Multi-objective optimization for stowage planning of large containership

Liu, Fan

2012

Liu, F. (2012). Multi-objective optimization for stowage planning of large containership.
Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/50631>

<https://doi.org/10.32657/10356/50631>

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

**MULTI-OBJECTIVE OPTIMIZATION FOR STOWAGE
PLANNING OF LARGE CONTAINERSHIP**

**LIU FAN
SCHOOL OF COMPUTER ENGINEERING
2012**

**MULTI-OBJECTIVE OPTIMIZATION FOR STOWAGE
PLANNING OF LARGE CONTAINERSHIP**

LIU FAN

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement for the degree of
Master of Engineering

2012

Acknowledgments

First of all, I would like to use this opportunity to express my sincere gratitude to my supervisor, Assistant Professor Low Yoke Hean, Malcolm, for offering me the opportunity to study and work in this project. His great guidance through the research and knowledge helped me a lot in this project. Without his constant support and patience, it would have been impossible for me to carry out this research work.

Secondly, I would like to thank Dr. Hsu Wen Jing, Dr. Huang Shell Ying, the teammates of this project Mr. Xiao Xiantao, Ms. Zeng Min, Mr. Win Cho Aye and our industry partners Mr. Nigam Tong Kin, Mr. Sew Kait Thong and Mr. Falan Chan for their continuous support and help to me in this research project.

Next, I specially thank the laboratory technicians Ms. Ng-Goh Siew Lai and Mr. Poliran Kenneth Caballes for their help to me in the daily research work.

Great thanks to all the colleagues in PDCC, I have been blessed with the friendly attitude of them since the first day I joined PDCC.

Finally, I heartily thank my father Mr. Liu Wenmin and mother Ms. Liu Xiaojin for supporting me through my study and work with their endless love and encouragement.

Contents

Acknowledgments	i
List of Figures	v
List of Tables	vi
Abstract	vii
Chapter 1 Introduction.....	1
1.1 Motivation	1
1.2 Objective and Scope.....	3
1.3 Organization of the Thesis	4
Chapter 2 Background and Literature Review	6
2.1 Structure of Containerships.....	6
2.2 Constraints and Objectives in Stowage Planning.....	8
2.2.1 Size of Containers	8
2.2.2 Weight of Containers	9
2.2.3 Distribution of Containers	9
2.2.4 Re-handles	10
2.2.5 Hatches.....	13
2.2.6 Quay Cranes' Utilization	13
2.3 Review on Related Works.....	16
2.4 Multi-Objective Optimization Problem of Large Containership Stowage Plans	19

2.4.1	Pareto Dominance	19
2.4.2	Pareto Optimality	20
2.4.3	Pareto-Optimal Set.....	20
2.4.4	Objectives considered in this project	21
2.5	Tabu Search Methodology	22
Chapter 3	System Framework	24
3.1	Initialization Module	24
3.2	Basic Plan Generation Module.....	26
3.3	Weight Adjustment Module	27
3.4	Final Stowage Plan Module	27
Chapter 4	Block Stowage Heuristics	30
4.1	Concept	30
4.2	Group the Containers	31
4.3	Partition the Containership.....	33
4.4	Block Selection	35
4.4.1	Loading Sequence of Containers Groups	35
4.4.2	Block Filtering Procedure	35
4.5	Stowing a Block	37
Chapter 5	Deterministic Block Selection Mechanisms	40
5.1	Block Ranking.....	40
5.2	Crane Intensity Control	42
Chapter 6	Randomized Block Stowage Algorithm with Tabu Search	49
6.1	Initial Stowage Patterns Generation Stage	50
6.2	Solutions Optimization by Tabu Search	51
Chapter 7	Experiment and Discussion	57
7.1	Introduction to the Experiment	57
7.2	Design of Tests.....	59

7.3	Results and Discussions	60
7.3.1	Experiments Results	60
7.3.2	Discussion.....	63
Chapter 8	Conclusions and Future Work	67
8.1	Conclusions	67
8.2	Recommendations for Future Work.....	69
8.2.1	Parallel Computing	69
8.2.2	Multi-Objective Evolutionary Algorithms.....	69
8.2.3	Branch and Bound Algorithms	70
Appendices.....		72
Appendix A	Publication List for this Research	73
Appendix B	Screenshots of Stowage Plans and the Corresponding Gantt Charts of Quay Cranes Generated by Automated Stowage Planning System and the Human Planner	75
Bibliography		89

List of Figures

Figure 1 : Structure of a Containership.....	7
Figure 2 : Structure of a Bay	8
Figure 3 : Forced Re-handles.....	11
Figure 4 : Stowage Plan before Loading Operation at Port 2	11
Figure 5 : Stowage Plan after Loading Operation at Port 2(Option 1)	12
Figure 6 : Stowage Plan after Loading Operation at Port 2(Option 2)	13
Figure 7 : Re-handles that Involve Hatch Movements	14
Figure 8 : Framework of the Automated Stowage Planning System.....	25
Figure 9 : Gantt Chart of Quay Cranes	28
Figure 10 : Flowchart of Block Stowage Heuristics.....	31
Figure 11 : Partition a Bay	33
Figure 12 : Stowing Algorithm for a Block.....	38
Figure 13 : Function of Crane Intensity Control Mechanism	43
Figure 14 : Example for Crane Intensity Control Module (a)	45
Figure 15 : Example for Crane Intensity Control Module (b)	46
Figure 16 : Example for Crane Intensity Control Module (c)	47
Figure 17 : Example for Crane Intensity Control Module (d)	48
Figure 18 : Example for Crane Intensity Control Module (e)	48
Figure 19 : Container Swapping only Prevent Reverse Swap	53
Figure 20 : Container Swapping with Tabu List Updated after Each Swap	54

List of Tables

Table 1 : Block Filtering Procedure.....	37
Table 2 : Pseudo Code of the Tabu Search Algorithm	55
Table 3 : Statistics of Containers Onboard before Planning at Port B for Test Case 1	58
Table 4 : Statistics of Container Load List at Port B for Test Case 1	58
Table 5 : Statistics of Containers Onboard before Planning at Port D for Test Case 2.....	59
Table 6 : Statistics of Container Load List at Port D for Test Case 2.....	59
Table 7 : Objective Values of Top 20 Solutions Obtained after the First Stage for Test Case 1	60
Table 8 : Objective Values of Plans Generated by Randomized Block Stowage Algorithm with Tabu Search, Deterministic Block Stowage Strategies and Human Planner at Port B for Test Case 1	61
Table 9 : Objective Values of Top 20 Solutions Obtained after the First Stage for Test Case 2	63
Table 10 : Objective Values of Plans Generated by Randomized Block Stowage Algorithm with Tabu Search and Deterministic Block Stowage Strategies Test Case 2.....	64

Abstract

The Master Bay Plan Problem (MBPP), which is to decide the stowage plan of large containerships, has been studied since 1990. Since the size of containerships has been growing dramatically from 1970's, this problem becomes one of the most important issues in shipping lines as the quality of a stowage plan affects the profit of the shipping lines significantly. Many different research works have been carried out to solve this problem, including mathematical modeling, decision support systems, rule-based expert systems and heuristic driven approaches. However, none of these works provide a satisfactory solution to the stowage planning problem. The focus of this project is to develop an automated multi-objective optimization stowage planning system which is capable of generating optimized stowage plans for large containerships in a short amount of time.

Optimized stowage plans here refer to: (1) the number of unnecessary shifts of containers is minimized, hence the operation cost is reduced; (2) the amount of workload (both loading and unloading of containers) across the ship is well distributed for every port, thus the efficiency of quay cranes for handling the containers is improved; (3) the number of unusable container slots is minimized, therefore there are more space to load other containers; (4) the weight distribution of containers on the containership is well arranged, so that the amount of ballast used to balance the ship is reduced and the fuel oil requirement is reduced. The optimality of these objectives is difficult to be

obtained simultaneously for one single stowage plan. Therefore instead of aiming to generate an optimal stowage plan for a containership, in this project a randomized block stowage algorithm with Tabu Search is proposed to obtain a set of stowage plans that emphasize on different objectives for the shipping lines, and help them make better decisions in the real world.

An automated stowage planning system is developed in this project. A randomized block stowage algorithm with Tabu Search is proposed and implemented in this system. The basic idea of the block stowage algorithm is to divide the containers in groups and partition the containership into blocks. The algorithm breaks the stowage planning process into two stages: (1) obtaining a set of stowage plans by selecting different blocks combinations in a randomized manner; (2) applying Tabu Search to each of the stowage plans to adjust every container's location in order to achieve an optimized final stowage plan.

The research issues of this project include: (1) designing and implementing an automated stowage planning system; (2) designing and implementing a basic block stowage algorithm to stow groups of containers into blocks; (3) designing and implementing different deterministic block stowage strategies; (4) designing and implementing a randomized block stowage algorithm with Tabu Search for multi-objective optimization of large containerships.

Chapter 1

Introduction

1.1 Motivation

Transportation of cargos has been facilitated increasingly by containerization since 1970s. Nowadays over 60 percent of deep-sea general cargo are transported in containers. Some routes, especially between economically strong and stable countries, are containerized up to 100 percent [1]. There are many shipping lines around the world competing to provide better container transportation service. In order to satisfy the growing shipping demands of the customers and also achieve higher operation profits, the size of containerships has increased dramatically since three decades ago. The size of containerships has grown from relatively small 350 TEUs (Twenty Foot Equivalent Unit) to 10000 TEUs. However, the increase in the size of containerships does not simply increase the profit of shipping lines; it also increases the complexity and difficulty of the arrangement of containers on a containership. Shipping lines are facing increasing challenge in generating feasible and economical stowage plans for large containerships while they sail from one port to another. The problem of arranging containers for large containerships is defined as the Master Bay Plan Problem (MBPP) by Ambrosino and Sciomachen in 2004 [2].

There are several objectives in the Master Bay Plan Problem (MBPP). Examples of these objectives include minimizing the total number of shifting of containers, minimizing containership's berthing time at terminals, balance the weight distribution of containers on board, maximizing the available space on

board, maximizing containership's utilization and prevent damages. MBPP is a difficult problem because of its combinatorial nature and the various operating constraints related to both the containership structure and container properties. It has already been proven to be a NP-complete problem [6] [8], which means that it is difficult or even impossible to guarantee an optimal solution in a reasonable amount of processing time for problems with large number of variables. In recent operations research and management science literatures, there are basically four main classes of methods to solve this problem: mathematical modeling, decision support systems, rule-based expert systems and heuristic driven methods.

Presently, the planning process in all major shipping lines is carried out by human planners. The human planners use computer aided planning software to help them in generating and fine tuning stowage plans for different containerships at different ports. A good stowage plan has to take into consideration numerous factors including container type, weight capacity of ship, ship stability and the port of origin and destination of containers. However, the function of the computer aided planning software is quite simple and limited, and they only check some hard constraints of the stowage plan, such as ship stability, line of visibility, stack limit and hazardous cargo safety issue. Therefore the main planning work of deciding which slot to stow which container is still carried out by the human planners. The quality of stowage plans would largely depend on the experience and performance of the human planners. Such experienced planners are in great shortage. To become a qualified ship planner, one must have several years of on-the-job training experience onboard containerships.

Because of the dramatically increase in capacity of containerships, and the increase in worldwide shipping volume, shipping lines all over the world are facing greater challenge to cope with the round-the-clock demand for generating efficient and stable stowage plans for containerships when they call between ports. The difficulties in hiring qualified professional ship planner for a seafaring career and maintaining a larger team of stowage planners also make the manual process of generating stowage plans increasingly difficult.

Therefore a computerized stowage plan generator would help shipping lines to save time, costs and increase safety by generating more plans. Hence, developing effective and efficient optimization methods for generating stowage plans will be useful in helping shipping lines to cope with the Master Bay Plan Problem.

1.2 Objective and Scope

In this project, an automated stowage planning system for large containerships is proposed and implemented. This system includes the following modules: (i) Initialization module which initializes the system for getting ready to start planning a containership; (ii) Basic Plan Generation module which generates a basic stowage plan for further refinement; (iii) Weight Adjustment module which optimizes the stowage plan by adjusting the containers' weight distribution on the containership; and (iv) Final Stowage Plan module which evaluates the objectives of the stowage plan and outputs them together with the containers and their stowage locations.

Since the stowage planning problem is a large scale multi-objective optimization problem. The objective of this project is to develop an automated stowage planning system for large containerships, which can generate stowage plans through multi-objective optimization in a short time period. This can be achieved by: (i) applying block stowage heuristics; (ii) employing proper stowage strategies; and (iii) making use of local search techniques.

By applying block stowage heuristics, the solutions space can be reduced significantly. Block stowage heuristics shorten the computation time by assigning containers to locations group by group instead of assigning containers to locations one by one. By employing a proper stowage strategy when assigning groups of containers to blocks of stowage locations, better basic stowage plans can be obtained through multi-objective optimization. Local search techniques can be used to adjust the location of each individual container on containership to obtain a better stowage plan considering the different objectives to be optimized.

To sum up, in order to achieve the objective of this project, an automated stowage planning system has to be designed and implemented. The block stowage heuristics must be proposed and implemented in this system. Different stowage strategies will be proposed and investigated in this project. A Tabu Search algorithm optimizing the stowage plans will be addressed in this thesis.

