12-15-2014

Scheduling of Container Handling Equipment in Marine Container Terminals

Narges Kaveshgar
University of South Carolina - Columbia

SCHEDULING OF CONTAINER HANDLING EQUIPMENT IN MARINE CONTAINER TERMINALS

by

Narges Kaveshgar

Bachelor of Science
University of Mohaghegh Ardebili, 2004

Master of Science
Islamic Azad University, 2008

Submitted in Partial Fulfillment of the Requirements

For the Degree of Doctor of Philosophy in

Civil Engineering

College of Engineering and Computing

University of South Carolina

2014

Accepted by:

Nathan Huynh, Major Professor

Robert Mullen, Committee Member

Juan Caicedo, Committee Member

Pelin Pekgun, Committee Member

Lacy Ford, Vice Provost and Dean of Graduate Studies

Dedication

To my Parents, Soroor and Habib and my sister Maryam.

ACKNOWLEDGEMENTS

I would like to sincerely thank my advisor, Dr. Nathan N. Huynh. Dr. Huynh is certainly a tremendous mentor for me. I would like to thank him for supporting my research and for encouraging me to grow as a research scientist. His advice, both on research and my career have been priceless. I am also grateful because there has been a very comfortable and friendly vibe in our research group, making me more productive.

I was fortunate to have accomplished faculties as committee members, Dr. Robert Mullen, Dr. Juan Caicedo and Dr. Pelin Pekgun. I am indebted to them for much needed advice and feedback on my work. I would also like to thank Drs. Kap Hwan Kim from Pusan National University and Young-Man Park from Korea Naval Academy for providing the valuable benchmark suites used for experiments in this research. I would also like to thank Drs. Frank Meisel and Christian Bierwirth from the Martin Luther University for their support in responding my questions and also for providing the optimal task sequences for the benchmark instances.

I also appreciate the support of my family, friends and colleagues in my research group who have always been a great inspiration to me.

ABSTRACT

To improve the competitiveness of marine container terminals, it is critical to minimize the makespan of a container vessel. The makespan is defined as the latest completion time among all handling tasks of the container vessel. Lower makespan (i.e. lower vessel turn time) can be achieved through better scheduling of the container handling equipment during vessel operations. The scheduling of terminal equipment is an operational problem, and a detailed schedule for each type of equipment operating in the terminal is necessary. Several studies have applied operations research techniques to optimize the processes of equipment in a terminal. This dissertation investigates three main operations in a marine container terminal, namely: quay crane scheduling, yard truck scheduling and yard crane scheduling.

The first study in this dissertation addresses the quay crane scheduling problem (QCSP), which is known to be NP-hard. A genetic algorithm (GA) was developed and tested on several benchmark instances. An initial solution based on the S-LOAD rule, a new approach for defining the chromosomes, and new procedures for calculating tighter lower and upper bounds for the decision variables were used to improve the efficiency of the GA search. In comparison with best available solutions, our method was able to find optimal or near-optimal solution in significantly shorter time for larger problems.

The second study of this dissertation addresses the quay crane scheduling problem with time windows (QCSPTW). A GA was developed to solve the problem. Unlike other

works, the proposed solution approach allows quay cranes (QCs) to move in directions independent of one another, and in certain situations, the QCs are allowed to change their directions. Using benchmark instances, it was shown that the developed GA can provide near optimal solutions in a faster time for medium and large-sized instances and provides an improvement in the solution quality for instances with fragmented time windows.

The equipment involved in each of three main operations of a container terminal are highly interrelated, and therefore, it is necessary to consider the operations of QCs, yard trucks (YTs), and yard cranes (YCs) in a holistic manner. The third study of this dissertation addresses the scheduling of QCs and YTs jointly. The integrated problem can be seen as an extension of the classical flow shop with parallel machines at each stage, which has been proved to be NP-hard. A mixed integer programming model was developed based on hybrid flow shop scheduling problem with precedence relationship between tasks, QC interference, QC safety margin, and blocking constraints. A GA combined with a greedy algorithm was developed to solve the problem. The GA solutions demonstrated that the developed integrated model is solvable within reasonable time for an operational problem.

