bay space allocation for transshipment containers in a transshipment hub like port of Singapore. Unlike space allocation under the entire yard overview and slot assignment with in a yard bay, bay allocation problem focuses on a block and is to allocate bay resource to fleets of containers in a more efficient way. As yard crane scheduling problem mainly deals with retrieving operation during loading process, we incorporate receiving operation and consider them as a whole process. A mixed integer programming model is developed for the integrated problem. Due to the complexity of the problem, a simulated annealing (SA) heuristic is proposed to find near optimal solutions. Both small and large scale numerical experiments are generated to test the performance of the designed heuristic and show that the proposed SA heuristic is a promising method to handle the integrated problem. A lower bound is obtained by decomposing the integrated problem into two reduced problems. The numerical experiments also show that the efficiency of yard crane for the integrated problem gets improved from traditional yard crane scheduling operation.

In this study, the operation information of transshipment containers is assumed to be known which is obtained from quayside operations including berth allocation and yard crane scheduling plans. A more comprehensive study could be to investigate the relationship between quayside operations and the efficiency of yard crane operation. Besides, this study considers only the situation of one block served by one yard crane. This could be extended to a more complicated situation such as multiple yard cranes.

Chapter 8

Conclusions

8.1 Concluding Remarks

Storage yard management for container transshipment terminals has been receiving more and more attention in practice as well as in the research community. Due to the increasing container throughput and shortage of land in some major transshipment hubs such as the Port of Singapore, the efficiency of the storage yard operations has been well regarded. This thesis has developed a comprehensive planning framework for the storage yard management at container transshipment terminals. Focusing on storage yard allocation, the planning framework consists of individual topics with various planning levels (from tactical to operational), various planning areas (from multi-terminal, single-terminal, yard section to single yard block), and various planning horizons (from months, weeks, days to hours). Integrated optimization has been extensively conducted by simultaneously considering storage allocation with other inter-related decision problems such as berth allocation and crane deployment.

In **Chapter 3**, two practical problems arising in a container transshipment hub with multiple terminals are studied: terminal allocation problem for vessels which is to assign home terminals for cyclically visiting vessels, and yard allocation problem which is to decide the storage locations for transshipment flows between vessels. In a multi-terminal transshipment hub, port operators need to determine the calling terminals for vessels and to manage the transshipment flows within as well as between terminals. Unlike the management of a single terminal, multi-terminal system puts forward a problem of reducing the inter-terminal container movement which is a major concern of port operators. An integer programming model is formulated integrating the two problems with the objective of minimizing total inter-terminal and intra-terminal handling costs generated by transshipment flows. Due to the computational complexity of the problem, A 2-level heuristic algorithm is developed to obtain high quality solutions in an efficient way.

Chapter 4 studies the feeder vessel management problem which consists of designing preferred berthing positions (i.e., berth template) and service time (i.e., schedule template) for cyclically visiting feeders, and allocating storage spaces (i.e., yard template) to the transshipment flows between mother vessels and feeders. We consider the above three tactical decision problems simultaneously for a container transshipment terminal with an eye toward the quayside congestion and the housekeeping cost of container movements. Unlike the previous literature, we adopt a proactive management strategy from the container terminals' perspective and plan the schedule template for feeders' calling in order to balance the temporal distribution of quayside workload. Meanwhile, the berth and yard template are designed to reduce the container movement cost between the quayside and storage yard. The integrated problem is formulated as a mixed integer programming model and solved by a memetic heuristic approach. The proposed memetic heuristic outperforms a commercial solver for large-scale instances as shown by the computational experiments. Scenario analysis demonstrates the effectiveness of adjusting the feeder calling schedules and the integration with the berth and yard template design.