1.3 Organization of the Thesis

The rest of this thesis are organized as follows:

- **Chapter 2** first describes the structure of a containership and illustrates how containers are stowed onboard. It also explains the constraints faced by stowage planners when generating stowage plans for large containerships. The objectives being considered in this project are explained in this chapter. The related research works in MBPP are then reviewed in this chapter, which provides a more detailed understanding of this problem. The principles of multi-objective optimization and Tabu Search methodology are explained in this chapter as well.
- **Chapter 3** describes the framework of the proposed automated stowage planning system. It contains: 1) Initialization module; 2) Basic Plan Generation module; 3) Weight Adjustment module and 4) Final Stowage Plan module.
- **Chapter 4** discusses the proposed basic block stowage heuristics in the automated stowage planning system. This block stowage heuristics is mainly focused on optimizing the number of re-handles and the containership berthing time for the stowage plans.
- **Chapter 5** discusses the two proposed deterministic block selection mechanisms. They are the: 1) block ranking mechanism and 2) crane intensity control mechanism. In the block ranking mechanism, two approaches: 1) *Workload-Base* approach and 2) *Distance-Base* approach are proposed that emphasize on optimizing the containership berthing time and the containership stability issue respectively. In crane intensity control mechanism, two approaches: 1) *Re-handle Driven* approach and 2) *Crane Intensity Driven* approach are proposed that emphasize on optimizing the number of re-handles and the containership berthing time respectively.

- **Chapter 6** discusses the proposed randomized block stowage algorithm with Tabu Search. It first explains the randomized initial solutions generation stage. After that, the Tabu Search algorithm for the solutions optimization stage is described.
- **Chapter 7** presents experiments containing two test cases to compare the stowage plans generated by the automated stowage planning system by applying different stowage strategies and stowage plans generated by a human planner. Discussion about the performance of the proposed strategies is provided in this chapter.
- **Chapter 8** summarizes the work that have been carried out in this project, and suggests some future work that can be explored for this project.

Chapter 2

Background and Literature Review

2.1 Structure of Containerships

The basic structure of a containership is shown in Figure 1 and Figure 2. The space for stowing containers in a containership consists of a given number of locations (the squares in Figure 1 and Figure 2) which can vary depending on the size of the containerships. However, the notation of each individual location and the structure of containerships are basically the same. Each location is identified by three indices with each consisting of two digits giving its position with respect to three dimensional x , y and z coordinates on the containership. In particular, each location is addressed by the following identifiers: (a) *bay*, which gives its position to the cross-section of the containership (counted from bow to stern); (b) *row*, which gives its position relative to the vertical section of the corresponding bay (counted from center to the outside of the ship); (c) *tier*, which gives its position related to the horizontal section of the corresponding bay and row (counted from the bottom to the top of the containership). Therefore, a location in a containership can be represented by a given bay, a given row and a given tier together.

Moreover, as can be seen from Figure 1, each 20Ft bay is numbered with an odd number, i.e. bay 01, 03, 05, etc., while two contiguous odd bays conventionally form one even bay, that is a 40Ft bay. For example, bay 02=bay 01+ bay 03. Though the effective even bays depends on the particular structure of the containership under consideration, each even bay must be associated with

two contiguous odd bays. For the index of row, if a location on containership is located on the starboard side, it is assigned with an odd number, i.e. row 01, 03, 05; if a location on containership is located on the port side, it will be assigned with an even number, i.e. row 02, 04, 06, etc. Finally, the index for tier are numbered from the bottom of the cargo hold to the top with even number, i.e. tier 10, 12, 14, etc., while in the upper deck the numbers start from 82, and increase with the levels, i.e. tier 82, 84, 86 etc. Thus, the shadowed location in Figure 2 is represented as 140484. Usually, from the tier number, the locations in a cargo hold can be easily distinguished from those on the upper deck. In this project, tier number less than or equal to 80 denotes locations in the cargo hold, while tier number greater than or equal to 82 denotes locations on the upper deck. Locations in the same bay and same row under deck or above deck form a stack respectively.

Figure 1 : Structure of a Containership

There are removable separators between cargo holds below deck and locations above deck, which are usually composed of a number of sections that interlock latitudinally. Each section is called a hatch in maritime term. A hatch usually lies across several rows and separates tiers in the cargo hold from those on the

upper deck. In one bay there are usually 2 to 3 hatches. In Figure 2, the 3 solid dark lines in the middle are hatches. The left hatch covers rows 04 and 06, the center hatch covers rows 01 and 02, and the right hatch covers rows 03 and 05.

Figure 2 : Structure of a Bay

2.2 Constraints and Objectives in Stowage Planning

Solving MBPP means determining a stowage configuration with certain objectives taking into consideration both the constraints related to the structure and the operational requirements of the particular containership under consideration and those of the containers. The following sections will elaborate on these constraints.

2.2.1 Size of Containers

There are two standard size of containers considered here in this project, which are 20 and 40 feet in length with a section of 8 feet by 8 feet and 6 inches. Moreover, the capacity of a containership is expressed in term of TEU (Twenty-foot Equivalent Unit). Each TEU represents a container/location that is 8 feet in width, 8 feet and 6 inches in height and 20 feet in length. Thus, a 40 feet container is equivalent to two TEUs.

A 40Ft container require two contiguous bow-stern 20Ft locations to stow, which indicates that 40Ft containers can only be stowed in even bays (for

example Bay 06 in Figure 1). As a consequence, the locations of the same row and tier corresponding to two contiguous odd bays (e.g. Bay 05 and Bay 07) are no longer available for stowing 20Ft containers.

Moreover, for safety reason it is required that 40Ft containers cannot be stacked either below locations where 20Ft containers are already stowed or above empty location, while 20Ft containers cannot be stowed above 40Ft containers or empty locations.

A high-cube container is a sub-class of 40Ft containers. It is with the same width and length of a normal 40Ft container, but is 1 feet longer in height compared to a normal 40Ft container. If a high-cube container is stowed in a row under deck, the slot in the top tier under deck of that row must be left empty otherwise the hatch cover cannot be closed.

2.2.2 Weight of Containers

Normally the weight of containers varies from 2 tons to over 30 tons, which depends on the size of containers and the goods inside them. There are two main constraints related to the weight of containers: 1) the total weight of all the containers on board cannot exceed the maximum weight capacity of the containership; 2) the total weight of the containers in the same stack cannot be greater than a predefined stack weight limit, which is determined by the size of containers stowed in that stack. Making sure that no stack exceeds the stack weight limit before the containership leaves the port is one of the objectives in MBPP.

2.2.3 Distribution of Containers

There are some constraints related to a proper containers' weight distribution over the containership, which is the basic requirement for feasible stowage plan. The difference of weight distribution across the horizontal dimension of the ship can result in a heel angle for the ship. A large heel angle will have safety implication when the ship is sailing. The difference of weight distribution across the longitudinal dimension causes a trim for the ship. For safety and fuel efficiency considerations, a target trim is desired for sailing. In this project, we

assume the target trim is zero. Therefore, when the loading operation of a containership is completed, the horizontal and cross moment equilibrium must be verified. Horizontal moment equilibrium ensures that the difference of moment caused by containers' weight between the starboard side (right) of the containership (including the odd rows on and under deck) and the port side (left) of the containership (including the even rows on and under deck) is as close to zero as possible. Cross moment equilibrium ensures that the moment caused by the weight of containers on the bow part (front) of the containership, including all the locations on and under deck, is as close as possible to the moment caused by the weight of containers on the stern part (back) of the containership.

2.2.4 Re-handles

Reducing the number of re-handles is one the important objectives associated with MBPP. A re-handle is a kind of container movement which is only performed in order to create access to another container, or to improve a stowage plan to take into account expected loads at downstream ports. There are basically two kinds of re-handles, which are described as follows.

- **Forced Re-handles**

The concept of forced re-handles is illustrated in Figure 3. The letters m and n of the notation $m-n$ used in Figure 3 represents the port of origin and the port of destination respectively, which means that the container is loaded onto the containership at *port m* and will be discharged at *port n*. This notation is used consistently throughout all figures for the rest of the chapter.

In Figure 3, the container labeled with “2-3” is required to be discharged at *port 3*. However, the container labeled with “2-4” is stacked upon it. Therefore, the container labeled with “2-4” has to be unloaded first before the container labeled with “2-3” can be discharged. After container labeled with “2-3” is discharged, the container labeled with “2-4” has to be loaded back to the containership. This kind of container movement is defined as forced re-handle.

Figure 3 : Forced Re-handles

Figure 4 : Stowage Plan before Loading Operation at Port 2

▪ Voluntary Re-handles

In order to illustrate the concept of voluntary re-handles, consider a loading scenario showed in Figure 4 at the current port, *port 2*.

Three 40Ft containers labeled with “2-4” are to be loaded at *port 2*. A container labeled with “1-3” is a 40Ft container loaded at *port 1* and is going to be discharged at *port 3*. The three 40Ft containers to be loaded at *port 2* could be loaded as in Figure 5. Then when the containership arrives at *port 3*, there will be 3 forced re-handles incurred because of the unloading of the container with label of “1-3”.

Figure 5 : Stowage Plan after Loading Operation at Port 2(Option 1)

There is another option to stow the three 40Ft containers, which is shown in Figure 6. If this option is applied, the container with the label of “1-3” must be unloaded first at *port 2*, then the 3 containers with the label of “2-4” can be loaded to the containership. Finally the container labeled with “1-3” can be loaded back. This operation reduces 3 forced re-handles at *port 3*, but creates 1 another re-handle at *port 2*. This kind of re-handles is defined as voluntary re-handles. Performing voluntary re-handle at the current port may prevent forced re-handle at downstream ports.

Figure 6 : Stowage Plan after Loading Operation at Port 2(Option 2)

2.2.5 Hatches

Because of the structure of hatches, sometimes re-handles may not only happen in one stack. For example, Figure 7 shows the stowage configuration for rows 01 and 02 in Bay 06 at *port 3* before the unloading operation. The two containers labeled with “1-3” in row 01 under deck have to be discharged at this port. Because both rows 01 and 02 are covered by the same hatch, in order to unload the two containers in row 01 under deck, the hatch must be opened first. As rows 01 and 02 above deck are both on the same hatch, the four containers labeled with “2-4” must be removed first before opening the hatch. Therefore, in order to discharge the two containers in row 01 under deck, the four containers in rows 01 and 02 above the same hatch must be re-handled.

In stowage planning, one of the objectives is to reduce the number of re-handles due to the opening of hatches, which is included in the overall objective of reducing the total number of re-handles.

2.2.6 Quay Cranes' Utilization

At each port, the containership will be served by a given number of (usually 3-5) quay cranes to unload and load containers. Each crane will be assigned to work in several bays. Because of the physical size of quay cranes, for operating

safety, there should be a safety distance between two adjacent operating cranes. The safety distance is determined by the port in accordance to the size of their quay cranes.

Figure 7 : Re-handles that Involve Hatch Movements

In this project the safety distance is defined as follows. If a crane is working in bay i , a neighboring crane may only work in bay $i \pm 8$ or even further. Therefore, if the operating areas of two adjacent cranes are too close, the situation of crane conflict arises and one crane has to wait until the other crane finishes its work and moves to the other bay further enough. The waiting time of a crane is called “idle time”. Too much “idle time” will result in a low utilization of cranes which may lengthen the vessel’s berthing time. A perfect crane workload allocation (or crane split) is the case where all cranes finish their work at the same time with minimum idle time.

The quality of a crane split is measured by the crane intensity (C.I.).

Let $T_{i-worktime}$ – denote the total time crane i spent on loading or unloading containers.

Let $T_{i-idletime}$ – denote the total time crane i spent on waiting due to crane conflict.

Let $T_{i-movetime}$ – denote the total time crane i spent on moving from one bay to another.

As cranes may serve multiple bays, sometimes they have to move to another bay after finishing loading or unloading containers in one bay.

Let $T_{i-complete}$ – denote the time crane i used to complete the loading and unloading of all the containers assigned to it. Thus,

$$T_{i-complete} = T_{i-worktime} + T_{i-idletime} + T_{i-movetime}.$$

Let T_{max} – denote the completion time of the longest crane.

Thus, $T_{max} = \max\{T_{i-complete} | i = 1, 2 \dots n\}$ across the n cranes allocated to the ship in the port.

The crane intensity, C.I., is thus defined as,

$$C.I. = \frac{\sum_{i=1}^n T_{i-worktime}}{T_{max}}$$

The duration a ship berthed in a port depends on the completion time of the longest crane and is mainly decided by the crane split. A good C.I. (close to the number of cranes allocated) indicates that the workloads of the cranes are evenly distributed with minimum idle and movement time.

In this project, instead of using C.I., T_{max} is chosen to evaluate the utilization of quay crane. This is because the total time that all quay cranes spend on unloading and loading containers are the same for different stowage plans, shorter T_{max} results in higher C.I. Observing T_{max} is a more straightforward way than calculating C.I.

2.3 Review on Related Works

Since the 1970s, the problem related to container stowage planning has been studied by shipping lines and researchers. The stowage planning problem is mainly referred to as the container loading problem, which is to decide the stowage configuration of a containership.

Initially the researchers mainly focused on establishing a set of 0-1 linear programming formulations which can express the stowage planning problem including all the constraints in a mathematical model. Theoretically if the set of linear programming formulations are well defined, an optimal solution can be obtained. However, the search space of the established mathematical model depends on the ship capacity, the number of containers being considered and the operational constraints imposed by the shipping company and container terminal at each port. Even for a medium size containership, e.g. a 2000 TEUs vessel, the problem becomes a non-trivial one due to the large number of variables and inequalities needed for the model.

