INNOVATIVE SIMULATION AND OPTIMIZATION STUDIES ON GRID SYSTEM FOR TRANSSHIPMENT TERMINAL

ZHOU CHENHAO
(B.Eng., Xi'an Jiaotong University)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF INDUSTRIAL SYSTEMS ENGINEERING & MANAGEMENT

NATIONAL UNIVERSITY OF SINGAPORE

2017

Supervisor:
Associate Professor Chew Ek Peng

Examiners:
Associate Professor Ng Kien Ming
Associate Professor Ng Tsan Sheng, Adam
Professor V. Jorge Leon, Texas A & M University

Declaration

I hereby declare that the 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.

Zhou Chenhao

April 7, 2017

Acknowledgement

First, I would like to thank my family, my father and mother, who have strongly supported me spiritually during my time oversea. With their wisdom, passion, and determination as my solid backing, I could concentrate on my research and preparation for the future.

Secondly, I would like to express my deepest gratitude to my supervisor, A/Prof. Chew Ek Peng, and advisor, A/Prof. Lee Loo Hay. They have extended their support and kind assistance to me selflessly during my candidature, teaching me many useful skills and ways of thinking. Their suggestions helped a lot in my research. During the past few years, I was constantly touched by their enthusiasm and love of life and research. From them I truly understand the meaning of loving what we do. In the meanwhile, I would like to present my warmest gratitude to the three examiners for their patience on reviewing the thesis, and their valuable comments and suggestions.

Next, my gratitude also goes out to all of our team members, such as Dr. Jiang Xinjia, Dr. Pedrielli Giulia, Dr. Li Haobin, Dr. Lee Byung Kwon, Dr. Xu Yanhua, Liu Yue, Qiu Zhipeng, Zhao Qitong, Lim Rong Sheng, etc. We had a great time, with many hours spent discussing, working, attending conferences, and enjoying research together. This communal life greatly enriched my life during these years. In addition, I am also grateful to all other faculty members, staffs, my peers, and fellow students in the Department of Industrial Systems Engineering and Management. Thank you for sharing your experiences on research and life in Singapore during these years.

Finally, I want to express my deepest feeling to Miss. Liu Yang, that I will never regret seeing you at Kent Ridge MRT station.

Table of Contents

Summary

With the increasing container traffic and the lack of land in port cities, it is imperative for transshipment terminals to achieve higher storage density (land utilization) and higher handling productivity at the same time. In addition, the cost of manpower, as well as the difficulty in getting skilled labors to do the job keeps increasing. This greatly impacts transshipment terminals, as transshipment activities are very labor intensive. A new conceptual equipment, Goods Retrieval and Inventory Distribution (GRID), based automated container terminal concept could be a promising solution to the above-mentioned challenges. To further understand the features of this system and its improvements to the system efficiency, innovative simulation and optimization studies on this conceptual equipment for transshipment terminals are developed in this thesis.

As the GRID is still a prototype, we propose two designs, the single GRID system and hybrid GRID system, for implementation in actual transshipment terminals. The details on layout design and working mechanism are elaborated. To control the vehicle movement on the mesh-like structure, a free routing traffic rule is proposed by identifying different conflict scenarios and subsequently providing rules to avoid conflicts. Thereafter, a simulation study investigates the performance of the single GRID system and its robustness with respect to horizontal and vertical expansion. The hybrid GRID system is then simulated for terminals demanding high capacity and productivity. The results reveal the advantages of the system, as well as the limitations which motivate us on following studies.

For the hybrid GRID system, we introduce a storage allocation strategy. A novel approach of developing an efficient storage allocation strategy for new terminal concepts is proposed. Specifically, the storage allocation strategy is derived from optimal allocation decision learnt from a Mixed-Integer Programming model. Certain input parameters of the MIP model are collected from a simulation model. An index measuring the convenience of the storage location is proposed and we regress this index with important variables to build an empirical model which recommends which storage locations to allocate containers to. Using an empirical approach has the advantage of fast computation speed, which is expected in the dynamic and uncertain environment in the port.

For the single GRID system, we seek to further improve the system throughput by fixing travel direction on the path which is defined by a single direction traffic rule. Due to the dynamic feature of the GRID system, it becomes very challenging to reduce the conflicts between vehicles in large scale scenarios. Therefore, we formulate the situation as a path design problem, and provide a novel algorithm motivated by the heat-map concept as the solution. A dynamic framework is also proposed to deal with the challenges in the continuously running system. The numerical experiment shows that the algorithm is efficient and scalable in different scenarios, and the single direction traffic rule outperforms the free routing traffic rule. In addition, the dynamic framework can effectively handle the changing of the path design.

List of Tables

List of Figures

List of Abbreviations

GRID	Goods retrieval and inventory distribution
TEU	Twenty-foot equivalent unit
GDP	Gross domestic product
ALV	Automated lifting vehicle
AGV	Automated guided vehicle
ACT	Automated container terminal
SGS	Single GRID system
HGS	Hybrid GRID system
TU	Transfer unit
GT	Ground transporter
IAS	Information-based allocation strategy
MIP	Mixed integer programming
AS/RS	Automated storage and retrieval systems
LMCS	Linear motor conveyance system
GR-ACT	Grid rail system based ACT
FB-ACT	Frame bridge based ACT
QC	Quay crane

LR	Least relocation
RTG	Rubber-tyred gantry
RMS	Reconfigurable manufacturing system
SDTR	Single direction traffic rule
TP	Transfer platform
CP	Control point
IOP	Input/output point
BFS	Bay-front section
IYS	Inner-yard section
QCS	Quay crane system
TS	Transportation system
DTS	Direct transfer system
YSS	Yard storage system
ASC	Automated Stacking Crane
SAP	Storage allocation problem
COI	Cube-per-order index
DoS	Duration-of-stay
I/O	Input/output

CSL	Convenient storage location
SI	Sorting index
RCSL	Relative convenient storage location
GSA	Greedy sequential allocation
BSA	Batch sequential allocation
PRD	Purely random
OLP	Operational level problem
FRTR	Free routing traffic rule
HA	Heat-map algorithm
HSEA	Heat-map-based simulation embedded algorithm
AP	Access point
PDP	Path direction problem
CTM	Control module
PDM	Path design module
RTM	Routing module

List of Notations

Storage Allocation Problem

N	The set of jobs, index n;
S	The set of modules, index s;
K	The set of sections, index k;
K_s	The set of section indexes that belong to module s, i.e. $K_1 = \{1,2,3\}$
T	The set of shifts, index t, $t \in \{1,2...T_{\max}\}$, $T_{\max}$ represents the last shift;
M	The maximum number of TUs each module can be assigned;
m	The number of TUs, $m \in \{0,1,2...M\}$;
c	Storage capacity, measured by the number of 40-foot containers;
β	Space utilization, a constant value less than 1;
$l_{o_n k}, l_{d_n k}$	Rectilinear distance from origin/destination of job n to section k;
w_n^{20}, w_n^{40}	Amount of 20-foot/40-foot containers in job n;
t_n^d, t_n^l	Discharging/loading shift of job n;
α_1, α_2	The weights on average travel distance and total number of TUs;

W_{nk}^{20}, W_{nk}^{40} Workload of job n in section k, continuous variable;

C_{kt} The storage level of section k at the beginning of shift t, $t \in \{1,2...T_{\max}\}$, continuous variable;

$C_{k,T+1}$ The storage level of section k at the end of planning period, continuous variable;

U_{st} The number of TUs assigned to module s in shift t, integer variable;

U The total number of TUs for the whole system, integer variable;

D The average container travel distance, continuous variable;

$f(\bullet)$ TU handling capacity function;

Path Direction Problem

N The set of nodes.

P The set of the directed paths. Each directed path k is uniquely represented by (ns_k, ne_k) which specifically goes from node $ns_k \in N$ to node $ne_k \in N$. The opposite of directed path k is the directed path k^-.

L_k The distance of path k, positive value.

PS_n The set of directed paths that start from node $n \in N$.

PE_n The set of directed paths that end at node $n \in N$.

pa_m A pair of adjacent directed paths. $pa_m \equiv \left(k_m^1, k_m^2\right)$ where k_m^1 and k_m^2 are the first and second elements in the pair pa_m. k_m^2 is right adjacent to k_m^1, and they have opposite direction to each other, as shown in Figure 5.3 (b).

PA The set of the pairs of adjacent directed paths, and $PA = \{ pa_m \mid \forall m \}$.

pt_h A pair of directed paths which forms a turn. $pt_h \equiv \left(k_h^1, k_h^2\right)$ where k_h^1 and k_h^2 are the first and second elements in the pair, as shown in Figure 5.3 (c).

PT The set of the pairs of directed paths which form a turn, $PT = \{ pt_h \mid \forall h \}$.

T Turning distance, positive value.

J The set of jobs. Each job $j \in J$ contains the information such as $\left(no_j, nd_j\right)$ where no_j is the origin node id and nd_j is the destination node id.

S_j The shortest distance of job *j*, positive value.

α a parameter for penalty.

G_r^U, G_r^D The up-going / down-going path group in row *r*, $r = 1, 2, 3...R$, R is the total number of path group in row.

G_c^L, G_c^R The left-going / right-going path group in column *c*, $c = 1, 2, 3...C$, C is the total number of path group in column.

δ^U, δ^D The minimal number of up/down in each path group in row.

δ^L, δ^R The minimal number of left/right in each path group in column.

M A large number.

x_k Binary variable. $x_k = 1$ if traffic goes from node ns_k to node ne_k; otherwise $x_k = 0$.

y_j Binary variable. $y_j = 1$ if job j cannot be delivered to its destination under current path design.

$a_{j,k}$ Binary variable. $a_{j,k} = 1$ if job j travelled on path k.

$b_{j,h}$ Binary variable. $b_{j,pt} = 1$ if job j turns at pt_h.

Reduced Path Direction Problem

N The set of nodes; node index is n;

I_{kn}^{σ} $\sigma \in \{L, R, U, D\}$, the 0-1 value which indicates that arc k should go left, right, up or down to provide an inbound connection to node n;

O_{kn}^{σ} $\sigma \in \{L, R, U, D\}$, the 0-1 value which indicates that arc k should go left, right, up or down to provide an outbound connection to node n;

PV The set of vertical arcs, index is k.

PH The set of horizontal arcs, index is k.

pa_m A pair of adjacent arcs. $pa_m \equiv \left(k_m^1, k_m^2\right)$ where k_m^1 and k_m^2 are arcs in the pair pa_m. k_m^2 is right adjacent to k_m^1, as shown in Figure 5.5 (c).

PA The set of the pairs of adjacent arcs, and $PA = \{ pa_m \mid \forall m \}$.

F_k^L, F_k^R The left flow and right flow on arc *k*, respectively.

F_k^U, F_k^D The up flow and down flow on arc *k*, respectively.

GR_r The arc group in row *r*.

GC_c The arc group in column *c*.

δ^U, δ^D The minimal number of up/down in each arc group in row.

δ^L, δ^R The minimal number of left/right in each arc group in column.

l_k $l_k = 1$ if arc *k* is set to go left, otherwise 0, for $k \in PH$.

r_k $r_k = 1$ if arc *k* is set to go right, otherwise 0, for $k \in PH$.

u_k $u_k = 1$ if arc *k* is set to go up, otherwise 0, for $k \in PV$.

d_k $d_k = 1$ if arc *k* is set to go down, otherwise 0, for $k \in PV$.

e_k $e_k = 1$ if arc *k* is closed, otherwise 0, for $k \in PV$.

Chapter 1. Introduction

This thesis contributes to a comprehensive study, at the planning and operational levels, of a new conceptual yard handling system – the GRID system. In the first two sections of this chapter, we briefly introduce the background of container terminals and the challenges in transshipment terminals. Subsequently, in Section 1.3, we will introduce the GRID system and highlight the research interests and gaps. The objective and scope of this thesis will be provided in Section 1.4.

1.1. Background of Container Terminals

The global port container traffic has experienced tremendous growth in the last decades since the modern intermodal container was first developed in 1955. According to the World Bank, the annual container port traffic has increased around 7.4 times, from 88.1 million TEUs (twenty-foot equivalent units) in 1990 to 651.1 million TEUs in 2013. With the growing world gross domestic product (GDP), the global container trade is expected to increase as well.

Container terminals can be categorized into two types, as an import/export terminal, where majority of the containers are transferred from sea-side to land-side via the yard-side, or vice versa, and as a transshipment terminal, where majority of the containers are moved from vessels to vessels, as illustrated in Figure 1.1.

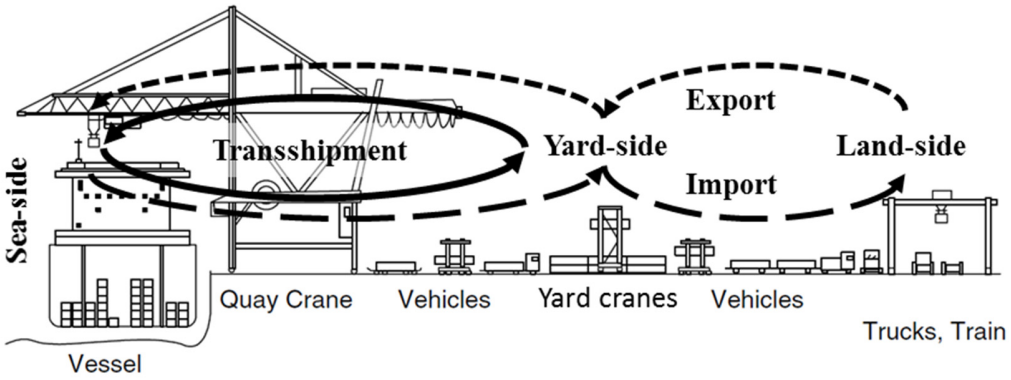


Figure 1.1 Flow in the container terminals

Container transshipment plays an important role in the container shipping industry. From a network perspective, the transshipment is used for connecting countries and regions not directly served by main-haul shipping services due to economic efficiency and/or physical restriction. With the growth of global supply chain and GDP, the volume of transshipment containers is increasing every year and is expected to continue growing in the future, as shown in Figure 1.2. However, as the majority of handling activities and traffic are concentrated between the quay-side and the yard-side, the running of transshipment terminals is increasingly challenging under the pressure of the growing container traffic.

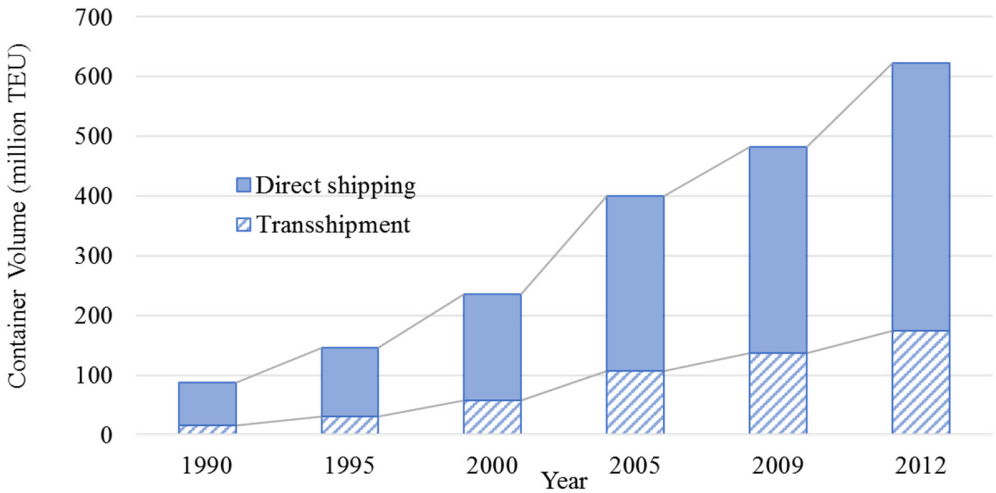


Figure 1.2 Global container trade from 1990-2012

1.2. Challenges in Transshipment Terminals

With the increasing container traffic and the scarcity of land in port cities, it is very important for the transshipment terminals to achieve both higher storage density (land utilization) and higher handling productivity at the same time. Terminals in Hamburg and Hong Kong cannot expand any more as the harbor is closely surrounded by the city, while terminals in Singapore and Rotterdam can only expand by reclamation of new land, which will incur high construction costs. There are three widely adopted terminal layouts as summarized by Carlo *et al.* (2014) and Zhou *et al.* (2015) as shown in Figure 1.3. In addition, Zhou *et al.* (2015) concluded that these traditional terminal layouts are either land utilization centered, or productivity centered. Design A (Figure 1.3A) can achieve a high land utilization but is limited by productivity of yard crane; design B (Figure 1.3B) uses straddle carriers or Automated Lifting Vehicles (ALVs) to move containers between yard and quay. By eliminating handshakes, straddle carriers apparently improve yard-side productivity but more space is required. Design C (Figure 1.3C) uses the parallel layout at the yard side which is predominant in Singapore. It allows trucks to move container under cranes, so as to reduce the movement of the cranes, thus resulting in a high overall productivity. However, design C has to reserve more space for vehicle paths to reduce traffic congestion. Hence, land utilization is much lower than in design A. In summary, trying to achieve both targets of high storage density (land utilization) and high handling productivity at the same time is quite challenging.

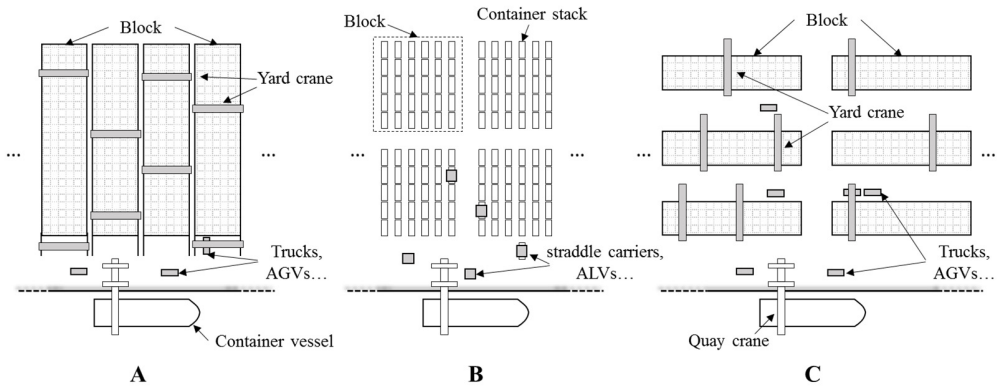


Figure 1.3 Three typical layouts for container terminals

Another challenge for container terminals is the ever-increasing cost of manpower, as well as the difficulty in getting skilled labors to do the job. This greatly impacts the transshipment terminals, as the transshipment activities are very labor intensive. As shown in Figure 1.4 (Castalia, 2012), the labor cost per TEU in Australia in 2011 has increased 10% over the years since 2003. This growing cost of labor makes up a very significant, and ever increasing proportion of the total costs.

Figure 1.4 Labor costs per TEU in Australian ports

A potential solution to the manpower challenge is to implement automation to replace the headcounts. One such example is the use of Automated Guided Vehicles (AGVs). AGVs are widely adopted in many terminals across the world, such as the Container Terminal Altenwerder in Hamburg, and the European Container Terminal in Rotterdam. While many new automated container terminal (ACT) concepts and designs have been introduced in the recent years, the majority of the ACTs are mainly catered for import/export terminals, and many terminals in Asia, especially the transshipment terminals such as Singapore and Hong Kong, are still semi-automated or human operated. However, these traditional handling systems may not be able to satisfy the future demand of the productivity and storage capacity (K. H. Kim *et al.*, 2012).

As the above challenges will eventually affect the efficiency and competitiveness of the terminals if they are not addressed adequately, operators need to find a new design of ACT which could improve both land utilization and system productivity at the same time.

1.3. Background of GRID System

Recently, BEC Industries LLC has designed a new conceptual container handling prototype named the single GRID system (SGS), as shown in Figure 1.5. The prototype is a fully automated, mesh-like overhead single rail structure. The system has two layers, a ground storage layer, and a sky transport layer. The transfer unit (TU) is the main interface for containers handling. It hangs on the sky layer and can travel at a high speed in both directions without rotation. With the separation of the

two distinct layers, the TUs have more space and higher flexibility to maneuver. As a result, fewer ground paths are needed, thus improving land utilization.

Figure 1.5 Prototype of the single GRID system

As an extension to the SGS, in order to serve millions of TEUs annually with high handling efficiency and storage capacity in a transshipment terminal, a hybrid GRID system (HGS) is proposed as shown in Figure 1.6. The system consists of three major components, GRID modules, transfer units (TUs), and ground transporters (GTs). In the HGS, the whole yard is divided into multiple modules. Each module has an independent SGS, and ground transporters are used for fast delivery between modules and the quay-side.

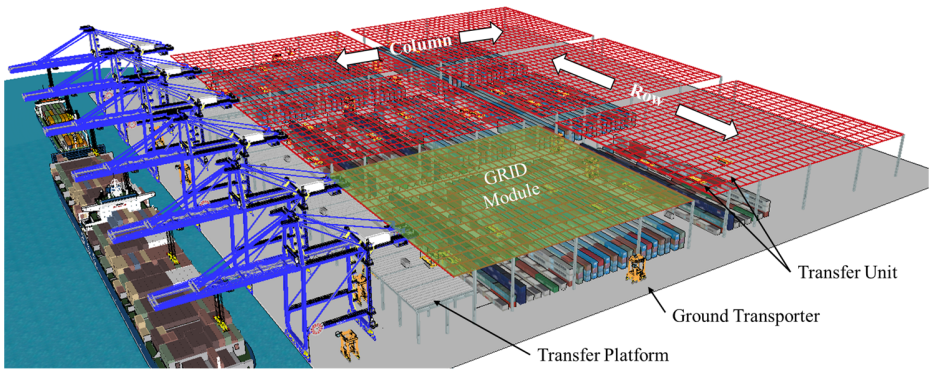


Figure 1.6 Concept of hybrid GRID system

However, as the system has not been deployed to any operating terminals, key information of the system such as the detailed configuration and productivity are unknown. In addition, due to its specific infrastructure, the vehicle control policy and container storage policy are quite different from previous terminal designs. Thus this thesis attempts to address three fundamental issues, (1) the system configuration and performance analysis, (2) yard storage management, and (3) transportation management.

1.4. Contributions

The main contribution of this thesis can be listed as follows.

- This is the first academic study on the new conceptual terminal handling system. Although it is still a prototype, the features of the GRID system could potentially be a solution to the challenges of current systems. This motivates us to propose new terminal designs based on the GRID system, and to develop new strategies for implementation in operational container terminals.
- **The study proposes a comprehensive framework reflecting the features and capabilities of a conceptual terminal system, which acts as the foundation of subsequent optimization studies.** As a new design without any prior fundamental study, it is important to understand its mechanism, performance, limitations, and applications, and thus, develop a comprehensive simulation work. Due to the complexity of the structure, the challenge of the system is to prevent and solve potential conflicts. As such, the free routing traffic rule is proposed which allows TUs to travel by the shortest path, and rerouting whenever a conflict is met.

- **The study proposes an intuitive, effective and data driven method to deal with the container allocation problem. By doing statistical analysis on the optimal solutions of the mathematical model, several key factors are identified to have significant impacts on the container allocation. From these findings, an index function is formulated. The study solves the problem from a different perspective and it can be easily implemented.** Since the previous strategies for warehouse and terminal is not suitable for GRID, and that solving a mathematical model is not practical, the information-based allocation strategy (IAS), a data driven method, is proposed. The proposed method can find a good solution quickly and achieve robustness in a dynamic and uncertain environment. The study also suggests a novel framework to deal with new conceptual designs. In order to describe a system, the simulation can be applied to obtain the necessary data, and a high-dimensional approximation method can then be developed to obtain simple empirical formulations which can be computed quickly.
- **The study also proposes an intuitive and effective method to reduce vehicle conflicts. Instead of formulating the situation as a routing or scheduling problem, it treats the problem as a path direction design problem which avoids solving a complex mathematical model. The "heat-map" concept is also applied innovatively as a heuristic algorithm for resolving the path direction design problem.** We propose the single direction traffic rule. The goal is to set the proper single direction on each path which eliminates the opposite conflicts. It is trivial that the success of the single direction traffic rule

follows the finding of a good path travel direction design over the whole system. A novel approach based on the heat-map concept is developed to solve the path design problem. Since the flow on the system can be very different at various times, the path design needs to be updated accordingly. Therefore, the proposed algorithm is applied to the framework to update the path design periodically.

1.5. Organization

This thesis consists of six chapters, which are organized as follows.

Chapter 2 reviews the studies dealing with the related problems in the GRID system. As an innovative and new system, a general idea on how to evaluate it can be established by reviewing previous studies of other new systems, and simulation has been a proven, efficient method for discovering new systems. To further improve the single GRID system, the path direction design can be a potential solution. Although there are many yard related problems, we decide to start the hybrid GRID system with the basis – the yard allocation problem and its solving methods.

Chapter 3 explores the single GRID system and hybrid GRID system in detail. A free routing traffic rule is proposed to control the vehicle movement on the mesh-like structure by identifying different conflict scenarios and subsequently providing the rule to avoid conflicts. Simulation studies are used to evaluate the performance of both GRID designs.

Chapter 4 introduces a storage allocation strategy for a transshipment container hub using the hybrid GRID system. A novel approach of developing an efficient storage allocation strategy for new terminal concepts is proposed. The storage allocation

strategy is derived from the optimal allocation decision learnt from an MIP model. Some input parameters of the MIP model are collected from simulation experiments. An index function which recommends the allocation of containers to storage locations is obtained from the optimal solutions of the MIP model. The advantage of using the empirical approach is that it allows for fast computation which is expected in the dynamic and uncertain environment in the port.

Chapter 5 seeks to further improve the system throughput of the GRID by fixing the travel direction of the path defined with the single direction traffic rule. Due to the dynamic feature and flexible design of the GRID system, it is very challenging to reduce vehicle conflict in large scale scenarios. Therefore, we formulate the situation as a path design problem and provide a novel algorithm motivated by the heat-map concept as the solution. In addition, a dynamic framework is proposed to deal with the challenges in a continuously running system. This study indicates a potential research area for the GRID system as well as AGV systems.

Chapter 6 concludes the important findings in previous chapters. The limitations of the current studies, and future research directions are also discussed.

Chapter 2. Literature Review

A container terminal is an integrated logistic system that offers the container handling and temporary storage between different parts such as the quay-side, the yard-side, and the land-side. Various operations are performed in a container terminal, and all operations are interrelated to some extent. Several literature reviews (Stahlbock & Voß, 2008; Steenken *et al.*, 2004; I. F. Vis & De Koster, 2003) have made comprehensive reviews of the research works related to various port operations.

According to the development of the GRID system, this literature review will focus on different problems as shown in Figure 2.1. This chapter firstly presents many innovative technologies for container terminals and the capabilities of these technologies. Since the performance of the system is of utmost interest to us, simulation is widely adopted among container terminal studies, regardless of traditional or new conceptual designs. Once we obtain the benchmark with simulation, we can further improve the performance via different focus and methods. Thus the vehicle management problems, path direction design problems, storage allocation problems, and related solving methods are reviewed.

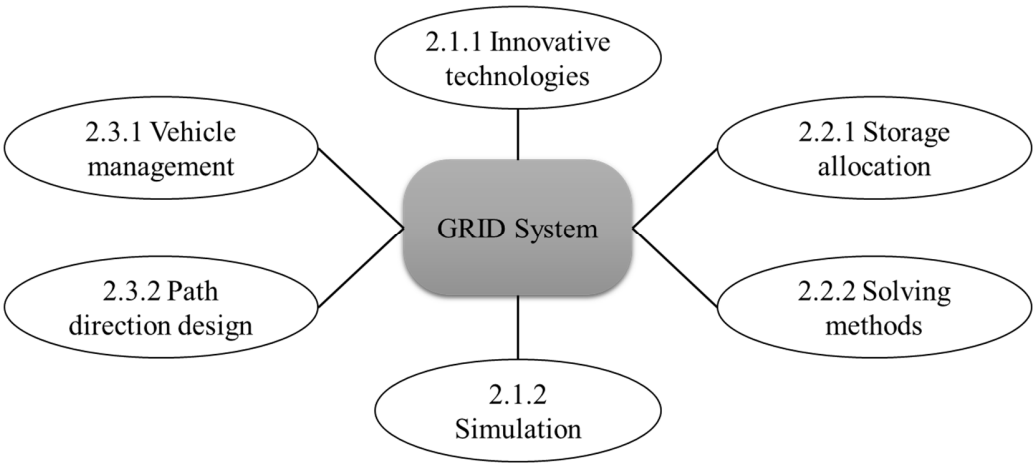


Figure 2.1 The structure of the literature review

2.1. The Design of Storage Yard

2.1.1. Innovative Technologies

The latest revolution in container terminals is gradual replacement of manual systems with automated systems, with traditional container terminals being upgraded to ACTs or semi-ACTs, where all tasks are carried out by automated devices in the field while operators only need to monitor and control the whole process from the office.

In order to help the automation, innovative devices and system designs are continuously proposed and discussed in academia. Choi and Ha have studied the Automated Storage and Retrieval Systems (AS/RS), and a prototype of the high stacking system (HSS), which is a variant of AS/RS, has been installed and tested by EZ-INDUS in Korea (Choi & Ha, 2005). The comparison between the automated guided vehicle system (AGV-ACT) and new systems such as linear motor conveyance system (LMCS-ACT), an overhead grid rail system (GR-ACT), and a high-rise AS/RS is first made in Liu *et al.* (2001; 2002). Zhen *et al.* (2012) showed that the frame-bridge based ACT (FB-ACT) has a higher throughput than

AGV-ACT. The same conclusions were obtained from using a decomposition method by Hu *et al.* (2014; 2013). A mathematical study on the GR-ACT for container routing is proposed by Zeng & Hsu (2008), and it presents a simple container routing algorithm guaranteeing freedom from conflicts for this mesh-like path topology.

2.1.2. Simulation

In large scale systems, such as container terminals, it is difficult to implement and test different configurations in the field. With increasing computing power, simulation has been widely adopted in container terminal related studies that pay particular attention to performance evaluation. Discrete event simulation has long been a useful tool to support container terminal decisions in a complex and stochastic environment (Nevins *et al.*, 1998). Saanen and Valkengoed (2005) compared the efficiency in terms of productivity, flexibility, area utilization, and cost, of three different automated container terminal concepts. Vis (2006) presented a simulation study to compare the straddle carrier with the automated stacking crane in terms of the expected total time to retrieve a set of storage and retrieval requests. Bae *et al.* (2011) compared the performances of AGVs and ALVs when using different quay cranes (QCs). Simulation is also a powerful method for testing and evaluation of innovative technologies. ALV-ACT, GR-ACT and AS/RS are shown to be more efficient than AGV-ACT (Bae *et al.*, 2011; C.-I. Liu *et al.*, 2002; I. F. Vis & Harika, 2004). Lee *et al.* (2014) shows that FB-ACT is unsuitable for the transshipment terminal due to a large number of handshakes and high volume of cross-section movement.