In **Chapter 5**, we develop a column generation based approach to the feeder vessel management problem studied in Chapter 4. We reformulate the problem via Dantzig-Wolfe decomposition and apply the column generation at the root node of the restricted master problem. A separate branch-and-bound procedure is employed to obtain integer solutions by CPLEX after the column generation procedure. Computational experiments have shown that the column generation based approach is more efficient than the memetic heuristic developed in Chapter 4 while achieving comparable solution quality.

Chapter 6 studies the daily storage yard manage problem arising in maritime container terminals, which integrates the space allocation and yard crane (YC) deployment decisions together with the consideration of container traffic congestion in the storage yard. The space allocation is conducted at the sub-block level and an YC deployment profile concept is introduced to model the YC activities in the storage yard. A particular attention is paid to the container traffic control so as to avoid potential traffic congestion in the storage yard. The integrated problem is formulated as an integer linear program with the objective of minimizing the YC operating cost and YC inter-block movement cost. We design a divide-and-conquer solution approach to solve the problem in an efficient manner in which harmony search and constraint satisfaction techniques are employed. Numerical results show that both of the optimization model and the heuristic approach are able to produce solutions with small optimality gap. Scenario analysis demonstrates the significant improvement from integrating the two decision problems.

Chapter 7 addresses the integrated problem for bay allocation and yard crane scheduling in transshipment container terminals. Unlike space allocation under the entire yard overview and slot assignment within a yard bay, bay allocation problem focuses on a block and aims to allocate bay resource to fleets of transshipment containers in a more efficient way. Receiving operation and retrieving operation in the storage yards are considered simultaneously to achieve a more efficient operation of yard crane. In this chapter, the bay allocation and the yard crane scheduling are integrated as a whole process. A mixed integer programming model is developed for the problem formulation with the objective of minimizing total costs including yard crane cost and delay cost. Considering the high complexity of the problem, a simulated annealing heuristic algorithm is proposed to obtain near optimal solutions.

8.2 Future Research

One promising direction of future research is to take uncertainty into consideration when modeling and solving the proposed individual research topics. In practice, terminal operations are highly dynamic and input information may not always be accurate. Therefore, stochastic optimization and robust optimization techniques could be applied for improving the robustness of the developed models and solution methods. Another interesting topic that deserves attention is container re-marshaling in the storage yard. In practice, container storage are not always in line with the planned scenarios due to various reasons. In such cases, re-marshaling for export and transshipment containers should be conducted. The planning and scheduling of the re-marshaling operations is an interesting optimization problem which remains to be explored.

Bibliography

Bierwirth, C., Meisel, F., 2010. A survey of berth allocation and quay crane scheduling problems in container terminals. European Journal of Operational Research 202 (3), 615–627.

Bish, E. K., 2003. A multiple-crane-constrained scheduling problem in a container terminal. European Journal of Operational Research 144 (1), 83–107.

Bish, E. K., Chen, F. Y., Leong, Y. T., Nelson, B. L., Ng, J. W. C., Simchi-Levi, D., 2005. Dispatching vehicles in a mega container terminal. OR Spectrum 27 (4), 491–506.

Bish, E. K., Leong, T. Y., Li, C. L., Ng, J. W. C., Simchi-Levi, D., 2001. Analysis of a new vehicle scheduling and location problem. Naval Research Logistics 48 (5), 363–385.

Cao, J. X., Lee, D.-H., Chen, J. H., Shi, Q., 2010. The integrated yard truck and yard crane scheduling problem: Benders' decomposition-based methods. Transportation Research Part E: Logistics and Transportation Review 46 (3), 344–353.

Cao, Z., Lee, D.-H., Meng, Q., 2008. Deployment strategies of double-rail-mounted gantry crane systems for loading outbound containers in container terminals. International Journal of Production Economics 115 (1), 221–228.

Chen, C. S., Lee, S. M., Shen, Q. S., 1995. An analytical model for the container loading problem. European Journal of Operational Research 80 (1), 68–76.