The fourth study of this dissertation developed a robust optimization model that considers all three equipment jointly. The unique difference between the fourth study and the existing literature is that it accounts for the non-deterministic nature of container processing times by the QCs, YTs, and YCs. To deal with the uncertainty in processing times, a model was formulated based on a recent robust optimization approach (p-robust). The objective function of the proposed model seeks to minimize the nominal scenario makespan, while bounding the makespan of all possible scenarios using the p-robustness constraints. To solve the robust integrated optimization model, a GA was developed. The experimental results demonstrated that the developed robust integrated model is solvable within reasonable time for an operational problem.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACT .... Automated Container Terminal

AGV ....Automated Guided Vehicle

ALV .... Automated Lifting Vehicle

FEU ....Forty-Foot Equivalent Unit

GA.. Genetic Algorithm

HFSS ....Hybrid Flow Shop Scheduling

IT...Internal Tuck

LB .... Lower Bound

MILP .... Mixed Integer Linear Programming

NUQCSP... Non-Unidirectional Quay Crane Scheduling Problem

QC... Quay Crane

QCSP ....Quay Crane Scheduling Problem

QCSPTW .... Quay Crane Scheduling Problem with Time Window

RTGC...Rubber Tyred Gantry Crane

TEU .... Twenty-Foot Equivalent Unit

UB...Upper Bound

UNCTD.. United Nations Conference on Trade and Development

UQCSP...Unidirectional Quay Crane Scheduling Problem

XT ....External Tuck

YC.. Yard Crane

YT .... Yard Truck

CHAPTER 1

INTRODUCTION

Containerization has grown dramatically in the last decade. UNCTAD (United Nations, 2012) report indicates that the world container trade, expressed in twenty-foot equivalent units (TEUs), has grown 7.1% in 2011. The world container terminal throughput has increased by 5.9% to its highest level ever (572.8 million TEUs) in 2011. To respond to this increase in container trade, marine container terminals need to improve their level of service. Container terminals play an important role in a nation's economy, infrastructure, and quality of life by providing the link between domestic markets to international customers and visa-versa. The success of a marine container terminal is critical to the success of the intermodal freight supply chain.

The primary objective of a container terminal is to achieve the minimum vessel turn time at a minimum cost. The operation cost of a modern container vessel is around $30,000 to $40,000 per day, and therefore, vessel turn time is known to be an important factor in the overall cost of container transportation. Container terminal operations involve various processes and deployment of expensive resources. Effective scheduling of equipment is crucial in each process to obtain optimal results.

This dissertation addresses three primary processes in marine terminal operations: 1) QC scheduling, 2) YT scheduling, and 3) YC scheduling.

All of these processes are concerned with resource optimization and share the common objective of minimizing the vessel turn time to enhance the terminal's competitiveness.

1.1 RESEARCH TOPIC I – QUAY CRANE SCHEDULING PROBLEM

Due to the significant impact of the QCs on terminal throughput, the QC scheduling problem (QCSP) has received considerable attention. The first study in this dissertation addresses the QC scheduling problem which has been proved to be NP-hard. A GA is proposed to tackle the problem. The efficiency of the GA search has been improved by using an initial solution based on the S-LOAD rule developed by Sammarra et al. (2007), using a new approach for defining the chromosomes to reduce the number of decision variables, and using new procedures for calculating tighter lower and upper bounds for the decision variables. Experimental results showed that the developed GA finds the solutions in faster time for larger problems compared to the available best-known solutions. A literature review of related studies is presented in Chapter 2. Readers are referred to Chapter 3 of this dissertation for an overview of the QC scheduling problem and the proposed GA for solving the problem.

1.2 RESEARCH TOPIC II – QUAY CRANE SCHEDULING WITH TIME WINDOWS

The assignment of the QCs to vessels may result in time windows for QCs that consist of different ready times and withdrawal times. In practice, higher priority vessels may require additional QCs at certain times to expedite operations. Thus, QCs are temporarily removed from lower priority vessels. The second study of this dissertation addresses the QC scheduling problem with time windows (QCSPTW) and aims to determine the task sequence for each QC to minimize the vessel turn time while satisfying time availability of the cranes. An efficient GA is proposed for solving the QCSPTW. The proposed approach contributes to the available literature in that the cranes are allowed to move in different directions independently and are allowed to change their directions in particular cases. Numerical experiments show that the developed GA can provide near optimal solutions in a faster time for medium and large-sized instances and that the developed GA improves the solution quality (lower vessel turn time) for instances with fragmented time windows. A review of related studies is presented in Chapter 2. Readers are referred to Chapter 4 of this dissertation for more information on the QC scheduling problem with time windows and the proposed solution methodology.