There are some papers related to simulation of the whole container terminal. Liu *et al.* (2002) developed a microscopic simulation model to simulate different ACT systems for the same operational scenario such as vessel capacity, container arrival rate, etc. Sun *et al.* (2013) proposed an integrated simulation framework to facilitate the design, simulation, and evaluation of a large number of terminal layouts and scenarios. Guldogan (2011) proposed a discrete-event simulation model that includes details on storage policies at operational level to investigate the effect of different storage location assignment policies on the overall performance of a terminal.

2.2. The Yard Storage Management

2.2.1. Storage Allocation Problem

The storage allocation problem is an important problem for container terminals and different strategies can have significant impacts on terminal performance in terms of productivity, resource usage, and/or land utilization.

The consignment strategy is proposed for transshipment terminals by Lee *et al.* and Murty (2006; 2007) which requires the terminal to store the containers that are loaded to the same vessel in the same sub-blocks to reduce the reshuffling rate. However, this caused high vehicle flows around sub-blocks. In order to reduce the potential traffic congestion, these sub-blocks are spread apart within the yard area (L. H. Lee *et al.*, 2006). Han *et al.* (2008) investigated the yard storage template which determines the sub-block reservation for a group of containers. In order to increase the land utilization of the storage yard using the consignment strategy, Jiang *et al.* (2012) and Jiang *et al.* (2013) suggested the space-sharing yard template

which allows space sharing with adjacent neighbors. However, since space is reserved for a long time in advance, the disadvantage of the consignment strategy is inefficient land usage.

Using stack/slot as allocation unit, two principals were described by Woo and Kim (2011), namely the nearest location principle, which prioritizes storage locations closer to the vessel berth, and the least relocation principle, which prevents mixing groups of containers in the same stack. In particular, the least relocation (LR) principle is most similar to the consignment strategy but at the stack level. Later, Petering (2013) applied the LR principle in a real-time environment which makes decisions at the latest possible moment for automatically selecting storage locations for rubber-tyred gantry (RTG)-based transshipment terminal. These storage methods are usually found in gateway ports, but cannot guarantee productivity for transshipment container terminals due to potential reshuffling or high frequency yard crane movement.

2.2.2. Solving Methodology

According to Carlo *et al.* (2014), there are many solutions proposed for the storage allocation problem. Nishimura *et al.* (2009) addressed the problem for the transshipment process from mega-containerships to feeder ships and proposed an MIP, the corresponding Lagrangian relaxation, and a heuristic to solve the problem. Chen and Lu (2012) proposed a two-stage solution for outbound containers. The first stage reserves the storage space for each yard bay by solving an MIP model, while the second stage determines the exact location by using a greedy-based algorithm. Jiang *et al.* (2013; 2012) developed a search algorithm which combines

MIP and heuristics to find the solution. Dynamic programming models and a two-stage heuristic were also suggested for outbound containers (Zhang *et al.*, 2014). A column generation-based heuristic method was proposed for an integrated problem including tactical berth and yard template design, which is formulated as a set covering model (Jin *et al.*, 2015).

2.3. The Transport Vehicle Management

2.3.1. Vehicle Management

Vehicle routing and dispatching problems are two critical issues in transport vehicle management.

The vehicle routing problem in container terminal is to avoid the collisions, and many studies have proposed different algorithms to guarantee conflict-free routing and shortest possible time (C. W. Kim & Tanchoco, 1993; Qiu & Hsu, 2001; Saidi-Mehrabad *et al.*, 2015).

Vehicle dispatching is concerned with the problem which determines the sequence of deliveries of the vehicles to achieve certain goals. In the port area, the vehicles are the trucks, AGVs, straddle carriers, or other types of transporters. One major issue in the general vehicle dispatching problem is regarding the exploitation of the dual cycle – if the loading (discharging) request can be followed by a discharging (loading) request then the total travel distance can be reduced. In container terminals, there are only a few studies that pay attention to the dispatching of individual vehicles. Usually, the container handling sequence of vehicles is considered as an input, and is determined by the quay crane and on-board storage

plan (D.-H. Lee *et al.*, 2009; Luo & Wu, 2015). Nishimura *et al.* (2005) proposed a dual cycle dispatching for a fleet of vehicles.

In recent years, more and more studies have started to pay attention to integrated problems with the terminal as an integrated logistic system. Many studies have paid attention to the integration of vehicle dispatching and conflict-free routing problems which can be optimally solved with very limited number of vehicles. Desaulniers *et al.* (2003) modelled the problem with a set partitioning formulation which can only be solved for up to four vehicles within a controllable time. Corréa *et al*. (2004) proposed an approach which combines constraint programming for vehicle dispatching and mixed integer programming for conflict-free routing.

2.3.2. Path Direction Design

Mathematical models and algorithms for mesh-like systems have been discussed in different domains. The studies of AGV scheduling in mesh-like path topology were summarized by a comprehensive review (Qiu *et al.*, 2002). Multiple variations of the guide-path design problem have been continually proposed and discussed in academic literature. These may be broadly categorized into three types of guide-path systems – conventional, single-loop, and tandem guide-path systems (Le-Anh & De Koster, 2006). The conventional system is most similar to the GRID system, as such previous studies of the conventional system can be explored.

Gaskins and Tanchoco (1987) first formulated the unidirectional conventional guide-path problem, also known as the flow path design problem, as a zero-one integer programming model to optimize the unidirectional path network and to

minimize total vehicle travelling distance with reachability and connectivity constraints. However, many studies have shown that the unidirectional flow path design problem is computationally difficult and does not solve well by standard integer programming models in large-scale applications. To improve the computational efficiency of the previous model, Kaspi and Tanchoco (1990) utilized a branch-and-bound procedure, though at the expense of path design quality. Kouvelis *et al.* (1992) tested five heuristic procedures of increasing complexity and compared their performances against different simulated annealing models; Seo *et al.* (2008) proposed an evolutionary computational approach combining genetic algorithm and tabu search methods to tackle large-scale path design problems; and Guan *et al.* (2011) proposed and tested a revised electromagnetism-like mechanism heuristic in the reconfigurable manufacturing system (RMS) context.

Single-loop guide-path systems, on the other hand, involve vehicles travelling in a single loop without alternative routes between workstations which could be simply controlled by the first-encountered-first-served (FEFC) dispatching rule (Qiu *et al.*, 2002). It was first proposed as an optimal procedure for designing a single-loop system (Tanchoco & Sinriech, 1992). Separately, other single-loop guide-path models and solution procedures have also been suggested. Banerjee and Zhou (1995) used a genetic algorithm-based approach to the problem; Chen *et al.* (1999) proposed a mixed integer programming model to design guide-paths for the single-loop dual rail system; and Asef-Vaziri *et al.* (2000) presented an alternative formulation to Tanchoco's optimal design procedure with fewer binary variables and a larger set of feasible solutions.

Lastly, tandem guide-path systems partition the entire AGV system into multiple non-overlapping single vehicle loops, with transfer stations used to interface between loops (Bozer & Srinivasan, 1992). It was first introduced by Bozer and Srinivasan (1991), who also proposed an algorithm for decomposing a system into non-overlapping single-vehicle zones. Tandem guide-path systems, like its single-loop counterparts, greatly simplify the vehicle control problem and route planning problem (Ho & Hsieh, 2004). However, several limitations of the tandem concept still exist, which include the time and cost inefficiencies of inter-zone load transfers (W.-L. Chen, 1995). Subsequent studies on variations of the tandem guide-path system have broadened to involve genetic algorithm, simulated annealing and tabu search, minimal spanning tree approaches amongst others, and have been comprehensively reviewed by Rezapour *et al.* (2011).

2.4. Research Gaps, Objectives and Scope

The GRID system is a new conceptual yard-side container handling system and no previous study can be found. Since the system has not been deployed to any operational terminals, the key information of the system such as the detailed configuration and productivity are unknown. In addition, due to its special infrastructure, the vehicle control policy and container storage policy are quite different from previous terminal designs. Thus, in this thesis, three fundamental issues (1) system configuration and performance analysis, (2) yard storage management, and (3) transportation management need to be addressed.

Based on the features of the GRID system and the previous literature review, we can conclude several research gaps as follows:

- Most current/proposed ACTs are mainly catered for gateway ports, and many terminals in Asia, especially transshipment terminals such as Singapore and Hong Kong, are still semi-automated or human powered. In addition, none of these systems are compatible with the increasing demand of the transshipment terminals in terms of land utilization or productivity.
- As the GRID system is still a prototype, the priority is to understand its throughput characteristics and potential bottlenecks in different configurations, layouts, and vehicle routing policies. Therefore, a simulation study on the GRID system is necessary. The unique design of the GRID system brings new challenges to the vehicle routing problem. Due to the use of single rail structure, TUs on the same or adjacent paths cannot cross over each other, as they share the same rail. When there is a large number of TUs, deadlock prevention becomes critical in maintaining system efficiency.
- The scale of the vehicle routing and scheduling problem cannot be very large in terms of the number of vehicles and jobs. One of the main causes of the computation complexity is that the travel direction is completely free, and thus the solution space is tremendously large.
- Due to the different structures and behaviors, the storage allocation problem in HGS requires different arrangements and strategies from the traditional terminals. For example, the consignment strategy may not work well in the GRID system since containers need to be spread out to avoid traffic congestion in one area from building up. Although many studies have proposed exact mathematical models and heuristic algorithms to compute the solution for

storage allocation problems, most of them are based on traditional terminals and on the consignment strategy. These algorithms are not easy to solve as well. Therefore, it is important to identify a practical and flexible storage allocation strategy which can have a significant impact on the overall terminal productivity for the HGS.

- Instead of solving the vehicle routing and scheduling problem, setting directions on the paths is another way to solve large scale vehicle routing problem. However, it is quite clear that previous studies cannot handle very large scale scenarios and did not consider changing the path design dynamically. Thus, an efficient algorithm and the dynamic framework are both expected. It should be noted that there are still many challenges present, such as ensuring that each node is accessible so that the path design is strongly connected, dealing with the unfinished delivery missions during the changing of the path design, etc.

These research gaps can help to understand the features of the GRID system and their significant impact on the productivity of the GRID system. The main purpose of this thesis is to apply simulation and optimization techniques to the decision support for the GRID system at the operation and planning levels. The specific goals of this thesis are as follows:

- As a new design without any prior fundamental study, it is important to understand its mechanism, performance, limitations, and applications, and thus, a simulation study on the GRID system, including SGS and HGS, is proposed with a free routing traffic rule where vehicles can travel freely and conflicts are

solved in real time. Although it is still a small prototype with only four slots, its features showed that the GRID system could be a solution to the above challenges. This motivates us to expand the prototype to a larger structure and to implement it for actual use.

- As the storage allocation problem is a major challenge in a general container terminal, we are interested in how the allocation strategy will affect the total cost, including the variable and setup costs, with respect to the TUs. To obtain the optimal allocation decision, an MIP model for a deterministic problem is formulated by optimizing operating and capital costs. However, this approach might not be practical when solutions need to be computed quickly, especially when the operational environment is often very dynamic and uncertain and thus, it is more realistic to use heuristic algorithms. Therefore, we develop a novel allocation strategy which exploits the optimal allocation decisions and eventually uses greedy and sequential based algorithms to allocate containers. It can also be used for operational decision when container arrival information is available only a few shifts ahead, or when the decision has to be made on a regular or ad-hoc basis.
- Motivated by the traffic rule in the urban city environment which sets a single travel direction on lanes and stops vehicles from different directions occupying the same crossroad, the single direction traffic rule (SDTR) is proposed for the GRID system. The concept is to set proper single direction on each path which eliminates the opposite conflicts. When two TUs meet at the crossings, succeeding TUs will wait for preceding TUs to pass, instead of detouring which

might cause extra conflicts. Therefore, the problem is about how to find a good path direction design (path design) efficiently over the whole system. Besides, once the path design is determined, it does not mean that it will be the one used all the time. In fact, since the flow on the system can be very different from time to time, the path design may need to be updated accordingly. Thus, a dynamic framework is proposed to work with SDTR to change the path directions and update the overall path design.

In the remaining chapters, detailed studies on the above objectives will be elaborated.

Chapter 3. Free-Routing Traffic Rule and Performance Evaluation on Single and Hybrid GRID Systems

In Section 1.3, the mechanism of the GRID system has been introduced. Although it is still a small prototype with four slots only, its features show that the GRID system could be a solution to the above challenges in Section 1.2. This motivates us to expand the prototype to a larger structure and implement it for actual use. Besides, as a new design with limited studies, it is important to understand its mechanism, performance, limitations and applications, and therefore, a simulation study has been carried out.

The remainder of this chapter is organized as follows: Section 3.1 introduces the single GRID system and the free routing traffic rule which establish the basis of the further studies on the GRID system. Simulation experiments are conducted on different sizes of the system for performance evaluation. The GRID architecture is extended in Section 3.2 where the hybrid GRID system is presented as a terminal solution. The performances are evaluated through a simulation study. Finally, conclusions are outlined in Section 3.3.

3.1. Single GRID System

3.1.1. Design Configuration

Without loss of generality, this study introduces a small layout as shown in Figure 3.1. The length and depth of the layout are 136 meters and 64 meters, respectively, and the layout covers 11 by 23 40-ft storage slots. The grid frame has 11 by 23 slots corresponding to the ground slots and it is further divided into two areas, transfer area, which covers 11 by 3 slots and interacts with quay cranes (QCs) via a transfer

platform (TP), and storage area, which covers 11 by 20 slots and allows up to 5 containers high so the total capacity is 2,200 TEUs. Considering the size of QC and safety distance, the terminal can at most deploy one QC every 70 meters.

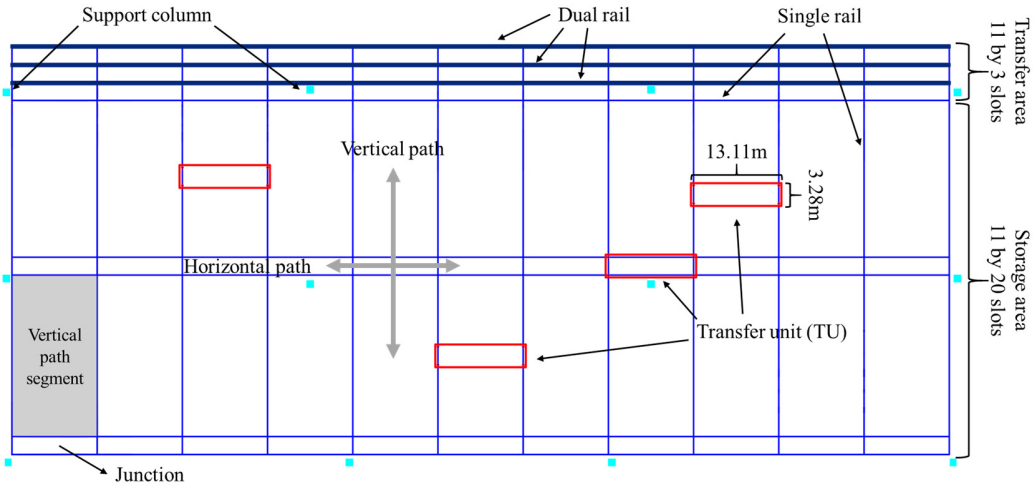


Figure 3.1 Small layout of single GRID system (top view)

Figure 3.2 illustrates how the transfer area in the single GRID system cooperates with TP and QC. In the single GRID system, TP uses a conveyor to move containers. To be specific, QC drops the containers on the conveyor and then the container is moved towards the transfer area by the conveyor. Whenever a TU is available, it will come to pick up and transfer the container to the assigned position/slot. The loading containers are also placed on the conveyor and the remaining operations are performed similarly in the reverse order.

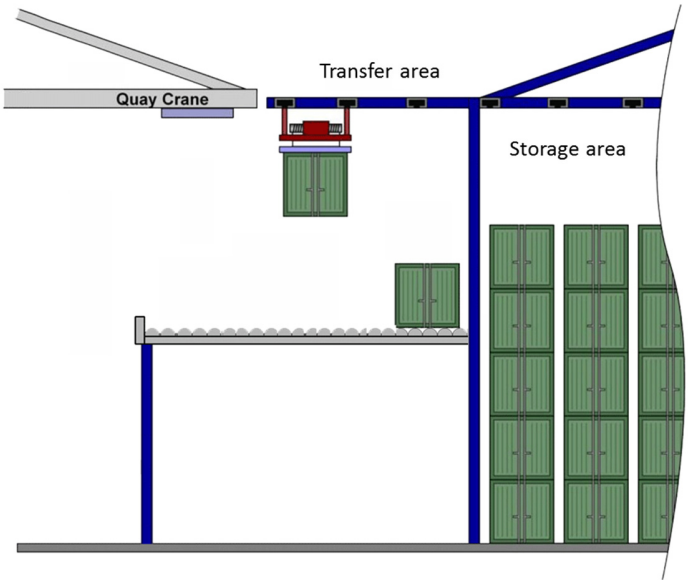


Figure 3.2 Structure of single GRID system (side view)

The layout has 11 vertical paths and only 5 horizontal paths instead of 23 horizontal paths due to cost reasons, although a complete grid frame has a higher flexibility in terms of TU movement. The vertical paths are formed by 12 single rails. TUs on the same or adjacent vertical paths share the same rail so they cannot cross over each other. Besides, due to the rail sharing and TU dimension as shown in Figure 3.1, when a TU occupies one storage slot, 8 neighboring slots are blocked and unavailable to other TUs, as shown in Figure 3.3.

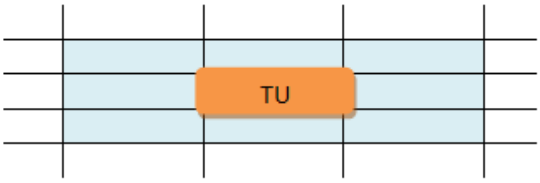


Figure 3.3 TU blocking area in storage area

The designs of horizontal path in the transfer area and storage area are different. As the activities concentrate in the transfer area, dual rails are adopted to reduce congestion and, as a consequence, improve the throughput. Transfer area has 3

horizontal paths formed by dual rails so TUs on adjacent horizontal paths will not influence each other and the TU blocking area will only cover the adjacent slots on the left and right of current slot. On the other hand, since the storage area has lower traffic intensity, only two horizontal single rail paths are designed for cross-column movement. However, as most TUs in storage area travel vertically, TU congestion on adjacent columns becomes a big issue. In fact, due to the particularity of GRID structure, TU routing becomes an important issue in this study which has to be handled carefully.

3.1.2. Conflict Scenarios

The biggest issue of shared rails for this structure is that it can easily lead to congestion and possible deadlocks, thus hindering the productivity of the system. For example, if one TU is occupying one slot in storage area, then the surrounding eight slots cannot have any TUs, otherwise a deadlock might occur, such as the case shown in Figure 3.10. Therefore, this study tries to establish a simple and efficient TU control logic – free routing traffic rule – to prevent congestion in advance and resolve the deadlock quickly. To be specific, the study firstly identifies eight potential conflict scenarios between two TUs as shown from Figure 3.4 to Figure 3.11 and then in the next section conflict free routing policies are proposed to handle these scenarios. It should be noted that conflicts among more than two TUs are not discussed as most of these conflicts can be decomposed into several conflicts between two TUs.

In each scenario, TU1 and TU2 cannot move further. Arrows in the figures indicate TUs' travelling direction. The shaded area is the area blocked by one TU and only

this TU can access it. The area shaded in stripes is blocked by two TUs and no TU can access it.

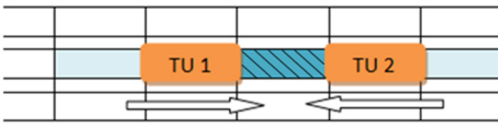


Figure 3.4 Conflict scenario 1

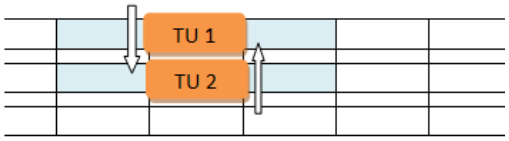


Figure 3.5 Conflict scenario 2

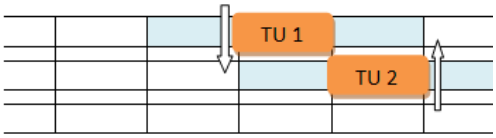


Figure 3.6 Conflict scenario 3

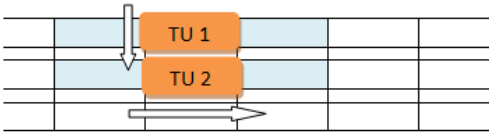


Figure 3.7 Conflict scenario 4

The first four scenarios are occurring in the transfer area (Figure 3.4 - Figure 3.7). In particular, under scenario 1, TU1 and TU2 both want to occupy the position between the two TUs but none can. In scenarios 2 and 4, one or both TUs are trying to occupy the other's position. In scenario 3, TU1 and TU2 cannot move further because they are using the same rail.

The remaining scenarios (Figure 3.8 – Figure 3.11) will occur in the storage area and they are similar to those in the transfer area. However, for scenarios 6 and 7, solving conflicts may take more efforts compared to other conflicts. For example, TU2 in scenarios 5 and 8 only need to move 2 slots then TU1 can pass, but TU1 or TU2 in scenarios 6 and 7 has to leave the segment first then the other TU can pass. Hence it is better to prevent deadlocks in scenarios 6 and 7 rather than solve them.

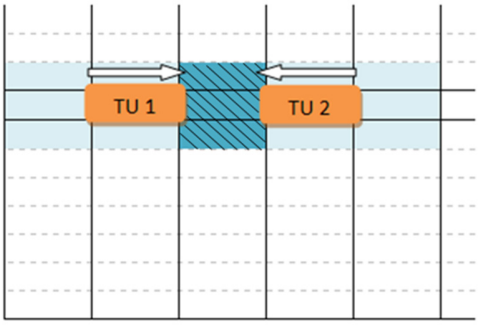


Figure 3.8 Conflict scenario 5

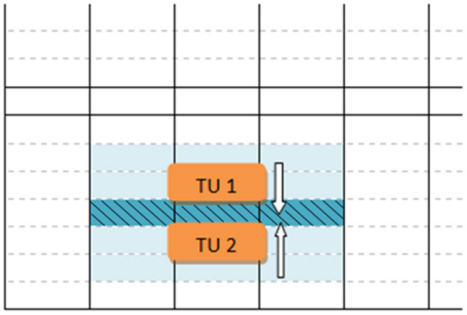


Figure 3.9 Conflict scenario 6

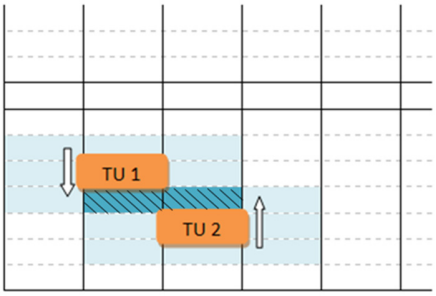


Figure 3.10 Conflict scenario 7

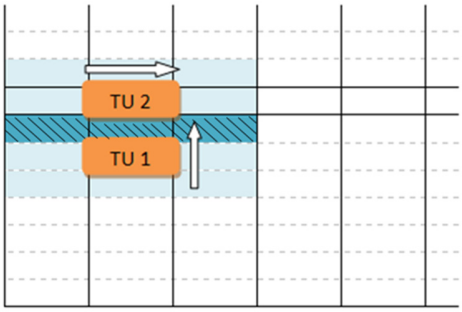


Figure 3.11 Conflict scenario 8

3.1.3. Free Routing Traffic Rule – TU Control Logic

Three main principles govern the traffic rule.

Principle 1: TUs are always seeking the shortest path to their destination.

Principle 2: TUs search horizontal path first then vertical path. If TUs are on the same column as the destination, TUs search vertical path first unless a conflict is encountered.

Principle 3: There is no pre-planned path for a TU. A TU makes its decision step by step, which enables TUs to adjust their path whenever conflicts happen.

Principle 1 allows TUs to travel on shortest path and *Principle 3* allows TUs to make real-time decision when conflicts are met. Moreover, *Principles 1* and *2* could help to reduce conflicts in vertical direction, for example, most loading containers will travel on one vertical path from storage area to transfer area in the same direction. However, routing principles cannot resolve conflicts and deadlocks. As such, we introduce the following three rules.

Rule 1: first come first act

- When a conflict between two TUs is detected, the TU which spots the conflict first will take action while the other TU will wait at its current location. But if the first TU cannot solve the problem for some period of time, then the other TU will act. In conflict scenarios 4 and 8, TU2 has already occupied its current location and TU1 wants to cross over, so TU2 comes first and it should act first. Once TU2 continues its path and releases its current location, TU1 can pass.

Rule 2: direction restriction in the storage area

- TUs with different directions are not allowed to travel on the same vertical path segment in the storage area (marked by dashed rectangle) and travelling direction of the current TUs will restrict the travelling direction of other coming TUs on adjacent vertical path segments. For example, as shown in Figure 3.12, TU1 is moving upwards on path segment 3 and TU2 is moving downwards on path segment 5. Now TU3 is coming. TU3 can either travel upwards or downwards on path segment 1 as it will not affect

any other TUs. If TU3 travels on path segment 2, 3, 5 or 6, it has to follow the direction of TU1 or TU2; otherwise scenario 6 or 7 is created. It should be noted that TU3 cannot travel on path segment 4 in both directions; otherwise scenario 7 is created. Therefore, applying *Rule 2* can prevent conflict scenarios 6 and 7. However, *Rule 2* will bring in two new conflict scenarios 9.1 and 9.2 as shown in Figure 3.13 because it only restricts the travel direction on path segment without considering junctions.

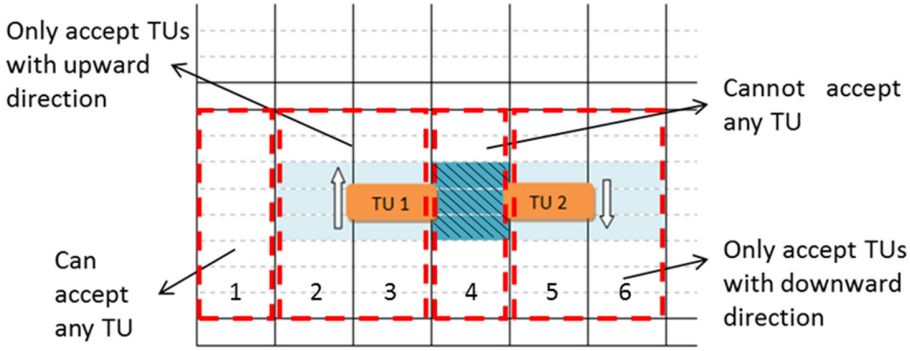


Figure 3.12 Example of Rule 2

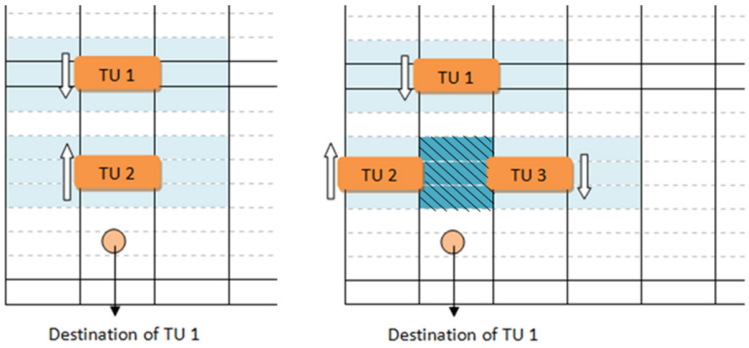


Figure 3.13 New conflict scenarios 9.1 and 9.2 produced by Rule 2

Rule 3: detouring

- For the rest of the conflict scenarios (1, 2, 3, 5, 9.1 and 9.2), the only solving method is to allow TU detouring by moving TU1 to next available row/column while delaying TU2 (and TU3) based on *Rule 1*. Allowing

detouring means that *Principle 1* is temporarily violated and it is acceptable because deadlock is the most unwanted case.

Table 3.1 summarizes how control rules deal with conflict scenarios.

Table 3.1 How control rules deal with conflict scenarios

Control rules	Conflict scenarios	Remarks
Rule 1	4, 8	
Rule 2	6, 7	Brings in two new cases 9.1 and 9.2
Rule 3	1, 2, 3, 5, 9.1 and 9.2	Violates Principle 1

However, as expected, conflict problems will occur when there are too many TUs. Although most of the conflicts among multiple TUs (more than two) can be decomposed into several conflicts between two TUs, current conflict free routing rules cannot guarantee that all complex cases can be eliminated.

3.1.4. Simulation Design

In this section, we are interested in converting a real system into a discrete event based simulation model. The model is built in AutoMod which is a leading graphical simulation software in manufacturing. The model consists of two parts, the visual model which describes the physical structure of the single GRID system and the logic system for container and TU control. However, the converting process has several difficulties to be solved.

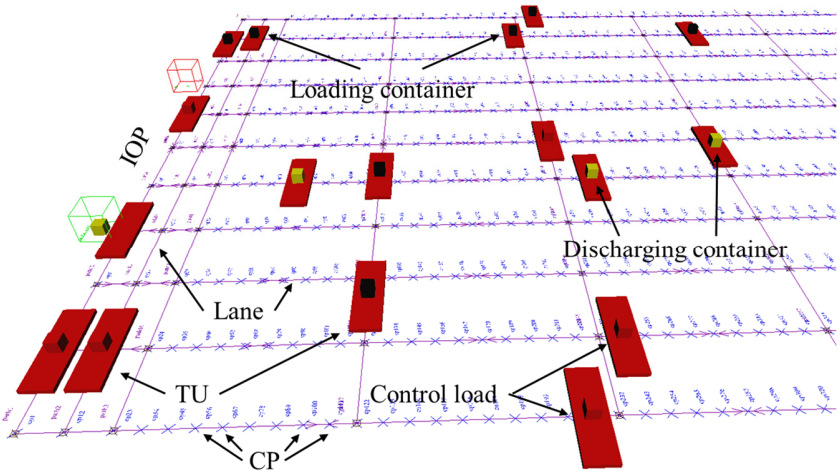


Figure 3.14 Visual model for single GRID system

Firstly, we need to convert the real structure in Figure 3.1 into the visual model in Figure 3.14. To be specific, the dimensions of the simulation model are based on the real dimension of the single GRID system. Each lane in the model represents a path which is formed by two parallel adjacent single rails in the real structure and the storage slots in the real structure are replaced by control points (CPs) attached to the lanes. For example, a TU traveling in the lane in the model means that the TU is moving between storage slots, and if a TU is loading/discharging in a storage slot, it will occupy the current CP and claim the surrounding CPs due to TU blocking. Besides, the concept of input/output point (IOP) is defined to simplify the interaction between QC, TP and corresponding CP in the transfer area. For example, the container will wait for picking up at the IOP until an empty TU reaches while a loading container will wait on the TU at the IOP until available.