Chen, J. H., Lee, D.-H., Cao, J. X., 2011. Heuristics for quay crane scheduling at indented berth. Transportation Research Part E: Logistics and Transportation Review 47 (6), 1005–1020.

Cheung, R. K., Li, C. L., Lin, W. Q., 2002. Interblock crane deployment in container terminals. Transportation Science 36 (1), 79–93.

Cordeau, J. F., Gaudioso, M., Laporte, G., Moccia, L., 2007. The service allocation problem at the Giola Tauro maritime terminal. European Journal of Operational Research 176 (2), 1167–1184.

Cordeau, J. F., Laporte, G., Legato, P., Moccia, L., 2005. Models and tabu search heuristics for the berth-allocation problem. Transportation Science 39 (4), 526–538.

Daganzo, C. F., 1989. The crane scheduling problem. Transportation Research Part B-Methodological 23 (3), 159–175.

Davies, A. P., Bischoff, E. E., 1999. Weight distribution considerations in container loading. European Journal of Operational Research 114 (3), 509–527.

Dong, J.-X., Song, D.-P., 2009. Container fleet sizing and empty repositioning in liner shipping systems. Transportation Research Part E: Logistics and Transportation Review 45 (6), 860–877.

Geem, Z., 2009. Music-inspired harmony search algorithm: theory and applications. Vol. 191. Springer Verlag.

Giallombardo, G., Moccia, L., Salani, M., Vacca, I., 2010. Modeling and solving the tactical berth allocation problem. Transportation Research Part B: Methodological 44 (2), 232–245.

Guan, Y. P., Cheung, R. K., 2004. The berth allocation problem: models and solution methods. OR Spectrum 26 (1), 75–92.

Guo, X., Huang, S. Y., 2012. Dynamic space and time partitioning for yard crane workload management in container terminals. Transportation Science 46 (1), 134–148.

Han, Y. B., 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 (4), 697–720.

Hendriks, M. P. M., Armbruster, D., Laumanns, M., Lefeber, E., Udding, J. T., 2012. The strategic allocation of cyclically calling vessels for multi-terminal container operators. Flexible Services and Manufacturing Journal, In Press.

Imai, A., Chen, H. C., Nishimura, E., Papadimitriou, S., 2008. The simultaneous berth and quay crane allocation problem. Transportation Research Part E: Logistics and Transportation Review 44 (5), 900–920.

Imai, A., Nagaiwa, K., Tat, C. W., 1997. Efficient planning of berth allocation for container terminals in asia. Journal of Advanced Transportation 31 (1), 75–94.

Imai, A., Nishimura, E., Papadimitriou, S., 2001. The dynamic berth allocation problem for a container port. Transportation Research Part B: Methodological 35 (4), 401–417.

Imai, A., Nishimura, E., Papadimitriou, S., 2003. Berth allocation with service priority. Transportation Research Part B: Methodological 37 (5), 437–457.

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

Kim, K., Bae, J., 1999. A dispatching method for automated guided vehicles to minimize delays of containership operations. International Journal of Management Science 5 (1), 1–25.

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

Kim, K. H., Bae, J. W., 2004. A look-ahead dispatching method for automated guided vehicles in automated port container terminals. Transportation Science 38 (2), 224–234.

Kim, K. H., Hong, G. P., 2006. A heuristic rule for relocating blocks. Computers & Operations Research 33 (4), 940–954.

Kim, K. H., Kim, H. B., 1998. The optimal determination of the space requirement and the number of transfer cranes for import containers. Computers & Industrial Engineering 35 (3-4), 427–430.

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

Kim, K. H., Kim, H. B., 2002. The optimal sizing of the storage space and handling facilities for import containers. Transportation Research Part B: Methodological 36 (9), 821–835.

Kim, K. H., Kim, K. Y., 1999b. An optimal routing algorithm for a transfer crane in port container terminals. Transportation Science 33 (1), 17–33.