The stowage planning problem has been proven to be NP-complete and is related to the circle graphs coloring problem (Avriel et al., 1998, 2000) [6] [8]. They showed that finding the minimum number of rows in one bay for stowing containers with different destinations with no re-handle is equivalent to finding the minimum number of colors needed to color circle graphs without color overlapping, which is a famous NP-complete problem. It means that it is very hard or even impossible to guarantee an optimal solution in a reasonable processing time for a real commercial containership. Thus, researchers have been trying to develop heuristic algorithms to provide workable solutions. A brief review of recent research follows.

The early study about the container loading problem can be traced back to the work by Aslidis in 1989 [1] and 1990 [4]. The author mainly focused on the problem related to the stacks. He developed an algorithm to calculate the number of re-handles, which is a mathematical model that reads in stowage configuration, then outputs the number of re-handles. This approach is more

like a criterion to measure the efficiency of a stowage plan, and not a method to generate stowage plans. Aslidis also proposed a heuristic algorithm to minimizing them. He suggested grouping the containers together according to their ports of destination, and then stowing them close to each other. However, his work only considered some special and small size cases, and also ignored the stability problem which is a very critical issue in the stowage planning problem. Though his approach cannot be used to solve MBPP, his heuristic has been thought useful by many researchers in this area.

The first reported attempt to derive some rules for determining good container stowage plans was made by Ambrosino and Sciomachen (1998) [1], where a constraint satisfaction approach is used to define and characterize the search space of feasible solutions.

In their follow-up work (Ambrosino and Sciomachen, 2004) [2], they described a 0-1 linear programming model for MBPP. They presented a heuristic approach before performing a 0-1 linear programming model, which consists of a set of heuristic preprocessing and pre-stowing procedures that allow the relaxation of some constraints of the exact model to reduce the searching space of the model. Based on the pervious works they proposed a three-phase algorithm for MBPP, which splits the ship into different portions and associates grouped containers with different subsets of bays without specifying their actual positions. Subsequently, they assign the actual position to each container by solving a 0-1 linear programming model. In the last phase, some local search exchanges are performed to check and remove the infeasible solutions due to the cross and horizontal stability issues. However they assumed that the ship starts its journey empty at a port and visits a given number of other ports where only unloading operations are allowed. This means that the loading problem is only considered at the first port. This assumption is also un-realistic. Also, as a 0-1 linear programming approach is used in this algorithm, the computation time is still high, about 20 minutes for one plan, for a small size problem (containership of 200 TEUs). Moreover, this research also erred in not providing a constraint that prevents free hanging containers in the mathematical model.

Avriel and Penn (1993) [6] described a heuristic called the “Whole Column Heuristic Procedure”. Through this model an optimal solution can be obtained. However, this heuristic involves binary linear programming. Although after some pre-processing of the data the size of the problem is smaller than the original exact model, it is still too large to be solved. Consequently, they developed another heuristic, called the “Suspensory Heuristic” (SH) Procedure [7], to solve this problem with the aim of reducing the number of re-handles. This heuristic procedure provided very impressive performance in term of computation time. However, the algorithm did not take stability issue into account, and also all the containers considered are of the same size, and no special containers (e.g. reefers, high cubes) are considered. These assumptions make the Suspensory Heuristic algorithm not flexible and thus cannot be used to solve the real-world stowage planning problem.

Wilson and Roach (1999, 2000) [15] [21] developed a methodology for generating computerized stowage plan. They break the stowage planning process into two sub-steps, called strategic and tactical levels, respectively. First they use branch-and-bound algorithms to solve the problem of assigning generalized containers to a bay’s block in a vessel. In the second step they use local search algorithm to assign specific locations for specific containers. Their approach is able to find a solution but optimality is not necessarily achieved. They only showed a solution of small sample problem in their paper, which takes 2 hours to obtain a feasible stowage plan for a 688 TEUs containership.

Dubrovsky et al. (2002) [11] used a genetic algorithm technique for stowage planning to minimize the number of re-handles. The authors developed a compact and efficient encoding of solutions to reduce the search space significantly. However, in their test case they used a containership that has only a single bay with 100 rows and 10 tiers without hatch cover. All the containers in their test cases are of homogenous weight. Because of this simplification, it becomes easier to make a balanced weight distribution. Also, the longitudinal balance issue was neglected in their research, and the algorithm requires 30 minutes to obtain a feasible (not optimum) stowage plan.

A recent research was carried out by Delgado and Jensen (2009) [9]. In this work, they applied Constraints Programming (CP) to solve the stowage planning problem. They reported that the CP approach outperformed an integer programming and column generation approach in a preliminary study. However, they only tested their approach in a single under deck bay of a containership. This simplistic test ignores the generality of the stowage planning problem and so cannot be applied to normal scenarios.

2.4 Multi-Objective Optimization Problem of Large Containership Stowage Plans

MBPP of large containerships is inherently a multi-objective optimization problem that some of the objectives can be conflicting (such as simultaneously minimizing the number of re-handles and maximizing the quay cranes' utilization). Since no single solution can be termed as an optimum solution to multiple conflicting objectives [15], the goal of multi-objective optimization problem is to look for a set of solutions which represents trade-offs with respect to the constraints. However, all the research work mentioned in Section 2.3 simplified this requirement. They either consider only one objective, the number of re-handles, or designed a single objective function which associated a weight to each objective factor that under consideration of the problem and obtain a single value to evaluate the solution. Using such methodology, a solution with optimum value in term of a single objective function may not be the optimal in term of multi-objective optimization. Therefore in this project, MBPP of large containerships is solved in term of multi-objective optimization. In the following sections, the concepts of Pareto Dominance, Pareto Optimal and Pareto Set in multi-objective optimization are discussed.

2.4.1 Pareto Dominance

Pareto Dominance is an important concept applied in multi-objective optimization problem as it provides means to distinguish good solutions from the bad ones in the presence of multiple conflicting objectives. The dominance relation between a pair of solutions is defined as follows,

Let x and y be two arbitrary solutions for a multi-objective optimization problem. Objectives vectors $F(x) = \{f_1(x), \dots, f_k(x)\}$ and $F(y) = \{f_1(y), \dots, f_k(y)\}$ are obtained using solution x and solution y . Thus solution x is considered to dominate solution y (also written as $x \succcurlyeq y$) if and only if

$$\forall i \in \{1, \dots, k\}: f_i(x) \leq f_i(y) \wedge \exists j \in \{1, \dots, k\}: f_j(x) < f_j(y)$$

It means that each objective value of x is no greater than the corresponding value of y and there exists at least one objective value of x is strictly less than that of y .

2.4.2 Pareto Optimality

The concept of Pareto Optimality was introduced by Vilfredo Pareto in the 19th century. It is defined as follows,

Let x be an arbitrary solution and X be the complete solutions space. The solution x is Pareto Optimal if and only if

$$\nexists x' \in X \mid x' \succcurlyeq x$$

It means if solution x is said to be Pareto Optimal, there is no solution in the whole solutions space that dominates it.

2.4.3 Pareto-Optimal Set

The goal of multi-objective optimization is to obtain a set of Pareto optimal solutions that cannot be improved in any objective without causing degradation other objectives. The concept of Pareto-Optimal Set is first introduced by K. Deb [17] in 1999.

For a given multi-objective optimization problem and set of solutions $X' \in X$ where X is the complete solutions space, X' is considered as a Pareto-Optimal Set when

$$\forall x' \in X' \wedge \nexists x \in X : x \succcurlyeq x'$$

It means each solution in Pareto-Optimal set X is non-dominated to any solution in the complete solutions space.

2.4.4 Objectives considered in this project

In this project, the following objectives in MBPP for large containerships are being considered for multi-objective optimization.

- Number of re-handles

Section 2.2.4 explains how re-handles occurs during unloading and loading a containership. This kind of operations brings additional quay crane movements for containers, which results in extra cost in both time and money. Therefore, reducing the number of re-handles is one of the key objectives in this research work.

- Completion time of the longest quay crane

Crane Intensity (C. I.) is introduced in Section 2.2.6 to measure the utilization of quay cranes. As the completion time of the longest quay crane determines C.I., minimizing the completion time of the longest quay crane is one important objective in MBPP of large containerships.

- Number of stacks that exceed the weight limit

Refer to Section 2.2.2.

- Number of slots killed

A slot killed represents an unusable empty stowage location on containership. It occurs in three different situations. First, a stack cannot be completely filled due to the weight of containers in that stack which has reached the stack weight limit. Second, if a stack under deck stows at least one high-cube container, the slot in the top tier of that stack must be left empty otherwise the hatch above cannot be closed. Third, for the empty slots under a hatch with containers stowed on top, in order to access these empty slots, extra re-handles are needed. One of the objectives in MBPP of large containerships is to maximize the use of all

the stowage locations on a containership and hence to minimize the number of slots killed.

- Horizontal moment difference

Refer to Section 2.2.3.

- Cross moment difference

Refer to Section 2.2.3.

2.5 Tabu Search Methodology

Tabu Search is a meta-heuristic that guides a local heuristic search procedure to explore the solution space beyond the local optimality. The basic form of Tabu Search is founded on ideas proposed by Fred Glover (1986) [12]. It has become an established optimization approach which has been successfully applied to obtain optimal or sub-optimal solutions to problems such as scheduling, time-tabling, travelling salesman, and layout optimization. The principal method, described by Glover, Taillard and de Werra (1993) [14], is to explore a search space of feasible solutions by making a sequence of moves, which can avoid being trapped by the boundaries of feasibility or local optimality instead of taking them as barriers. Thus dead loop is avoided and global optimality is obtained. At some certain iteration, some moves are categorized as illegal or tabu. The moves that meet the pre-defined criteria will be assigned to a tabu status. For instance, one move can be classified as tabu if the reverse move of it has been made recently (within a given number of iterations) or frequently (exceed a given number of times during a certain number of iterations). However, the move being defined as tabu is not strictly forbidden. It may sometimes be desirable to make an otherwise tabu move and a particular implementation may include aspiration criteria that override the tabu status of the move.

A comparison between Tabu Search and simple descent method where the goal is to minimize $f(x)$ can be made. Such simple descent method only allows moves to neighbor solutions $N(x)$ which improve the current objective function value and terminates when no improving solutions can be found. The final x

obtained by the simple descent method is called a local optimal, since it is at least as good as or better than all solutions in its neighborhood. The evident shortcoming of the simple descent method is that such a local optimal in most cases will not be the global optimal, i.e., it usually will not minimize $f(x)$ over all $x \in X$. Tabu Search allows moves that deteriorate the current objective function value and selects the moves from a modified neighborhood $N^*(x)$. Short and long term memory structures are responsible for the specific composition of $N^*(x)$. In other words, the modified neighborhood is the result of maintaining a selective history of the states encountered during the search procedure. In Tabu Search strategies based on short term considerations, $N^*(x)$ characteristically is a subset of $N(x)$, and the tabu classification serves to identify elements of $N(x)$ excluded from $N^*(x)$. In Tabu Search strategies that include longer term considerations, $N^*(x)$ may also be expanded to include solutions not ordinarily found in $N(x)$, such as solutions found and evaluated in past search, or identified as high quality neighbors of these past solutions. Characterized in this way, Tabu Search may be viewed as a dynamic neighborhood method. This means that the neighborhood of x is not a static set, but rather a set that can change according to the history of the search.

Chapter 3

System Framework

The framework for the proposed automated stowage planning system is shown in Figure 8. It consists of four main modules: (i) Initialization module; (ii) Basic Plan Generation module; (iii) Weight Adjustment module; (iv) Final Stowage Plan module. This system is designed and implemented in this project to simulate the unloading and loading process of large containerships and obtain stowage plans for different kinds of containerships and different inputs of containers lists. This framework is used to evaluate the effectiveness of different stowage strategies. The functions of each module are described in this chapter.

3.1 Initialization Module

One of the functions of the Initialization module is to provide a ship entity, which can be used as a model to simulate a real containership sailing from ports to ports with containers being unloaded and loaded at each port. This system is proposed to solve stowage planning problems for different types of containerships. However, the structures of different containerships are different from the capacities to bay and location configurations, though most of the containerships share some structural similarities. Therefore, it is important to model the physical structure of a containership correctly.

A standard format of ship structure file is designed in this project, which contains information fields such as the stowage locations, the stack weight limits and the hatch covers. The physical structure of any containership can be translated and saved in a ship structure file in this standard format. The system

is developed to read in the ship structure file in the standard format and create a ship entity according to it.

Figure 8 : Framework of the Automated Stowage Planning System

Another function of the Initialization module is to provide the sailing voyage of the containership. A containership always visits a number of ports in one journey, it is important for the automated stowage planning system to know which port it is planning for and what the sequence of the downstream ports is. A voyage file provides these information.

The third function of the Initialization module is to initialize the container lists. There are two container lists that need to be initialized. The first one is the

onboard container list. This list provides all the containers that are onboard the containership before the unloading and loading process and their stowage locations on the containership. The automated stowage planning system first creates the containers entities according to the list and then assigns them to the corresponding stowage locations of the ship entity. The second container list is the containers loading list, which records all the containers that need to be planned by the automated stowage planning system. The system reads in this list of containers and creates the containers entities according to it.