Secondly, we need to get the TU to move. Instead of defining the TU route, this study uses the load to control the TU movement. In this study, the load represents the container and three types of containers are introduced: loading containers,

which moves the TU from CP in the storage area to IOP, discharging containers, which moves the TU from IOP to CP in storage area, and the dummy container, which is simply used to control an empty TU moving between CPs. To be specific, initially a dummy container is on board the TU. Once a working (loading/discharging) container calls for the TU, the dummy container will know the position of the working container and then drives the TU to the destination. Upon arriving, the TU will drop the dummy container and wait at the current location for 40 seconds to simulate the process of lowering the grappler and picking up the working container. Since the working container already knows its destination, it will drive the TU towards the destination once it is on board. When the TU drops the working container, a new dummy container or another working container will get on board. The whole process is shown in Figure 3.15.

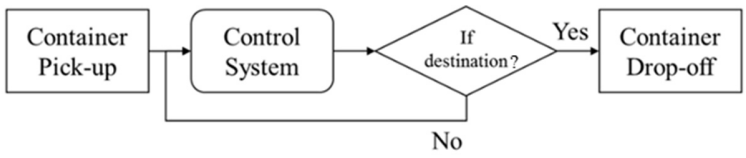


Figure 3.15 Flow of the TU movement

Thirdly, the control system is very important to the model. By giving instruction to the on-board container, the control system determines the movement of each TU step by step. To be specific, whenever the TU reaches the current CP, the control system will search for the next CP based on the shortest path principle and check the availability of the next CP. If the next CP is available, the TU will move; otherwise, the control system will determine either to search for alternative CP based on control logic or waiting at its current CP (for 0.3 seconds). Whenever the time for waiting action is up or the TU reaches the next CP, the control system will

send new command until the TU arrives at the destination. The details of the control system are shown in Figure 3.16.

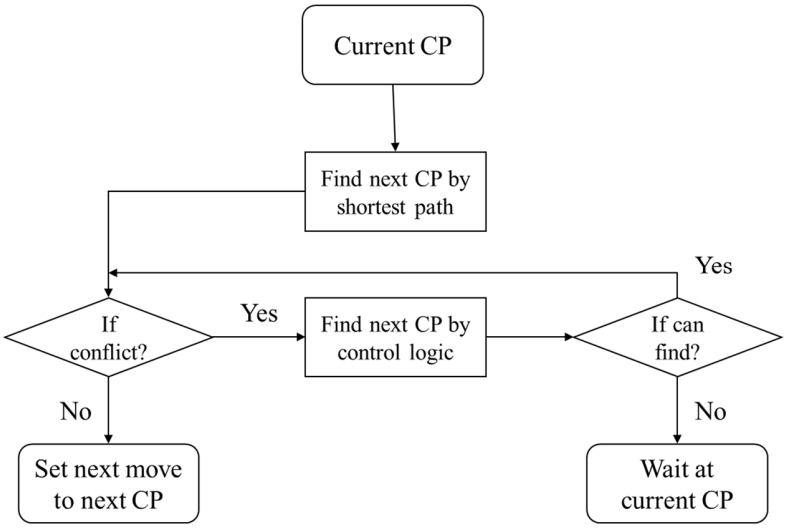


Figure 3.16 Details of control system

Lastly, the way of generating working containers significantly affects the throughput. At the beginning of the simulation, a container list with 100 initial containers is created. In order to make the simulation more conservative, we assume that the type, origin and destination of the container are randomly determined. To be specific, each container has an equal chance to be a discharging or loading container. If it is a discharging (loading) container, its origin (destination) is selected from one of the IOPs with equal chance and its destination (origin) is selected from one of the CPs in the storage area with equal chance. Once the container is picked up by the TU at its origin, it is removed from the list and if the container is dropped at its destination, the TU will be assigned to the first available container on the list. If the number of containers on the list is not enough, another 100 containers will be generated until the end of the simulation.

3.1.5. Performance Analysis

The performance analysis obtained from the simulation is the maximum ideal system throughput of the single GRID system since the QC behavior and storage capacity are ignored. In particular, it is assumed that each IOP always has containers available for discharging or is ready for loading. Besides, it is assumed that the storage capacity is infinite in each layout and containers can continue moving in or out without considering if there is space for storage. It should be noted that these two assumptions are reasonable in this study as the capability of TUs using the proposed control logic is a key parameter of the GRID system. Once the maximum ideal throughput is obtained, the remaining sub-systems can be designed according to it.

The first system analyzed is a small layout having a width of 136 meters, depth of 64 meters and two IOPs on the quayside. We tested the control logic with a number of TUs ranging from 1 to 8. The simulation length is eight hours, which is one shift in real port operations. The system performance is measured in terms of container throughput per IOP per hour and it is calculated using the formula

$$\frac{\text{Total discharging containers} + \text{Total loading containers}}{\text{Number of IOPs} \times \text{Time length}}.$$

In the beginning, all TUs are idle and randomly located among the whole yard. A warm-up analysis is used to estimate how long it takes to reach a steady throughput. In the warm-up analysis, the throughput of every 15 minutes is recorded for each test, as shown in Figure 3.17. It can be noticed that the warmup period of the case

with less TUs is shorter than the case with more TUs. To be consistent for all the tests, the first 2 hours are considered as a warm-up period and they are not included in the estimation of the simulation response.

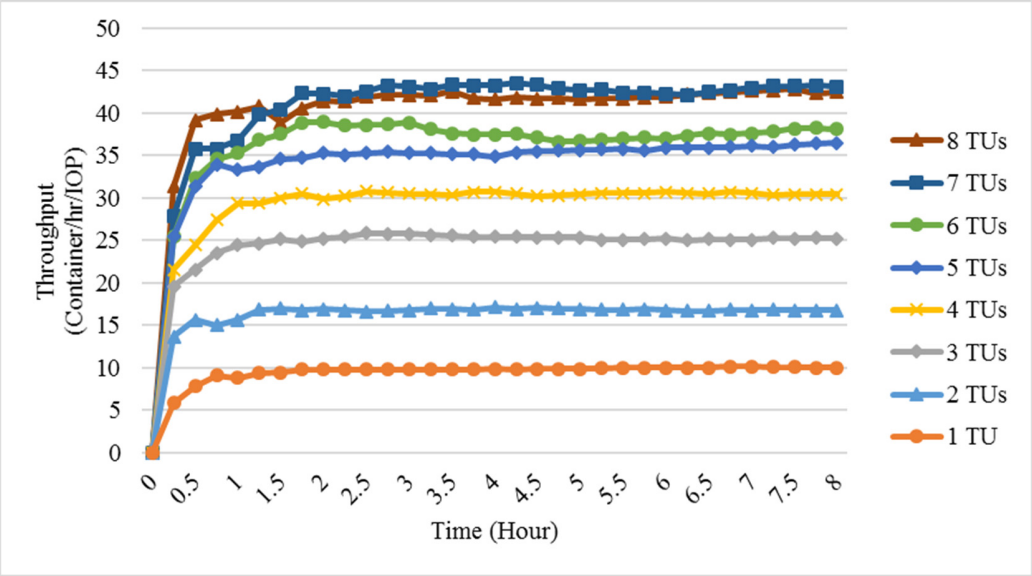


Figure 3.17 Warmup analysis

For each test, we performed five replications and then calculate the mean and variance as the test result. The details can be found in Table 3.2 and Figure 3.18. Figure 3.18 shows that the system throughput increases until its maximum value of 42.9 containers per IOP per hour when the number of TUs reaches 7. This value is slightly larger than the throughput of a typical QC, which is around 40 containers per hour. This indicates that the single GRID system in a small layout can perform quite well in ideal situations. Besides, the standard deviation of the mean in Table 3.2 shows that the model is quite robust to different TU numbers.

If the number of TUs keeps increasing, the conflicts become more and the overall throughput starts decreasing as the TUs end up spending more time on solving conflicts, such as, waiting for other TUs to pass by or detouring.

Table 3.2 Detailed simulation results of small layout

Number of TUs	1	2	3	4	5	6	7	8
Throughput (Containers/hour/IOP)	9.4	16.8	24.4	30.2	35.7	39.5	42.9	42.3
Standard Deviation	0.16	0.33	0.55	0.46	0.89	1.08	0.74	0.83

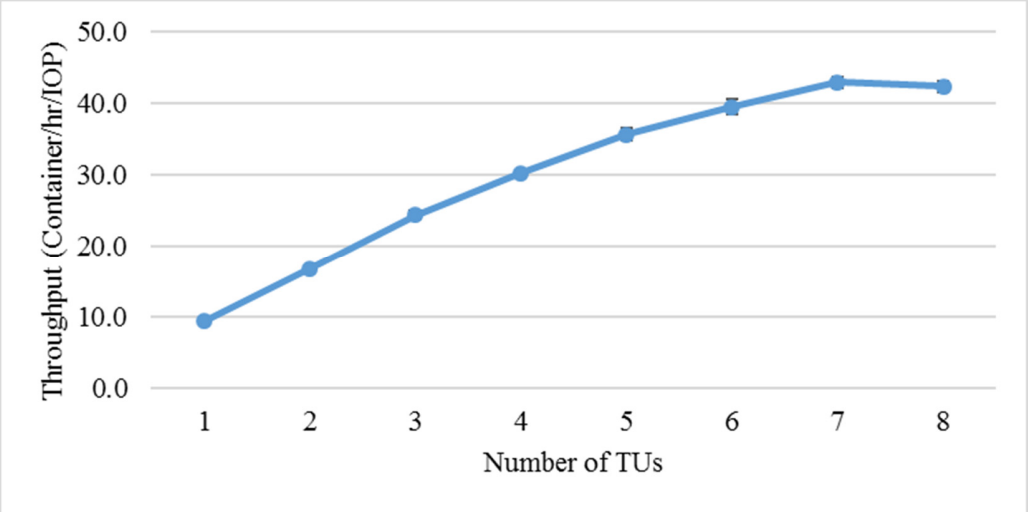


Figure 3.18 Throughput of small layout for different TU numbers

3.1.6. Layout Expansion

The ideal throughput of a small layout shows that the GRID is quite efficient. However, when taking storage capacity into consideration, the capacity is far from enough to match the throughput. For example, each of the two IOPs could reach 42.9 containers per hour. It is assumed that half of the containers move in and half move out. It is also assumed that half of the containers are 20-ft size and half are 40-ft. Besides, for a general transshipment terminal, the average duration-of-stay of a container is 5 days and 20% space is reserved for container fluctuation and potential congestions. If the system is highly utilized, the estimated space is around

$$[(42.9 \times 1.5) \times 2 \div 2] \times 24 \times 5 \div 0.8 \approx 9,653\ TEUs\ .$$

However, since the small layout only has 2,200 TEUs, the average throughput is 9.8. Although the maximum throughput could reach 42.9, the low average throughput indicates that the big advantage of the GRID system is wasted. Besides, a small layout can at most have two QCs attached at the same time, which cannot meet the demand of real terminals. In fact, the small layout is expected to expand for larger capacity, higher throughput or have more QCs serving together and it is necessary to figure out the effect of layout size on system throughput. Therefore, the following two expansions are introduced.

The first expansion is the vertically expanded layout that keeps a width of 136 meters, but increases the depth of storage area so only the total capacity is increased, as shown in Figure 3.19 (left). The arrow in the figure represents the expansion direction. The second expansion is the horizontally expanded layout that is introduced by increasing the width by three times, which reaches 408 meters wide and increases capacity to 6,600 TEUs. It allows six QCs to work together, as shown in Figure 3.19 (right). The simulation models were modified accordingly based on the system dimensions and experiments were conducted using the same procedures as the small layout.

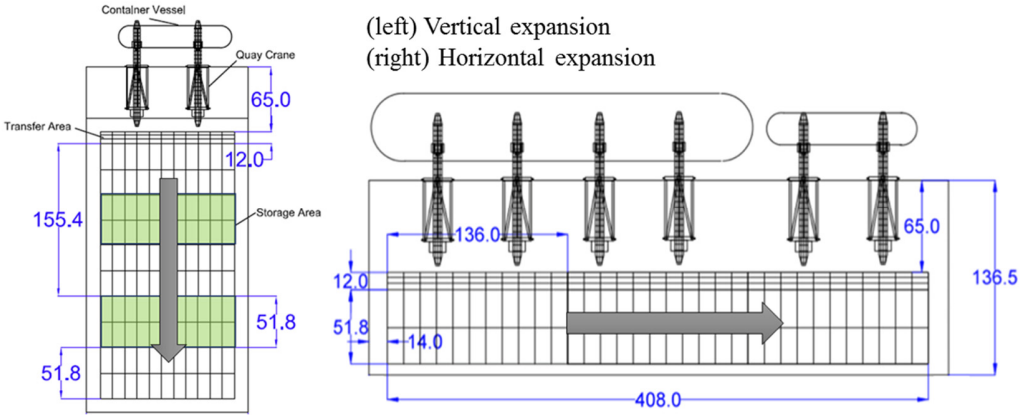


Figure 3.19 Vertical (left) and horizontal (right) expansion layout

In the case of vertical expansion, the design keeps the width of 136 meters unchanged which can at most have two IOPs and it increases the depth of storage area by two to five times and the capacity keeps increasing as depth increases. As observed in Figure 3.20, as depth increases, more TUs are required in order to achieve the maximum throughput while the system throughput is barely affected. Since the capacity is finite, to achieve the maximum throughout the ideal size of the single GRID system should be the 5*Depth layout.

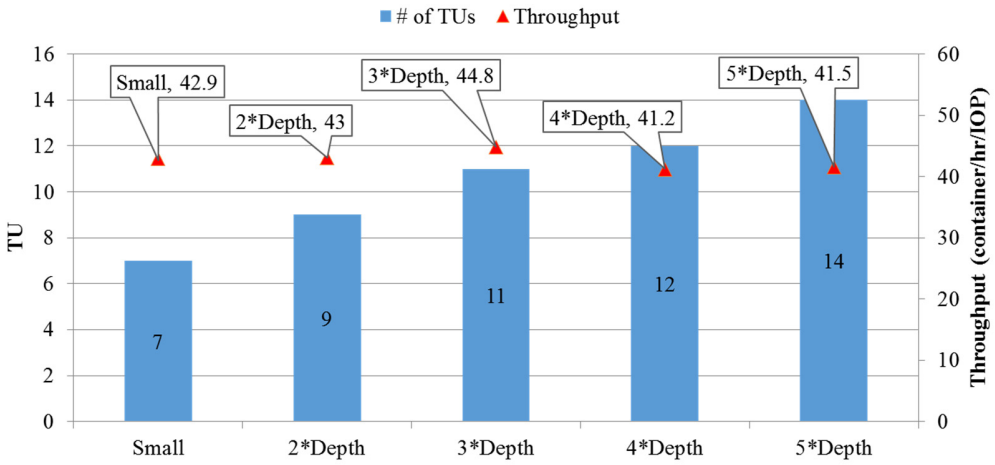


Figure 3.20 Throughput comparisons of the small and vertically expanded layouts

The system performance of the horizontally expanded layout is discussed next. The dimension increases the width by three times, which reaches 408 meters wide, 64

meters deep and allows six IOPs at the quay side. The capacity is also increased to 6,600 TEUs. However, as shown in Figure 3.21, the system can only achieve a maximum throughput of 27.1 per hour per IOP with 18 TUs. Comparing the small layout and horizontally expanded layout, it is obvious that increasing the width significantly decreases system throughput. Besides, when comparing horizontal expansion and vertical expansion with the same capacity, horizontal expansion requires more TUs to achieve maximum throughput while the throughput is far worse than vertical expansion.

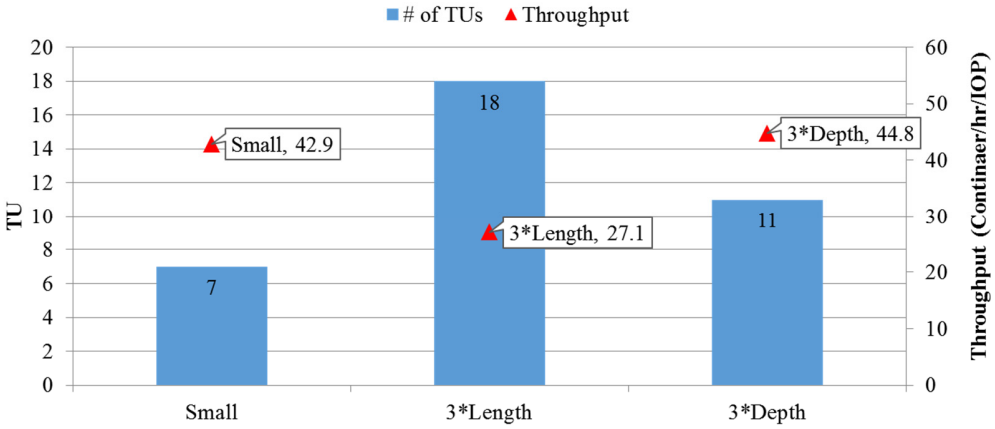


Figure 3.21 Throughput comparison between different expansions with the same capacity

As can be seen from the experiments, the system throughput of the single GRID system drops as the width increases, but it remains almost the same when the depth increases. This is due to the following reasons: Horizontal expansion has more IOPs than vertical expansion, which means that horizontal container flow, or equivalently cross-section movement, will be very large. Besides, conflicts increase in the transfer area as all points can handle both discharging activities and loading activities. Furthermore, the storage area only has two horizontal paths so the TUs

have to use the transfer area for horizontal movement. Since too many TUs have to pass through the transfer area or access IOPs, heavy congestion will occur.

In summary, the vertical expansion is mainly designed for increasing capacity due to the robustness of throughput while the horizontal expansion is for serving with more QCs. Although the horizontal expansion could probably satisfy the capacity constraint, the increasing congestion in the transfer area will limit its further expansion in vertical direction. For transshipment terminals that require high capacity and productivity, it is obvious that both vertical and horizontal expansions of the single GRID system do not satisfy the demand and therefore, the hybrid GRID system concept is proposed.

3.2. Hybrid GRID System

3.2.1. Design Configuration

Due to the limitations of the single GRID system, this new concept is proposed to serve terminals demanding high capacity and productivity. The hybrid GRID system includes three parts: quay-side, transportation system and storage yard. On the quay-side, standard QCs are deployed.

As shown in Figure 3.22 and Figure 3.23, the storage yard is divided into *n* sections with *r* tiers and *c* columns and an independent transportation system is proposed to support the storage yard. Paths are surrounding sections for vehicles to handle all ground movements on the quay-side, yard-side and in between. In this design, we do not use the housekeeping concept, which brings in a lot of handshakes, double handlings and inter-section movement. In most cases, the containers are discharged

from the vessel, sent to the assigned location for storage and loaded to another vessel in the near future. In addition, since the throughput of the single GRID system is high, the independent transportation system uses ALVs to move containers. Indeed, it has been analytically proven that the ALV is superior to the AGV in productivity by reducing handshakes and waiting time (I. F. Vis & Harika, 2004; Yang *et al.*, 2004).

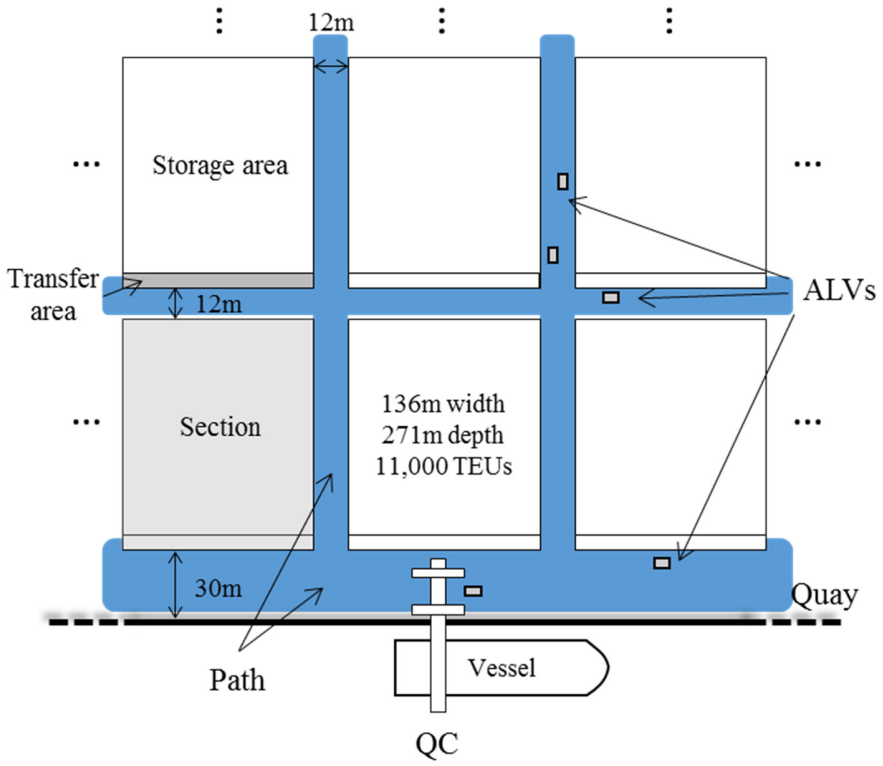


Figure 3.22 Plane structure of hybrid GRID system

The storage yard has several sections and each section is covered by a single GRID system. Since the capacity is finite and based on the findings of the single GRID system, vertically expanded layout is preferred to be used in the hybrid GRID system. Sections are connected with each other via a bidirectional link that the TUs can use to move to other sections. We distinguish two section categories: Bay-Front

section (BFS), which is close to quay side and can directly access QCs, and Inner-Yard section (IYS), which is behind BFS.

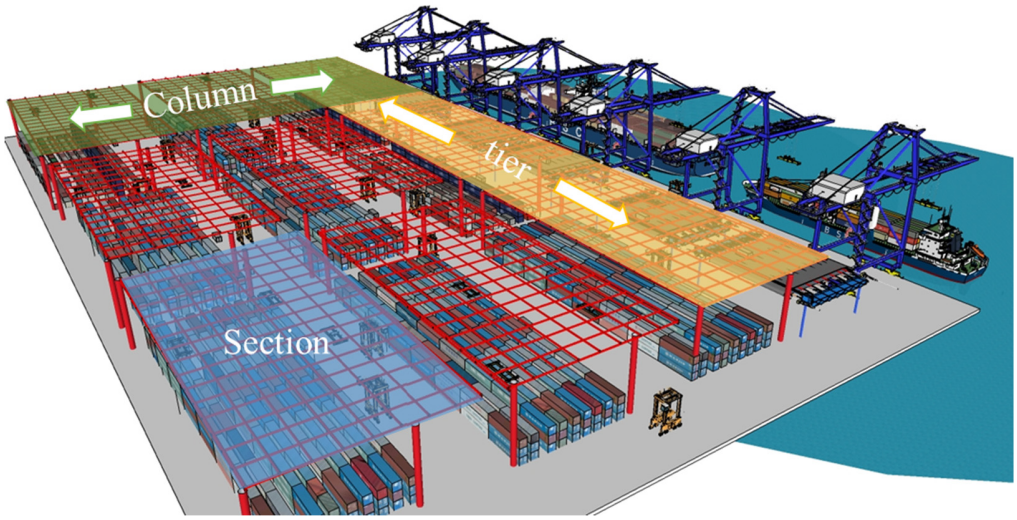


Figure 3.23 Demonstration on hybrid GRID system

Figure 3.24 demonstrates how the QC and ALV cooperate with each single GRID system in BFS (left) and IYS (right). In the BFS design, the QC can access both the ALV and GRID, i.e. the QC can drop the container on the ground and then the ALV comes to pick up, or, QC puts the container on the TP (direct transfer system) and then by conveyor, the containers are moved to the GRID for the TU to pick up or in reverse order. Since the transfer area of BFS has to serve the TP and ALV, it is deeper than the transfer area of the single GRID system. On the ground, there are three types of vehicle paths: (1) the path under QC, (2) the path under GRID and (3) direct transfer path, which is under the TP and not connected with any equipment. The IYS design is much simpler than the BFS as it cannot be accessed by QCs directly and so the TP is removed. The ALVs will move containers in/out to the yard/quay side. On the ground, this structure only has two types of paths, which are the path under GRID and direct transfer path.

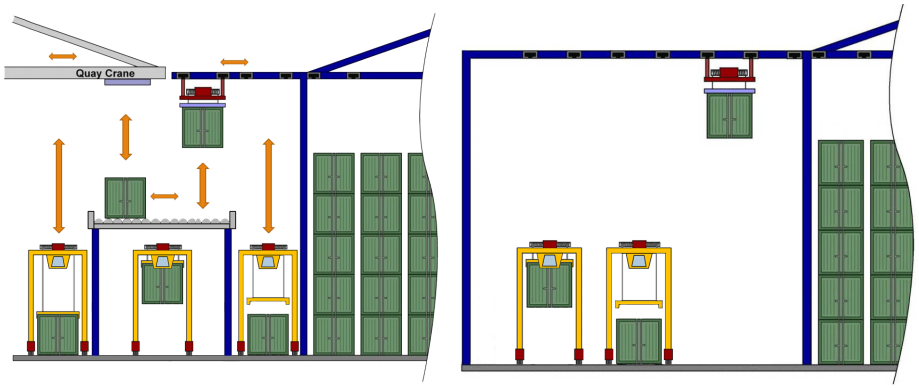


Figure 3.24 Structure of the transfer area of BFS (left) and IYS (right)

The following example shows how the hybrid GRID system works. For a discharging container, if the discharging QC is right next to the assigned single GRID system, the container will be dropped on the TP and transferred to the assigned position for storage. If the discharging QC is not close to the assigned GRID, for example, the GRID is in the first tier but far from this QC, or the GRID is in other tiers, the container will be dropped on the ground for the ALV to pick up and deliver to the assigned GRID. Loading of containers operates in a similar way but in reverse order.

3.2.2. Comparisons on Land Utilization

In this section, land utilization is compared between the hybrid GRID system and several current terminal configurations. In Euromax Terminal Rotterdam, AGVs are used in combination with stack yards served by Automated Stacking Cranes (ASC). AGVs do not drive into the storage area, but only interchange with the ASC at the waterside. In Norfolk International Terminal, containers are transported and stored using straddle carriers while in Pasir Panjang Terminal, RTGs and trucks are deployed. Trucks drive into the storage area and interchange with the RTG at the side of every stack.

Because of their different port sizes, the land utilization is calculated using

$$\frac{RSA}{YA} \times \frac{SH}{5}$$

where *RSA* stands for the total real storage area (net space occupied by containers), *YA* represents the total yard area (includes gaps between containers, space for ground vehicles but excludes the area between the yard side and quay side) and *SH* is the stacking height which is 5 by default.

Table 3.3 Land utilization comparison

Terminal	Euromax Terminal Rotterdam	Norfolk International Terminal	Pasir Panjang Terminal	Hybrid GRID 6 Sections as in Figure 3.22
Depth	38 TEUs 234 meters	12-14 TEUs 74 – 87 meters	34 TEUs 210 meters	100 containers 259 meters
Length	10 containers 25 meters	24 containers 103 meters	6 containers 15 meters	22 TEUs 136 meters
Height	5 containers	3 containers	5 containers	5 containers
Capacity (CP) (TEUs)	87,400	38,016	107,100	33,000
Land Utilization	65%	40%	40%	85%

As shown in Table 3.3, both Norfolk terminal and Pasir Panjang terminal have low land utilization. In Norfolk, this is due to the fact that every stack is at most three containers high, limited by the height of the straddle carrier, and the gap is wider

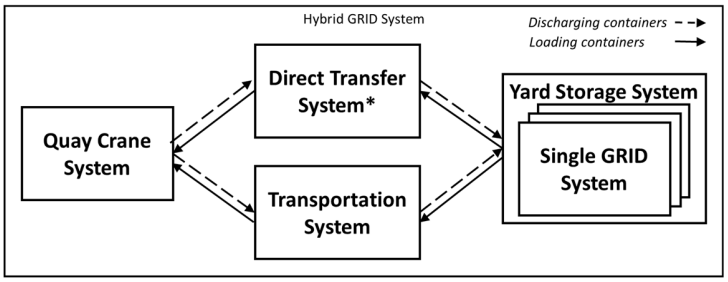
between every stack for straddle carriers to pass. In Pasir Panjang, it is due to the wide path needed for trucks to travel on. Euromax performs the best among all current terminal designs because AGVs do not drive into the storage area. However, the increasing block length in Euromax will decrease crane productivity. It is this reason that Euromax can only expand its space horizontally. It is quite obvious that the hybrid GRID system is the best design in terms of storage capacity and land utilization. This is because the ALVs do not move into the storage area and there are less aisles between sections. Furthermore, the section capacity can be further improved by increasing the height of the single GRID to 8 containers.

3.2.3. Simulation Model

Starting from this section, the effectiveness and efficiency of the hybrid GRID system will be validated by simulation experiments. However, the single GRID model is no longer suitable as it integrates all entities, including GRID and QC together which makes it difficult to expand. Therefore, a new simulation model considering flexibility and scalability is introduced.

The model for the hybrid GRID system is modularized into four sub-models: quay crane system (QCS), transportation system (TS), direct transfer system (DTS) and yard storage system (YSS) which consists of dozens of single GRID systems. The structure and flow are shown in Figure 3.25. To be specific, each quay crane in QCS is also treated as a fixed IOP; YSS uses the same model of the single GRID system, but YSS is connected to TS (and DTS in some cases) instead of IOPs. DTS uses a buffer to hold containers until it can move them to QCS or YSS; TS is an independent vehicle system which could directly pick up from the buffer between

TS and YSS or discharge container to the buffer. The other configurations such as real-time TU routing process and conflict solving process remain the same.



* Only the first tier single GRID systems can connect to transfer platform in DTS.

Figure 3.25 Simulation model for hybrid GRID system

In container generation, the type, origin and destination of a container are still randomly determined as the simulation model for single GRID system. However, the flow of the containers varies in terms of the origin and destination. The origin of the discharging (loading) container is randomly set at one of the IOPs in QCS (one of the CPs in a single GRID) and its destination is randomly set at one of the CPs in a single GRID (one of the IOPs in QCS). If origin IOP (destination IOP) is attached to its destination GRID (origin GRID), then the container is sent to the YSS via DTS, otherwise it uses TS to transport.

Table 3.4 ALV specifications

Minimum Turn Radius (m)		10
Time to load/unload (s)		20
Length, Width, Height (m)		9.5, 5, 11.5
	Speed (m/s)	Time from 0 to highest speed (s)
Loaded	3.9	11
Empty	5.8	15

Lastly, the ALV specifications can be found in Table 3.4 and other parameters will be the same as the single GRID system.

3.2.4. Experiments and Discussion

We conducted a series of experiments to validate the efficiency of the hybrid GRID system. It should be noted that the result represents an ideal maximum throughput as it is still assumed that containers are always available for handling and each section has infinite capacity.

Firstly, the influence of the number of ALVs is examined. A design is proposed as shown in Figure 3.26. It has three sections in one tier and each single GRID is 136 meters wide and 271 meters deep. Single-directional round ALV path is placed on the ground between the GRID and QCs and bypath is added at every IOP where ALVs can temporarily park without blocking the way. 14 TUs are given to each GRID with different number of ALVs. Each case runs five replications and each replication is 8 hours long.