Kim, K. H., Lee, K. M., Hwang, H., 2003. Sequencing delivery and receiving operations for yard cranes in port container terminals. International Journal of Production Economics 84 (3), 283–292.

Kim, K. H., Moon, K. C., 2003. Berth scheduling by simulated annealing. Transportation Research Part B: Methodological 37 (6), 541–560.

Kim, K. H., Park, K. T., 2003a. Dynamic space allocation for temporary storage. International Journal of Systems Science 34 (1), 11–20.

Kim, K. H., Park, K. T., 2003b. A note on a dynamic space-allocation method for outbound containers. European Journal of Operational Research 148 (1), 92–101.

Kim, K. H., Park, Y. M., 2004. A crane scheduling method for port container terminals. European Journal of Operational Research 156 (3), 752–768.

Kim, K. H., Park, Y. M., Ryu, K.-R., 2000. Deriving decision rules to locate export containers in container yards. European Journal of Operational Research 124 (1), 89–101.

Kim, K. Y., Kim, K. H., 1997. A routing algorithm for a single transfer crane to load export containers onto a containership. Computers & Industrial Engineering 33 (3-4), 673–676.

Lee, D.-H., Cao, J. X., Shi, Q., Chen, J. H., 2009. A heuristic algorithm for yard truck scheduling and storage allocation problems. Transportation Research Part E: Logistics and Transportation Review 45 (5), 810–820.

Lee, D.-H., Cao, Z., Meng, Q., 2007. Scheduling of two-transtainer systems for loading outbound containers in port container terminals with simulated annealing algorithm. International Journal of Production Economics 107 (1), 115–124.

Lee, D.-H., Chen, J. H., Cao, J. X., 2010a. The continuous berth allocation problem: A greedy randomized adaptive search solution. Transportation Research Part E: Logistics and Transportation Review 46 (6), 1017–1029.

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

Lee, D.-H., Jin, J. G., Chen, J. H., 2011. Integrated bay allocation and yard crane scheduling problem for transshipment containers. Transportation Research Record: Journal of the Transportation Research Board 2222, 63–71.

Lee, D.-H., Jin, J. G., Chen, J. H., 2012. Terminal and yard allocation problem for a container transshipment hub with multiple terminals. Transportation Research Part E: Logistics and Transportation Review 48 (2), 516–528.

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

Lee, L. H., Chew, E. P., Tan, K. C., Han, Y. B., 2006. An optimization model for storage yard management in transshipment hubs. OR Spectrum 28 (4), 539–561.

Lee, L. H., Chew, E. P., Tan, K. C., Wang, Y., 2010b. Vehicle dispatching algorithms for container transshipment hubs. OR Spectrum 32 (3), 663–685.

Lim, A., 1998. The berth planning problem. Operations Research Letters 22 (2-3), 105–110.

Linn, R. J., Zhang, C. Q., 2003. A heuristic for dynamic yard crane deployment in a container terminal. IIE Transactions 35 (2), 161–174.

Liu, J. Y., Wan, Y. W., Wang, L., 2006. Quay crane scheduling at container terminals to minimize the maximum relative tardiness of vessel departures. Naval Research Logistics 53 (1), 60–74.

Maersk, 2012. Maersk line schedules. `http://www.maerskline.com/link/?page=brochure&path=/routemaps`, accessed on June 12, 2012.

Martello, S., Toth, P., 1992. Generalized assignment problems. In: Algorithms and Computation. Vol. 650 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 351–369.

Meisel, F., Bierwirth, C., 2009. Heuristics for the integration of crane productivity in the berth allocation problem. Transportation Research Part E: Logistics and Transportation Review 45 (1), 196–209.

Moccia, L., Astorino, A., 2007. The group allocation problem in a transshipment container terminal. In: Proceedings of World Conference on Transport Research. University of California at Berkeley.