The last function of the Initialization module is to initialize the loading constraints of stowage planning. There are many constraints related to the stowage planning, such as dangerous goods segregation requirements, 20Ft containers stowage positions and special stowage requirements. These constraints are built inside the automated stowage planning system as guide lines of the planning process.

3.2 Basic Plan Generation Module

After the tasks in the Initialization module is completed. The Basic Plan Generation Module in the automated stowage planning system starts. There are two tasks to be carried out sequentially in this module, (i) Basic Allocation and (ii) Hazardous and Special Stowage Adjustment.

The Basic Allocation task applies a block stowage heuristics to stow groups of containers to blocks of stowage locations on a containership. The block stowage heuristics and the definitions of the groups of containers and the blocks of stowage locations are described in Chapter 4. Besides that, different block selection strategies, which can provide different basic allocation plans, are also designed and implemented in this module. The detailed descriptions of these strategies are presented in Chapter 5 and Chapter 6.

The Hazardous and Special Stowage Adjustment task is performed after the Basic Allocation task. Based on the plan obtained from the Basic Allocation task, the Hazardous and Special Stowage Adjustment task introduces

adjustments in terms of swapping containers so that the basic stowage plan can satisfy the dangerous goods and special stowage constraints.

3.3 Weight Adjustment Module

The Basic Plan Generation module generates a stowage plan that takes into consideration of the number of re-handles, number of slots killed and crane intensity, and also try to satisfy the dangerous goods and special stowage constraints. However, the issues related to container weights are left unsolved. The purpose of the Weight Adjustment module is to adjust the container stowage locations according to their weights, and thus solve the stowage problems caused by container weights. There are three main problems that need to be solved in the Weight Adjustment module: (i) the total weight of containers in each stack; (ii) the container weight distribution in the cross dimension of the containership and (iii) the container weight distribution in the horizontal dimension of the containership.

Through swapping containers, Stack Weight Adjustment reduces the total weight of containers in the stack that exceeds the weight limit; Trim Adjustment reduces the cross moment difference caused by container weight onboard; Heel Angle Adjustment reduces the horizontal moment difference caused by container weight onboard. Different kinds of search and swapping algorithms can be used in this module to achieve the goal. The search and swapping algorithm designed and implemented in this system is presented in Chapter 6.

3.4 Final Stowage Plan Module

After the tasks in the Weight Adjustment module are completed, the stowage planning procedure is completed. However, the objective values of the stowage plan are still unknown. Therefore, an objectives evaluation procedure is designed and implemented in the Final Stowage Plan Module. After the objective values of the stowage plan are obtained, they are output with the final stowage plan, which states each container's stowage location on the containership, to external files provided to the system user.

The different objectives being considered in this project have been discussed in Section 2.4.4. Except for the completion time of the longest quay crane, the values of the other objectives are straightforward to calculate.

In order to investigate the actual completion time of the longest quay crane of the stowage plans generated by the different strategies, a Crane Simulator is designed and implemented in this module. The Crane Simulator simulates the operating sequences of the quay cranes at the terminal of each port. It can dynamically adjust the workload assigned to each quay crane, detect conflict between two adjacent cranes, and calculate the actual crane intensity including the time for quay cranes to move from one bay to another and the possible idle time for each crane. Finally a Gantt chart of the quay cranes will be obtained by the Crane Simulator to show the operation status of each quay crane.

Figure 9 : Gantt Chart of Quay Cranes

Figure 9 shows an example of the Gantt chart of quay cranes. Each row represents a bay on the containership. The number assigned to each row indicates the total workload (including loading and unloading) of that bay. Three colored line represent three quay cranes are serving this containership. If a colored line switches from one row to another row, it means the quay crane has finished the workload of one bay and moves to another bay. The grey colored rectangular between the colored lines indicates the quay crane is moving from one bay to another. The break of a colored line, for instance in the yellow line in Figure 9, indicates that the quay crane encounters a conflict with

another quay crane (the red one in Figure 9) and is waiting for that quay crane to move out of the conflicting area. A square in the figure represents a time interval of 10 minutes. If it is less than 10 minutes, a number will be written in the square to indicate the actual time. Therefore, summing up all the time for a colored line gives the completion time of that quay crane. The completion time of the longest quay crane comes from the longest colored line.

The output of the Final Stowage Plan module are three external files. The first one lists all the containers onboard and their corresponding stowage locations. The second file is a Gantt chart of quay cranes shown in Figure 9. The last file lists all the values of the objectives of the stowage plan.

Chapter 4

Block Stowage Heuristics

From the framework of the proposed automated stowage planning system described in Chapter 3. It can be seen that the most important part of this system is the Basic Plan Generation module. The Basic Plan Generation module lays the foundation for the Weight Adjustment module to do further fine tuning of the stowage plan. It is mentioned in Chapter 3 that the Basic Allocation task in the Basic Plan Generation module applies a Block Stowage heuristics to stow groups of containers to blocks of stowage locations on a containership. In this chapter, a detailed description of the Block Stowage heuristics, which includes its concept, the definition of the groups of containers and the blocks of stowage locations, and the logic of stowing process within a block will be provided.

4.1 Concept

Instead of stowing the containers one by one, which is tedious and time consuming, the Block Stowage heuristics stows the containers group by group. It first divides the containers to be loaded to the containership into several groups according to some rules, and then stow the groups onto the containership. The flowchart of this procedure is shown in Figure 10. In this way, the search space of this problem and the processing time can be reduced significantly.

The general structure of the Block Stowage algorithm is to first divide the containers to be loaded into several groups, and partition the locations on the

containership into several blocks as well. Next, the algorithm selects one group of containers from the containers list, and then according to the port of discharge of the group of containers, selects a block from the containership and stow the containers into it. If the number of locations in one block is not enough to stow all the containers in that group, another block will be selected from the containership to stow the rest of the containers in the group. After that, if all the containers have been loaded to containership, the process terminates. Otherwise the algorithm continues to select the next group of containers to stow.

Figure 10 : Flowchart of Block Stowage Heuristics

4.2 Group the Containers

A basic heuristics in container stowage is to stow the containers with the same port of destination together, which can prevent unnecessary re-handles, and also result a reasonable quay cranes' utilization. Therefore, in the proposed Block Stowage algorithm the containers are grouped according to their ports of destinations.

Another general heuristics in stowage planning problem is to stow the containers with a further port of destination below those containers with a nearer port of destination. This approach can prevent forced re-handles. Thus, after dividing containers into different groups of destination port, the proposed Block Stowage algorithm selects the group of containers with a further port of destination to be stowed first.

For example, if the containership is on a voyage of n ports including the current port, there will be up to $n - 1$ groups of containers to be loaded and transported to the $n - 1$ ports. *Group $n - 1$* are the containers going to the furthest port in the voyage. *Group $n - 2$* are the containers going to the second furthest port and *group 1* is going to the next port from the current port.

For instance, a list of containers at *Port AAA* (the 3 characters port code) is ready to be loaded to the containership with a voyage of *AAA – BBB – CCC – DDD – EEE – FFF – GGG – HHH*. The list of containers will be divided by destination ports as:

- *Group 1*: include the containers with destination of *Port BBB* (the next port from the current port in the voyage),
- *Group 2*: include the containers with destination of *Port CCC*,
- *Group 3*: include the containers with destination of *Port DDD*,
- *Group 4*: include the containers with destination of *Port EEE*,
- *Group 5*: include the containers with destination of *Port FFF*,
- *Group 6*: include the containers with destination of *Port GGG*,
- *Group 7*: include the containers with destination of *Port HHH* (the furthest port from the current port in the voyage).

The stowing process will start from the group with the greatest index (*Group 7* in this case) to the group with smallest index (*Group 1* in this case).

Because of the operational constraint that 40Ft container can be stacked on top of both 20Ft and 40Ft containers, but 20Ft containers can only be stowed above 20Ft containers, another step is needed to be performed after grouping the containers by destination ports. Within each group of containers of the same

port of destination, it is divided into two sub-groups according to sizes of the containers, named 20Ft group and 40Ft group. 20Ft group is stowed before 40Ft group within the same port of destination group.

4.3 Partition the Containership

For a containership with hatches, a container in a row below a hatch can affect all the rows above the same hatch. Therefore the hatch is an important factor when considering re-handles. In order to reduce the number of hatch movements and re-handles caused by hatch movements, the proposed Block Stowage heuristics uses hatches as references to partition the location on the containership.

Figure 11 : Partition a Bay

Figure 11 shows a 40Ft bay which has 6 rows and 3 hatches. Each hatch covers 2 rows, for example, the left hatch covers *row 06* and *row 04*, the center hatch covers *row 02* and *row 01*, the right hatch covers *row 03* and *row 05*. First, the rows below or above the same hatch are defined as a block respectively, which means the bay in Figure 11 is first partitioned into 6 blocks:

- *Block 1—row 06 and row 04 below hatch*
- *Block 2—row 06 and row 04 above hatch*

- *Block 3—row 02 and row 01 below hatch*
- *Block 4—row 02 and row 01 above hatch*
- *Block 5—row 03 and row 05 below hatch*
- *Block 6—row 03 and row 05 above hatch*

The stability issue is one of the most important considerations in stowage planning problem. Hence another heuristics to achieve good weight balance between the starboard side and port side is to stow containers in one side of the containership after stowing containers in the other side. Therefore, the proposed Block Stowage heuristics links the two side blocks together, which means the rows below the left hatch and the rows below the right hatch are considered to be in the same block. The same logic also applies to the rows above deck. After the blocks are linked, the bay in Figure 11 is partitioned into 4 blocks:

- *Block 1—row 06,row 04,row 03 and row 05 below hatch*
- *Block 2—row 06,row 04,row 03 and row 05 above hatch*
- *Block 3—row 02 and row 01 below hatch*
- *Block 4—row 02 and row 01 above hatch*

After the locations on a containership have been partitioned into blocks, all the blocks have to be further divided into different groups. The Block Stowage heuristics divides the blocks into different groups according to the destination ports of containers that remaining onboard. An additional group is added for empty blocks (blocks with no containers). For the convenience of description, “destination port of a block” is defined as the nearest destination port of all the containers stowed in the block. When new containers are stowed into a block, the destination port of the block must be updated.

For instance, if a containership with a voyage of *AAA – BBB – CCC – DDD – EEE – FFF – GGG – HHH* is at *Port AAA* and is ready for loading new containers. We can divide the blocks onboard into groups as follows:

- *BlockBelow [0]:* include the Blocks below hatch which are empty,
- *BlockBelow [1] :* include the Blocks below hatch stowed with containers whose destination ports are the nearest from the current port in the voyage, *Port BBB* in this case.

- *BlockBelow* [2] : include the Blocks below hatch stowed with containers whose destination ports are the second nearest from the current port in the voyage, *Port CCC* in this case.
 - ...
 - *BlockBelow* [$n - 1$]: include the Blocks below hatch stowed with containers whose destination ports are the furthest from the current port in the voyage, *Port HHH* in this case.
-
- *BlockAbove* [0]: include the Blocks above hatch which are empty,
 - *BlockAbove* [1] : include the Blocks above hatch stowed with containers whose destination ports are the nearest from the current port in the voyage, *Port BBB* in this case.
 - *BlockAbove* [2] : include the Blocks above hatch stowed with containers whose destination ports are the second nearest from the current port in the voyage, *Port CCC* in this case.
 - ...
 - *BlockAbove* [$n - 1$]: include the Blocks above hatch stowed with containers whose destination ports are the furthest from the current port in the voyage, *Port HHH* in this case.

4.4 Block Selection

4.4.1 Loading Sequence of Containers Groups

The aim of Block Stowage heuristics is to minimize the number of re-handle in MBPP of large containerhips. Hence, one of the basic strategies in this heuristics is to load the containers with the furthest port of destination first, and load the containers with the nearest port of destination last. For example, the loading sequence of groups of containers in the case presented in Section 4.2 should be *Group 7* first, then *Group 6* ... and finally *Group 1*.

4.4.2 Block Filtering Procedure

While stowing a group of containers, the Block Stowage heuristics first filter all the blocks on the containership and find a set of suitable candidate blocks for

stowing the group of containers. Usually the set contains one or a few blocks. The filtering procedure is described in Table 1.

Assume that several containers of *Group k* is being considered for finding one or a few blocks to stow. The greater *k* is, the further the destination port is.

If the procedure terminates before reaching step 8, this means that a set of blocks that satisfy the same block filtering requirement has been obtained. Any block chosen in the set to stow the containers can achieve the goal of minimizing the number of re-handles, because no re-handle will be incurred.

However, if the procedure proceeds to step 8, it means that there is no available block for stowing the group of containers without causing any re-handles. There are two possible reasons for this situation:

- 1) All the blocks are occupied by containers with a nearer destination than the group of containers being considered;
- 2) There are some blocks with a further destination than the group of containers being considered, but all these blocks are stowed with 40Ft containers in the top most tier and the group of containers going to be stowed are 20Ft containers, which cannot be stacked on top of 40Ft containers.

The following actions can be carried out to solve these two problems,

- 1) Over-stow, which means stowing containers with a further port of destination above containers with a nearer destination. Forced re-handles will be incurred in a downstream port because of this over-stow.
- 2) Voluntary re-handle, which means discharging some containers to obtain enough space for the un-loaded containers, then loading these containers back to the containership again.