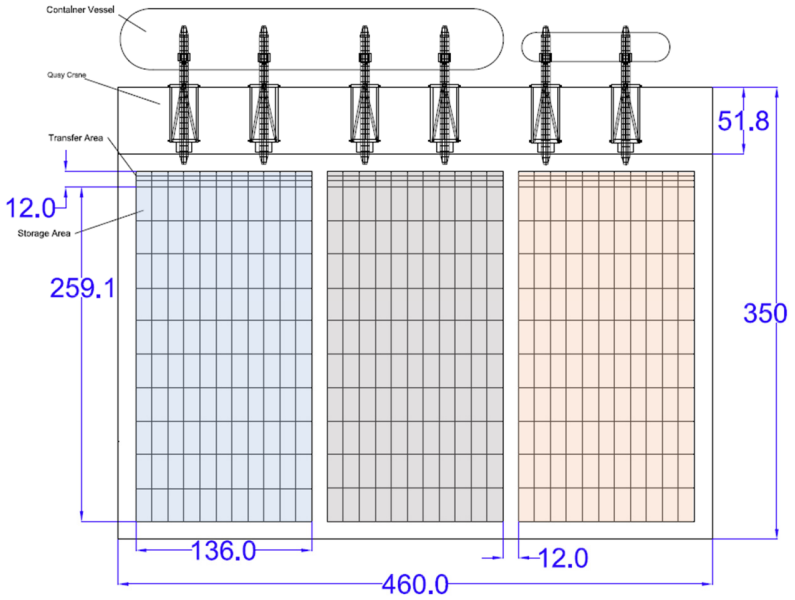


Figure 3.26 Design of hybrid GRID system (3 sections, 1 tier, 271 meters deep)

The result in Figure 3.27 shows that the system throughput of the hybrid GRID system is affected by the number of ALVs. Obviously enough ALVs are required to handle cross-section movements.

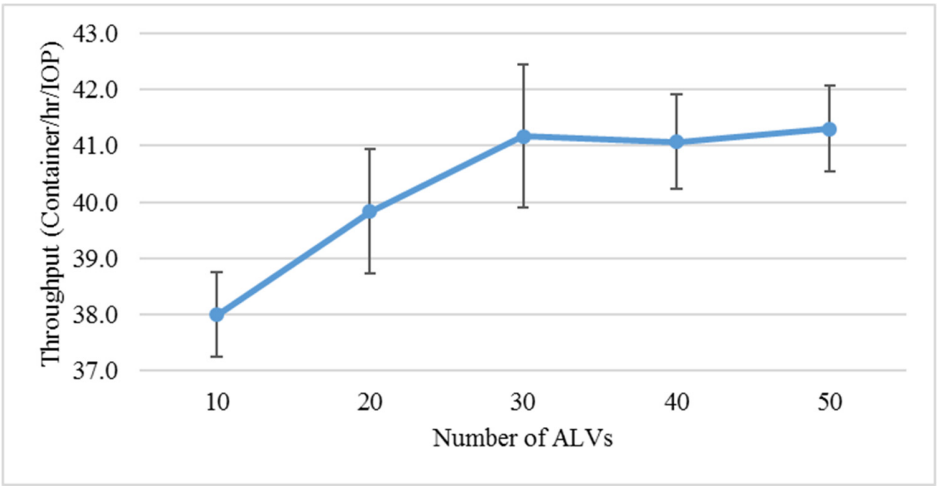


Figure 3.27 Influence of ALVs in the hybrid GRID system

Secondly, the bottleneck analysis is carried out by two comparisons. The first comparison is made between the throughput of the hybrid GRID system and throughput of the single GRID system with the same depth. Since the former

represents the whole system while the latter represents each section, if the former throughput is slightly smaller than or equal to the latter throughput, it indicates that the sub-systems between single GRID system and IOPs are not the bottlenecks or otherwise new designs are required. The second comparison is made between the throughput of the hybrid GRID system and the throughput of a typical QC which is 40 containers per hour. If the former throughput is larger than or equal to the latter throughput, it indicates that the overall throughput can satisfy the general terminal requirement and the hybrid GRID system can be deployed to current terminals. The simulation results are given in Figure 3.28.

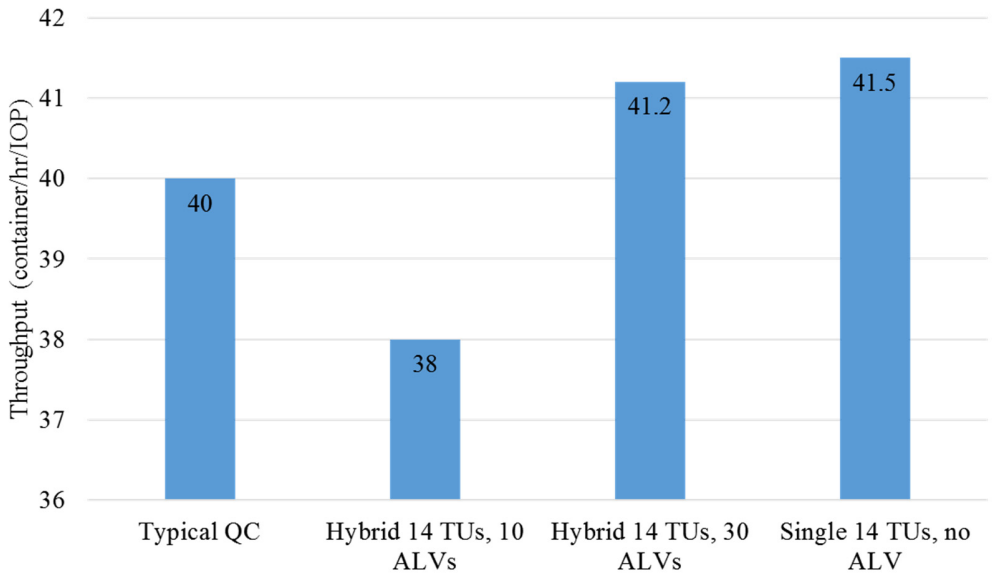


Figure 3.28 Throughput comparisons between single, hybrid GRID systems and typical QC

As can be observed, the system throughput of the hybrid GRID system with sufficient ALVs is very close to the corresponding single GRID system and is higher than that of the typical QC. It shows that all sub-systems in the hybrid GRID system can efficiently cooperate with each other and overall productivity satisfies the terminal requirement.

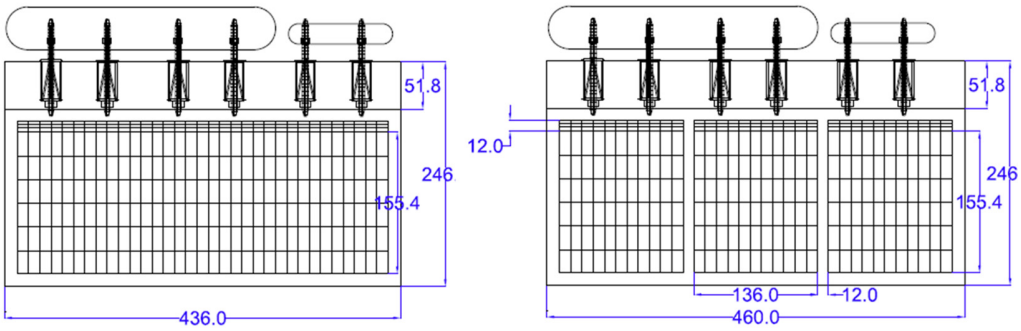


Figure 3.29 A design of single GRID system and hybrid GRID system

The last experiment is to compare a single GRID design and a hybrid GRID design which have the same capacity. The single GRID system (Figure 3.29, left) is 408 meters wide, 167 meters deep and with capacity of 19,800 TEUs. It can have up to 6 QCs attached at the quay side. The hybrid GRID system (Figure 3.29, right) has 1 tier with 3 sections. Each section has a single GRID with 136 meters wide, 167 meters deep and capacity of 6,600 TEUs, so the total capacity is 19,800 TEUs. Without loss of generality, 20 ALVs are deployed in the TS. Since the hybrid GRID system has to reserve space between sections for ground vehicles, it takes 5.5% more space than the single GRID layout. The simulation result is shown below in Figure 3.30.

The single GRID system reaches its maximum throughput with 24 TUs while the hybrid GRID system needs 12 TUs in each section. Although the number of TUs and space required are more than the single GRID design, the throughput of the hybrid GRID design is much higher than the single GRID design. It is because the hybrid GRID inherits the feature of 3*Depth single GRID (which is 44.8 containers per IOP per hour). Besides, it efficiently cooperates with the TS and DTS. On the other hand, the increment on width greatly contributes to the low throughput. The

result provides a strong support on using the hybrid GRID system instead of the single GRID system for large terminal design.

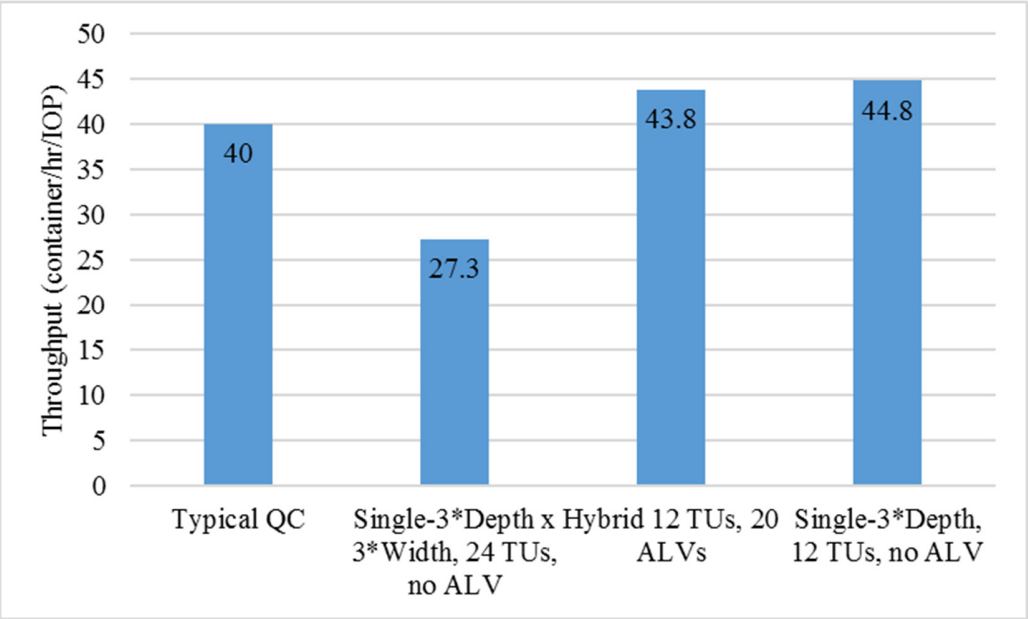


Figure 3.30 Throughput comparison between single and hybrid GRID with same capacity

When the system comes to actual use, the capacity constraint needs to be considered again. Since the capacity is finite, to achieve the maximum throughput under 6 QCs and 3 sections configuration, the ideal size of each single section should be 1*5 Depth dimension as in Figure 3.26. This dimension is based on the average of 80% utilization of storage space and 5 days of duration-of-stay time.

3.3. Conclusion

The single GRID system directly interacted with QCs and this Section introduced a small layout, horizontally and vertically expanded layouts. Due to the particularity of the GRID structure, TU routing becomes a critical issue so 8 representative conflict scenarios were firstly identified and then the free routing traffic rule was proposed, including 3 principles and 3 rules, for conflict prevention and resolution.

The simulation experiments revealed that the small layout could achieve quite a high throughput and vertical expansion would increase storage capacity while keeping its good performance. As for horizontal expansion, although it could increase storage capacity, the productivity declined dramatically. It should be noted that the capacity constraints are omitted so the results are the maximum ideal throughput. In summary, the limitations of the single GRID system prevented it from being applied in large terminals and motivated us to propose the concept of the hybrid GRID system.

The hybrid GRID system was proposed to serve terminals demanding high capacity, high land utilization and productivity. It consisted of several single GRID systems and used independent ground transportation system to deliver containers between quay-side and yard-side. A new modular simulation model was designed for the new system. The experiments proved that all sub-systems in the hybrid GRID system can efficiently cooperate with each other and overall productivity satisfies the terminal requirement. It means that the performance of the hybrid GRID system is promising and can be applied to transshipment terminals in the future.

Chapter 4. Information-based Allocation Strategy for Storage Allocation Problem in Hybrid GRID System

The concept of the hybrid GRID system has been introduced in Chapter 3. There are two major differences between the HGS and the conventional transshipment terminals. Firstly, the size of the HGS is flexible and scalable. It is scalable in both the width and the length and the number of TUs can be increased as the size of the GRID increases. For example, 3*Depth single GRID can achieve the maximum handling capacity with 11 TUs while 5*Depth single GRID needs 14. For the conventional system, the size of the block is limited by the yard cranes: the width of the block is restricted by the length of the yard cranes, and the length of the block is limited by how many yard cranes can be placed in the same block which is usually two at the maximum. Secondly, in the HGS, since the number of TUs can be a lot, the containers with the same profile are preferred to be stacked in the same slot and the slots should be spread out within the storage area to reduce traffic congestion during the loading activities; In the conventional terminal, since there are at most two yard cranes per block, it is preferred that containers slots with the same profile are located next to each other so that yard cranes need not move a large distance during the loading activities as yard crane movement is slow. This strategy is known as the consignment strategy and has been widely adopted, such as in Pusan Newport (Woo & Kim, 2011) and Singapore Port (L. H. Lee *et al.*, 2006). Although the simulation experiments in Chapter 3 have shown that it is a promising design in terms of overall productivity, due to the different behaviors, one important

problem HGS brings in is the storage allocation problem, which requires different arrangements and strategies from the traditional terminals.

In this study, we are interested in how the allocation strategy will affect the total cost including the variable cost and the setup cost with respect to the TUs. To obtain the optimal allocation decision, a MIP model for the deterministic problem is formulated by optimizing operating cost and capital cost. However, this approach might not be practical when solutions are needed to be computed quickly, especially when the environment is often very dynamic and uncertain, and so it is more realistic to use heuristic algorithms. Therefore, we develop an information-based allocation strategy (IAS) which uses greedy and sequential based algorithms to allocate containers. As the strategy requires the understanding of the optimal container allocation of the new system, solutions of the solvable terminal sizes are collected for analysis. In the analysis, the concept of convenient storage locations is defined for the terminal system and a regression model is proposed to analyze the collected data.

The proposed storage allocation strategy could be applied to different scenarios. For example, it is suitable for ports with larger terminal size. It can also be used for operational decision when arriving container information is available only a few shifts ahead or when the decision has to be made on a regular or ad-hoc basis.

The remaining chapter is organized as follows. The storage allocation problem in the hybrid GRID system is described in Section 4.1 and its MIP model is presented in Section 4.2. In Section 4.3, the new allocation strategy is introduced. The new

strategy is compared with traditional strategies in the numerical experiments described in Section 4.4. Finally, conclusions are summarized in Section 4.5.

4.1. Problem Description

In this study, we are interested to address the following research questions: Where should the containers be stored and how the allocation strategy will affect the total cost. The total cost consists of the variable cost and the setup cost with respect to the TUs. To be specific, the variable cost mainly comes from the unit container delivery cost incurred by the ground vehicles and transfer units, which depends on average container traveling distance, such as the fuel or power consumption cost. The setup cost is incurred from the transfer unit in terms of the capital and other overhead costs. Hence, the objective function could be written as:

$$\min \quad \alpha_1 D + \alpha_2 U$$

where D is the average container travel distance, U is the total number of TUs required for the whole system and the corresponding coefficients α_1 and α_2 represent the costs. Both coefficients can be evaluated and tuned according to the needs of a container terminal.

In the terminal configuration, there are S modules with S_r rows and S_c columns as shown in Figure 4.1. In the initial module design, Chapter 3 assumes that containers are allocated randomly in the storage area. However, this allocation strategy assumes each container is treated equally. This strategy can be very inefficient if the containers have different characteristics. For example, incoming containers can come with different dwelling times or different sizes and if containers with smaller dwelling time are stored in locations far from the quayside, then these storage

locations will be frequently visited, which imposes a lot of unnecessary travelling distance for the TUs. It will result in low productivity of the TUs. Hence, container profiling is an important factor for the storage allocation strategy. To address this issue, we apply a zoning strategy by equally dividing each module into S_k sections and so the yard has $K = S_r \times S_c \times S_k$ sections. For the whole yard, we have to decide the type and the number of containers that should be assigned in order that the overall productivity can be maximized. Intuitively, the system would need less TUs to maintain high productivity than when containers are randomly allocated, if the containers with right characteristics are assigned to the area close to the transfer area.

The space utilization is considered in this study to reduce potential reshuffling and avoid high stacking. For example, if the module storage capacity is 6,600 TEUs and the utilization is 0.8, then the available storage capacity is 5,280 TEUs. The remaining storage capacity can be used to provide extra space to deal with reshuffling. It is assumed that the berth positions of all vessels are known, such as the discharging/loading position (P_n^d, P_n^l) of each container. The travel distance of a container from quay-side to storage location is defined as the rectilinear distance from the discharging/loading position to the center point of the section. We assume that in each section, containers are randomly located so the center point of the section is used as pick-up and loading locations.

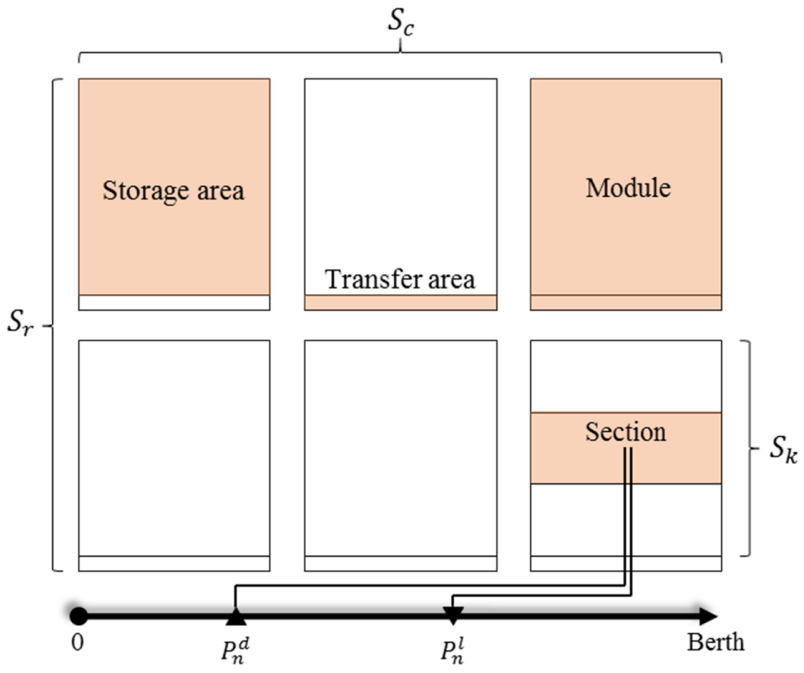


Figure 4.1 Plane Structure of HGS

It should be noted that GTs are not critical to the design of the HGS so the traffic congestion of GTs will not be considered. Chapter 3 has shown that the ground vehicles would not be a bottleneck of the HGS if the number of GTs is sufficient. The reasons are as follows: 1) due to the high storage capacity per module, the HGS has less ground paths so the traffic flow on the ground can be much simpler and easier to control. 2) the speed of GTs can be much faster than the TUs, e.g. the AGV can achieve 6 m/s while the TU can only achieve 2 m/s. 3) there are many access points between GTs and the GRID module so multiple containers can be handled simultaneously.

It is defined that the time unit in this study is a working shift, i.e. 8 hours, and the port operates on a 24/7 basis. The planning horizon is fixed at one week or 21 shifts. During each shift, a fixed number of TUs are assigned to each module and the TUs

can move freely within the module during the shift. If the workload changes in the next shift, the TUs can be redeployed to other modules.

A job is defined as a group of containers that are firstly discharged from the same original vessel O (o_n) at position p_n^d in shift t_n^d and will be eventually loaded to destination vessel D (d_n) at position p_n^l in shift t_n^l. It is assumed that the loading/discharging activity can be finished within one shift. The amount of 20-foot containers in this job is w_n^{20} and 40-foot containers is w_n^{40}. Usually the terminal operator will treat the 20-foot and 40-foot containers separately so each job can only have one type of container. Eventually, a job can be written as $\left(o_n, d_n, t_n^d, t_n^l, p_n^d, p_n^l, w_n^{20}, w_n^{40}\right)$. It should be noted that each job can either be a group of containers or an individual container. Usually for a large container terminal which serves millions of containers per year, the majority of the jobs consist of a group of containers. There might be cases where reshuffling is required. However, since the containers are grouped in a slot in terms of the same destination vessels, the reshuffling is kept minimal. As such it is not necessary to look at individual containers as this will make the problem too large and complex.

In the transshipment terminals, usually vessels arrive at and leave the port periodically. Although the container workload varies from time to time, as it is for planning purpose, we assume that the job schedule is deterministic and periodical, that is, the pattern of discharging and loading containers is the same every week. For example, if a job arrives in the current horizon and departs in the next horizon, the schedule will wrap-around and the job is defined as a wrap-around job, as illustrated by jobs 2 and 4 in Figure 4.2; otherwise, it is named as a normal job.

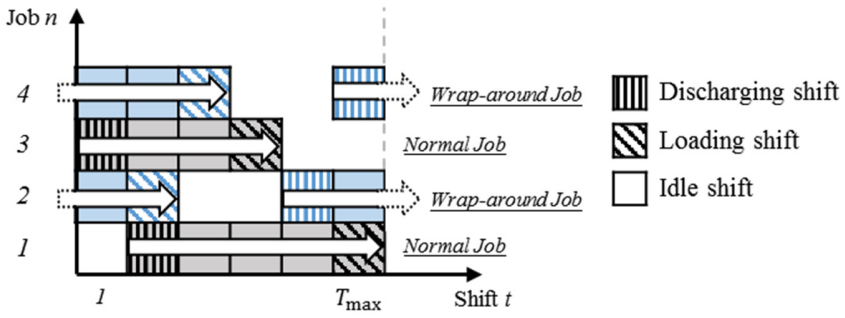


Figure 4.2 Wrap-around Job Schedule

In order to minimize the reshuffling, the containers in the same job are stacked in the same slots and the containers in different jobs are not allowed to share the slot. The slots for storing each job are reserved at the beginning of the discharging shift and will be immediately released right after the loading shift. Since the space needs to be reserved at a slot level only for a duration of stay unlike the consignment strategy in which the space has to be reserved for the entire planning horizon, the space utilization is greatly improved.

In summary, the assumptions in the study are listed as follows:

- Once the container is allocated to specific position, it will not be reallocated. This is helpful especially for ports that are very busy and operating on a 24/7 basis;
- The containers with the same profile are stacked in the same slot and mixed stacking is not allowed;
- In each section containers are randomly located so the center point of the section is used to represent the pick-up and loading locations;
- The ground transporter system is efficient and has sufficient handling capacity;

- The berthing schedule and the job schedule are pre-determined and will repeat itself every week; the loading/discharging activity of each job can be finished within one shift.

4.2. Modeling

We formulate the problem described above as a MIP model. Three sets of constraints which are the flow conservation, storage capacity and TU handling capacity have to be considered.

4.2.1. Model Development

Parameters

N	The set of jobs, index n;
S	The set of modules, index s;
K	The set of sections, index k;
K_s	The set of section indexes that belong to module s, i.e. $K_1 = \{1,2,3\}$
T	The set of shifts, index t, $t \in \{1,2...T_{\max}\}$, $T_{\max}$ represents the last shift;
M	The maximum number of TUs that each module can be assigned;
m	The number of TUs, $m \in \{0,1,2...M\}$;

c	Storage capacity, measured by the number of 40-foot containers;
β	Space utilization, a constant value less than 1;
$l_{o_n k}, l_{d_n k}$	Rectilinear distance from origin/destination of job n to section k;
w_n^{20}, w_n^{40}	Amount of 20-foot/40-foot containers in job n;
t_n^d, t_n^l	Discharging/loading shift of job n;
α_1, α_2	The weights on average travel distance and total number of TUs;

Decision variables

W_{nk}^{20}, W_{nk}^{40}	Workload of job n in section k, continuous variable;
C_{kt}	The storage level of section k at the beginning of shift t, $t \in \{1, 2...T_{\max}\}$, continuous variable;
$C_{k,T_{\max}+1}$	The storage level of section k at the end of planning period, continuous variable;
U_{st}	The number of TUs assigned to module s in shift t, integer variable;
U	The total number of TUs for the whole system, integer variable;
D	The average container travel distance, continuous variable;

Other notations

$f(\bullet)$ TU handling capacity function;

The *SAP* can be formulated as follows

(SAP) Objective:

$$\min SAP = \alpha_1 D + \alpha_2 U \tag{4.1}$$

Subject to:

$$D \geq \frac{\sum_{n=1}^{N}\sum_{k=1}^{K}\left(l_{o_n k} + l_{d_n k}\right)\left(W_{nk}^{20} + W_{nk}^{40}\right)}{\sum_{n=1}^{N}\left(w_n^{20} + w_n^{40}\right)} \tag{4.2}$$

$$U \geq \sum_{s=1}^{S} U_{st}, \quad t \in T \tag{4.3}$$

$$\sum_{k=1}^{K} W_{nk}^{20} = w_n^{20}, \quad n \in N \tag{4.4}$$

$$\sum_{k=1}^{K} W_{nk}^{40} = w_n^{40}, \quad n \in N \tag{4.5}$$

$$C_{kt} + \sum_{n \in \left\{n | t_n^d = t\right\}} \left(\frac{W_{nk}^{20}}{2} + W_{nk}^{40}\right) \leq \beta \cdot c, \quad k \in K, t \in T \tag{4.6}$$

$$C_{kt} + \sum_{n \in \left\{n | t_n^d = t\right\}} \left(\frac{W_{nk}^{20}}{2} + W_{nk}^{40}\right) - \sum_{n \in \left\{n | t_n^l = t\right\}} \left(\frac{W_{nk}^{20}}{2} + W_{nk}^{40}\right) = C_{k,t+1}, \quad k \in K, t \in T \tag{4.7}$$

$$C_{k,T+1} = C_{k,1}, \quad k \in K \tag{4.8}$$

$$C_{k,1} = \sum_{n \in \left\{n | t_n^d > t_n^l\right\}} \left(\frac{W_{nk}^{20}}{2} + W_{nk}^{40}\right), \quad k \in K \tag{4.9}$$

$$\sum_{k\in K_s}\left(\sum_{n\in\{n|t_n^d=t\}\cup\{n|t_n^l=t\}}\left(W_{nk}^{20}+W_{nk}^{40}\right)\right)\le f(U_{st}),\quad s\in S,t\in T \tag{4.10}$$

$$C_{kt},C_{k,T_{\max}+1},\quad k\in K,t\in T\text{ , non-negative continuous} \tag{4.11}$$

$$W_{nk}^{20},W_{nk}^{40},\quad n\in N,k\in K\text{ , non-negative continuous} \tag{4.12}$$

$$U_{st},\quad s\in S,t\in T\text{, non-negative integers} \tag{4.13}$$

Constraint (4.2) represents the average container travel distance. Constraint (4.3) guarantees that U is the maximum number of TUs of the whole yard among each shift. Constraints (4.4) and (4.5) allow a job to be split. Constraint (4.6) ensures that the current storage level and the incoming containers will not exceed the storage capacity. The variable C_{kt} is used to monitor the storage level of section *k* at the beginning of shift *t*. This value is updated in each shift based on the amount of arriving and leaving containers in each shift. To reduce reshuffling, some space is reserved and the space utilization is set to be β which is less than 1. Constraint (4.7) updates the storage level shift by shift, considering all incoming and leaving containers. Constraint (4.8) is used to represent the wrap-around job schedule. $C_{k,T_{\max}+1}$ is used to capture the wrap-around jobs. This variable indicates the storage volume of section *k* at the end of the planning horizon, or after shift $T_{\max}$. It is equal to the storage volume at the beginning of the 1st shift in the next planning horizon. Constraint (4.9) sets the initial storage level for the 1st shift. It should be noted that the initial containers belong to wrap-around jobs, or containers that stay across two planning horizons. Constraint (4.10) ensures that all inbound and outbound

containers can be handled by the GRID system and $f(U_{st})$ is the TU handling capacity function.

In addition, there are three propositions that we could learn from the model:

Proposition 1: Given different (α_1, α_2)***, the problem may have the same the optimal solution*** (D,U).

Proposition 2: D is monotonic non-increasing with U increasing.

Proof: Suppose (D_1,U_1) is the optimal solution of (α_1^1, α_2^1) and for a different (α_1^2, α_2^2), its optimal solution is (D_2,U_2). It is obvious that if $U_1 = U_2$ then $D_1 = D_2$. Suppose $U_1 > U_2$ and $D_1 \geq D_2$ then $\alpha_1^1 D_1 + \alpha_2^1 U_1 > \alpha_1^1 D_2 + \alpha_2^1 U_2$, so (D_1,U_1) is not an optimal solution of (α_1^1, α_2^1). Therefore $D_1 < D_2$ must be true. Intuitively, by adding TUs, it is possible to assign more jobs close to the quay side so the average travelling distance decreases. Therefore, D is monotonic non-increasing with U increasing.

Proposition 3: The number of optimal solutions (D,U) ***is finite.***

Proof: For the worst case, each module will assign with the maximum number of TUs M, and $S_{\max}$ represents the maximum number of module, so the upper bound of U can be $M \times S_{\max}$. Since the solution of U is finite, so optimal solution (D,U) is finite.

4.2.2. Approximation of TU Handling Capacity Function

The difficulty of the above model is the unknown TU handling capacity function in Constraint (4.10) which requires the understanding of the GRID system.

This study divides each GRID module into many sections with TUs moving freely between sections. Since TUs have to cross the front sections to access later sections, the number of activities in each section will affect the traffic in other sections. Therefore, we want to understand the relationship between the number of TUs, the TU handling capacity and the distribution of the container activities across different sections $\left(x^1, x^2 ... x^{ki} ... x^{S_k}\right)$. In particular, the handling capacity function can be represented as $y = f(m, x^1, x^2 ... x^{ki} ... x^{S_k})$, where m is the number of TUs, y is the TU handling capacity per shift and x^{ki} is the ratio of the containers that need to be handled in section ki (ki is the index of the section in the module), which can be expressed as

$$x^{ki} = \frac{\text{The container workload in section } ki}{\text{The container workload in module}}$$

With the above definitions, we can easily obtain the handling capacity in each specific section by calculating $y \cdot x^{ki}$. In particular, the summation of x^{ki} equals to one. For example, if there are 3 sections and 60 containers, including 30 containers in the first section and 30 in the second section, the ratio will be $x^1 = 50\%, x^2 = 50\%, x^3 = 0$.