1.3 RESEARCH TOPIC III – INTEGRATED QUAY CRANE AND YARD TRUCK SCHEDULING FOR UNLOADING INBOUND CONTAINERS

Most studies have optimized the processes in a marine container terminal independently. Given that there is no buffer area available at the berth, a QC cannot proceed with the next task until a truck is available to accept the container and vice versa. Thus, the operations of QCs and YTs are highly interrelated. It is necessary to develop and solve these operations in an integrated manner that reflects the characteristics of the marine container terminals. The third study of this dissertation develops a mixed integer programming model for scheduling QCs and YTs jointly. The developed model is based on the hybrid flow shop scheduling technique and it extends the existing body of work by considering multiple QCs, non-crossing constraint, and safety margins between QCs. A GA combined with a greedy algorithm is developed to solve the model. A review of related studies is presented in Chapter 2. Readers are referred to Chapter 5 of this dissertation for more information about the hybrid flow shop scheduling method, the developed integrated QC and YT scheduling model, the implementation of the solution approach, and relevant results.

1.4 RESEARCH TOPIC IV – ROBUST SCHEDULING OF TERMINAL CONTAINER HANDLING EQUIPMENT

Chapter 6 of this dissertation addresses the integrated scheduling of QCs, YTs, and YCs. An integrated model is developed that considers all three stages of vessel operation: the unloading of the containers by QCs, transport of the containers by YTs, and finally, the stacking of the containers by YCs. Important operational constraints like precedence relationship among tasks, QC interference, safety margin, and blocking are taken into account. To the best of our knowledge, this is the first study that considers non-deterministic task processing times and proposes a robust model to solve the integrated scheduling problem. A solution approach is developed to solve the robust problem, and its effectiveness is tested using numerical experiments. A literature review on related studies is presented in Chapter 2. Readers are referred to Chapter 6 for more information on the developed robust integrated model and the proposed solution approach.

1.5 LIST OF PAPERS AND STRUCTURE OF DISSERTATION

This dissertation includes four research papers published, accepted or submitted to peer-reviewed journals. The author of this dissertation is the "first author" of these articles:

1. Kaveshgar, N., Huynh, N., & Khaleghi Rahimian, S. (2012). An efficient genetic algorithm for solving the quay crane scheduling problem. *Expert Systems with Applications*, *39*(18): 13108-13117.

2. Kaveshgar, N., & Huynh, N. A Genetic Algorithm Heuristic for Solving the Quay Crane Scheduling Problem with Time Windows. Accepted by *Maritime Economics and Logistics*, 11/04/2014.

3. Kaveshgar N., & N. Huynh. Integrated Quay Crane and Yard Truck Scheduling for Unloading Inbound Containers. Accepted by *International Journal of Production Economics*, 09/17/2014.

4. Kaveshgar N., & N. Huynh. Robust Scheduling of Terminal Container Handling Equipment. Submitted to *Computers & Operations Research*, 09/17/2014.

The remaining chapters are organized as follows: Chapter 2 provides a review of marine container terminal operations and highlights the related studies. Chapters 3 to 6 include the four research topics mentioned above. Lastly, chapter 7 provides summary and concluding remarks.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

This chapter provides a broad overview of marine container terminal operations. It also presents a literature review on the four studies in this dissertation.

2.1 MARINE CONTAINER TERMINAL BACKGROUND

The cargo transported in ocean vessels could be classified into two main categories: bulk and containerized cargo. Bulk cargo is shipped using specialized vessels called bulk carriers in large quantities. Crude oil, coal, ores and grains are some examples of bulk cargo. Containerized cargo includes a variety of goods that are packed into standard-size steel containers and shipped on container vessels. The current dissertation focuses on containerized cargo. A marine container terminal is the place where vessels berth and unload inbound (import) containers and pick up outbound (export) containers. Containers are steel boxes with dimensions of $20\times8\times8.5$ (ft^3) or $20\times8\times9.5$ (ft^3), measured in TEU (20 ft. equivalent units) and $40\times8 \times8.5$ (ft^3) or $40\times8\times9.5$ (ft^3) measured in FEU (40 ft. equivalent units). Specialized containers, like refrigerated containers (used for cargo that must be kept at special cold temperatures during transit), may have slightly different size. The TEU used to be the most common container size, but the FEU is beginning to be more common (Murty et al. 2005).