Table 1 : Block Filtering Procedure

<p>Step 1. Select all the blocks with empty available locations in <i>BlockAbove</i> [k], if no such block is found go to the next step;</p> <p>Step 2. Select all the blocks with empty available locations in <i>Blockbelow</i> [k], if no such block is found go to the next step;</p> <p>Step 3. Select all the blocks with empty available locations in <i>BlockAbove</i> [0] and the block below is in <i>Blockbelow</i> [k], if no such block is found go to the next step;</p> <p>Step 4. Select all the blocks with empty available locations in <i>Blockbelow</i> [0], if no such block is found go to the next step;</p> <p>Step 5. Select all the blocks with empty available locations in <i>Blockbelow</i> [t] ($t > k$), if no such block is found go to the next step;</p> <p>Step 6. Select all the blocks with empty available locations in <i>BlockAbove</i> [0] and the block below is in <i>Blockbelow</i> [t] ($t > k$), if no such block is found go to the next step;</p> <p>Step 7. Select all the blocks with empty available locations in <i>BlockAbove</i> [t] ($t > k$), if no such block is found go to the next step;</p> <p>Step 8. Switch to Re-handle stage.</p>

Since over-stow only can solve problems caused by reason 1, while voluntary re-handle can solve problems caused by both reason 1 and reason 2, the system performs voluntary re-handle when the procedure proceeds to step 8.

4.5 Stowing a Block

When a set of blocks has been obtained after Block Filtering procedure, one block is selected from it to stow the group of containers. If one block is not enough to stow all the containers in the group, then a further block is chosen from the set. All the blocks in the same set are considered to be of the same preference for stowing the group of containers with regards to the objective of

minimizing the number of re-handles. Thus, the set of blocks follows a first-in-first-out (FIFO) manner.

Once the group of containers and the exact block haven been decided, the individual containers are stowed into the individual locations. Because of the operational constraint that 20Ft containers cannot be stowed above 40Ft container and 40Ft container can be stowed above 20Ft containers, we filter all the 20Ft containers and stow them first. The 40Ft containers are loaded after all 20Ft containers have been loaded. Within each 20Ft or 40Ft containers group the containers are sorted according to individuals' weight by the system. Containers with heavier weight are stowed earlier and are thus generally stowed in a lower location than the containers with lighter weight. This approach is helpful to the stability issue of the containership by reducing the meta-centric height (GM) of the ship.

Figure 12 : Stowing Algorithm for a Block

When stowing a block, the system stows the containers row by row. This means that the algorithm fills all the tiers in one row before starting to fill another row. This approach can also prevent possible re-handles problem.

The stowing algorithm fills a row in the right part of the containership first, and then fills a row in the left part, alternatively. This approach helps to maintain a good weight balance in the horizontal dimension of the containership. Figure 12 shows an example of the stowing algorithm. The block contains *row* 03, 05, 04 and 06 under deck is selected to stow a group of containers. Containers are stowed vertically in *row* 03 first, and then in *row* 04, 05 and 06 at last.

Chapter 5

Deterministic Block Selection Mechanisms

It is described in Section 4.4.2 that a set of blocks will be obtained after the Block Filtering procedure. The block stowage algorithm has no preference in selecting any block in that set for stowing containers with regards to the objective of minimizing the number of re-handles. However, the MBPP for large containerhips is inherently a multi-objective optimization problem. Beside minimizing the number of re-handles, other objectives such as the utilization of quay cranes, the number of stacks that exceed the stack weight limit, the number of slots killed, the horizontal moment difference and the cross moment difference also have to be taken into consideration during the optimization. Therefore, two mechanisms emphasizing the optimization of the quay crane utilization and container weight distribution are proposed in this chapter to help make better decisions during block selection.

5.1 Block Ranking

After Block Filtering, a set of blocks is obtained. If there is only one block in the set, the containers have to be stowed into that block. However, if there is more than one block in the set, the block ranking mechanism provides deterministic approaches to rank the blocks with regards to the objectives of maximizing the quay crane utilization and balancing container weight distribution respectively.

We propose two block ranking approaches to decide how blocks should be ranked.

▪ **Approach 1 –Workload-Based**

Before applying this approach, the number of moves of each bay must first be obtained. One “move” is defined as one container handling operation, either loading or unloading. For instance, the loading of a container to the containership is counted as one “move”, and the unloading of a container from the containership is also defined as one “move”.

For the *Workload-Based Approach*, the Block in the bay with the lowest number of moves is ranked higher.

Let $L(Bay_i)$ denote the number of moves for Bay_i .

Let $R(Block_i)$ denote the rank of $Block_i$, $Block_i \in Bay_i$.

If $L(Bay_i) < L(Bay_j)$, then $R(Block_i) > R(Block_j)$.

In order to obtain a high utilization of quay cranes, it is desired to spread the workload evenly across the ship. This approach is designed to stow more containers to the block whose bay has a lower number of moves.

▪ **Approach 2 –Distance-Based**

In the *Distance-Based Approach*, the block nearer to the longitudinal center of the ship is ranked higher.

Let $D(Bay_i)$ denote the distance between Bay_i to the longitudinal center of the containership.

If $D(Bay_i) < D(Bay_j)$, then $R(Block_i) > R(Block_j)$

In order to avoid significant moment difference caused by containers' weight across the ship which may result in an undesirable trim, containers (especially heavy ones) are allocated to the longitudinal center of the containership to

provide better stability. This approach mainly aims to minimize the cross moment difference caused by containers' weight.

These two approaches are mutually exclusive in the block ranking mechanism. Only one of them can be applied in a single stowage plan generation process. In the experiment and discussion chapter, the effects of different block ranking approaches on stowage planning will be studied.

5.2 Crane Intensity Control

The quay cranes' utilization is one of the important objectives in MBPP of large containerhips. Thus a crane intensity control mechanism is proposed to obtain a relatively high utilization of quay cranes. The core of this mechanism is called the Crane Intensity Control module, which serves as the final stage before stowing containers into the selected block. Figure 13 shows the function of the crane intensity control mechanism.

As two quay cranes cannot serve the same containership simultaneously within an "8 bays" distance, two heuristic rules were proposed by Xiao et al.(2009) [22] to prevent excessive workload being assigned to two adjacent 40Ft bay (the distance of two adjacent 40Ft bays equals to the pre-defined distance of "8 bays"). The Crane Intensity Control module implemented the following two heuristic rules:

- Rule 1. The total number of moves of two adjacent 40' bays should not exceed the average number of moves per crane across all bays at the current port.
- Rule 2. The number of containers with the same port of destination in two adjacent 40' bays should not exceed the average unloading moves per crane at the port of destination.

Each block being considered by the Crane Intensity Control module is verified with the above two rules. If the block passes the verification of the module, containers can be stowed in it. If the block fails to pass the module, it is

discarded and the next block in the blocks group will be sent to the module for consideration.

Figure 13 : Function of Crane Intensity Control Mechanism

In some instances, these two heuristic rules maybe too strong and thus result in no block satisfying both requirements. A parameter C is introduced to the Crane Intensity Control module to relax the constraints. If $C=2$, both requirements must be satisfied; If $C=1$, only the first rule needs to be satisfied; If $C=0$, none of the rules needs to be satisfied. The constraints are relaxed gradually if all the blocks in the group cannot pass the module.

It can be seen from the scheme of the function of crane intensity control mechanism that, if the size of the blocks group is very small, it is highly probable that none of the blocks satisfied both rules and C will eventually

become 0. Therefore the Crane Intensity Control module may not be able to effectively control the crane intensity of the generated stowage plan.

Two approaches are proposed in the crane intensity control mechanism to determine the size of the blocks group.

- **Approach 1- *Re-handle Driven***

In the *Re-handle Driven* approach, the group of blocks contains only one set of blocks obtained in a certain step of Block Filter procedure. The group of blocks is verified by the Crane Intensity Control module one by one. If a block fails to meet the requirements set by the rules, the next block in the group will be tested. If all the blocks in the group fail to meet the requirements set by the rules, then the parameter C is reduced and the blocks in this group are re-tested. The group of blocks will eventually meet the requirements when C is reduced to 0. The rank of the blocks in the group is decided by the Block Ranking approach described in Section 5.1.

In this approach, if a set of blocks is found in step N of the Block Filtering procedure, then blocks in steps $>N$ will never be used to stow the containers, even if stowing containers in the later selected block can obtain better crane intensity. By forcing a selection of a block from group N , this approach can achieve better space utilization on containership which prevents possible re-handles.

- **Approach 2- *Crane Intensity Driven***

In the *Crane Intensity Driven* approach, the blocks selected in all steps of the Block Filtering procedure are considered as a group and verified by the Crane Intensity Control module one by one. For example, if all the blocks from the set, set N , obtained from step N of the Block Filtering procedure fail to meet the requirements of the rules, then the blocks in the next set, set $N+1$, obtained from step $N+1$ of the Block Filtering procedure will be tested. If none of the blocks in all the sets meets the requirements of the rules, then the parameter C is reduced and the blocks from set 1 of the Block Filtering procedure are re-

tested. The blocks in each set are ranked using one of the approaches proposed in block ranking mechanism described in Section 5.1.

In this approach it is possible that blocks selected from a later Block Filtering step, step N , can be used to stow containers, even if there are sets with one or a few blocks in an earlier Block Filtering step, step $N-1$. This is because a block in a later set may meet the requirements of the rules while none of the blocks in the sets before it could. Selecting this block thus results in a better crane intensity.

The example below explains the different effects of these two approaches:

Figure 14 : Example for Crane Intensity Control Module (a)

Figure 14 shows a scenario where the containership is being loaded at Port 1. There is a group of containers with the destination of Port 4 to be stowed, and only 3 empty blocks can be used to stow them. One empty block (the one selected from step 3) is in Bay 22 above the hatch, and the block below it stowed containers with the destinations of Port 4 as well. The second empty block (the one selected from step 4) is in Bay 42 under the hatch. The block above it is the third empty block. Suppose that the number of moves in Bay 22

has already exceeded both the limits set by rule 1 and rule 2 in the Crane Intensity Control module, while the number of moves in Bay 42 is still within the limits set by both rules.

The figures below show the stowage plans generated by applying different crane intensity control approaches.

Figure 15 : Example for Crane Intensity Control Module (b)

Figure 15 and Figure 16 show the stowage configurations of Bay 22 and Bay 42 at Port 1 by applying the *Re-handle Driven* approach and the *Crane Intensity Driven* approach respectively after stowing the containers. The plan in Figure 16 can be seen as having better quay crane utilization.

Suppose the containership sails to the next port. At Port 2 there are a group of containers with the destination of Port 5 to be stowed. Unfortunately, all the newly created empty blocks at Port 2 have already been occupied by the containers with the destination further than Port 5.

Figure 17 and Figure 18 show the scenarios of the stowage plans applying the *Re-handle Driven* approach and the *Crane Intensity Driven* approach respectively at Port 2 before stowing containers. For the plan in Figure 17, the containers with the destination of Port 5 can be stowed into Blocks in Bay 42 without causing any re-handles. However, for the plan in Figure 18, voluntary re-handles must be carried to stow the containers with the destination of Port 5, or forced re-handles have to be performed when the containership sails to Port 4.

Figure 16 : Example for Crane Intensity Control Module (c)

From the example above, it can be seen that the *Re-handle Driven* approach aims to reduce the number of possible re-handles, while the *Crane-Intensity Driven* approach aims to obtain better quay cranes utilization.

These two approaches are mutually exclusive in crane intensity control mechanism. Only one of them can be applied in a single stowage plan generating process. In the experiment and discussion chapter the effects of the two different approaches in crane intensity control mechanism on stowage planning will be shown.

Figure 17 : Example for Crane Intensity Control Module (d)

Figure 18 : Example for Crane Intensity Control Module (e)

Chapter 6

Randomized Block Stowage Algorithm with Tabu Search

The two block selection mechanisms proposed in Chapter 5 provide some deterministic approaches in deciding a group of containers to be stowed in which block may obtain a better quay cranes' utilization and containers' weight distribution. However, due to the complex nature of MBPP for large containerships, no single approach is found to be adequate for obtaining a good stowage plan when considering the different objectives to be optimized.

The solutions space of MBPP for large containerships can be quite large. The number of possible solutions can easily reach to tens of thousands for a containership with a capacity of 500 TEUs. Even if one only considers the different groups of containers being assigned to different blocks, the number of combinations for block selection can be very large as well. For example, if there are 6 groups of containers to be stowed to the containership and for each group there are 3 suitable blocks for selection and only one of the three can be selected. This results in $3^6 = 729$ different combinations. Therefore, deterministic approaches that use static block selection strategy to obtain solutions cannot fully explore the entire solutions space to find optimum or near optimum solutions.

To overcome the drawback of such deterministic block selection approaches, in this project a randomized two-stage block stowage multi-objective optimization algorithm with Tabu Search (TS) is proposed, in which not only one but a large number of stowage plans are generated in a “randomized” block selection way in the first stage, and then a small number of stowage plans with better characteristics such as quay crane utilization, number of slots killed and number of stacks that exceed weight limit are selected for further optimization in the second stage. TS is implemented in the second stage to optimize each stowage plan in terms of the number of stacks that exceed weight limit, horizontal moment difference and cross moment difference. The detailed description of the randomized two-stage multi-objective optimization algorithm with TS for the large containership stowage planning problem is given below.

6.1 Initial Stowage Patterns Generation Stage

In this stage, the aim is not to generate a stowage plan which is ready for use by the decision maker, but to obtain some good stowage patterns for further optimization.