Since the system has not been implemented, we cannot obtain a precise function. Therefore, we use simulation to find the handling capacity function of the GRID system under different scenarios in terms of the number of TUs and the distribution

of the container activities across different sections. The simulation model is modified from the one developed in Chapter 3 which focuses on the handling capacity of the module.

However, if we want to get the true function which is non-linear and unknown, we need an infinite number of simulation runs. To resolve this issue, we estimate this function by fitting a piecewise linear function with limited number of data points obtained from the simulation.

To demonstrate, we use our preceding example which has three sections in a module. With the given number of TUs, say m, we need to estimate the handling capacity function $y = f(x^1, x^2, x^3)$. Since x^1, x^2 and x^3 are dependent, the function can be represented as a surface as shown in Figure 4.3(a). To estimate this function and without loss of generality, we first select some (x^1, x^2, x^3) points such that they are equally spaced and then divide the domain space into smaller regions as shown in Figure 4.3(b). The number of points needed to represent the region equals to the number of sections, which is 3 in our example. With simulation runs, we can obtain the values of the handling capacity at these (x^1, x^2, x^3) points. Hence Figure 4.3(c) shows the estimated handling capacity function of the small region by using the piecewise linear interpolation. For each *m*, we can use the same procedure to divide the surface into smaller regions and estimate the function, as shown in Figure 4.3(d). In addition, the regions of each m are uniquely defined and indexed, so we can derive the value of m from the selected region.

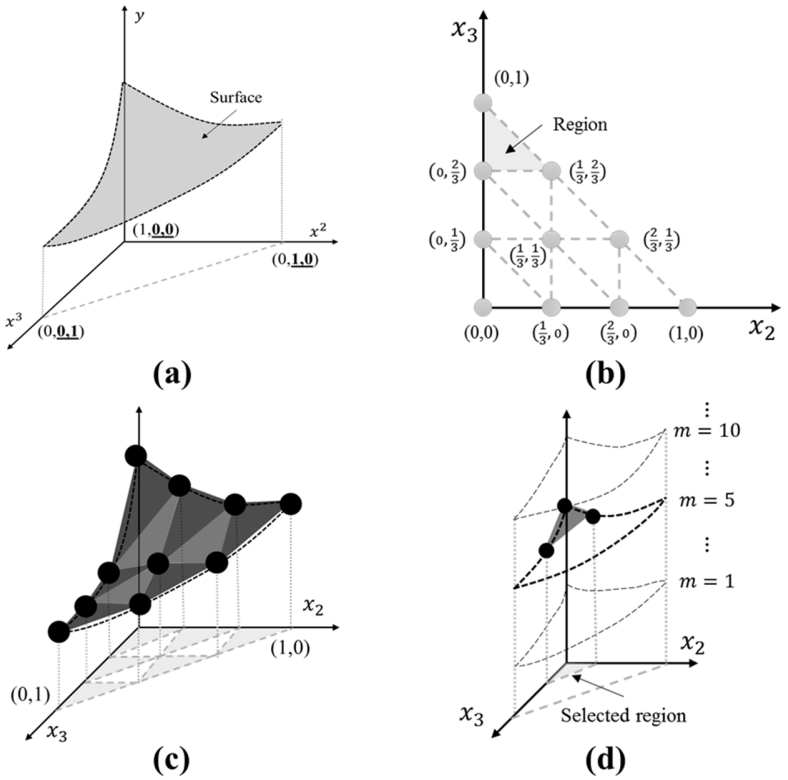


Figure 4.3 Example of linear combination in 3-section module

It should be noted that there is a trade-off between accuracy and computing complexity. Although the smaller the region, the more accurate the estimation will be for fitting the actual handling capacity function, it will also increase the time to find the optimal solution.

Finally, we can represent the TU handling capacity constraint as follows:

$$W_{ki}^{K} \le \sum_{\forall r} \sum_{j \in J_r} \delta_j y_j x_j^{ki}, \quad \forall ki \tag{4.14}$$

$$\sum_{j \in J_r} \delta_j = Z_r, \quad \forall r \tag{4.15}$$

$$\sum_{\forall r} Z_r = 1 \tag{4.16}$$

where W_{ki}^{K} represents the container workload in section ki, r is the index of the region, j is the index of the data points within each region, J_r is the set of data points of region r, Z_r is the indicator variable where $Z_r = 1$ if the region r is selected and δ_j is a real variable between 0 and 1 representing the weight associated to the extreme point j in region r. Constraint (4.14) represents that the container workload cannot exceed the TU handling capacity. Constraints (4.14) and (4.15) are for the linear interpolation of the handling capacity in region r. Constraint (4.16) represents that only one region can be chosen.

4.2.3. Revised Model

The original Constraint (4.10) can be replaced by the new constraints and the revised model can be written as follows:

(SAP-R) Objective:

$$\min SAP\text{-}R = \alpha_1 D + \alpha_2 U \tag{4.17}$$

Subject to:

Constraint (4.2) to Constraint (4.9).

$$\sum_{n \in \left\{ n | t_n^d = \mathrm{t} \vee t_n^l = \mathrm{t} \right\}} \left(W_{nk}^{20} + W_{nk}^{40} \right) \le \sum_{\forall r} \sum_{j \in J_r} \delta_{stj} y_j x_j^{I_1(s,k)}, \quad s \in S, t \in T, k \in K_s \tag{4.18}$$

$$\sum_{j \in J_r} \delta_{stj} = Z_{str}, \quad s \in S, t \in T, \forall r \tag{4.19}$$

$$\sum_{\forall r} Z_{str} = 1, \quad s \in S, t \in T \tag{4.20}$$

$$U_{st} = \sum_{\forall r} I_2(r) \cdot Z_{str}, \quad s \in S, t \in T, \tag{4.21}$$

Constraints (4.11) to (4.13).

δ_{stj}, $s \in S, t \in T, \forall j$, non-negative continuous variable (4.22)

Z_{str}, $s \in S, t \in T, \forall r$, binary variable (4.23)

In the above constraints, Constraints (4.18) – (4.21) are expanded from Constraints (4.15) – (4.16) by considering time and module. $I_1(s,k)$ is a mapping function which returns the index of section *k* in module *s*. For example, if section 6 is the 3rd section in module 2, then $I_1(2,6)=3$. Since the yard configuration is known, $I_1(s,k)$ will return a unique value. $I_2(r)$ is a mapping function which returns the number of TUs for region *r*. For example, if region 8 has 2 TUs, then $I_2(8)=2$. Since each region is uniquely indexed and known, $I_2(r)$ will return a unique value.

4.3. Information-based Allocation Strategy

Due to the difficulty in finding the optimal solution for *SAP-R* in large scale scenarios, the above model might not be practical when solutions need to be computed quickly. This is especially true at the operational level where the environment is dynamic and uncertain, which often results in changes to the plan on the actual day.

The common practice to allocate items in a quicker way is either to use a greedy or sequential based algorithm. Some of these allocation strategies can be found in the warehousing literatures, such as cube-per-order index (COI), duration-of-stay (DoS) and Popularity (ABC), and they have been demonstrated to be effective (Gu *et al.*, 2007; Roodbergen & Vis, 2009). In fact, the terminal allocation problem is similar to the warehouse allocation problem. Each storage module could be considered as

a storage slot and each quay crane could be considered as an input/output (I/O) point. Therefore, it is possible to adopt similar concepts in terminal systems.

However, there are some differences between the port system and the warehouse system. For the warehouse strategies, it is assumed that the I/O points are fixed on numbers and positions. But in the terminal system, any position along the quay side can be the I/O points for loading or discharging and so it is dynamic. Besides, most of the warehouse problems will not consider storage capacity but terminal problems have to include both storage capacity and handling capacity due to the traffic congestions in the GRID system. Because of the differences, the warehouse strategy cannot be directly used and there is a need to incorporate the factors mentioned above. By further by using the information on the optimal allocation of containers, we develop a new storage allocation strategy which is called the Information-based Allocation Strategy (IAS).

The overall framework of this section is as shown in Figure 4.4.

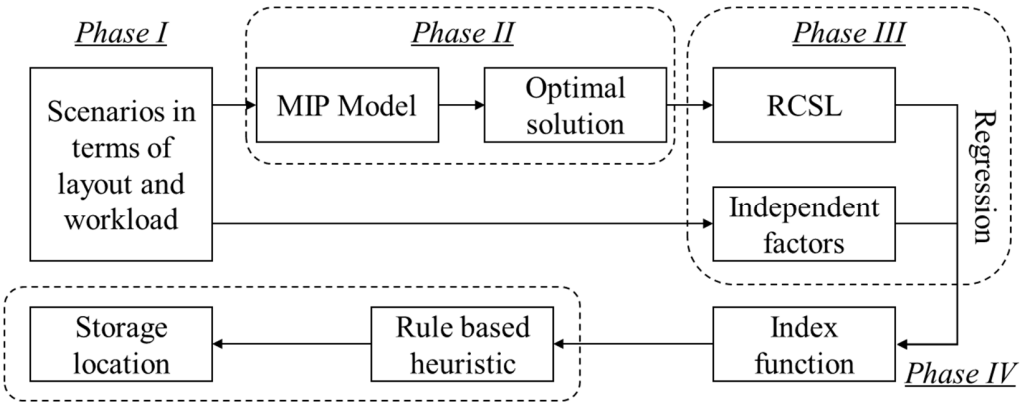


Figure 4.4 Overall framework of this section

4.3.1. Convenient Storage Location and Sorting Index

In warehouse problems, the common practice to allocate items is by using a greedy and sequential based algorithm. Basically, the algorithm will rank both the storage locations and the items, and then sequentially assign the items to the best location based on ranking.

To be specific, the ranking of the storage locations is named as the convenient storage location (CSL), which represents the preferred storage locations of items and is measured by the rectilinear distance between the storage location to the I/O point, or the exit point of the warehouse. The locations are ranked in an increasing order and the location with lower CSL value means that the location is closer to the I/O point. Usually, the warehouse has one or a few fixed I/O points and so the CSL to each item is fixed. On the other hand, the ranking of the items is named as the sorting index (SI) which is calculated based on a specific strategy such as DoS and COI. Usually the items are ranked in an increasing order and the items with lower SI value are assigned the higher ranking locations (lower CSL).

However, the previous algorithms and strategies for the warehouse are not suitable for the terminal system. Firstly, since any position along the quay side can be the I/O point (discharging point or loading point), the concept of the convenient locations has to be redefined; secondly, more than one factor, e.g. job size, job type and dwell time, may affect the SI and have to be evaluated.

In this case, the CSL is specifically redefined for the terminal system and therefore, the concept – relative convenient storage location (RCSL) - is proposed. RCSL relies on the discharging and loading positions $\left(p_n^d, p_n^l\right)$ and it could represent the

relative distance from $\left(p_n^d, p_n^l\right)$ to different sections. Figure 4.5 illustrates how the RCSL is determined. $RCSL = 1$ means that this location is closest to discharging and loading positions in rectilinear distance. A job with lower RCSL means it will occupy a better storage location and it has a higher priority to be handled. If more than two locations have the same RCSL, then it means that these locations are of no difference to the job in terms of distance, as shown in Figure 4.5 (right). If one job is split to multiple locations, then the weighted average is calculated for the RCSL. For example, if a job is split to two locations, with 20 containers stored in area A whose rank is 3 and 40 containers stored in area B whose rank is 1, then the RCSL for this job will be $\frac{20\times 3+40\times 1}{20+40}=1.67$.

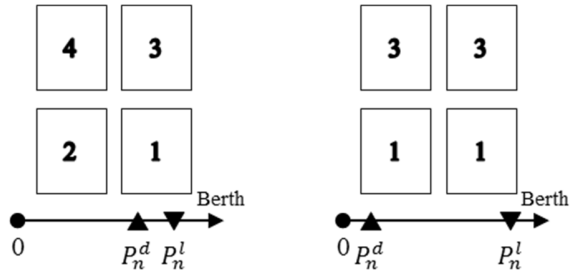


Figure 4.5 Definition of RCSL

There are many factors that may affect the job ranking. These include the individual factors such as $o_n, d_n, t_n^d, t_n^l, p_n^d, p_n^l, w_n^{20}, w_n^{40}$ and the combined factors such as the linear distance between OD positions $OD = \left|p_n^d - p_n^l\right|$ and duration of stay $DoS = t_n^l - t_n^d$ if n is a normal job or $DoS = T - t_n^l + t_n^d$ if n is a wrap-around job.

Since different jobs may have different RCSL values, the way of assigning jobs to the best location is also different. Basically the jobs are ranked in an increasing order and then the job with the lower SI value is firstly assigned to the location with

lower RCSL if the location is available in terms of both handling and storage capacities.

4.3.2. Framework for Developing IAS

To develop the IAS, the following phases are required, as described below:

Phase I: Configuration Design

In this phase, we will determine the scenario for experiment. The scenario has two aspects: layout configurations in terms of yard row number S_r, yard column number S_c and berth length, and the workload information. For the layout configuration, it should be noted that S_k – the number of sections in each module – has to be a fixed value since it determines the TU handling capacity function. If S_k changes, the characteristics of the function will be different which makes the original strategy inaccurate. The workload information is based on wrap-around job schedule and berth length.

The goal of this strategy is to be as general as possible since a lot of different scenarios are expected for further analysis.

Phase II: Modeling and Experiment

Based on the problem description, a MIP model is proposed for the problem. As mentioned in Section 4.1, the data or known information is directly used to formulate the model if it is available to describe the system. However, if the information is not available as in the case of the GRID system, simulation can be used to generate data for modeling.

In order to understand the optimal allocation decision, we need to solve the MIP model in as many different scenarios as possible. There are three groups of

scenarios, solvable, cannot be solved to optimality but feasible solution can be obtained, and infeasible. If the scenario is solvable, its optimal solution as well as the detailed allocation decision is collected for further analysis and these data will be used to derive the storage allocation strategy. If the scenario cannot be solved to optimality but feasible solution can be obtained, we will use the relaxed model to solve and the result of this lower bound solution will be used to compare the effectiveness of the proposed storage allocation strategy. If the scenario is infeasible, then it will be dropped.

Phase III: Analysis

The target of this phase is to use the collected data from solvable scenarios to determine the primary factor(s) that affect container allocation and obtain the index function which could be applied to different layouts and scenarios. Therefore, the RCSL is used to standardize the optimal storage locations between different terminal layouts. The input includes the individual factors such as $o_n, d_n, t_n^d, t_n^l, p_n^d, p_n^l, w_n^{20}, w_n^{40}$, and combined factors such as OD and DoS. Finally, we can form the following regression function to compute the values of each factor:

$$\ln\left(RCSL\right) = \sum_i \beta_i \cdot \ln\left(factor_i\right)$$

Phase IV: Implementation

Based on regression analysis, only significant factors are kept in the regression function. Therefore, the SI can be calculated from the regression function which can be written as below:

$$SI = \prod_i \left(factor_i\right)^{\beta_i}, \quad i \in \left\{i \mid factor_i \text{ is a primary factor}\right\}$$

Once the SI is obtained, the allocation procedures could start by assigning low SI jobs to the convenient locations, e.g. the locations that are close to the quay-side.

4.3.3. Allocation Procedures

In this section, two procedures are designed for the wrap-around job schedule.

Greedy Sequential Allocation (GSA)

Basically, GSA will sort all jobs first, and then sequentially and greedily assign the jobs to the available locations. In order to sort the jobs, the SI is used and calculated based on a specific strategy. It is obvious that the SI is the link between the allocation procedure and allocation strategies. As mentioned in the previous section, the key to the strategy is to create a mapping between the job ranking and the convenient locations. Since the jobs are assigned sequentially, the first assigned jobs will automatically occupy better locations. Therefore, it is defined that the lower SI value means that the job should be assigned first and the job set will be ranked in an increasing order. The details are described as follows.

---

1. Calculate SI for each job using IAS and sort the job set N by SI in increasing order;
2. Search for a storage location for each job:
   a) Select the section by comparing $\alpha_1 D + \alpha_2 U$. Since previous jobs have been fixed, the job will search the section that is close to the transfer area in each module.
   b) Repeat a) until a section (minimal value) is found; if no section is available, stop the whole process. It means that there is no space or not enough TU and there is no solution.
   c) Assign the job to the selected section;

---

3. Repeat Step 2 until all jobs are assigned.

The challenge of GSA is that it does not allow a job to be split into different areas. Intuitively, when storage capacity is more than enough, assigning a job to one location will not be an issue. However, when storage capacity becomes scarce, assigning the whole job into one location will usually lead to poor performance. In order to tackle this issue, we propose another framework – Batch Sequential Allocation.

Batch Sequential Allocation (BSA)

Similarly, BSA will sort all jobs first, and then sequentially assign the jobs to available locations. However, instead of greedy assignment, BSA uses a model to make better decisions. To be specific, based on the job ranking, a batch of the jobs is solved together at a time with the modified MIP model – *SAP-M*.

- ***The modified MIP model – SAP-M***

Additional Parameters

B	Batch size;
N_s	The set of selected jobs, index n, $1 \le n \le B$;
N_f	The set of fixed jobs, index n_h, $1 \le n_h \le \frac{B}{2}$;
f^s	Sum of flow of fixed jobs;

w^s Sum of containers of fixed jobs;

w_{kt}^0 Storage capacity of fixed jobs in section k in shift t;

f_{kt}^0 Flow of fixed jobs in section k in shift t;

(SAP-M) Objective:

$$\min SAP-M = \alpha_1 D + \alpha_2 U \tag{4.24}$$

Subject to:

Constraints (4.3), (4.7) to (4.9) and (4.19) to (4.21).

$$D = \left(\frac{\sum_{n \in N_s} \sum_{k=1}^{K} (l_{o_n k} + l_{d_n k})(W_{nk}^{20} + W_{nk}^{40}) + f^s}{\sum_{n \in N_s} (w_n^{20} + w_n^{40}) + w^s} \right) \tag{4.25}$$

$$\sum_{k=1}^{K} W_{nk}^{20} = w_n^{20}, \quad n \in N_s \tag{4.26}$$

$$\sum_{k=1}^{K} W_{nk}^{40} = w_n^{40}, \quad n \in N_s \tag{4.27}$$

$$w_{kt}^0 + C_{kt} + \sum_{n \in \left\{ n | t_n^d = t \right\}} \left(\frac{W_{nk}^{20}}{2} + W_{nk}^{40} \right) \leq \beta \bullet c, \quad k \in K, t \in T \tag{4.28}$$

$$f_{kt}^0 + \sum_{n \in \left\{ n | t_n^d = \mathrm{t} \vee t_n^l = \mathrm{t} \right\}} \left(W_{nk}^{20} + W_{nk}^{40} \right) \leq \sum_{\forall r} \sum_{j \in J_r} \delta_{stj} y_j x_j^{I_1(s,k)}, \quad s \in S, t \in T, k \in K_s \tag{4.29}$$

Since not all jobs are involved at the same time, the MIP model could be solved in a short amount of computation time. Based on the solution, the allocation of the half batch is fixed and used to update the storage status from time to time. This idea can greatly reduce computation complexity and allow job splitting into different locations for storage if necessary. The whole procedure is described below.

---

1. Calculate SI for each job and sort the job set N by SI in increasing order;
2. Move the first *B* (batch size) jobs into the set N_s; if remaining jobs are less than *B*, then move all jobs and mark this iteration as the last iteration;
3. Solve job set N_s with *SAP-M* and CPLEX; if this is the first iteration, then $f^s = 0$, $w^s = 0$ and $f_{kt}^0 = 0$ for all *k* and *t*; if infeasible, stop the whole process.
4. Move the first half jobs from N_s to N_f and move the rest of the jobs to the beginning of the set N_k; if this is the last iteration, move all jobs from N_s to N_f.
5. Update f^s and w^s using

$$f^s = \sum_{n \in N_f} \sum_{k=1}^{K} (l_{o_n k} + l_{d_n k})(W_{nk}^{20} + W_{nk}^{40})$$

$$w^s = \sum_{n \in N_f} (w_n^{20} + w_n^{40})$$

Update p_{kt} and f_{kt} using

$$w_{kt}^0 = \sum_{n \in \{n | n \in N_f, t \in \{\text{job } n \text{ staying time}\}\}} \left(\frac{W_{nk}^{20}}{2} + W_{nk}^{40} \right)$$

$$f_{kt}^0 = \sum_{n \in \{n | n \in N_f, t_n^d = t \vee t_n^l = t\}} \left(W_{nk}^{20} + W_{nk}^{40} \right)$$

6. Repeat steps 2 – 5 until the job set is empty.

4.4. Numerical Experiment

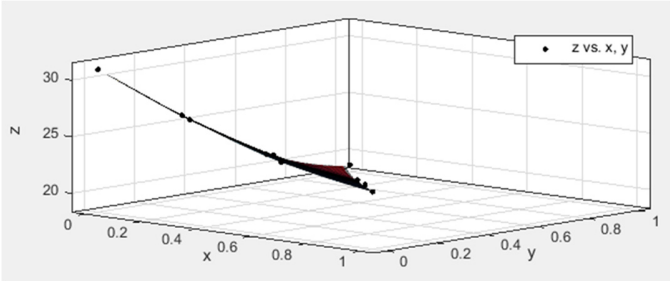
4.4.1. Experiment Settings

Firstly, we will focus on developing the IAS by solving the MIP model in a series of scenarios. Then using the allocation frameworks, the performance of the algorithms will be compared. Both the MIP model and allocation procedures are implemented by Visual Studio 2015, C# on a PC (Intel i5, 3.30GHz with 8GB memory) and the solver of the model is CPLEX 12.6.1.

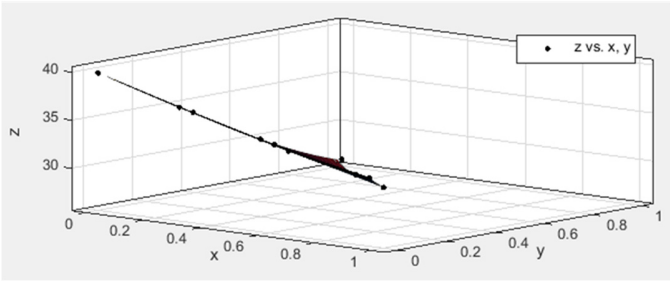
The scenarios and parameters of the numerical study are listed below:

- The planning horizon in this study is one week (7 days). The time unit is "shift" and each day has 3 8-hour shifts. Hence, each week has 21 shifts and $T_{\max} = 21$.
- Each section is 136 meters wide, 52 meters deep. The stacking height is 5 containers so the storage capacity could reach 2,200 TEUs. The transfer area is 12 meters with 3 horizontal dual paths. We fix $S_k = 3$ for each module so the depth is 168 meters and the storage capacity reaches 6,600 TEUs. The space utilization is set to be 0.8. The example of the simulation result of $S_k = 3$ layout is as follows. In this study, we use the layout with three sections as shown in Figure 4.1. According to Section 4.2.2, for each *m*, we divide the domain space into 9 regions with 10 data points, as shown in Figure 4.3(c). Since the maximum number of TUs is 13, there are eventually 117 regions. To obtain the data points, we run the simulation with 10 cases of the distribution of the container activities, including $(1,0,0)$, $(0,1,0)$, $(0,0,1)$, $\left(0,\frac{1}{3},\frac{2}{3}\right)$, $\left(\frac{1}{3},0,\frac{2}{3}\right)$, $\left(\frac{1}{3},\frac{2}{3},0\right)$, $\left(\frac{2}{3},\frac{1}{3},0\right)$, $\left(\frac{2}{3},0,\frac{1}{3}\right)$, $\left(0,\frac{2}{3},\frac{1}{3}\right)$ and $\left(\frac{1}{3},\frac{1}{3},\frac{1}{3}\right)$, and the number of

TUs varies from 1 to 13, and thus we have a total of 130 data points. Figure 4.6 shows the simulation result of the TU handling capacity function (per hour per I/O point) with respect to the container assignment plan when the TU number is 4 and 6. It should be noted that since the data in the figure is using per hour per I/O point, the value used in the constraints should involve multiplying by 16 (8 hour-shift and 2 I/O points)



GRID in 3*Depth layout, TU = 4



GRID in 3*Depth layout, TU = 6

Figure 4.6 TU handling capacity chart by the distribution of the container activities

- 10 scenarios are proposed in Table 4.1. Scenarios 1 to 8 are solvable cases, while Scenarios 9 and 10 are regarded as insolvable but feasible cases if they have feasible solution(s) but cannot be solved to optimality by the solver within 4 hours. In addition, the scenarios for IAS analysis should have a heavy container flow, otherwise there are less competitions for the space and resources.

Table 4.1 Numerical experiment scenarios

Scenario	S_r	S_c	S_k	***# of 20-foot containers***	***# of 40-foot containers***	***Optimal in 4 hours?***
1	1	5	3	6,900	16,100	Yes
2	1	8	3	7,700	17,700	Yes
3	1	10	3	12,700	28,600	Yes
4	2	3	3	8,400	18,000	Yes
5	2	3	3	8,500	16,800	Yes
6	2	4	3	10,800	23,000	Yes
7	3	2	3	8,200	17,700	Yes
8	3	3	3	10,600	23,200	Yes
9	2	8	3	17,700	37,400	No
10	3	5	3	18,700	42,500	No

Secondly, the performance of the new strategy will be compared with the traditional strategies, such as COI, DoS and Purely Random (PRD) as described below

1) COI: the COI of a job is defined as the ratio of the job's total required space to the number of trips required to satisfy its demand per period. In the GRID system, a 20-foot container only occupies half unit of the storage capacity and one TU can only pick up one container regardless of its type. Therefore, for each job, the SI is defined as

$$SI = \frac{w_n \times \Gamma_n}{\left(\frac{w_n}{DoS_n}\right)} = DoS_n \times \Gamma_n$$

where w_n is the job volume, which either equals to w_n^{20} or w_n^{40} since each job can only have one type of container; DoS_n is defined as $DoS_n = t_n^l - t_n^d$ if n is a normal job or $DoS_n = T_{\max} - t_n^l + t_n^d$ if n is a wrap-around job; Γ_n is defined as job type and $\Gamma_n = 1$ is for 20-foot containers and $\Gamma_n = 2$ is for 40-foot containers.

2) DoS: the duration-of-stay of a job is defined as the dwell time between the job's arrival and departure. Typically, shorter DoS jobs are preferred to be placed closer to the exit point to achieve a higher turnover rate. Therefore, for each job, the SI is defined as

$$SI = DoS_n$$

It should be noted that $DoS_n = t_n^l - t_n^d$ if n is a normal job and $DoS_n = T_{\max} - t_n^l + t_n^d$ if n is a wrap-around job.

3) PRD: in purely random strategy, jobs are randomly sorted and then assigned to the available space. Therefore, for each job, the SI is defined as

$$SI = random\ value .$$

Since these are greedy based strategies, GSA will be applied. Based on the different procedures and strategies, we can use the following algorithms to solve the allocation problems.

Table 4.2 List of algorithms

Algorithm	***Procedure***	***Strategy***
IAS-GSA	GSA	IAS
IAS-BSA	BSA	IAS
COI-GSA	GSA	COI
DoS-GSA	GSA	DoS
PRD-GSA	GSA	PRD

Thirdly, the algorithms will be validated in the operational level problem (OLP). In previous sections, it is assumed that the job schedule is deterministic, which means that the arrival and departure time of each job is known in advance and the allocation decision can be optimized. Although it is reasonable for planning purpose, the assumption might not be valid in actual operations. Usually the terminal will receive a discharging and loading scheme hours or days ahead before a vessel's arrival, and then the operator will reserve the space and the resources based on the scheme. Therefore, in the OLP, we will study the effectiveness of the storage allocation strategy for short term rolling horizon by comparing it with the optimal solution or lower bound solution when we have the whole 21 shift information. The details of the OLP are described as below.

Additional Parameters

N^t Set of the jobs which arrive at shift t, $t \in \{1,2,3...T_{\max}\}$

A	Number of shifts that the operator will know the scheme in advance; for example, if current shift is 3, then $N^3,...,N^{3+A}$ will be considered as known.

---

1. Pre-processing of input data – the job set N;
   a) Solve the relaxed model;
   b) Fix the allocation of the wrap-around jobs and remove those jobs from corresponding job set N^t;
   c) Initialize the yard storage state in each shift based on fixed jobs;
2. Job allocation in shift *t*;
   a) Set $N = \{N^t,...,N^{t+A}\}$;
   b) Apply GSA or BSA to the set N;
   c) Fix the allocation of the jobs that belongs to N^t;
   d) Update the yard storage state in each shift based on fixed jobs;
   e) Set $t = t+1$;
3. Repeat Step 2 until the end of planning horizon.

4.4.2. Model Solution

In this section, we will solve the model in Scenario 4 with CPLEX. As the objective function can be written as $\alpha_1 D + \alpha_2 U$ where *D* is a continuous variable and *U* is an integer variable, the solutions of the MIP model can be represented as in Figure 4.7 with the x-axis representing *U* and y-axis representing *D*.

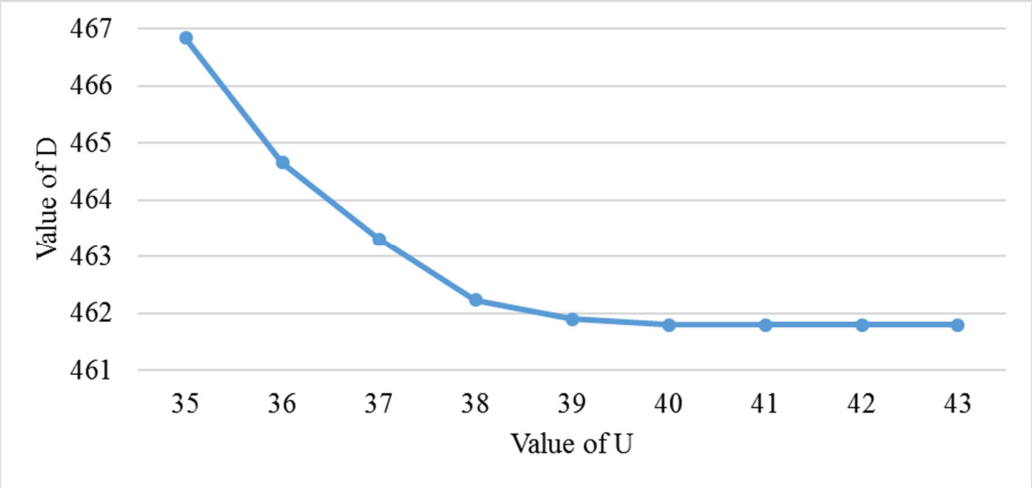


Figure 4.7 Solution space of the MIP model

If the number of TUs is less than 34, the model is infeasible as there is not enough TUs. However, if the TU number is larger than 40, the average distance will not decrease any more. This result verifies two propositions, i.e., the number of optimal solutions is finite and D is monotonic non-increasing with U increasing. Besides, it also verifies that each (α_1,α_2) pair will correspond to a unique (D,U) but each (D,U) will correspond to a range of (α_1,α_2). For example, our results have shown that the (α_1,α_2) pairs (2000,1), (200,1) (50,1) have the same result (461.801,40) while (1,20) and (1,100) share the result (466.837,35).