Moccia, L., Cordeau, J. F., Gaudioso, M., Laporte, G., 2006. A branch-and-cut algorithm for the quay crane scheduling problem in a container terminal. Naval Research Logistics 53 (1), 45–59.

Moccia, L., Cordeau, J. F., Monaco, M. F., Sammarra, M., 2009. A column generation heuristic for a dynamic generalized assignment problem. Computers & Operations Research 36 (9), 2670–2681.

Monaco, M. F., Moccia, L., Sammarra, M., 2009. Operations research for the management of a transhipment container terminal: The Gioia Tauro case. Maritime Economic & Logistics 11 (1), 7–35.

Monaco, M. F., Sammarra, M., 2007. The berth allocation problem: A strong formulation solved by a lagrangean approach. Transportation Science 41 (2), 265–280.

Moorthy, R., Teo, C. P., 2006. Berth management in container terminal: the template design problem. OR Spectrum 28 (4), 495–518.

Narasimhan, A., Palekar, U. S., 2002. Analysis and algorithms for the transtainer routing problem in container port operations. Transportation Science 36 (1), 63–78.

Ng, W. C., 2005. Crane scheduling in container yards with inter-crane interference. European Journal of Operational Research 164 (1), 64–78.

Ng, W. C., Mak, K. L., Li, M. K., 2010. Yard planning for vessel services with a cyclical calling pattern. Engineering Optimization 42 (11), 1039–1054.

Ng, W. C., Mak, K. L., Zhang, Y. X., 2007. Scheduling trucks in container terminals using a genetic algorithm. Engineering Optimization 39 (1), 33–47.

Nishimura, E., Imai, A., Papadimitriou, S., 2005. Yard trailer routing at a maritime container terminal. Transportation Research Part E: Logistics and Transportation Review 41 (1), 53–76.

Park, Y. M., Kim, K. H., 2003. A scheduling method for berth and quay cranes. OR Spectrum 25 (1), 1–23.

Slack, B., Comtois, C., McCalla, R., 2002. Strategic alliances in the container shipping industry: a global perspective. Maritime Policy & Management 29 (1), 65–76.

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

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

UNCTAD, 2008. Review of maritime transport. Technical report 0566-7682, United Nations.

Vacca, I., Bierlaire, M., Salani, M., 2007. Optimization at container terminals: Status, trends and perspectives. Tech. Rep. TRANSP-OR 071204, Transport and Mobility Laboratory, EPFL.

Vacca, I., Salani, M., Bierlaire, M., 2010. Optimization of operations in container terminals: hierarchical vs integrated approaches. In: Swiss Transport Research Conference.

Vacca, I., Salani, M., Bierlaire, M., 2012. An exact algorithm for the integrated planning of berth allocation and quay crane assignment. Transportation Science, In Press.

Wan, Y. W., Liu, J. Y., Tsai, P. C., 2009. The assignment of storage locations to containers for a container stack. Naval Research Logistics 56 (8), 699–713.

Wang, S., Meng, Q., 2012. Liner ship fleet deployment with container transshipment operations. Transportation Research Part E: Logistics and Transportation Review 48 (2), 470–484.

Won, S. H., Zhang, X., Kim, K. H., 2012. Workload-based yard-planning system in container terminals. Journal of Intelligent Manufacturing doi:10.1007/s10845-011-0565-x.

Zhang, C., Liu, J., Wan, Y.-W., Murty, K. G., Linn, R. J., 2003. Storage space allocation in container terminals. Transportation Research Part B: Methodological 37 (10), 883–903.

Zhang, C., Wan, Y. W., Liu, J. Y., Linn, R. J., 2002. Dynamic crane deployment in container storage yards. Transportation Research Part B: Methodological 36 (6), 537–555.

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 (4), 483–504.

Zhu, Y., Lim, A., 2006. Crane scheduling with non-crossing constraint. Journal of the Operational Research Society 57 (12), 1464–1471.