The Block selection step described in Section 4.4 is a critical step in the “Block Stowage” heuristics. If a deterministic block selection heuristic rule is applied in this step, only one stowage pattern can be obtained through this procedure. Some other potential stowage patterns with better objective values are neglected. Thus, a randomized approach in the block selection step is proposed. The idea is that a large number of stowage patterns are generated in a “randomized” way instead of generating a single stowage pattern in a deterministic way.

It should be noted that the term “randomized” here does not mean totally random. Following the basic heuristics proposed in Section 4.4.2, the blocks selected should still satisfy the requirement of minimizing the number of re-handles. Sometimes there is more than one block that satisfies the block selection requirement. In this situation a decision must be made on which block to select. We define this situation as a “decision point”. The randomized block selection only happens at each “decision point”. Choosing different blocks at

different “decision points” can result in totally different stowage patterns. This is because the consequence of a decision can propagate to a later “decision point”, and the number of combinations of different decisions can be quite large. In order to speed up the process of generating such a large amount of stowage patterns, only physical size constraints, which includes constraints such as 20 foot containers must be stowed in 20 foot locations, 40 foot containers must be stowed in 40 foot locations, and no 20 foot containers are allowed to be stowed on top of 40 foot ones, are considered when assigning each individual container to a specified stowage location regardless of the weight of each container. Because at this stage only stowage pattern, which determines the number of re-handles, the completion time of the longest crane and the number of slots killed, is important. All the other objectives relating to the weight of containers will be optimized in the next stage.

After a desired number of stowage patterns are generated, Pareto sorting is performed on the stowage patterns based on three objectives: number of re-handles, completion time of the longest crane and number of slots killed. Then a Pareto set of stowage plans are obtained for further optimization in the next stage.

6.2 Solutions Optimization by Tabu Search

A “move” is one of the important elements in Tabu Search (TS), as the idea of TS is to explore the search space of feasible solutions by making a series of moves and finally reach an optimal solution.

In our proposed optimization algorithm, a “move” in this problem is defined as a swap of two containers with the same size and port of destination. Merely moving one container from its original stowage location to another new empty slot on the ship or swapping two containers with different sizes or ports of destination is not a legal move in our algorithm. This is because only a swap of two containers with the same size and port of destination can maintain the stowage pattern obtained from the first stage of the algorithm, which is already designed to optimize the number of re-handles, the completion time of the longest quay crane and the number of slots killed. All the other illegal moves

will change the stowage pattern obtained from the first stage. There are two possible outcomes from a sequence of illegal move:

1. Result in a stowage pattern with more re-handles, longer completion time of the longest quay crane or more slots killed;
2. Result in another stowage pattern which has already been discovered in the first stage.

Another reason for not allowing the move of a single container to an empty location or swapping two containers with different sizes or ports of destination is that it will significantly increase the size of the search space of feasible solutions. Since these kinds of “moves” will not improve the quality of stowage patterns, but only increase the computation complexity, it is necessary to forbid these moves.

Tabu move is another critical element in TS. A move is defined as tabu when its reverse move has just been made recently. If a move is regarded as tabu in one iteration, it cannot be used as a legal move to lead to another solution in the neighborhood even if the solution is the best one available in that iteration. This mechanism prevents the exploration process from being stuck in a local optimum, and allows some degradation moves to be made in order to explore new neighborhood which may contain global optimum. In our proposed algorithm, a tabu move is not only a reverse of a move which has just been made recently, which is not sufficient to allow the exploration process to escape from the local optimum. Figure 19 shows an example, there are four containers C_a, C_b, C_c, C_d stowed in location L_a, L_b, L_c, L_d respectively. Then given the following swaps: C_a swaps with C_b , C_c swaps with C_d , C_a swaps with C_d , C_b swaps with C_c , C_a swaps with C_c , and C_b swaps with C_d . After the sequence of moves, the four containers all return to their original locations. However, during the sequence of moves no reverse move occurs.

Thus, the “move” and “reverse move” have been defined for this problem to prevent this kind of cycling occurring. Each “move” consists of two half moves, each half contains the identity of the container and its newly assigned location. For instance, a “move” of C_a in L_a swapping with C_b in L_b is presented in the form of $[(C_a, L_b), (C_b, L_a)]$, the “reverse move” is a move to let two half moves (C_a, L_a) and (C_b, L_b) occur simultaneously.

Figure 19 : Container Swapping only Prevent Reverse Swap

A tabu list of size T_{max} is needed to record the T_{max} moves made most recently, which prevents cycles of length less than or equal to T_{max} from occurring in the search procedure. In our proposed algorithm a basic unit in the tabu list is a half move of a “reverse move” associated with the “move” performed recently. Therefore, a tabu list actually contains $2T_{max}$ half moves. If both the two half moves of one “move” are found in the tabu list, the move is considered tabu and cannot be performed.

Figure 20 shows an example by using the same case presented in Figure 19 with a tabu list, which updates the tabu moves after each swap. It can be observed that after the first swap of C_a in L_a with C_b in L_b , the tabu list becomes $T = \{(C_a, L_a), (C_b, L_b)\}$. After the fourth swap of C_c in L_d with C_b in L_a , the tabu list becomes $T = \{(C_a, L_a), (C_b, L_b), (C_c, L_c), (C_d, L_d), (C_a, L_b), (C_d, L_c), (C_c, L_d), (C_b, L_a)\}$. The fifth swap of C_a in L_c with C_c in L_a is forbidden by the TS mechanism, because both half moves in $[(C_a, L_a), (C_c, L_c)]$ are contained in the tabu list. Hence, this design prevents the above mentioned cycling problem from happening.

Figure 20 : Container Swapping with Tabu List Updated after Each Swap

However, sometimes it is desirable to let a “move” marked as tabu to be carried out. Under this condition, an aspiration mechanism is needed to overrule the tabu status of the “move”, and allow it to happen. In this research we defined an aspiration criterion to allow a “tabu move” to be made. The aspiration criterion

is such that if the “move” can lead to a solution which dominates the best global solution found so far, then it can be made regardless of its tabu status.

In this project, TS is applied to the multi-objective optimization. The concept of Pareto dominance introduced in Section 2.4.1 is applied to determine the dominance relation between two solutions. The objectives being optimized in this stage are the number of stacks that exceed weight limit, horizontal moment difference and cross moment difference.

Table 2 : Pseudo Code of the Tabu Search Algorithm

```

// Select an initial solution s from the Pareto set of solutions obtained in the first stage,
// and assign it to the best global solution found so far S*
S*:= s and k:=0
// Divide the containers in the load list according to their sizes, and ports of destinations into
// n arrays, each array only contains containers of the same size and port of destination
Create array[0], ..., array[n-1] and set i:=0
Create an empty FIFO (first in, first out ) tabu list T of size Tmax
While k < K do
    Create a set M(s) = all possible moves in array[i]
    Define a move A
    For  $\forall m \in M(s)$  do
         $s' = s \oplus m$  // A move m being applied to solution s results a new solution s'
        If  $s' \succcurlyeq S^*$  // s' dominates S*
            Then  $S^*:=s'$ ,  $s:=s'$ , and  $A:=m$ 
        Else If  $s' \succcurlyeq s$  and  $m \notin T$  // s' dominates s and m does not exist in T
            Then  $s:=s'$ ,  $A:=m$ 
        End If
    End For
     $T = T \cup A$  // add A to the end of T
    If the size of T > Tmax
        Then remove the first element in T
    End If
    If S* is updated
        Then k:=0
    Else k:=k+1
    End If
    i:=i+1 // move to the next array
    If i > n-1
        Then i:=0 // restart from array[0]
    End if
End While

```

The TS algorithm in this research also requires a termination condition. Each time a solution that dominates the best global solution found so far is obtained, the best global solution will be updated with it. If the number of iterations performed since the best global solution last updated is greater than a pre-determined maximum number of iterations, the procedure terminates and the best global solution is regarded as the optimal or sub-optimal solution.

The proposed TS algorithm is applied to each solution in the Pareto set obtained in the first stage. Pareto ranking will be again applied to the set of solutions obtained after TS. The pseudo code of TS is presented in Table 2. In this algorithm, K is a pre-determined value used as the stopping condition.

Chapter 7

Experiment and Discussion

In this chapter, we carried out a set of experiments to evaluate the randomized multi-objectives optimization algorithm with Tabu search. A containership with a capacity of 5436 TEUs is used in the experiment. The proposed randomized block stowage algorithm with Tabu Search for multi-objective optimization is tested with two test cases using this ship. A total of 500 randomized initial stowage plans are obtained in the first stage. Then the top n ($n=\max [20, \text{number of solutions in Pareto set}]$) are passed to the second stage. In the TS stage, the termination condition K is set to 100, the size of tabu list T_{max} is set to 100. Finally, a Pareto set of stowage plans are generated in the second stage. The stowage plans obtained by applying different approaches in block ranking mechanism and crane intensity control mechanism are also provided for comparison. All the experiments are carried out on an Intel Core2 PC with 2.66GHz CPU and 2GB RAM.

7.1 Introduction to the Experiment

In our experiments, two test cases with the same containership are used. The two test cases differ in the shipping data used to generate the stowage plan.

- **Test Case 1: Real World Test Case**

In the first test case, we used a set of real world data from a shipping company. The ship sails from port A to port B, and the stowage planning process is done at port B. Table 3 shows the number of containers being discharged and the number of containers remaining onboard at Port B before loading new

containers. Table 4 shows the statistics of container load list at port B. Ports C to G are downstream ports ordered according to alphabetical order. Note that the containers loaded in Port B that are destined for Ports E and G are all empty containers with a weight of 4 tons. Because this is a real stowage planning problem for large containership provided by our industry partner, the stowage plan generated by the human planner is available for comparison.

Table 3 : Statistics of Containers Onboard before Planning at Port B for Test Case 1

	20Ft	40Ft
Number of Container Discharged	618	986
Number of Container Remaining Onboard	465	781

Table 4 : Statistics of Container Load List at Port B for Test Case 1

Port		C	D	E	F	G
20 Ft	Number of Containers	143	18	0	109	0
	Average Weight (ton)	23.17	16.37	-	21.22	-
	Standard deviation of weight (ton)	5.90	5.73	-	3.54	-
40 Ft	Number of Containers	166	73	69	39	158
	Average Weight (ton)	18.47	15.85	4	26.67	4
	Standard deviation of weight (ton)	7.53	7.16	0	4.81	0

▪ Test Case 2: Syntactic Test Case

Because the average container weight for the test case 1 which involves real data is not very heavy, it is relative easy to keep all the stacks within the weight limit. Thus, in the second experiment we obtained a set of real stowage planning data from a shipping company and only modified the weight of these containers, and expect to observe some stacks that exceed the weight limit after applying TS. We modified the weight of containers by dividing the containers into groups according to their sizes and destination ports. 4 different types of weight 10 tons, 20 tons, 30 tons and 40 tons were assigned to containers in each group with the ratio of 1:1:1:1. The ship sails from port C to port D, and the stowage planning process is done at port D. Table 5 shows the number of containers being discharged and the number of containers remaining onboard at

Port D before loading new containers. Table 6 shows the statistics of the container load list at port D. Ports E to H are downstream ports.

Table 5 : Statistics of Containers Onboard before Planning at Port D for Test Case 2

	20Ft	40Ft
Number of Container Discharged	491	491
Number of Container Remaining Onboard	244	796

Table 6 : Statistics of Container Load List at Port D for Test Case 2

Port		E	F	G	H
20Ft	Number of Containers	52	99	207	96
	Average Weight (ton)	25	24.85	24.93	25
	Standard deviation of weight(ton)	11.18	11.13	11.16	11.18
40Ft	Number of Containers	245	6	37	-
	Average Weight (ton)	25.06	25	25.41	-
	Standard deviation of weight(ton)	11.20	9.57	11.29	-

7.2 Design of Tests

The proposed randomized block stowage algorithm with Tabu Search in Chapter 6 is used to generate a set of stowage plan. The approaches in the block selection mechanisms proposed in 0 are also applied to generate stowage plans for comparison purpose.

There are two approaches (*Workload-Based* and *Distance-Based*) in block ranking mechanism, and two approaches (*Re-handle Driven* and *Crane Intensity Driven*) in crane intensity control mechanism. Thus, 4 different deterministic block stowage strategies can be formed: *Workload-Based Re-handle Driven* stowage strategy, *Distance-Based Re-handle Driven* stowage strategy, *Workload-Based Crane Intensity Driven* stowage strategy and *Distance-Based Crane Intensity Driven* stowage strategy.

These 4 deterministic block stowage strategies will also apply a Tabu Search multi-objective optimization module for reducing the number of stacks that exceed the weight limit, minimizing the horizontal moment difference and

minimizing the cross moment difference. Therefore, for each test case besides the stowage plans generated by the randomized block stowage algorithm with Tabu Search, another 4 more stowage plans will also be generated. Objectives such as the number of re-handles, the completion time of the longest quay crane, the number of stacks that exceed the weight limit, the number of slots killed, the difference of horizontal moment and the difference of cross moment are presented for comparison.