4.4.3. Developing IAS

Based on the framework, regression analysis was conducted on different scenarios with different (α_1,α_2) pairs. The four independent variables chosen for the regression are the number of 20-foot containers of a job (w_n^{20}), the number of 40-foot containers of a job (w_n^{40}), duration of stay (*DoS*), and linear berth distance between OD positions (*OD*). The regression function can be written as:

$$\ln\left(RCSL\right) = a \cdot \ln\left(w_n^{20}\right) + b \cdot \ln\left(w_n^{40}\right) + c \cdot \ln\left(DoS\right) + d \cdot \ln\left(OD\right)$$

The SI function can then be written as:

$$SI = \left(w_n^{20}\right)^a \left(w_n^{40}\right)^b \left(DoS\right)^c \left(OD\right)^d$$

Some discussions on the specific values of the four coefficients are as follows. Firstly, we want to figure out if we can find a specific set of coefficients for each scenario regardless of the value of $\left(\alpha_1, \alpha_2\right)$ or $\left(D, U\right)$. To illustrate, Scenario 4 is taken as an example. As discussed in the previous section, there is a finite number of meaningful optimal solutions of Scenario 4 and regression is then conducted on each solution. Using the regression function, Table 4.3 can be obtained.

Table 4.3 Coefficient comparison for different (D, U)

(D,U)	***a***	***b***	***c***	***d***	***R Square***
(466.837,35)	-0.09	0.11	0.77	0.09	0.45
(464.651,36)	-0.08	0.13	0.76	0.10	0.43
(463.312,37)	-0.09	0.12	0.78	0.07	0.45
(462.236,38)	-0.07	0.14	0.75	0.06	0.46
(461.898,39)	-0.07	0.14	0.72	0.08	0.44
(461.801,40)	-0.08	0.13	0.73	0.05	0.47
Average	-0.08	0.13	0.75	0.08	
Stand deviation	0.009	0.012	0.023	0.018	

It can be seen that the average value of coefficients can be applied to the IAS for this scenario directly since the standard deviation is small. The result can be explained as below:

(1) A job with more 20-foot containers should be placed in preferred locations as smaller storage space is required;

(2) A job with less 40-foot containers should be placed in preferred locations as more storage space is required;

(3) A job with small duration-of-stay time should be placed in preferred locations as the space occupied by the job can be released faster for the next job;

(4) A job with small OD distance should be placed in preferred locations. The reason is that a job with large OD distance has more preferred locations compared to the job with small OD distance.

Secondly, we want to know if we can have a set of robust coefficients which can be applied to different scenarios. In the experiment, two (D,U) cases of each scenario are selected and Table 4.4 can be obtained.

It can be seen that the standard deviations of the above coefficients are small and the R square varies within an acceptable range. As shown in Table 4.4, the R square value is not that high since some dynamic factors such as TU number and module storage volume are not being captured. However, since the coefficient deviation is small and we are performing ordinal ranking, using the coefficient values from the mixed data set should give a sufficiently good estimation. In fact, later experiments show that the result of the new strategy is close to optimality.

Table 4.4 Coefficient comparison for different scenarios

Case	*Scenario*	*TU*	*a*	*b*	*c*	*d*	*R Square*
1	1	52	-0.12	0.11	0.51	0.12	0.47
2	1	50	-0.13	0.09	0.54	0.13	0.41
3	2	46	-0.20	0.05	0.64	0.06	0.51
4	2	32	-0.17	0.09	0.66	0.07	0.47
5	3	66	-0.17	0.06	0.54	0.11	0.39
6	3	58	-0.17	0.05	0.52	0.11	0.36
7	4	40	-0.08	0.13	0.73	0.05	0.47
8	4	35	-0.09	0.11	0.77	0.09	0.45
9	5	49	-0.12	0.10	0.65	0.07	0.36
10	5	47	-0.12	0.10	0.67	0.04	0.35
11	6	62	-0.10	0.15	0.65	0.06	0.38
12	6	54	-0.13	0.12	0.73	0.03	0.41
13	7	51	-0.21	0.07	0.68	0.05	0.40
14	7	47	-0.14	0.13	0.66	0.09	0.38
15	8	63	-0.14	0.10	0.92	0.05	0.43
16	8	48	-0.12	0.11	0.90	0.05	0.40
Average			-0.14	0.10	0.67	0.07	
Stand deviation			0.04	0.03	0.12	0.03	

4.4.4. Algorithm Comparison

In this section, we firstly compare the performance of different algorithms listed in Table 4.2 under Scenario 4. It should be noted that since a larger batch size will increase the solving time in each iteration but have less iterations, without loss of generality, we set the batch size to be 50 for BSA in the following experiments. The experiment cases are listed in Table 4.5.

Table 4.5 Instances for algorithm comparison

Case	(α_1, α_2)	***Corresponding*** (D, U)
1	(200,1)	(461.801,40)
2	(50, 1)	(461.801,40)
3	(10, 1)	(461.898,39)
4	(1, 1)	(463.312,37)
5	(1, 20)	(466.837,35)
6	(1,100)	(466.837,35)

The following indicators are used for comparison:

GAP on D = (D by Algorithm – D by CPLEX) / D by CPLEX

Difference of U = U by Algorithm – U by CPLEX

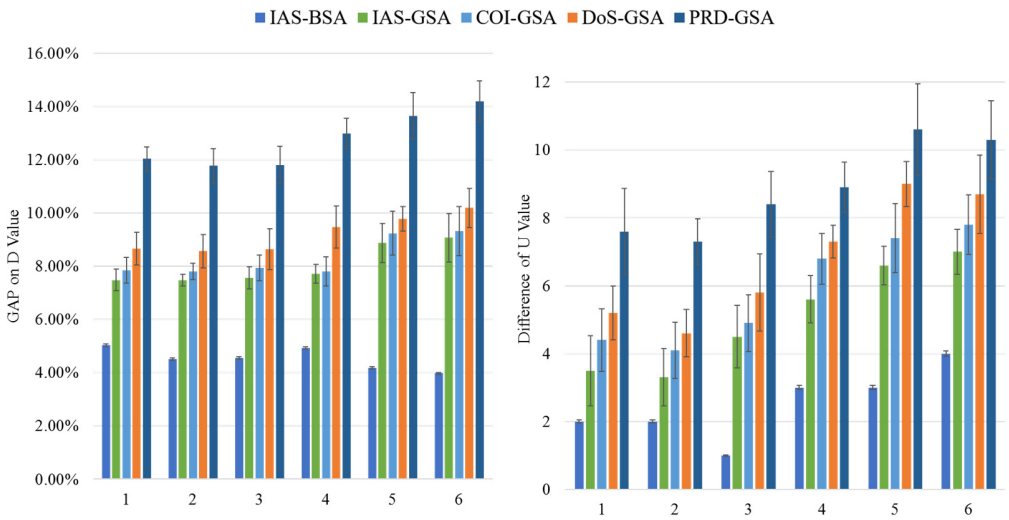


Figure 4.8 Comparison between algorithms in different cases of a solvable scenario

The comparison between different algorithms is shown in Figure 4.8 and the sequence of the bars for each case follows the same order. Basically, IAS-BSA is 3%-5% and 2-4 TUs better than IAS-GSA. It is because the BSA allows splitting of jobs so the space can be highly utilized. Comparing with different strategies, IAS is better than COI, COI is better than DoS and DoS is far better than purely random. It means that the new strategy which uses additional information can provide a very good allocation plan.

Secondly, the algorithm performance under Scenarios 4, 9 and 10 is examined. For each scenario, the (α_1, α_2) pair is set to (1, 1). Since Scenarios 9 and 10 cannot be solved within a certain time period, the relaxed model will be solved instead of the original model and the following comparing indicators are used:

$$GAP\ on\ D = (D\ by\ Algorithm - D\ by\ Relax) / D\ by\ Relax$$

$$Difference\ of\ U = U\ by\ Algorithm - U\ by\ Relax$$

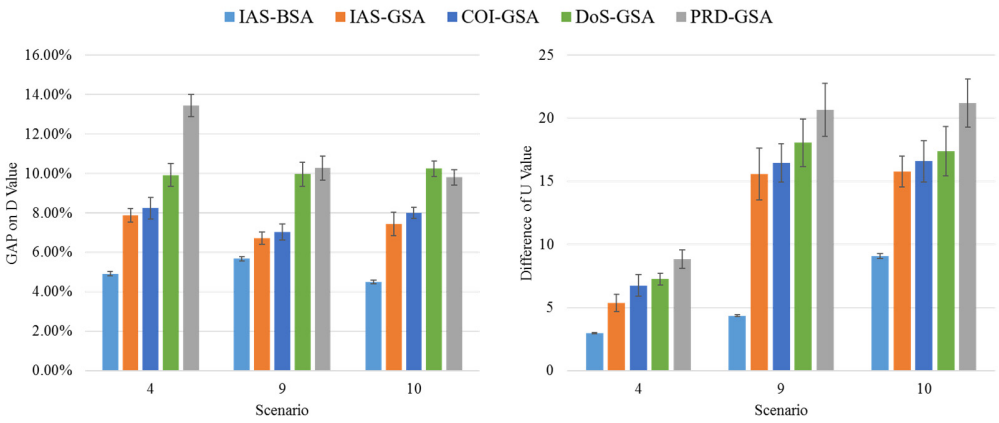


Figure 4.9 Comparison between algorithms in unsolvable scenarios

Apparently, Figure 4.9 shows similar results as Figure 4.8 and IAS-BSA is the best while PRD-GSA is the worst. It should be noted that with the increasing storage capacity and total container volume, the limitation of GSA becomes obvious. Since the job cannot be split, much TU handling capacity is wasted and the required number of TUs increases rapidly.

Lastly, the effectiveness of the proposed strategy is tested under operational level which has short term rolling horizon. The rolling horizon is set to be 4 shifts, which means that the operator knows the information of the jobs that arrive in the current shift and next 3 shifts. The experiment uses Scenario 4 and sets the (α_1, α_2) pair to (1, 1). Since the initial yard states are based on the relaxed model of *SAP-R*, the following indicators are used in this experiment.

GAP on D = (D by Algorithm – D by Relax) / D by Relax

Difference of U = U by Algorithm – U by Relax

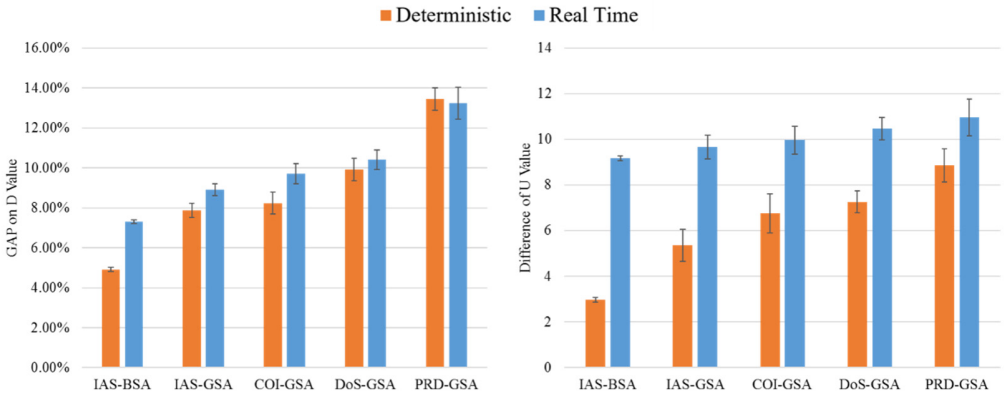


Figure 4.10 Comparison between algorithms in deterministic and real-time problems

Figure 4.10 compares the algorithms in deterministic and operational level problems. It is obvious that the algorithms generally perform better in the deterministic problem as all job information is known in advance, and TU usage is particularly high in the real-time problem. Similar to the deterministic problem, IAS-BSA is the best in the real-time problem and PRD-GSA is the worst. It indicates that both the new strategy and allocation procedures are able to perform sufficiently well under operational level.

4.5. Conclusion

It is known that the allocation strategy will significantly affect system performance in both the terminal system and warehouse system. Due to the unique features of the GRID based terminal system, previous strategies may not be suitable and so a new allocation strategy is needed.

In this study, we are interested in developing a good container allocation strategy which will affect the total cost. To describe the problem, a MIP model was formulated and simulation was used to find the TU handling capacity function of the GRID system under different scenarios in terms of the number of TUs and the distribution of the container activities across different sections. With limited

number of simulation runs, this function was estimated by using fitting a piecewise linear function.

However, the above model was not practical considering the computation time in large scale scenarios and this was especially true at the operational level with dynamic and uncertain environment. Therefore, we developed an information-based allocation strategy and used greedy and sequential based algorithms to allocate containers. The strategy required the understanding of the optimal container allocation of the new system and so solutions of the solvable terminal sizes were collected for analysis. In the analysis, the concept of convenient storage locations was redefined for the terminal system and a regression model was proposed to analyze the collected data. It was found that four primary factors would strongly influence allocation, which were the number of 20-foot containers of a job (w_n^{20}), the number of 40-foot containers of a job (w_n^{40}), duration of stay (*DoS*) and OD positions (*OD*). The results showed that the strategy was independent of the yard layout, workload and cost configuration (i.e. the values of α_1 and α_2) and so it could be applied to many different scenarios. Compared with traditional strategies such as COI, DoS and purely random allocation, the new strategy performed better in average container travel distance *(D)* and the total number of TUs *(U)*. Besides, the allocation procedure BSA could further improve the performance of the new strategy. In addition, this strategy could also be used for operational decisions where the information on arriving containers is available for only a few shifts ahead. This study thus provided a novel approach to develop good storage allocation strategies for new terminal concepts. Due to the system complexity, simulation and

approximation methods were suggested to obtain empirical formulations to describe the system and convert complex formulations to simple ones which can be computed quickly. In addition, the study showed that how the information could help to improve the efficiency of traditional industries such as the allocation problem in container terminals. It indicated that more information-based or data-driven studies could be expected in the future.

Chapter 5. A Heat-map Based Algorithm for Path Direction Design Problem in GRID System

Chapter 3 has highlighted the primary challenge of the GRID system, that is to prevent and solve the conflicts in real time. As a solution, the free routing traffic rule (FRTR) is proposed. FRTR allows TUs to travel by the shortest path and reroutes the TUs whenever a conflict occurs. The simulation experiment shows that due to the conflicts of the opposite movement, the throughput drops when the number of TUs increases.

Motivated by the city traffic design which sets single travel direction on specific roads and creates loops to make sure that all vehicles are travelling in the same direction on some roads, we propose the single direction traffic rule (SDTR) for the GRID system. The concept is to set proper flow direction on each path so that all vehicles on the same path will travel in the same direction and eliminate the opposite conflicts. When two TUs meet at the crossings, the succeeding TUs will wait for the preceding TUs to pass instead of detouring, which may cause more conflicts. It is trivial that the success of the SDTR depends on finding a good path flow direction design over the whole system (simply referred to as path design). Besides, once the path design is determined, it does not mean that it will be the one used all the time. Since the flow on the system can be very different from time to time, the path design will need to be updated accordingly. Therefore, the objective of this study is to efficiently determine the path design in the large scale system and the dynamic environment.

Similar problems have been raised in factory literatures such as the unidirectional AGV path direction design problem (Kaspi & Tanchoco, 1990) and tandem AGV path design problem (Bozer & Srinivasan, 1991). These studies treated the situation as a path-job problem by formulating mixed integer programming (MIP) models to optimize the AGV flow direction for the given jobs which are uniquely represented by origin node, destination node, and volume. Due to the computational complexity of solving the mathematical programming problem, different heuristic algorithms were proposed. For example, Seo *et al.* (2008) proposed an evolutionary computational approach combining genetic algorithm and tabu search methods to tackle path design problems with around 20 nodes and 30 jobs; and Guan *et al.* (2011) proposed and tested a revised electromagnetism-like mechanism heuristic in the reconfigurable manufacturing system context with at most 100 nodes and 380 jobs. In the previous studies, due to the limited number of workstations or processes in the factories, the scale of the related AGV problem is not that large. Since we target to solve the problem with much more nodes and jobs in a short period of computation time, the above mentioned heuristic algorithms cannot work. Besides, the path/flow direction will be implemented between workstations for a very long time without changing.

The concept of heat-map is well known for representing temperatures in an area in meteorology. It was initially used in 2 dimensional displays of the values in a data matrix where larger values were represented by small dark squares and smaller values by lighter squares. Later, it is widely used in many different areas, such as molecular biology, meteorology, and computer science, etc. However, the heat-map

is still used to illustrate or display values, and has no big difference from the original usage. Some maritime studies implement the heat-map as a way of data display as well. Boer and Saanen (2012) developed a terminal operating system integrated with simulation. The usage frequency-based heat-map was used to represent the bottlenecks inside the terminal. Vatin and Napoli (2013) used the heat-map to show the major traffic zones with high density of vessels in the Mediterranean Sea. In general transportation systems, the heat-map is also commonly used to visualize the statistical data, such as the crowded areas in a city by the number of taxis for a given time period (Liu *et al.*, 2011). In our study, we need to determine the path direction wisely since each path is shared among many jobs. Intuitively, we would like to set the direction which can serve the most number of TUs. Thus, we use the heat-map concept to represent the traffic flow on each path. By converting a path-job problem into a path-flow problem, there is no need to solve a very complex mathematical model and the number of jobs can be very flexible.

As a continuously running system, it will have three main challenges if the path design is to be updated dynamically. Firstly, the trigger or the timing of updating path design has to be set properly. If the update is executed very frequently, the proposed algorithm has to be very efficient, and it will need a lot of computing resources. Besides, the effectiveness may not be significant if the number of jobs is very small. Secondly, the actions in the transition period have to be determined, since the system always has working and scheduled jobs in progress. Thirdly, the routing policy has to be determined. To deal with these challenges, we propose a dynamic framework.

This study provides a novel approach to solve the path design problem. In the first step, the heat-map algorithm (HA) is proposed to convert the given jobs into a flow-based heat-map. Then, the heat-map-based simulation embedded algorithm (HSEA) is proposed to search and evaluate the path design iteratively. Finally, the HSEA is applied in the framework to update the path design dynamically.

The remaining sections are organized as follows: Section 5.1 defines the path design problem based on the GRID background; Section 5.2 proposes novel approaches – HA and HSEA – to solve the large scale path design problem; and then the dynamic framework is introduced in Section 5.3. Numerical results and related analysis are provided in Section 5.4, while conclusions are presented in Section 5.5.

5.1. Problem Description

5.1.1. Definitions

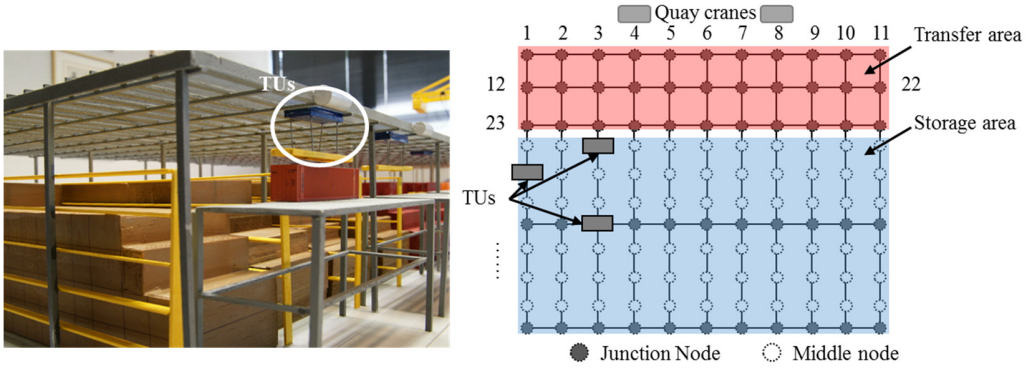


Figure 5.1 An example of GRID structure

The target of this study is to determine the path direction in the GRID system. Thus, we need to convert the physical structure shown in Figure 5.1 (a) into a node-path graph which is a typical mesh-like structure, as shown in Figure 5.1 (b).

Node definition. In this graph, the nodes can be categorized into two types – junction nodes, where TUs can turn to another direction, and middle nodes, where TUs pass by or handle containers. Each node is uniquely indexed. In order to be consistent with previous work by Zhou *et al.* (2015), we also assume that the TUs will only pick-up and drop containers at certain nodes, which are defined as access points (APs), for example, *Node 4* and *Node 8*.

Path and directed path. Figure 5.2 (a) shows that the path is defined as the line segment between two adjacent nodes. Since the direction of the path is our focus, the path graph can be converted into the directed path graph as shown in Figure 5.2 (b). Figure 5.3 (a) shows that since each path can at most have one direction at a time, only one directed path (1 or 2) can be activated. According to the start node and end node, each directed path can be uniquely indexed by following the sequence of vertical first then horizontal, from left to right, and from up to down. Each directed path is indexed by k and each path has a distance L_k.

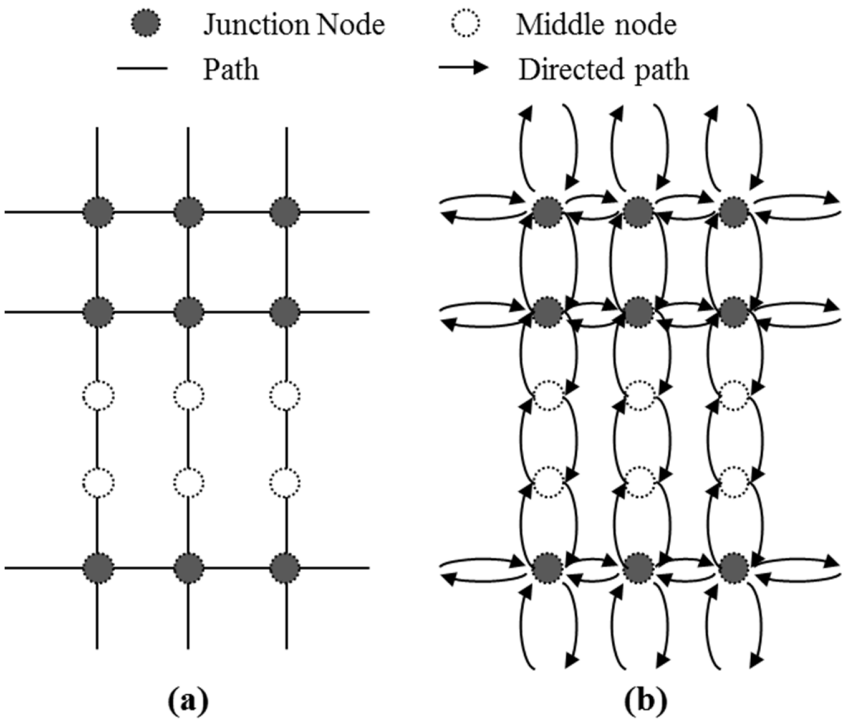


Figure 5.2 Path and directed path graph

As introduced in Chapter 3, a TU traveling on the horizontal path will not interfere the TUs on its adjacent horizontal paths. But due to the use of single rails along the vertical paths, TUs cannot travel in parallel along adjacent vertical paths in opposite direction. Therefore, if one direction of the path is chosen, then its adjacent path cannot have an opposite direction. Figure 5.3 (b) shows that if directed path 1 is chosen, then directed path 4 cannot be chosen. For convenience, we define such adjacent conflict pair $pa_m \equiv \left(k_m^1, k_m^2\right)$ where m is the index of the pair, and k_m^1, k_m^2 are two directed paths. For example, $pa_1 = \left(1,4\right)$, $pa_2 = \left(2,3\right)$. Since the graph is given, by looking for all the adjacent conflict pairs in the directed graph from left to right, and from up to down, they can be uniquely indexed.

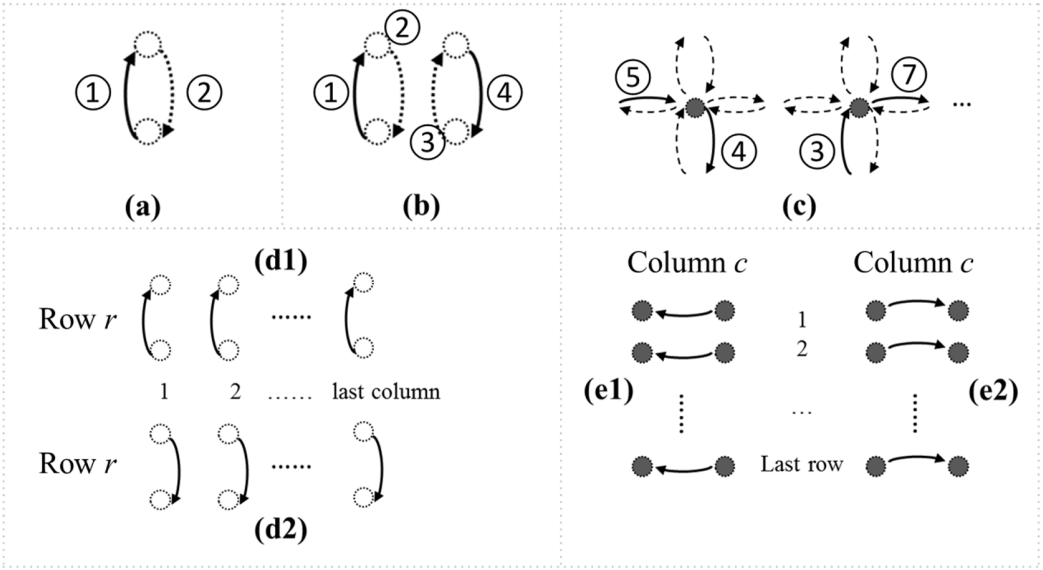


Figure 5.3 Definitions of directed path pair or group

In the GRID system, since the TU turning takes five seconds which is significant compared with its speed (1 meter per second for loaded TU and 2 meters per second for empty TU), we do not want to have a lot of turns on a job's route. Therefore, we take the turning of the TU into account in this study and we assume that one turn is equivalent to turning distance T. From a modeling perspective, two directed paths, which rotate the TU from vertical to horizontal or in the reverse way, form one turn, as presented in Figure 5.3 (c). For example, directed paths 5 and 4, directed paths 3 and 7 form two turns respectively. For convenience, we define such turning pair $pt_h \equiv \left(k_h^1, k_h^2\right)$ as a unique pattern of the turn in the directed graph where h is the index of the pair, k_h^1 and k_h^2 are the directed paths, for example, $pt_1 = (5,4)$, $pt_2 = (3,7)$. Since the graph is given, by looking for all the turning pairs in the directed graph from left to right, and from up to down, they can be uniquely indexed.

To improve the connectivity of the whole directed graph, we need to guarantee that each direction has at least one path going in that direction. Therefore, we define the path group as shown in Figure 5.3 (d1, d2, e1, e2). (d1) is the up-going path group in row *r* and (d2) is the down-going path group in row *r*. They are notated as G_r^U and G_r^D respectively. Similarly, (e1) G_c^L and (e2) G_c^R are the left-going and right-going path groups in column *c,* respectively.

Job definition. A job is defined as a single TU movement from its origin node (*O*) to its destination node (*D*). So each job *j* is uniquely denoted as $\left(no_j, nd_j\right)$; e.g. *(4, 120)* means the job is moving from *Node 4* to *Node 120*. It can either be a discharging task, which delivers a container from the quay crane to a node in the storage area, a loading task, which delivers a container from a node in the storage area to the quay crane, or an empty move, which has no container onboard. The shortest distance of job j is notated as S_j and is defined as the Manhattan distance between $\left(no_j, nd_j\right)$ with one turning distance T .

5.1.2. Modeling

The objective of this study is to find a good path design which can minimize the total job travel distance. If we do not have a path design, then we cannot know the exact travel route for each job, and on the other hand, if we do not know the route, then we cannot decide the direction on each path. To achieve the objective, a typical path design problem can be formulated by solving for the path design and the job travel route simultaneously. Based on the above definitions, we formulate the Path Direction Problem (PDP) as below.

Path Direction Problem (PDP)

Parameters:

N The set of nodes.

P The set of the directed paths. Each directed path k is uniquely represented by $\left(ns_k, ne_k\right)$ which specifically goes from node $ns_k \in N$ to node $ne_k \in N$. The opposite of directed path k is the directed path k^-.

L_k The distance of path k, positive value.

PS_n The set of directed paths that start from node $n \in N$.

PE_n The set of directed paths that end at node $n \in N$.

pa_m A pair of adjacent directed paths. $pa_m \equiv \left(k_m^1, k_m^2\right)$ where k_m^1 and k_m^2 are the first and second elements in the pair pa_m. k_m^2 is right adjacent to k_m^1, and they have opposite direction to each other, as shown in Figure 5.3 (b).

PA The set of the pairs of adjacent directed paths, and $PA = \left\{ pa_m \mid \forall m \right\}$.

pt_h A pair of directed paths which forms a turn. $pt_h \equiv \left(k_h^1, k_h^2\right)$ where k_h^1 and k_h^2 are the first and second elements in the pair, as shown in Figure 5.3 (c).

PT The set of the pairs of directed paths which form a turn, $PT = \left\{ pt_h \mid \forall h \right\}$.

T Turning distance, positive value.

J The set of jobs. Each job $j \in J$ contains the information such as $\left(no_j, nd_j\right)$ where no_j is the origin node id and nd_j is the destination node id.

S_j The shortest distance of job j, positive value.

α a parameter for penalty.

G_r^U, G_r^D The up-going / down-going path group in row r, $r = 1,2,3...R$. R is the total number of path group in row.

G_c^L, G_c^R The left-going / right-going path group in column c, $c = 1,2,3...C$. C is the total number of path group in column.

δ^U, δ^D The minimal number of up/down in each path group in row.

δ^L, δ^R The minimal number of left/right in each path group in column.

M A large number.

Decision variables:

x_k Binary variable. $x_k = 1$ if traffic goes from node ns_k to node ne_k ; otherwise $x_k = 0$.

y_j Binary variable. $y_j = 1$ if job j cannot be delivered to its destination under current path design.

$a_{j,k}$ Binary variable. $a_{j,k} = 1$ if job j travelled on path k.

$b_{j,h}$ Binary variable. $b_{j,pt} = 1$ if job j turns at pt_h .