Appendix

Awards during PhD Study

1. President's Graduate Fellowship, National University of Singapore, 2012

    - Awarded to graduate students with outstanding research accomplishment
    - One of 18 PhD students selected from across 16 faculties and schools in NUS

2. Honorable Mention Award, INFORMS, 2011

    INFORMS Railway Application Section 2011 Problem Solving Competition

Recent Research Accomplishments: Journal Papers

[1] Lee, D.-H., Jin, J.G., 2013. Feeder Vessel Management at Container Transshipment Terminals. *Transportation Research Part E: Logistics and Transportation Review* 49(1), 201-216.

[2] Jin, J.G., Zhao J., Lee, D.-H., 2013. A Column Generation based Approach for the Train Network Design Optimization Problem. *Transportation Research Part E: Logistics and Transportation Review* 50, 1-17.

[3] Lee, D.-H., Jin, J.G., Chen, J.H., 2012. Schedule Template Design and Storage Allocation for Cyclically Visiting Feeders in Container Transshipment Hubs. *Transportation Research Record* 2273, 87-95.

[4] Lee, D.-H., Jin, J.G., Chen, J.H., 2012. Terminal and Yard Allocation Problem for a Container Transshipment Hub with Multiple Terminals. *Transportation Research Part E: Logistics and Transportation Review* 48(2), 516-528.

[5] Lee, D.-H., Jin, J.G., Chen, J.H., 2011. Integrated Bay Allocation and Yard Crane Scheduling Problem for Transshipment Containers. *Transportation Research Record* 2222,

63-71.

[6] Jin, J.G., Cao, J.X., Chen, J.H., Lee, D.-H., 2011. A Service-Oriented Model for the Yard Management Problem in Container Terminals. *Lecture Notes in Computer Science* 6971, 233-242.

[7] Jin, J.G., Lee, D.-H., Cao, J.X., 2012. Storage Yard Management in Maritime Container Terminals. (Submitted)

[8] Jin, J.G., Lee, D.-H., 2012. A Column Generation based Approach to Feeder Vessel Management Problem. (Submitted)

Recent Research Accomplishments: Conference Presentations

[1] Jin, J.G., Lee, D.-H., 2012. Storage Yard Management with Integrated Consideration of Space Allocation and Crane Deployment in Container Terminals. *3nd International Conference on Computational Logistics, September 24-26, Shanghai, China.*

[2] Lee, D.-H., Jin, J.G., 2012. Tactical Feeder Scheduling Problem in a Container Transshipment Hub. *5th International Workshop on Freight Transportation and Logistics, May 21-25, Mykonos, Greece.*

[3] Lee, D.-H., Jin, J.G., Chen, J.H., 2012. Schedule Template Design and Storage Allocation for Cyclically Visiting Feeders in Container Transshipment Hubs. *91st Transportation Research Board Annual Meeting, January 22-26, Washington D.C., USA.*

[4] Lee, D.-H., Wu, X., Jin, J.G., 2012. Microsimulation Model for Analysis of Traffic Flow in Container Port. *91st Transportation Research Board Annual Meeting, January 22-26, Washington D.C., USA.*

[5] Jin, J.G., Cao, J.X., Chen, J.H., Lee, D.-H., 2011. A Service-Oriented Model for the Yard Management Problem in Container Terminals. *2nd International Conference on*

Computational Logistics, September 19-22, Hamburg, Germany.

[6] Lee, D.-H., Jin, J.G., Chen, J.H., 2011. A Tabu Search Heuristic for Group Allocation Problem in Transshipment Hubs. *90th Transportation Research Board Annual Meeting, January 23-27, Washington D.C., USA.*

[7] Lee, D.-H., Jin, J.G., Chen, J.H., 2011. Integrated Bay Allocation and Yard Crane Scheduling Problem for Transshipment Containers. *90th Transportation Research Board Annual Meeting, January 23-27, Washington D.C., USA.*