7.3 Results and Discussions

7.3.1 Experiments Results

- Test Case 1: Real World Test Case

Table 7 : Objective Values of Top 20 Solutions Obtained after the First Stage for Test Case 1

Solution ID	Dominated Number	Completion time of longest crane (min)	Number of slots killed	Horizontal moment difference (ton×m)	Cross moment difference (ton×m)
341	0	1375	220	483.1	33743.6
428	0	1326	268	3072.8	10000.0
33	0	1326	268	2344.2	530.4
457	0	1373	226	245.7	7829.0
396	0	1442	212	1774	81382.0
65	0	1373	226	758.5	29259.0
387	0	1407	218	614.5	31476.6
30	0	1326	268	2612.5	7660.2
304	0	1262	278	2785.4	11628.8
119	1	1262	358	526.1	35531.6
259	1	1266	280	1321.3	47340.8
267	1	1379	220	1412.9	42897.8
215	2	1284	284	289.6	21854.2
1	2	1380	220	1008.4	41918.2
280	2	1264	452	1010.8	49487.0
440	2	1266	298	1257.9	56757.2
66	3	1314	288	2530.2	34659.0
140	3	1322	286	2871.7	35713.0
365	3	1342	272	994.2	32269.6
211	3	1266	304	1703.5	37570.0

For test case 1, it takes 6 minutes to obtain the 500 initial solutions for stage 1 of the multi-objective optimization. Table 7 shows the top 20 solutions among

the 500 initial solutions, including 9 solutions in the Pareto set. It takes another 28 minutes to apply TS to the top 20 solutions in the second stage. Finally, 9 non-dominated solutions are obtained.

Table 8 : Objective Values of Plans Generated by Randomized Block Stowage Algorithm with Tabu Search, Deterministic Block Stowage Strategies and Human Planner at Port B for Test Case 1

Solution ID	Completion time of longest crane (min)	Number of slots killed	Cross moment difference (ton×m)
304	1262	278	135.8
30	1326	268	0
428	1326	268	0
33	1326	268	0
65	1373	225	2009.4
457	1373	226	0
341	1375	220	10694.2
387	1407	218	14600.8
396	1442	212	69699.8
Workload-Based Re-handle Driven	1428	376	30365.2
Distance-Based Re-handle Driven	1485	282	2682.6
Workload-Based Crane Intensity Driven	1252	391	45835.8
Distance-Based Crane Intensity Driven	1261	336	3885.8
Human Planner	1567	288	39347.6

Table 8 shows the objective values of the 9 non-dominated stowage plans generated by applying the randomized block stowage algorithm with TS, the 4 stowage plans generated by applying deterministic block stowage strategies and the stowage plan provided by a human planner. The values related to weight are calculated based on a ship configuration with zero ballast. The number of stacks that exceed the weight limit, the number of re-handles and the horizontal moment difference (ton×m) are not listed in the table as the plans generated by both the stowage planning system and human planner have no stack that exceeds the weight limit.

For the number of re-handles, the 9 solutions generated by applying the randomized block stowage algorithm with TS and the 4 stowage plans generated by applying deterministic block stowage strategies all have no re-

handles, while the plan provided by the human planner has 11 re-handles at the current port.

For the horizontal moment difference, all the 13 plans generated by the stowage planning system are able to obtain an even horizontal moment, while the plan provided by the human planner has a horizontal moment difference of 5736.4 ton×m.

Solutions with IDs 304, 30, 428, 33, 65, 457, 341, 387 and solutions obtained by applying the *Distance-Based Re-handle Driven* block stowage strategy and the *Distance-Based Crane Intensity Driven* block stowage strategy dominate the solution generated by the human planner for all objectives. The stowage plan generated by applying the *Workload-Based Re-handle Driven* block stowage strategy is dominated by 8 stowage plans generated by applying the randomized block stowage algorithm with Tabu Search (Solutions ID: 304, 30, 428, 33, 65, 457, 341, 387), while the stowage plan generated by applying the *Distance-Based Re-handle Driven* block stowage strategy is dominated by 6 stowage plans generated by applying the randomized block stowage algorithm with Tabu Search (Solutions ID: 304, 30, 428, 33, 65, 457).

The stowage plan generated by the human planner, stowage plans generated by applying the four deterministic stowage strategies and stowage plans with ID 30 and ID 33 generated by applying the randomized block stowage strategy with their corresponding Gantt charts are shown in Appendix B.

▪ Test Case 2: Syntactic Test Case

For test case 2, it takes 4.8 minutes to obtain 500 initial solutions in the first stage. Table 9 shows the top 20 solutions among the 500 initial solutions, including 8 solutions in the Pareto set. It takes 30.5 minutes to apply TS to the top 20 solutions in the second stage. Finally, 10 non-dominated solutions are obtained.

Table 10 shows the objective values of the 10 non-dominated stowage plans generated by applying the randomized block stowage algorithm with TS and the 4 stowage plans generated by applying deterministic block stowage strategies. As the loading list data is syntactically generated, the statistics for the stowage plan from human planner is not available. The number of re-handles is not listed in the table as all the plans obtained have no re-handle at the current port.

Table 9 : Objective Values of Top 20 Solutions Obtained after the First Stage for Test Case 2

ID	Dominat ed Number	Completion time of longest crane (min)	Number of Stacks Exceed Weight Limit	Number of slots killed	Horizontal moment difference (ton×m)	Cross moment difference (ton×m)
212	0	1128	57	281	4907.5	25435.7
124	0	979	72	297	3597.5	13115.7
23	0	937	65	331	5907.5	23484.3
253	0	945	54	315	4497.5	22004.3
287	0	1110	71	285	4827.5	34805.7
461	0	1107	69	295	6407.5	635.7
353	0	976	82	303	4617.5	10884.3
434	0	1110	67	285	4702.3	24545.7
182	1	989	63	297	4637.5	3934.3
129	1	1134	61	281	6457.5	27995.7
339	2	980	79	305	4197.5	1804.3
487	2	1002	67	297	3567.5	30835.7
360	2	1163	55	281	4677.5	9874.3
347	2	1116	71	285	4397.5	26055.7
443	2	998	57	301	3527.5	5345.7
449	2	972	73	333	3377.5	24045.7
424	3	1122	71	289	6047.5	3774.3
24	3	984	71	307	5287.5	28755.7
46	3	1124	65	285	4747.5	14725.7
400	3	1114	65	295	3757.5	5495.7

The stowage plans generated by applying the four deterministic stowage strategies and the stowage plans with ID 23 and ID 24 generated by applying the randomized block stowage strategy with their corresponding Gantt charts are shown in Appendix B.

7.3.2 Discussion

From these experiments we can see that the proposed randomized block stowage algorithm with Tabu Search for multi-objective optimization of large

containership stowage plans is able to generate multiple non-dominated solutions (9 in test case 1 and 10 in test case 2 based on a setting of 500 randomized initial solutions generated in the first stage) in a reasonable time period (less than 40 minutes), which compared with the previous research work and the human planners (it usually takes 2 hours to generate one stowage plan by a human planner) is relatively short. In test case 1, 8 out of 9 (88.9%) of the non-dominated solutions generated dominate the plan provided by the human planner.

Table 10 : Objective Values of Plans Generated by Randomized Block Stowage Algorithm with Tabu Search and Deterministic Block Stowage Strategies Test Case 2

Solution ID	Completion time of longest crane (min)	Number of Stacks Exceed Weight Limit	Number of slots killed	Horizontal moment difference (ton×m)	Cross moment difference (ton×m)
23	937	23	293	-0.1	4.3
253	945	8	264	-0.1	4.3
353	976	48	273	0	4.3
24	984	41	282	0	7435.7
443	998	17	263	0.2	5.7
487	1002	44	278	0	13955.7
434	1110	38	258	-0.1	5.7
287	1110	39	252	0	16845.7
347	1116	32	247	0	3115.7
212	1128	29	257	0	10815.7
Workload-Based Re-handle Driven	1076	50	287	0.1	265.7
Distance-Based Re-handle Driven	1030	14	307	-0.2	5.7
Workload-Based Crane Intensity Driven	931	85	469	0	70245.7
Distance-Based Crane Intensity Driven	928	80	346	0.1	67225.7

In both test cases, none of the stowage plans generated by applying the randomized block stowage algorithm with Tabu Search dominate the stowage plans generated by applying both the *Workload-Based Crane Intensity Driven*

block stowage strategy and the *Distance-Based Crane Intensity Driven* block stowage strategy. This is because the *Crane Intensity Driven* approach in crane intensity control mechanism is quite strong in controlling the quay cranes' utilization, which may cause blocks in a later set obtained from the block filter procedure (described in Section 4.4.2) to be selected for stowing a group of containers.

Both the randomized block stowage algorithm with Tabu Search and the *Re-handle Driven* approach in the crane intensity control mechanism will not select a block from a less promising set of blocks obtained from the block filter procedure in order to reduce the number of re-handles. This is because selecting such a block from a less promising set may bring about unpredictable re-handles in the downstream ports, which is related to the number of slots killed in the current port. If too many slots are killed at the current port, there will be fewer stowage locations that can be used for stowing containers in the downstream ports without causing re-handles.

From the test cases we can see that the completion time of the longest quay crane of the stowage plan by applying the *Crane Intensity Driven* approach is slightly shorter (less than 10 minutes) than the stowage plans with the shortest completion time of the longest quay crane generated by the randomized block stowage algorithm with Tabu Search, but the number of slots killed is much more (at least 50) than that of stowage plans generated by the randomized block stowage algorithm.

It can also be observed that the *Re-handle Driven* approach can obtain a lower number of slots kill than the *Crane Intensity Driven* approach in the crane intensity control mechanism. This is because the *Re-handle Driven* approach not only considers reducing the number of re-handles of the current port, but also tries to reduce the number of re-handles in the downstream ports. A lower number of slots killed at the current port will lead to a lower probability of having re-handles in the downstream ports. In the block ranking mechanism, the *Distance-Based* approach can obtain a lower cross moment difference than the *Workload-Based* approach. This is because the *Distance-Based approach*

tends to stow more containers near the center part of the containership, which leads to a lower cross moment difference than stowing more containers to the front part and back part of the containership.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

In this thesis, an automated stowage planning system for large container ships is proposed and implemented. This system contains (i) an Initialization module, which creates a ship entity and containers entities for the investigation of the proposed stowage strategies; (ii) a Basic Plan Generation module, which implements the proposed stowage strategies to generate basic stowage plans; (iii) a Weight Adjustment module, which implements the proposed Tabu Search algorithm to optimize the stowage plans; (iv) a Final Stowage Plan module, which evaluates the objectives of the stowage plans and outputs them together with the stowage location of each container onboard to external files.

In this thesis, a Block Stowage heuristics is proposed and implemented. It divides the containers to be loaded into different groups according to their ports of destinations and physical sizes, and partitions the stowage locations on the container ship into different blocks according to their relative positions to hatch covers. Assigning the groups of containers from the furthest port of destination to the nearest port destination to each block achieves the objective of minimizing the number of re-handles.

Two deterministic block selection mechanisms, the block ranking mechanism and the crane intensity control mechanism, are proposed and implemented in this thesis. In the block ranking mechanism, two ranking approaches are

proposed. The *Workload-Based* approach and the *Distance-Based* approach achieve the objectives of minimizing the completion time of the longest quay crane and minimizing the difference of cross moment respectively. In the crane intensity control mechanism, two approaches are proposed. The *Crane Intensity Driven* approach and the *Re-handle Driven* approach achieve the objectives of minimizing the completion time of the longest quay crane and minimizing the number of re-handles respectively. The combination of the two approaches in these two mechanisms provides four different stowage strategies for the Basic Plan Generation module of the automated stowage planning system. Each stowage strategy is implemented in the system and investigated.

In this thesis, beside the deterministic stowage strategies, a multi-objective randomized block stowage algorithm with Tabu Search is also proposed and implemented in the automated stowage planning system. This algorithm obtains a Pareto set of basic stowage plans by applying the randomized block stowage strategies at the first stage, and optimizes each basic stowage plan in that set by applying Tabu Search in the second stage. Eventually, a Pareto set of final stowage plans are obtained. By applying this algorithm, instead of providing a single stowage plan to the shipping lines, multiple stowage plans that emphasize on different objectives can be generated. This can help the shipping lines make better decisions in the real world.

From the test results of the experiment in Section 7.3.1, conclusion can be drawn that the proposed automated stowage planning system is capable of obtaining stowage plans through multi-objective optimization. The objectives optimized include the number of re-handles, the completion time of the longest quay crane, the number of stacks that exceed the weight limit, the number of slots killed, the horizontal moment difference and the cross moment different.

The proposed deterministic stowage strategies are capable of obtaining stowage plans that dominate the stowage plan generated by the human planner. Each plan is generated within 2 minutes, compared to the 2 hours required by a human planner to generate a plan.

The proposed randomized block stowage algorithm with Tabu Search is capable of obtaining a set of non-dominated stowage plans (9 in test case 1 and 10 in test case 2 based on a setting of 500 randomized initial solutions generated in the first stage) in less than 40 minutes, compared to 2 hours required by a human planner to complete only one stowage plan. In the test case with the comparison of the stowage plan generated by the human planner, 8 out of 9 (88.9%) of the non-dominated solutions dominate the plan provided by the human planner.

8.2 Recommendations for Future Work

8.2.1 Parallel Computing

Increasingly, parallel computing technology is being seen as a cost-effective method for solving computationally large and data-intensive problems. Nowadays, the emergence of inexpensive parallel computers such as commodity desktop with multiple processors, cluster of workstations or PCs and software standards for portable parallel programming have made parallel computing generally applicable and accessible [18].