Objective:

$$\min \quad \sum_{j}\left(\sum_{k} L_k \cdot a_{j,k} + \sum_{h} T \cdot b_{j,h} + \alpha \cdot S_j \cdot y_j\right) \tag{5.1}$$

Constraints:

$$x_k + x_{k^-} \le 1, \quad k \in P,\ k^- \text{ is directed path which is opposite to } k \tag{5.2}$$

$$x_{k_m^1} + x_{k_m^2} \le 1, \quad \forall pa_m \equiv \left(k_m^1, k_m^2\right) \in PA \tag{5.3}$$

$$\sum_{k \in PS_n} a_{j,k} = 1 - y_j, \qquad n = no_j, j \in J \tag{5.4}$$

$$\sum_{k \in PE_n} a_{j,k} = 0, \qquad n = no_j, j \in J \tag{5.5}$$

$$\sum_{k \in PS_n} a_{j,k} = 0, \qquad n = nd_j, j \in J \tag{5.6}$$

$$\sum_{k \in PE_n} a_{j,k} = 1 - y_j, \qquad n = nd_j, j \in J \tag{5.7}$$

$$\sum_{k_1 \in PE_n} a_{j,k_1} = \sum_{k_2 \in PS_n} a_{j,k_2}, \quad n \in N - \{no_j, nd_j\}, j \in J \tag{5.8}$$

$$a_{j,k_h^1} + a_{j,k_h^2} - 1 \le b_{j,h}, \qquad \forall pt_h \equiv \left(k_h^1, k_h^2\right) \in PT, j \in J \tag{5.9}$$

$$\sum_{j \in J} a_{j,k} \le M \cdot x_k, \qquad k \in P \tag{5.10}$$

$$\sum_{k \in G_r^U} x_k \ge \delta^U \qquad r = 1,2,3...R \tag{5.11}$$

$$\sum_{k \in G_r^D} x_k \ge \delta^D \qquad r = 1,2,3...R \tag{5.12}$$

$$\sum_{k \in G_c^L} x_k \ge \delta^L \qquad c = 1,2,3...C \tag{5.13}$$

$$\sum_{k \in G_c^R} x_k \geq \delta^R \qquad c = 1,2,3...C \tag{5.14}$$

The objective of the problem is to find the directional path design which can minimize the total TU travel distance. The total travel distance includes the sum of the actual distance and the sum of the penalized distance among all jobs. The penalized distance is given to the jobs which cannot be delivered or have to travel a very long distance to reach the destination. In both cases, if the job has a large S_j, any changes related to this job will lead to more changes to other jobs in the system. Therefore, we introduce the penalized travel distance as $\alpha \cdot S_j$, and α should be a value larger than 1. The indicator variable, y_j, will be equal to one if job *j* cannot be delivered to its destination under current path design. The objective function can be represented as equation (5.1).

Due to physical restrictions, Constraint (5.2) represents that each path can at most have one travel direction, and Constraint (5.3) represents the direction restriction on adjacent vertical paths. Due to the rail sharing nature of the vertical paths in the storage area, the adjacent vertical paths cannot have opposite direction. Thus, the vertical path in between the two opposite direction vertical paths is closed to avoid deadlock. Once the path is closed, the corresponding nodes are closed and so the TUs cannot can access the nodes.

Constraints (5.4) to (5.7) represent the traffic flow of each job at the starting and ending nodes respectively. Constraint (5.8) represents the flow conservation of each job at the intermediate nodes between origin and destination. Constraint (5.9) determines the turns on the job route. Constraint (5.10) guarantees that the job flow

direction follows the path direction. Lastly, Constraints (5.11) to (5.14) ensure that at least a certain number of paths will be set to each direction.

5.2. Methodologies

Due to the difficulty of solving the PDP in large scale scenarios, the model might not be practical when solutions need to be computed quickly, especially in the dynamic environment. In addition, without considering the dispatching of the TUs, the PDP can only determine the path design for discharging and loading jobs. To solve these problems, we use the heat-map concept to represent the traffic flow on each path. By converting a path-job problem into a path-flow problem, there is no need to solve a very complex mathematical model, and the number of jobs can be very flexible. Therefore, the heat-map algorithm (HA) is proposed to convert the given jobs into a flow based heat-map and then generate the path design.

However, the above model and the HA do not take time into account. So we still need a method to evaluate the path design based on the throughput in terms of the number of containers per hour. Hence, the heat-map based simulation embedded algorithm (HSEA) is proposed to search and evaluate the path design iteratively. The flow charts of the HA and HSEA are shown in Figure 5.4. The elaborations of both algorithms can be found in Section 5.2.1 and Section 5.2.2, respectively.

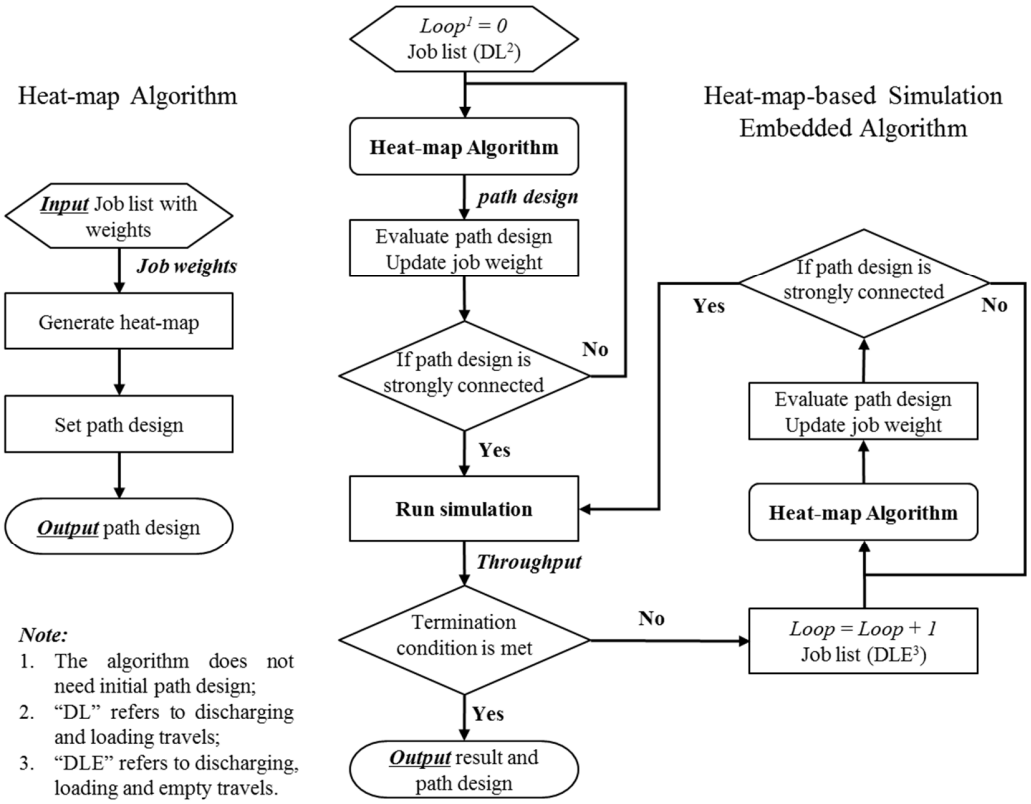


Figure 5.4 Demonstration of HA and HSEA

5.2.1. Heat-map Algorithm

In the original PDP, we need to find the route from the exact origin node to the exact destination node, and so we define the "path" as the line segment between two adjacent nodes. Actually, the paths along the whole path between two junction nodes have the same direction, as shown in Figure 5.5 (a). Since the purpose of the algorithm is to determine the direction along the whole path, it will be more convenient and simpler to define this whole path with another term to avoid confusion. Therefore, we define the whole path between two adjacent junction nodes as the arc, as shown in Figure 5.5 (b). Then we define the adjacent arc pair $pa_m \equiv \left(k_m^1, k_m^2\right)$ where k_m^1 and k_m^2 are the two arcs and m is the index of the pair. Figure 5.5 (c) shows two examples $pa_1 = (1,2)$, $pa_2 = (2,3)$. Since the arc graph is

given, each pair is known and can be uniquely indexed. Similar to Figure 5.3 (d1, d2, e1, e2), we define the arc group in column *c* as shown in Figure 5.5 (d) and the arc group in row *r* as shown in Figure 5.5 (e). They are notated as GC_c and GR_r respectively.

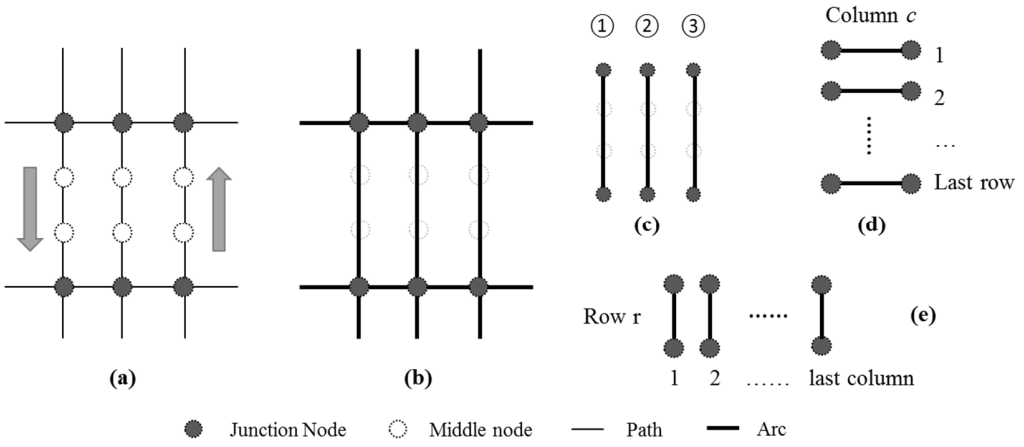


Figure 5.5 Definition of the arc

The process of the HA is demonstrated in Figure 5.4. The heat-map concept is commonly used to represent how heavy the traffic in a specific area is. In our study, we need to determine the arc direction which can provide the most traffic capacity. Therefore, the heat-map is used to represent how heavy the traffic for each direction on each arc is. However, if we do not have a path design, we cannot know the exact travel route of each job, on the other hand, if we do not know the exact route, we cannot have an accurate heat-map for traffic flow.

To deal with this issue, we define two concepts – potential travel area and potential travel direction – to estimate the traffic flow, as shown in Figure 5.6. Figure 5.6 (a) shows a job that needs to travel from its origin node to destination node. Based on the OD of the job, we can draw the shaded rectangle area which indicates the

potential area it will travel through. The job may not need to travel through this area, but from the distance perspective, it has a high chance of passing through.

Next, we can set the potential travel direction on the arcs within this area. In order to represent the impact of different jobs, the job weightage w_j is defined as the

$$w_j = \frac{d_j}{S_j}$$

where d_j is the actual distance of job j and S_j is the shortest distance of job j. The actual distance is the distance of the actual route including the turns. Initially, the path design does not exist so the weightage is set to 1, which is the lower bound. After the first iteration, the weightage starts to change, with a higher weightage indicating that the job is traveling longer than its shortest route and has a higher priority to be dealt with. Since different jobs have different effects on the arc due to the job weightage, each job will apply a vector on each arc within the area.

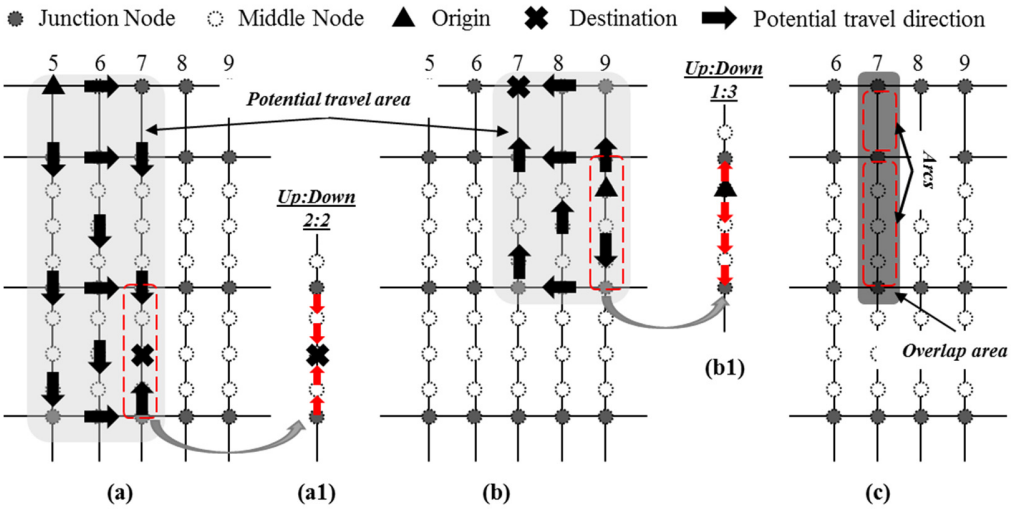


Figure 5.6 Demonstration on heat-map concept

Each job will have a potential travel area and each job has an effect on the arcs covered by the area. It should be noted that some vectors have two directions, such as the highlighted arcs in Figure 5.6 (a1) and (b1). The ratio of up and down flows of the arc in (a1) and (b1) is 2:2 and 1:3 respectively. If the weightage of the job in Figure 5.6 (a) equals to 1.5, the effect of this job for an arc in (a1) is 0.75 for both the up flow and down flow.

Since all the potential travel areas will be superimposed on the heat-map, the effect of all jobs on each arc is the sum of the corresponding vectors. For example, if the system only has two jobs as shown in Figure 5.6 (a) and (b), only the arcs within the overlap area as shown in Figure 5.6 (c) will be affected by the two jobs, and the rest of the arcs will be affected by one job at most. If we assume that the job weightage in Figure 5.6 (a) is 1.5 and the job weightage in Figure 5.6 (b) is 1.3, then the arcs in Figure 5.6 (c) will have an up flow of 1.3 and a down flow of 1.5. Eventually, we can easily convert all jobs into the flow-based heat-map.

Once the heat-map is generated, we can determine the path design with a reduced PDP model. The objective is to maximize the traffic flow. The restrictions introduced in Section 5.1 are formulated as constraints. Since the model only needs to decide arc directions, it can be optimally solved by CPLEX easily. The detailed model is presented below.

Reduced Path Direction Problem

Parameters:

N The set of nodes; node index is n;

I_{kn}^{σ} $\sigma \in \{L, R, U, D\}$, the 0-1 value which indicates that arc *k* should go left, right, up or down to provide an inbound connection to node *n*;

O_{kn}^{σ} $\sigma \in \{L, R, U, D\}$, the 0-1 value which indicates that arc *k* should go left, right, up or down to provide an outbound connection to node *n*;

PV The set of vertical arcs, index is k .

PH The set of horizontal arcs, index is *k*.

pa_m A pair of adjacent arcs. $pa_m \equiv \left(k_m^1, k_m^2\right)$ where k_m^1 and k_m^2 are arcs in the pair pa_m. k_m^2 is right adjacent to k_m^1, as shown in Figure 5.5 (c).

PA The set of the pairs of adjacent arcs, and $PA = \{pa_m \mid \forall m\}$.

F_k^L, F_k^R The left flow and right flow on arc *k*, respectively.

F_k^U, F_k^D The up flow and down flow on arc *k*, respectively.

GR_r The arc group in row *r*.

GC_c The arc group in column *c*.

δ^U, δ^D The minimal number of up/down in each arc group in row.

δ^L, δ^R The minimal number of left/right in each arc group in column.

Decision variables:

l_k $l_k = 1$ if arc *k* is set to go left, otherwise 0, for $k \in PH$.

r_k $r_k = 1$ if arc k is set to go right, otherwise 0, for $k \in PH$.

u_k $u_k = 1$ if arc k is set to go up, otherwise 0, for $k \in PV$.

d_k $d_k = 1$ if arc k is set to go down, otherwise 0, for $k \in PV$.

e_k $e_k = 1$ if arc k is closed, otherwise 0, for $k \in PV$.

Objective:

$$\max \quad \sum_{k \in PV} \left(F_k^U \cdot u_k + F_k^D \cdot d_k \right) + \sum_{k \in PH} \left(F_k^L \cdot l_k + F_k^R \cdot r_k \right) \qquad (5.15)$$

Subject to:

$$u_k + d_k + e_k = 1 \qquad \forall k \in PV \qquad (5.16)$$

$$l_k + r_k = 1 \qquad \forall k \in PH \qquad (5.17)$$

$$u_{k_m^1} + d_{k_m^2} \le 1 \qquad \forall pa_m \equiv \left(k_m^1, k_m^2 \right) \in PA \qquad (5.18)$$

$$d_{k_m^1} + u_{k_m^2} \le 1 \qquad \forall pa_m \equiv \left(k_m^1, k_m^2 \right) \in PA \qquad (5.19)$$

$$\sum_{k \in GR_r} u_k \ge \delta^U \qquad r = 1, 2, 3 \ldots R \qquad (5.20)$$

$$\sum_{k \in GR_r} d_k \ge \delta^D \qquad r = 1, 2, 3 \ldots R \qquad (5.21)$$

$$\sum_{k \in GC_c} l_k \ge \delta^L \qquad c = 1, 2, 3 \ldots C \qquad (5.22)$$

$$\sum_{k \in GC_c} r_k \ge \delta^R \qquad c = 1, 2, 3 \ldots C \qquad (5.23)$$

$$\sum_{k \in PV} I_{kn}^U \cdot u_k + \sum_{k \in PV} I_{kn}^D \cdot d_k + \sum_{k \in PH} I_{kn}^L \cdot l_k + \sum_{k \in PH} I_{kn}^R \cdot r_k \ge 1 \qquad n \in N \quad (5.24)$$

$$\sum_{k \in PV} O_{kn}^{U} \cdot u_k + \sum_{k \in PV} O_{kn}^{D} \cdot d_k + \sum_{k \in PH} O_{kn}^{L} \cdot l_k + \sum_{k \in PH} O_{kn}^{R} \cdot r_k \geq 1 \qquad n \in N \qquad (5.25)$$

The objective function is represented by equation (5.15), which is to maximize the traffic flow of the heat-map. Constraint (5.16) guarantees that each arc can only have one state. Constraint (5.17) guarantees that the arc goes either left or right. Constraints (5.18) and (5.19) make sure that the adjacent vertical arcs will not have opposite directions. Constraints (5.20) to (5.23) set the minimal number of arcs which should go on each direction. Constraints (5.24) and (5.25) guarantee that there is at least one inbound connection and one outbound connection at each node.

We can thus obtain the path design, which is the solution to the above model and the output of the HA. If all nodes are accessible by starting from any node in the graph, then the path design is defined as strongly connected. It should be noted that the model cannot guarantee that the path design is strongly connected. Thus, we still need to resolve this issue so that the path design can be implemented to direct the movement of the TUs.

5.2.2. Heat-map-based Simulation Embedded Algorithm

The HA could efficiently provide a path design. However, three major challenges remain open: firstly, the path design provided by the HA may not be strongly connected, and thus it is possible that some jobs cannot reach their destination; secondly, we still need a method to evaluate the path design based on throughput in terms of the number of containers per hour; and thirdly, we need a method to improve the solution. Therefore, the HSEA, as shown in Figure 5.4, is proposed to deal with these challenges.

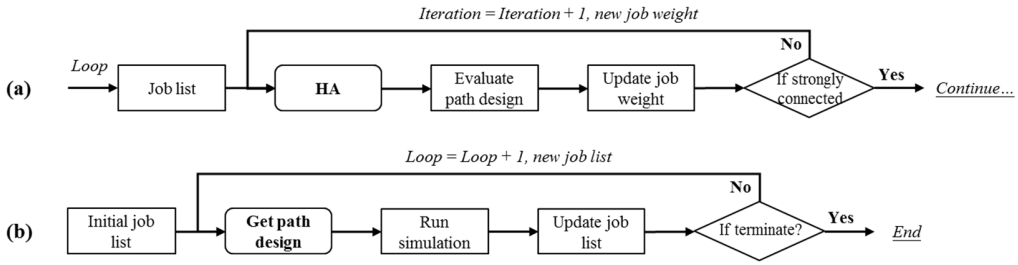


Figure 5.7 Illustration on iteration and loop levels

For the first challenge, the model in Section 5.1 cannot ensure that a strongly connected path design is obtained. Besides, there is no theory that can guarantee the connectivity, and no efficient algorithm to find a strongly connected directed graph. Although the connectivity of the path design can be tested by enumerating routes from node to node, it will be very time consuming in a large scale system. Thus, instead of solving this issue at once, we solve it iteratively by updating the job weightages. In the first iteration, the path design does not exist and the weightage of all jobs is one. Then the HA is applied to generate the path design. After that, the shortest path algorithm (e.g. the Dijkstra's algorithm) is applied to all jobs to evaluate the path design and update the job weightages by the formulation in Section 5.1. If some jobs cannot be delivered due to the lack in connectivity, the weightages of these jobs will equate to a large value (job penalty), so the related arcs will be "flagged" in the heat-map in the next iteration. Based on the new job weightages, a new heat-map will be generated again by the HA, and eventually, after a few iterations, we can obtain a strongly connected path design. The detailed demonstration can be found in Figure 5.7 (a).

For the second challenge, instead of proposing a time dependent model for scheduling and dispatching, we apply discrete event simulation, which can closely

describe the actual system, to evaluate the path design. The simulation model is simplified from Zhou *et al.* (2015). Since the SDTR has guaranteed that there are no opposite movements on the same and adjacent vertical arcs and no detouring of TUs, the simulation model only needs to find the shortest route for each TU based on the path design, and then control its movement. The key performance indicator of the simulation model is the same as that of the previous study, which is the throughput in terms of the number of containers per hour per AP. In addition, the dispatching policy in the simulation model is first-come-first-served, which also follows the previous study.

It should be noted that the job list remains the same in the same loop, but will be slightly different between loops. Initially, the job list contains all loading and discharging activities at *Loop 0*. Starting from *Loop 1*, the job list will include the original loading and discharging jobs, and the newly generated empty jobs from the last simulation run. Due to the difference in path designs and the randomness of the simulation, the empty jobs are different between loops. By taking the empty jobs into account, the path design is expected to improve, which meets the demand of the third challenge. Eventually the whole process will stop repeating when certain termination condition is met. The detailed demonstration can be found in Figure 5.7 (b).

In this study, we define that the HSEA will terminate if the throughput of each loop hits a certain range for θ times. The range is defined as $\pi\%$ of the maximal throughput to the maximal value. The maximal throughput can be updated if higher output occurs. For example, if the current maximal value is 40, $\theta = 3$ and $\pi = 90$,

the algorithm will terminate if the three loops have the outputs ranging from 36 to 40. If the latest output is higher than 40, for example 42, the simulation will then set the current maximal value to 42 and check the historical outputs if there are three outputs ranging from 37.8 to 42. If the condition is met, then the algorithm terminates and outputs the heat-map which provides the maximal throughput.

5.3. Dynamic Framework

In order to dynamically update the path design in a continuously running system, there are three challenges to be dealt with: firstly, the trigger or the timing of updating of the path design has to be set properly; next, both working jobs and scheduled jobs have to be handled properly; and lastly, the routing policy has to be determined. The overview of the solution of the framework is shown in Figure 5.8. The framework consists of three parts: the control module, which is to determine the trigger of updating the path design and how the system acts at the trigger; the path design module, which is used to determine the path design according to the job information; and the routing module, which is used to determine the route for each TU based on a specific path design. The details of each part are explained in the following sections.

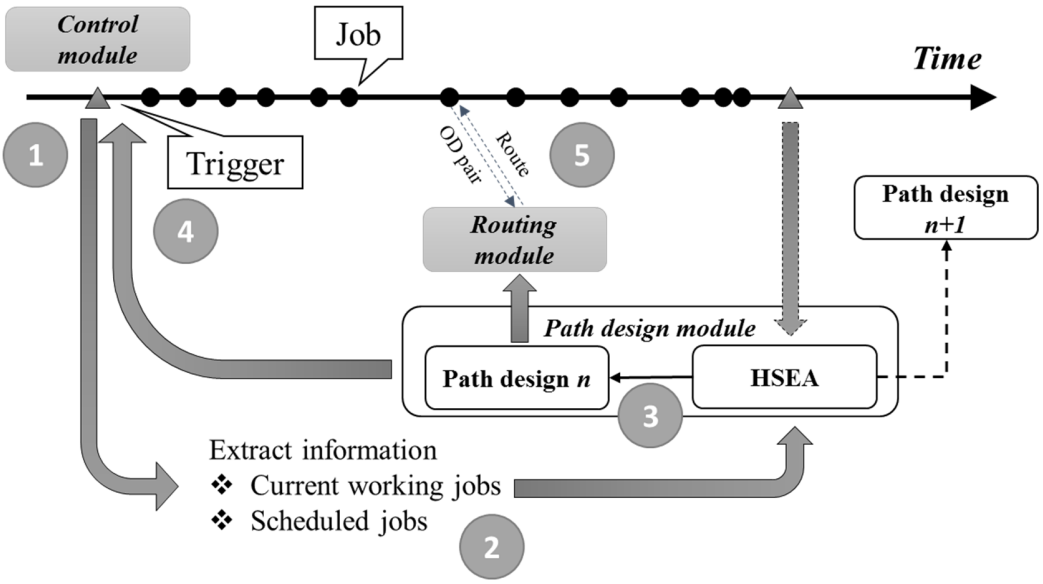


Figure 5.8 The structure of the dynamic framework

5.3.1. Framework Structure

Control Module (CTM). When changing the path design, there are three major problems that need to be resolved: firstly, if and when the path design should be updated; next, what information will be used to update the path design; and lastly, how to deal with the TUs during the transition period between two path designs, e.g., when the new path design has temporarily closed a column but TUs are still running on it. Therefore, three important components are defined in the control module to deal with the above problems.

The first component is the trigger, which defines the event when the path design needs to be updated. There are many different events which can be the trigger, for example, vessel arrival, container flow changing, operator decision, etc. It should be noted that when updates are triggered, there might be still some existing jobs in progress. Therefore, when the event triggers the path updating, the job details,

including current unfinished jobs, future scheduled jobs, will be needed by the path design module.

The second component is the evaluator, which compares the old path design with the new design and determines whether to implement the new design. Here we will compare the throughput of the two designs by the simulation model proposed in Section 5.2. If the new design is used, then the transition control will be activated, otherwise, the whole system will keep running.

The third component is the transition control which defines how the system should react when a new path design is obtained from the path design module. As illustrated in Figure 5.9, at the juncture where path designs are updated, a suitable transition mechanism is required to move TUs out of newly closed columns before the routing module can take over with the new path design. Once the transition is over, all TUs will then be rerouted from their existing positions to their original destinations at the point of the path design change.

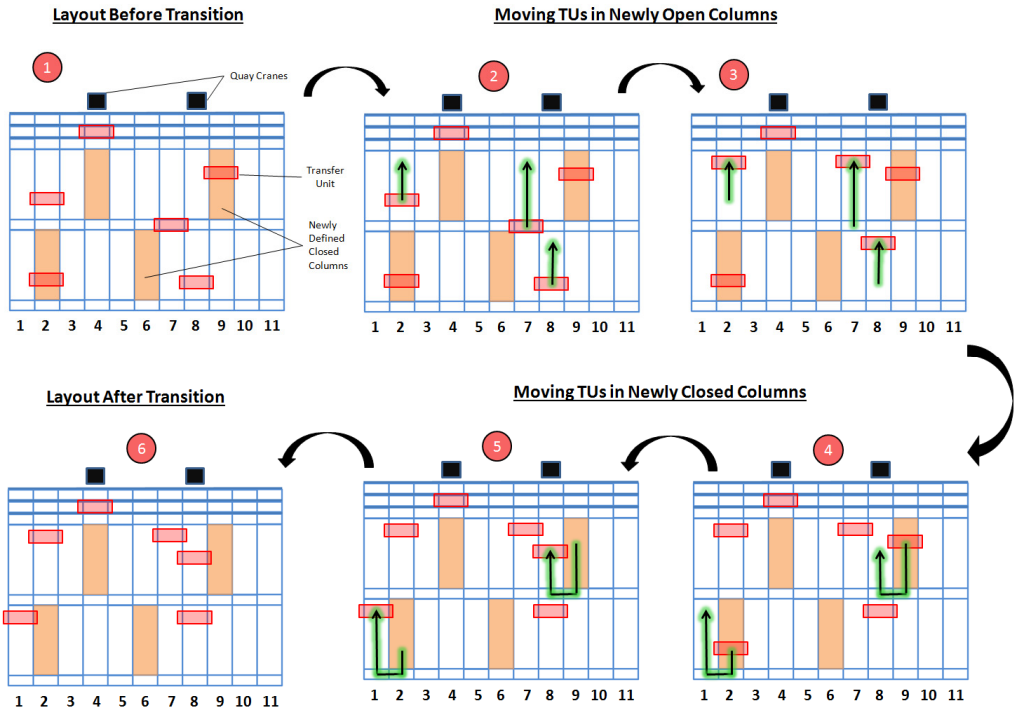


Figure 5.9 Example of the transition control

Path Design Module (PDM). The path design module is used to determine the path direction based on given job list, including working and scheduled jobs. The HSEA is implemented and its output will be set as the current path design. Once the path design is determined, it will not change until next trigger.

Routing Module (RTM). The routing module is designed to determine TU routes based on current path design. Whenever a job is ready to move or needs to be rerouted, the TU will request the route from this module. The basic routing principle is the shortest path.

5.3.2. Framework Process Flow

Before the system runs, it needs to be initialized by executing the PDM once with given scheduled jobs as shown by Step 3 in Figure 5.8. The output of the PDM will be used as the current path design as in Step 5. Then the system can start running.

When a job is initializing as in Step 5, it will request a route from RTM. Once the job receives the route, it is ready for pick up by any TU. Once the TU picks up a job, it will travel on the given route. The procedure is the same for empty travels.

When the first trigger condition is met, i.e., a certain number of containers have been delivered, then the system will call the CTM and extract the information as in Step 1. As a continuous system, there will be jobs in progress still traveling in the system. Thus, we will need to extract the information of working and scheduled jobs from the system and then send to PDM for processing, as in Step 2. Once the path design is generated by the PDM, it will be sent back to CTM for evaluation. If the path design requires an update, then the transition control will take over to assist TUs in moving to safe locations; otherwise the system will resume working with the original path design.

5.4. Numerical Experiment

In order to compare with the previous study by Zhou *et al.* (2015) while considering the practicality in terms of storage capacity, the HSEA experiment uses the same GRID layout – the 1x5 GRID layout with eleven columns and 103 rows (including three rows for transfer area, as shown in Figure 5.10). The storage capacity of this layout reaches 11,000 TEUs if the stacking height is five containers. Since the TU speed is around 1 meter per second for loaded TUs and 2 meters per second for empty TUs, and turning takes 5 seconds, we assume that the turning distance T is 2.8 meters.

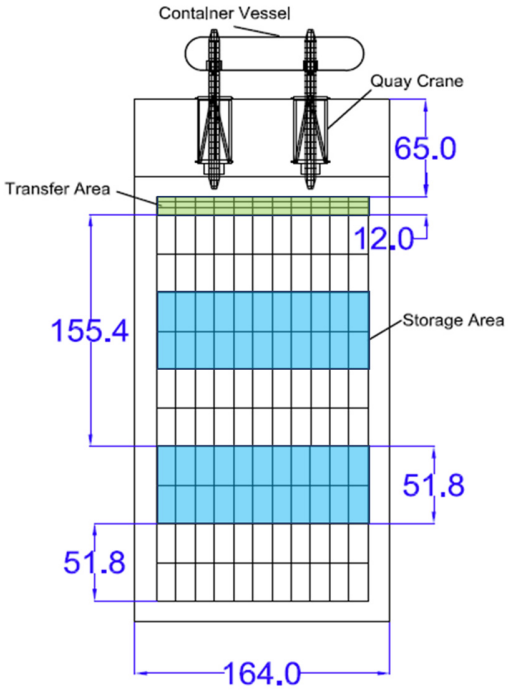


Figure 5.10 The 1x5 GRID layout with 11 columns and 103 rows

For HSEA, the discrete event simulation (DES) model used to evaluate the path design is built with commercial software AutoMod. The system throughput will be averaged across three replications conducted with different random number sets. The system throughput is defined as

$$\text{Throughput} = \frac{N_D + N_L}{N_{AP} \cdot t} \quad \text{(Containers per AP per hour)}$$

where N_D is the number of discharging containers, N_L is the number of loading containers, N_{AP} is the number of APs, and t is the simulation length. To be consistent with the previous study, we assume that the system only has two access points, which are *Node 4* and *Node 8*.

For the experiment environment, MIP models are coded in C# with Microsoft Visual Studio 2015 and solved with CPLEX 12.6. The HA and HSEA are also written in C#. The computer used to run the experiment has Intel i7 CPU and 16GB memory.

5.4.1. The Performance of HSEA

Without loss of generality, we use a medium scale scenario, which cannot be optimally solved under PDP, to demonstrate the performance of HSEA. We randomly generate 500 jobs for Case 1 and Case 2; the simulation length is set to five hours and the number of TUs is ten. We will not terminate the algorithm until fifteen runs are completed. The result is presented in Figure 5.11. The x-axis represents the index of the run and the y-axis is the throughput.

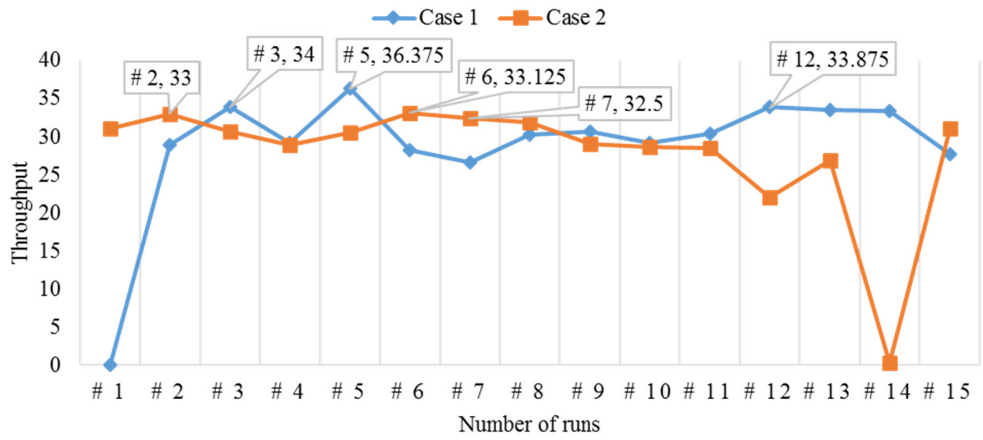


Figure 5.11 The performance of HSEA in medium scale scenario

The result indicates that the path design will be improved within several runs with some fluctuations. If the termination condition is set properly, the algorithm can find a good path design quickly. Eventually, the path design which has the maximum throughput will be chosen as the output of the algorithm. For Case 1 and Case 2, the maximum throughput is 36.38 and 33.13 respectively. It should be noted that

occasionally, the simulation hangs or the traffic is blocked. Since the job weightages have been updated by the latest path design, the algorithm is still able to continue. Next, we run another three cases with 500 jobs. The maximum throughputs are found to be 33.50, 33.86 and 35.13. Therefore, the average throughput of the 1x5 GRID with ten TUs is 34.40 with a standard deviation of 1.34.

To demonstrate the scalability, another experiment is designed by generating a 1,000-job case with a ten-hour simulation time. Similarly, we will not terminate the algorithm until fifteen runs are completed and the result is presented in Figure 5.12. It should be noted that while the path design is strongly connected, there are still some cases which stop the simulation from running properly, leading to very low throughputs. Since the job weightages will still be updated based on the path design generated in the previous loop (run), the algorithm can continue searching for new path design. The overall trend of the result shows that the algorithm performs similarly for different scales of problem.

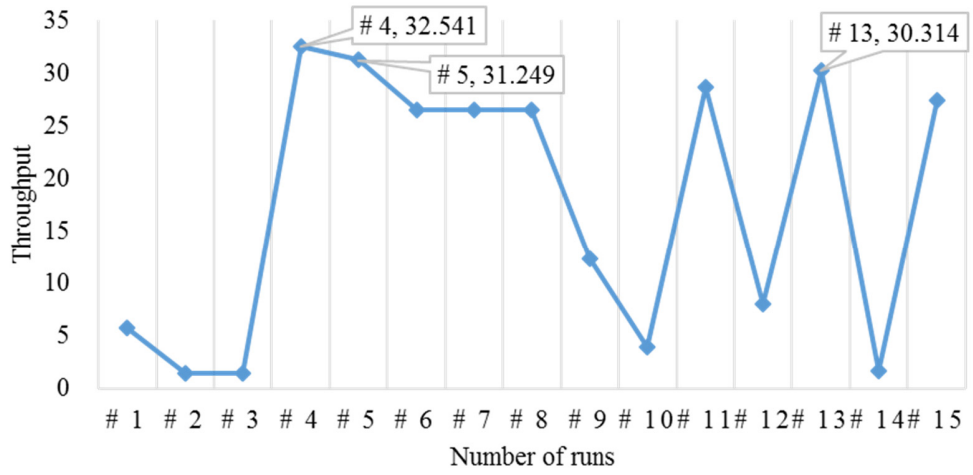


Figure 5.12 The performance of HSEA in larger scale scenario

To demonstrate the algorithm efficiency, we compare the time per iteration and the iteration per loop for solving cases with 500, 1000, and 2000 jobs. For each case, we conducted fifty runs. As demonstrated in Figure 5.7 (a), each iteration outputs a path design and each loop has at least one iteration, as illustrated in Figure 5.7 (b). Since the simulation time depends on the simulation parameters, we exclude the simulation time to have a fair comparison. The result is presented in Table 5.1. Although the computation time increases with the number of jobs, it is negligible compared with the actual time needed to handle all jobs, i.e., seconds versus approximately 29 hours for handling 2000 jobs (with 10 TUs and 2 APs, 34.40 jobs per hour).

Table 5.1 Algorithm efficiency

Job number	500	1000	2000
Time per iteration (seconds)	13.2	19.4	51.7
Iteration per loop	1.15	1.21	1.19

In the next section, we will compare the HSEA for SDTR with FRTR by Zhou *et al.* (2015) in the same manner.

5.4.2. Comparison with Free Routing Traffic Rule

Using the FRTR, we tested the throughput of the 1x5 GRID for different numbers of TUs. The result is listed in Table 5.2. Due to the TU conflicts, the throughput is limited to 41.5 with fourteen TUs.

Next, we tested the HSEA with different numbers of TUs. We randomly generate five cases with 800 jobs each. To be consistent with the previous study, the

simulation length is set to five hours. Also, the termination condition is $\theta = 3$, $\pi = 90$, and the numbers of TUs are 10, 12, 14, 16, 18 and 20.

Table 5.2 Result of FRTR in 1x5 GRID

Number of TUs	10	11	12	13	14	15
Throughput	36	38.9	40	40.9	41.5	33.7

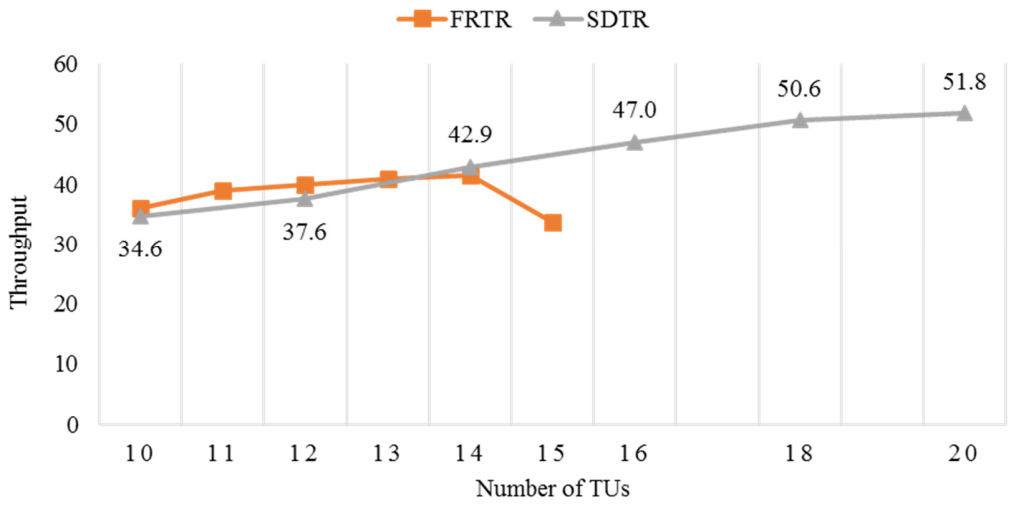


Figure 5.13 Comparison with FRTR

The result is shown in Figure 5.13, indicating that the SDTR is more suitable for the scenarios with large TU count. In fact, conflicts rarely happen in FRTR when the TU number is small, and as such, the majority of the jobs can travel on the shortest route. When the number of TUs increases, the SDTR will greatly reduce TU conflicts, which is much more significant than the shortest route.

5.4.3. Performance of the Dynamic Framework

The dynamic framework is originally designed to run in a real system. However, since the GRID system is still a prototype, we use another DES model built in

AutoMod to validate the framework. To validate our framework, we use the "job count" as the trigger event. With this trigger, the update will be activated whenever a certain number of jobs has been delivered. To incorporate the working jobs, we create new jobs with their current position and destination. Eventually, a number of scheduled jobs and a few "created" working jobs will be inputs into the algorithm to find the path design. It should be noted that the job count cannot be too small or too large; if the number is very small, the update is executed very frequently, while if the number is very large, it may not reflect the actual situation. For example, the pattern of the container flow has changed during the 29 hours to finish 2,000 jobs.

In the experiment, the job count is set as 500. We set the total number of jobs to be 5,000 jobs which are randomly generated and the number of TUs as ten. Whenever the CTM is triggered, the simulation time from previous event to present is captured to calculate the throughput, based on the formulation

$$\text{Throughput} = \frac{500}{2 \bullet \left(t^{present} - t^{previous} \right)} \quad \text{(Containers per AP per hour)}$$

where $t^{present}$ represents the current simulation clock time and $t^{previous}$ represents the simulation clock time when previous event is triggered. The parameters for the HSEA and the termination condition are $\theta = 3$, $\pi = 90$, and the simulation length is set to five hours. Although the HSEA also requires the use of simulation, the two models serve different purposes. The result is shown in Figure 5.14 with comparison to the static path design, which only runs the HSEA once and uses the obtained path design for all 5,000 jobs. It indicates that the overall performance of the path design updated by the dynamic framework is better than the static path design. Since the

path design will be updated for every 500 jobs finished, the throughput can be pushed to a high value and yet maintain stability with path changing. In some cases, the dynamic framework and static path design have similar performance. Since the scale of the GRID structure is not very big, the variations of the good path design are limited and so it may happen that the path design obtained from the dynamic framework is quite close to the static one. Although there are still some limitations on the transit mechanism, the overall framework has been shown to be useful and promising.

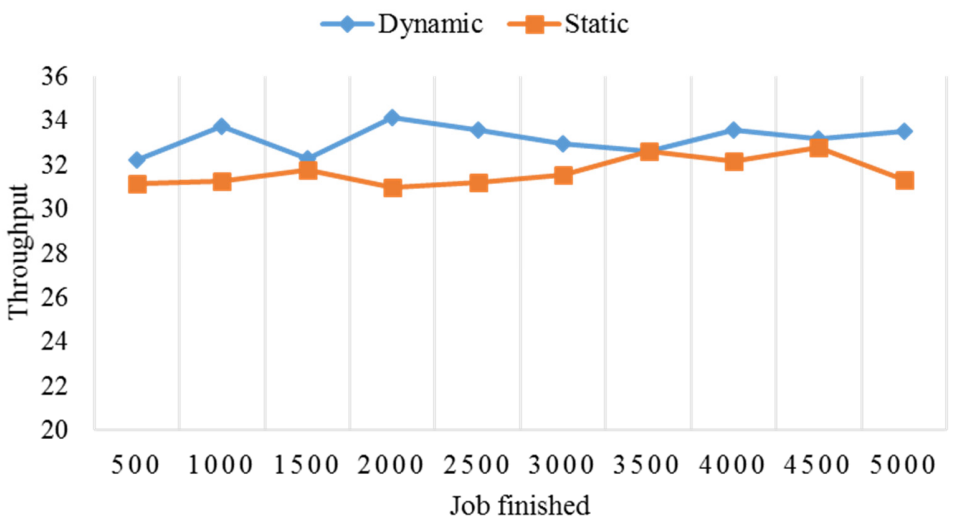


Figure 5.14 Comparison between dynamic framework with static path design

5.5. Conclusion

Due to the dynamic feature and flexible design of the GRID system, it is very challenging to reduce the vehicle conflicts in large scale scenarios. Instead of proposing a conflict-free routing or scheduling problem, we proposed the single direction traffic rule and formulated a path design problem. Although the path design

problem is not new, previous solutions are not applicable for large scale scenarios and dynamically changing environment.

Hence, motivated by the heat-map concept used in other domains, this study provided a novel algorithm, HSEA, to solve the path design problem. As a part of the HSEA, the algorithm HA converts a path-job problem into a path-flow problem so that it does not need to solve a very complex mathematical model, and hence, the scale of the problem can be very flexible. The integration of the simulation allows for the performance analysis of the algorithm, using the meaningful system throughput which is comparable with previous works. A dynamic framework was also proposed to update the path design using the HSEA. To deal with the challenges in the continuously running system, three modules were introduced.

The numerical experiment has proven that the algorithm is efficient and scalable in different scenarios. Although the computation time increases with the number of jobs, it is negligible compared to the actual time needed to handle all jobs. By comparing with the previous work, the SDTR is shown to be effective in the scenarios with large TU count, since the SDTR greatly reduces TU conflicts. In addition, the dynamic framework has been proven to be effective by using a simulation model proposed to simulate the GRID system.

Chapter 6. Conclusions and Future Research

The current trend now is pushing transshipment terminals to achieve both higher yard utilization for storage, and higher handling productivity in the future, thus driving terminals to look for new designs, configurations, and/or equipment. BEC Industries LLC introduced a fully automated handling prototype named GRID. For actual use, this study implemented the prototype into a single GRID system concept and a hybrid GRID system concept. In this thesis, we study three actual problems faced by using the GRID system in transshipment terminals. As the first study on the GRID system, the priority is to understand its throughput characteristics and potential bottlenecks, and as such, a simulation study is necessary. Due to the limitation of the GRID system on scalability and productivity, the vehicle management problem in the single GRID system and the container allocation problem in the hybrid GRID system are urging new studies.

In Chapter 3, a discrete-event simulation model is proposed for the GRID system to evaluate the system performance. Due to the nature of the GRID structure, congestion increases rapidly when the number of TUs increases. In addition, we concluded that the single GRID system is not suitable for horizontal expansion, making it incapable of being applied in large terminals, and this motivates us to propose the concept of a hybrid GRID system. A new modular simulation model was designed for the hybrid GRID system and the experiments proved that the system can be very efficient on overall productivity. It is a promising design for the future transshipment terminals.

In Chapter 4, we proposed a novel approach of developing storage allocation strategies. The idea is that we develop a mathematical model for the allocation problem and try to solve it in small instances. By analyzing the optimal solutions, the factors that determines the allocation can be identified. Finally, by using the above knowledge, an information-based allocation strategy is developed. The numerical experiments show that the new strategy performs better in both terminal operation efficiency and resource usage compared to the traditional strategies such as COI, DoS and purely random. In addition, the new strategy can also be used for operational decisions where the information on arriving containers will be available for only a few shifts ahead.

In Chapter 5, due to the dynamic feature and flexible design of the GRID system, it becomes very challenging to reduce the vehicle conflicts in large scale scenarios. Instead of proposing a conflict-free routing or scheduling problem, we propose the single direction traffic rule and formulate a path design problem. Motivated by the heat-map concept used in other domains, a novel algorithm, HSEA, is proposed to solve the path design problem. As a part of the HSEA, the algorithm HA converts a path-job problem into a path-flow problem so that it does not need to solve a very complex mathematical model, and hence, the scale of the problem could be very flexible. The integration of the simulation allows the performance analysis of the algorithm in terms of the meaningful system throughput which is comparable with previous works. A dynamic framework is also proposed to update the path design using the HSEA to deal with the challenges in the continuously running system. This indicates a potential research area for GRID systems as well as AGV systems.

Since the GRID system is still a new concept, there are several topics related to this thesis that can allow further research to be conducted. For the hybrid GRID systems, the current integrated problem is more suitable for planning level decision making, with the container handling requests usually given as inputs. Future research may need to consider operation level decision making such as requests arriving in a rolling horizon manner. Secondly, the information-based allocation strategy uses optimal solutions of the solvable scenarios to perform regression, and uses solutions of the relaxed model of the insolvable scenarios for validation. Since the relaxed model is not part of the regression, it may be possible to use the solution of the relaxed model to improve the strategy. In the single GRID system, the dynamic framework currently needs to pause the simulation process while waiting for the path design module to finish. However, pausing the whole system while waiting for further decision may not be a good option in reality. Therefore, an improvement on transition control is expected in further studies. Besides, instead of finding conflict-free routing and scheduling, this study defines single direction path design for the GRID system to avoid conflicts. However, this concept cannot guarantee a conflict-free scenario, especially when there is a very large number of TUs. Further studies on specific strategies or rules can be expected to solve TU conflicts in heavy traffic condition.

References

Asef-Vaziri, A., Laporte, G., & Sriskandarajah, C. (2000). The block layout shortest loop design problem. *IIE Transactions, 32*(8), 727-734.

Bae, H. Y., Choe, R., Park, T., & Ryu, K. R. (2011). Comparison of operations of AGVs and ALVs in an automated container terminal. *Journal of Intelligent Manufacturing, 22*(3), 413-426.

Banerjee, P., & Zhou, Y. (1995). Facilities layout design optimization with single loop material flow path configuration. *International Journal of Production Research, 33*(1), 183-203.

Boer, C. A., & Saanen, Y. A. (2012). Improving container terminal efficiency through emulation. *Journal of Simulation, 6*(4), 267-278.

Bozer, Y. A., & Srinivasan, M. M. (1991). Tandem configurations for automated guided vehicle systems and the analysis of single vehicle loops. *IIE Transactions, 23*(1), 72-82.

Bozer, Y. A., & Srinivasan, M. M. (1992). Tandem AGV systems: a partitioning algorithm and performance comparison with conventional AGV systems. *European Journal of Operational Research, 63*(2), 173-191.

Carlo, H. J., Vis, I. F., & Roodbergen, K. J. (2014). Storage yard operations in container terminals: Literature overview, trends, and research directions. *European Journal of Operational Research, 235*(2), 412-430.

Castalia. (2012). *The Effect of Wages on Australian Port Costs and their Competitiveness in an International Context* Retrieved from

http://www.castalia-advisors.com/files/Port_Wage_Cost_report_20120604.pdf

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

Chen, M.-T., McGinnis, L., & Zhou, C. (1999). Design and operation of single-loop dual-rail inter-bay material handling system. *International Journal of Production Research, 37*(10), 2217-2237.

Chen, W.-L. (1995). *A planning model for the tandem automated guided vehicle system.* MS thesis, National Yunlin Institute of Technology, Taiwan, ROC.

Choi, S., & Ha, T. (2005). A study on high-efficiency yard handling system for next generation port. *Ocean Policy Research, 20*, 81-126.

Corréa, A. I., Langevin, A., & Rousseau, L. M. (2004, April). Dispatching and conflict-free routing of automated guided vehicles: A hybrid approach combining constraint programming and mixed integer programming. In *International Conference on Integration of Artificial Intelligence (AI) and Operations Research (OR) Techniques in Constraint Programming* (pp. 370-379). Springer Berlin Heidelberg.

Desaulniers, G., Langevin, A., Riopel, D., & Villeneuve, B. (2003). Dispatching and conflict-free routing of automated guided vehicles: An exact approach. *International Journal of Flexible Manufacturing Systems, 15*(4), 309-331.

Gaskins, R., & Tanchoco, J. (1987). Flow path design for automated guided vehicle systems. *International Journal of Production Research, 25*(5), 667-676.

Gu, J., Goetschalckx, M., & McGinnis, L. F. (2007). Research on warehouse operation: A comprehensive review. *European Journal of Operational Research, 177*(1), 1-21.

Guan, X., Dai, X., & Li, J. (2011). Revised electromagnetism-like mechanism for flow path design of unidirectional AGV systems. *International Journal of Production Research, 49*(2), 401-429.

Guldogan, E. (2011). Simulation-based analysis for hierarchical storage assignment policies in a container terminal. *Simulation, 87*(6), 523-537.

Han, Y., Lee, L. H., Chew, E. P., & Tan, K. C. (2008). A yard storage strategy for minimizing traffic congestion in a marine container transshipment hub. *OR Spectrum, 30*(4), 697-720.

Ho, Y.-C., & Hsieh, P.-F. (2004). A machine-to-loop assignment and layout design methodology for tandem AGV systems with multiple-load vehicles. *International Journal of Production Research, 42*(4), 801-832.

Hu, H., Huang, Y., Zhen, L., Lee, B. K., Lee, L. H., & Chew, E. P. (2014). A decomposition method to analyze the performance of frame bridge based automated container terminal. *Expert Systems with Applications, 41*(2), 357-365.

Hu, H., Lee, B. K., Huang, Y., Lee, L. H., & Chew, E. P. (2013). Performance Analysis on Transfer Platforms in Frame Bridge Based Automated Container Terminals. *Mathematical Problems in Engineering, 2013*.

Jiang, X., Chew, E. P., Lee, L. H., & Tan, K. C. (2013). Flexible space-sharing strategy for storage yard management in a transshipment hub port. *OR Spectrum, 35*(2), 417-439.

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

Jin, J. G., Lee, D.-H., & Hu, H. (2015). Tactical berth and yard template design at container transshipment terminals: A column generation based approach. *Transportation Research Part E: Logistics and Transportation Review, 73*, 168-184.

Kaspi, M., & Tanchoco, J. (1990). Optimal flow path design of unidirectional AGV systems. *International Journal of Production Research, 28*(6), 1023-1030.

Kim, C. W., & Tanchoco, J. (1993). Operational control of a bidirectional automated guided vehicle system. *International Journal of Production Research, 31*(9), 2123-2138.

Kim, K. H., Phan, M.-H. T., & Woo, Y. J. (2012). New conceptual handling systems in container terminals. *Industrial Engineeering & Management Systems, 11*(4), 299-309.

Kouvelis, P., Gutierrez, G. J., & Chiang, W.-C. (1992). Heuristic unidirectional flowpath design approaches for automated guided vehicle systems. *International Journal of Production Research, 30*(6), 1327-1351.

Le-Anh, T., & De Koster, M. (2006). A review of design and control of automated guided vehicle systems. *European Journal of Operational Research, 171*(1), 1-23.

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, L. H., Chew, E. P., Jiang, X., & Zhou, C. (2014). *A simulation study for next generation transshipment port.* Paper presented at the Proceedings of the 2014 Winter Simulation Conference.

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

Liu, C., Jula, H., & Ioannou, P. (2002). Design, simulation, and evaluation of automated container terminals. *IEEE Transactions on Intelligent Transportation Systems, 3*(1), 12-26.

Liu, C., Jula, H., & Ioannou, P. (2001). *A simulation approach for performance evaluation of proposed automated container terminals.* Paper presented at the Intelligent Transportation Systems, 2001. Proceedings. 2001 IEEE.

Liu, S., Liu, C., Luo, Q., Ni, L. M., & Qu, H. (2011). *A visual analytics system for metropolitan transportation.* Paper presented at the Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems.

Luo, J., & Wu, Y. (2015). Modelling of dual-cycle strategy for container storage and vehicle scheduling problems at automated container terminals. *Transportation Research Part E: Logistics and Transportation Review, 79*, 49-64.

Murty, K. G. (2007). Yard crane pools and optimum layouts for storage yards of container terminals. *Journal of Industrial and Systems Engineering, 1*(3), 190-199.

Nevins, M. R., Macal, C. M., & Joines, J. C. (1998). A discrete-event simulation model for seaport operations. *Simulation*, *70*(4), 213-223.

Nishimura, E., Imai, A., Janssens, G. K., & Papadimitriou, S. (2009). Container storage and transshipment marine terminals. *Transportation Research Part E: Logistics and Transportation Review, 45*(5), 771-786.

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.

Petering, M. E. (2013). Real-time container storage location assignment at an RTG-based seaport container transshipment terminal: problem description, control system, simulation model, and penalty scheme experimentation. *Flexible Services and Manufacturing Journal* , *27*(2-3), 351-381.

Qiu, L., & Hsu, W.-J. (2001). A bi-directional path layout for conflict-free routing of AGVs. *International Journal of Production Research, 39*(10), 2177-2195.

Qiu, L., Hsu, W.-J., Huang, S.-Y., & Wang, H. (2002). Scheduling and routing algorithms for AGVs: a survey. *International Journal of Production Research, 40*(3), 745-760.

Rezapour, S., Zanjirani-Farahani, R., & Miandoabchi, E. (2011). A machine-to-loop assignment and layout design methodology for tandem AGV systems with single-load vehicles. *International Journal of Production Research, 49*(12), 3605-3633.

Roodbergen, K. J., & Vis, I. F. (2009). A survey of literature on automated storage and retrieval systems. *European Journal of Operational Research, 194*(2), 343-362.

Saanen, Y. A., & Valkengoed, M. V. (2005, 4-4 Dec. 2005). *Comparison of three automated stacking alternatives by means of simulation.* Paper presented at the Simulation Conference, 2005 Proceedings of the Winter.

Saidi-Mehrabad, M., Dehnavi-Arani, S., Evazabadian, F., & Mahmoodian, V. (2015). An Ant Colony Algorithm (ACA) for solving the new integrated model of job shop scheduling and conflict-free routing of AGVs. *Computers & Industrial Engineering* , *86*, 2-13.

Seo, Y., Moon, C., Moon, Y.-H., Kim, T., & Kim, S. S. (2008). *Adapting genetic algorithm and tabu search approaches for unidirectional AGV flowpath design problems.* Paper presented at the 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence).

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.

Sun, Z., Tan, K. C., Lee, L. H., & Chew, E. P. (2013). Design and evaluation of mega container terminal configurations: An integrated simulation framework. *Simulation, 89*(6), 684-692.

Tanchoco, J., & Sinriech, D. (1992). OSL—optimal single-loop guide paths for AGVS. *International Journal of Production Research, 30*(3), 665-681.

Vatin, G., & Napoli, A. (2013). *Guiding the controller in geovisual analytics to improve maritime surveillance.* Paper presented at the GEOProcessing 2013: the Fifth International Conference on Advanced Geographic Information Systems, Applications, and Services.

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

Vis, I. F., & Harika, I. (2004). Comparison of vehicle types at an automated container terminal. *OR Spectrum, 26*(1), 117-143.

Vis, I. F. (2006). A comparative analysis of storage and retrieval equipment at a container terminal. *International Journal of Production Economics, 103*(2), 680-693.

Woo, Y. J., & Kim, K. H. (2011). Estimating the space requirement for outbound container inventories in port container terminals. *International Journal of Production Economics, 133*(1), 293-301.

Yang, C., Choi, Y., & Ha, T. (2004). Simulation-based performance evaluation of transport vehicles at automated container terminals. *OR Spectrum, 26*(2), 149-170.

Zeng, J., & Hsu, W.-J. (2008). Conflict-free container routing in mesh yard layouts. *Robotics and Autonomous Systems, 56*(5), 451-460.

Zhang, C., Wu, T., Zhong, M., Zheng, L., & Miao, L. (2014). Location assignment for outbound containers with adjusted weight proportion. *Computers & Operations Research, 52*, 84-93.

Zhen, L., Lee, L. H., Chew, E. P., Chang, D.-F., & Xu, Z.-X. (2012). A comparative study on two types of automated container terminal systems. *IEEE Transactions on Automation Science and Engineering, 9*(1), 56-69.

Zhou, C., Chew, E. P., Lee, L. H., & Liu, D. (2016). An introduction and performance evaluation of the GRID system for transshipment terminals. *Simulation, 92*(3), 277-293.

Appendix 1 Candidate's Publications

JOURNALS

Zhou, C., Chew, E. P., Lee, L. H., & Liu, D. (2016). An introduction and performance evaluation of the GRID system for transshipment terminals. *Simulation*, 92(3), 277-293. *Link: https://doi.org/10.1177/0037549715623845*

Zhou, C., Chew, E. P., & Lee, L. H. Information-based allocation strategy for grid-based transshipment automated container terminal. *Transportation Science*, Articles in Advance. *Link: http://dx.doi.org/10.1287/trsc.2017.0736*

Zhou, C., Lee, L. H., Chew, E. P., & Poh, Z. C. A heat-map based simulation embedded optimization approach for path direction problem in grid system. *Working paper.*

CONFERENCE PROCEEDINGS

Zhou, C., Chew, E. P., & Lee, L. H. (2015, December). A comprehensive study on new conceptual container handling system. In *Proceedings of the 2015 Winter Simulation Conference* (pp. 3094-3095). IEEE Press.

Lee, L. H., Chew, E. P., Jiang, X., and **Zhou, C.** (2014). A simulation study for next generation transshipment port. In Proceedings of the 2014 Winter Simulation Conference, pp. 1831-1842. IEEE Press, 2014.

CONFERENCE PRESENTATIONS

Zhou, C., Chew, E. P., & Lee, L. H. (2016). Optimal path design for mesh-like structure in GRID system. Talk presented at the 2016 International Conference on Logistics and Maritime Systems, Sydney, Australia.

Zhou, C., Chai, Y., Chew, E. P., & Lee, L. H. (2015). Traffic rule design for GRID system. Talk presented at the 2015 International Conference on Logistics and Maritime Systems, Hong Kong, China.

Zhou, C., Chew, E. P., & Lee, L. H. (2014). System evaluation and storage allocation problem in grid-act based transshipment hub. Talk presented at the 2014 International Conference on Logistics and Maritime Systems, Rotterdam, Netherlands.

Zhou, C., Chew, E. P., & Lee, L. H. (2015). A Comprehensive study on new conceptual container handling system. Talk and poster presented at the Winter Simulation Conference 2015, Huntington Beach, CA, United States.

Zhou, C., Chew, E. P., & Lee, L. H. (2014). System evaluation and storage allocation problem in grid-act based transshipment hub. Talk presented at the 2014 International Conference on Logistics and Maritime Systems, Rotterdam, Netherlands.