Due to the numerous combinations of possible blocks selection options in the Basic Plan Generation module, the initial solutions generation stage in the proposed randomized block stowage algorithm with Tabu Search is a computation intensive problem, which would require a long time for obtaining sufficient number of initial solutions. It is worthwhile to experiment with using parallel computing technology in the initial solutions generating stage, since PCs with multiple processors and cluster are easily accessible, and the framework of this system is built on Java platform, which supports multi-threaded parallel programming. If parallel computing is implemented successfully in this initial solutions generation stage, more initial solutions can be obtained within a shorter time period.

8.2.2 Multi-Objective Evolutionary Algorithms

The basic idea for single objective Evolutionary Algorithms (EAs) is to imitate the natural process of biological evolution. It first creates a group of solutions and selects several best solutions of the group as the parents for the next

generation. After that, a new group of offspring is created by applying mutation on the parents. After the first iteration of EAs is completed, the algorithm loops back to the evaluation of the offspring for selecting the parents of the next iteration.

Evolutionary Algorithms (EAs) is particularly suited for optimization problem with multiple objectives as they process a set of solutions in parallel, eventually exploiting similarities of solutions by crossover [12]. Some researchers suggest that multi-objective optimization problem might be an area where EAs do better than other blind search strategies [9] [18]. Therefore, by applying Multi-Objective Evolutionary Algorithms (MOEAs) together with parallel computing to the proposed randomized block stowage algorithm with Tabu Search, the solutions space with promising solutions can be explored more intensively and better solutions will be obtained.

8.2.3 Branch and Bound Algorithms

Branch and Bound (B&B) algorithms is by far the most widely used method for solving large scale NP-hard combinatorial optimization problems. The B&B algorithm searches the entire solutions space by using bounds for the function to be optimized combined with the value of the current best solution to enable the algorithm to search parts of the solutions space only implicitly [15].

Therefore, B&B algorithm can also be used in the initial solutions generating stage of the proposed randomized block stowage algorithm with Tabu Search. The search process calculates the bound at each node (decision point where a block selection has to be made in this project), and then decides whether to branch to that node or not. By applying this method, nodes that lead to the subspaces that do not contain optimal solutions will not be branched and thus not explored. Thus, computing resource that would otherwise spent on exploring un-productive solutions subspaces can be saved to explore more promising subspaces.

However, the problem in applying B&B to the proposed randomized block stowage algorithm with Tabu Search is in calculating the bound at each

decision point where a block selection has to be made. As the objective values of a stowage plan can only be obtained after the plan is completed, techniques for bounding the objective values of a partial stowage plan will need to be investigated.

Appendices

Appendix A

Publication List for this Research

- **Fan Liu**, Malcolm Yoke Hean Low, Shell Ying Huang, Wen Jing Hsu, Min Zeng and Cho Aye Win. “Stowage Planning of Large Containership with Tradeoff between Crane Workload Balance and Ship Stability”. Proceedings of the 2010 IAENG International Conference on Industrial Engineering. pp. 1537-1543, 17-19 March, 2010, Hong Kong.
- **Fan Liu**, Malcolm Yoke Hean Low, Wen Jing Hsu, Shell Ying Huang, Min Zeng, Cho Aye Win. “Randomized Algorithm with Tabu Search for Multi-Objective Optimization of Large Containership Stowage Plans”. (Submitted to 2nd International Conference on Computational Logistics, 19-22 September, Hamburg, Germany)
- Malcolm Yoke Hean Low, Min Zeng, Wen Jing Hsu, Shell Ying Huang, **Fan Liu** and Cho Aye Win. "Improving Safety and Stability of Large Containerships in Automated Stowage Planning". IEEE System Journal, Vol. 5, No. 1, pp. 50-60, March 2011.
- Min Zeng, Malcolm Yoke Hean Low, Wen Jing Hsu, Shell Ying Huang, **Fan Liu** and Cho Aye Win. "Automated Stowage Planning for Large Containership with Improved Safety and Stability". Proceedings of the 2010 Winter Simulation Conference. pp. 1976-1989, 5-8th December 2010, Baltimore, MD, USA

- Min Zeng, Malcolm Yoke Hean Low, Wen Jing Hsu, Shell Ying Huang, **Fan Liu** and Cho Aye Win. “Improving Ship Stability in Automated Stowage Planning for Large Containerships”. Proceedings of the 2010 IAENG International Conference on Industrial Engineering. pp. 1838-1843, 17-19 March, 2010, Hong Kong.
- Malcolm Yoke Hean Low, Xiantao Xiao, **Fan Liu**, Shell Ying Huang and Wen Jing Hsu. “An Automated Stowage Planning System for Large Containerships”. Proceedings of the 4th Virtual International Conference on Intelligent Production Machines and Systems (IPROMS). July, 2009.
- Xiantao Xiao, Malcolm Yoke Hean Low, **Fan Liu**, Shell Ying Huang, Wen Jing Hsu and Zhengping Li. “An Efficient Block-Based Heuristic Method for Stowage Planning of Large Containerships with Crane Split Consideration”. Proceedings of the International Conference on Harbor, Maritime & Multimodal Logistics. Modeling and Simulation, 23-25 September, 2009, Tenerife-Canary Islands, Spain.

Appendix B

Screenshots of Stowage Plans and the

Corresponding Gantt Charts of Quay Cranes

Generated by Automated Stowage Planning System

and the Human Planner

Figure 21 : Stowage Plan Generated by Human Planner at Port B in Test Case 1

Figure 22 : Gantt Chart of Quay Cranes for Stowage Plan Generated by Human Planner at Port B in Test Case 1

Figure 23 : Stowage Plan Generated by Workload-Based Crane Intensity Driven Stowage Strategy at Port B in Test Case 1

Figure 24 : Gantt Chart of Quay Cranes for Stowage Plan Generated by Workload-Based Crane Intensity Driven Stowage Strategy at Port B in Test Case 1

Figure 25 : Stowage Plan Generated by Distance-Based Crane Intensity Driven Stowage Strategy at Port B in Test Case 1

Figure 26 : Gantt Chart of Quay Cranes for Stowage Plan Generated by Distance-Based Crane Intensity Driven Stowage Strategy at Port B in Test Case 1

Figure 27 : Stowage Plan Generated by Workload-Based Re-handle Driven Stowage Strategy at Port B in Test Case 1

Figure 28 : Gantt Chart of Quay Cranes for Stowage Plan Generated by Workload-Based Re-handle Driven Stowage Strategy at Port B in Test Case 1

Figure 29 : Stowage Plan Generated by Distance-Based Re-handle Driven Stowage Strategy at Port B in Test Case 1

Figure 30 : Gantt Chart of Quay Cranes for Stowage Plan Generated by Distance-Based Re-handle Driven Stowage Strategy at Port B in Test Case 1

Figure 31 : Stowage Plan of Solution No. 30 Generated by Randomized Block Stowage Algorithm with Tabu Search at Port B in Test Case 1

Figure 32 : Gantt Chart of Quay Cranes for Stowage Plan of Solution No. 30 Generated by Randomized Block Stowage Algorithm with Tabu Search at Port B in Test Case 1

Figure 33 : Stowage Plan of Solution No. 33 Generated by Randomized Block Stowage Algorithm with Tabu Search at Port B in Test Case 1

Figure 34 : Gantt Chart of Quay Cranes for Stowage Plan of Solution No. 33 Generated by Randomized Block Stowage Algorithm with Tabu Search at Port B in Test Case 1

Figure 35 : Stowage Plan Generated by Workload-Based Crane Intensity Driven Stowage Strategy at Port D in Test Case 2

Figure 36 : Gantt Chart of Quay Cranes for Stowage Plan Generated by Workload-Based Crane Intensity Driven Stowage Strategy at Port D in Test Case 2

Figure 37 : Stowage Plan Generated by Distance-Based Crane Intensity Driven Stowage Strategy at Port D in Test Case 2

Figure 38 : Gantt Chart of Quay Cranes for Stowage Plan Generated by Distance-Based Crane Intensity Driven Stowage Strategy at Port D in Test Case 2

Figure 39 : Stowage Plan Generated by Workload-Based Re-handle Driven Stowage Strategy at Port D in Test Case 2

Figure 40 : Gantt Chart of Quay Cranes for Stowage Plan Generated by Workload-Based Re-handle Driven Stowage Strategy at Port D in Test Case 2

Figure 41 : Stowage Plan Generated by Distance-Based Re-handle Driven Stowage Strategy at Port D in Test Case 2

Figure 42 : Gantt Chart of Quay Cranes for Stowage Plan Generated by Distance-Based Re-handle Driven Stowage Strategy at Port D in Test Case 2

Figure 43 : Stowage Plan of Solution No. 23 Generated by Randomized Block Stowage Algorithm with Tabu Search at Port D in Test Case 2

Figure 44 : Gantt Chart of Quay Cranes for Stowage Plan of Solution No. 23 Generated by Randomized Block Stowage Algorithm with Tabu Search at Port D in Test Case 2

Figure 45 : Stowage Plan of Solution No. 24 Generated by Randomized Block Stowage Algorithm with Tabu Search at Port D in Test Case 2

Figure 46 : Gantt Chart of Quay Cranes for Stowage Plan of Solution No. 24 Generated by Randomized Block Stowage Algorithm with Tabu Search at Port D in Test Case 2

Bibliography

- [1]. Ambrosino, D. and Sciomachen, A., 1998. A Constraints Satisfaction Approach for Master Bay Plans, In: Sciutto, G., Brebbia, C.A. (Eds.), Maritime Engineering and Ports. WIT Press, Boston, pp. 155-164.
- [2]. Ambrosino, D., Sciomachen, A. and Tanfani, E., 2004. Stowing a Containership: the Master Bay Plan Problem, Transportation Research, 38, pp. 81-99.
- [3]. Aslidis, T., 1989. Combinatorial Algorithms for Stacking Problems, Ph.D. Thesis, MIT.
- [4]. Aslidis, T., 1990. Minimizing of Overstowage in Containership Operations, Operational Research, 90, pp. 457-471.
- [5]. Steenken, D., VoB, S., Stahlbock, R.: Container Terminal Operation and Operations Research: a Classification and Literature Review. OR Spectrum 26, pp 3–49 (2004)
- [6]. Avriel, M., Penn, M., 1993. Exact and Approximate Solutions of the Container Ship Stowage Problem, Computers and Industrial Engineering, 25, pp. 271-274.
- [7]. Avriel, M., Penn, M., Shpirer, N. and Witteboon, S., 1998. Stowage Planning for Container Ships to Reduce the Number of Shifts, Annals of Operation Research, 76, pp 55-71.
- [8]. Avriel, M., Penn, M., Shpirer, N., 2000. Containership Stowage Problem: Complexity and Connection Capabilities, Discrete Applied Mathematics, 103, pp. 271-279.
- [9]. Carlos M. Fonesca and Peter J. Fleming. Multiobjective Genetic Algorithm Made Easy: Selection, Sharing and Mating Restrictions. First

- International Conference on Genetic Algorithm in Engineering Systems: Innovation and Applications (GALESIA 95), pp. 45-52.
- [10]. Delgado A., Jensen Schulte C., 2009. Generating Optimal Stowage Plans for Container Vessel Bays, Principles and Practice of Constraint Programming - CP pp. 6- 20.
 - [11]. Dubrovsky, O., Levitin, G. and Penn, M., 2002. A Genetic Algorithm with Compact Solution Encoding for the Container Ship Stowage Problem, Journal of Heuristics, 8, pp. 585-599.
 - [12]. Eckart Zitzler, Lothar Thiele, 1998. An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach. TIK-Report
 - [13]. Glover F., 1986, Future Paths for Integer Programming and Links to Artificial Intelligence, Computers and Operations Research, 13, pp. 533-54
 - [14]. Glover F., Taillard E., de Werra D., 1993. A User's Guide to Tabu Search, Annals of Operations Research, 41, pp. 3-28
 - [15]. Jens Clausen, 1999. Brand and Bound Algorithms – Principles and Examples, Department of Computer Science, University of Copenhagen, Denmark.
 - [16]. K. Deb, 2001. Multi-Objective Optimization using Evolutionary Algorithms, Wiley.
 - [17]. K. Deb, 1999. Multi-objective Genetic Algorithms: Problem Difficulties and Construction of Test Problem, Evolutionary Computation, vol. 7, no. 3, pp. 205-230.
 - [18]. Manuel Valenzuela-Rendon, Eduardo Uresti-Charre, 1997. A non-Generational Genetic Algorithm for multiobjective optimization. Proceeding of the Seventh International Conference on Genetic Algorithm, pp. 658-665.
 - [19]. Vipin Kumar, 2002. Introduction to Parallel Computing, 2nd edition, Addison-Wesley.
 - [20]. Wilson, I.D. and Roach, P.A., 1999. Principles of Combinatorial Optimization Applied to Container-Ship Stowage Planning, Journal of Heuristics, 5, pp. 403- 418.

- [21]. Wilson, I.D. and Roach, P.A., 2000. Container Stowage Planning: a Methodology for Generating Computerized Solutions, *Journal of Operational Research Society*, 51, pp. 1248-1255.
- [22]. Xiao X., Low M. Y. H., Liu F., Huang S. Y., Hsu W. J., Li Z., 2009. An Efficient Block-Based Heuristic Method for Stowage Planning of Large Containerships with Crane Split Consideration. *Proceedings of the International Conference on Harbour, Maritime & Multimodal Logistics Modelling and Simulation*, pp. 93-99.