Figure 2.1 illustrates the typical layout of a marine container terminal: a quay side area with berths for vessels to dock and a container yard to store containers. The storage yard is usually divided into rectangular regions called blocks.

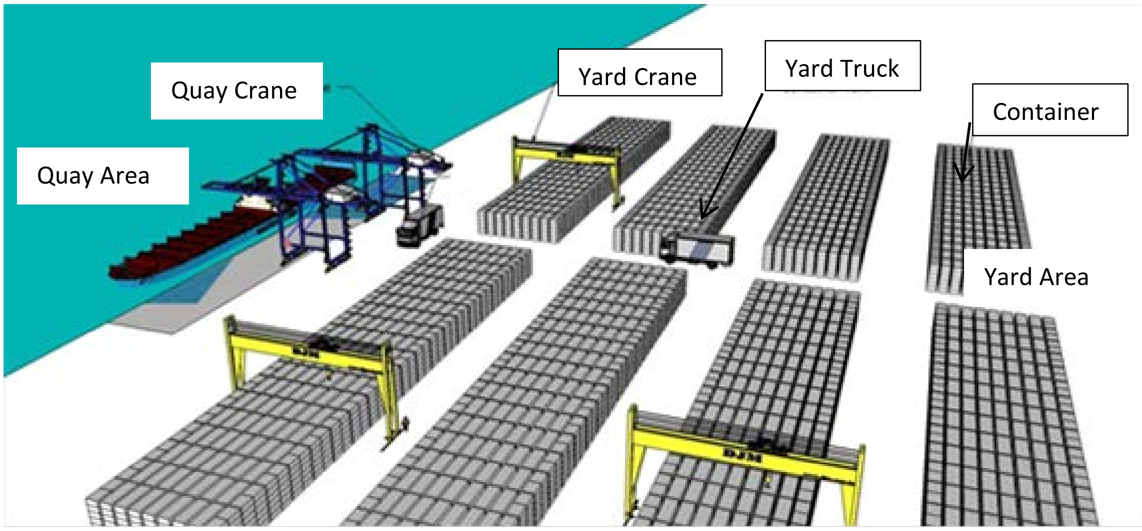


Figure 2.1 Layout of a marine container terminal.

Inside a container vessel, containers are stacked on top of each other. Figure 2.2 shows the side view of a container vessel. The vessel is divided along its length to several storage areas. The storage areas are known as holds or bays. A bay is divided vertically into two sections, below deck (hold) and on deck. The number of bays depends on the size of the container vessel and might be as high as 15. Some large container vessels are capable of carrying over 7000 TEUs (Murty et al, 2005).

Figure 2.2 Side view of a container vessel.

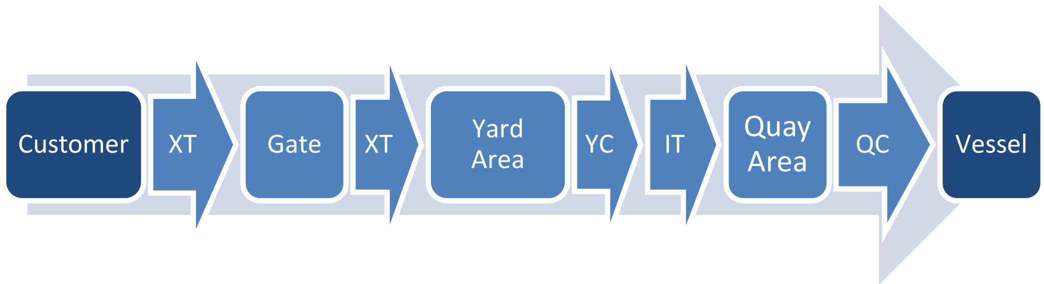
Containers are typically divided into groups. This is based on the container size, port of discharge, and container weight. A *task* is defined as a group of containers which are usually located on neighboring bays in the vessel. Operationally, there is a precedence relationship between container groups. As an instance, the unloading operation must precede the loading operation for tasks/containers located in the same ship bay. Also, when the tasks share the same bay, the unloading operation must start with the tasks located on the deck before proceeding to the ones in the hold (below deck). During the loading operation the tasks in the hold must be loaded before those on the deck.

The main functions of a container terminal are to 1) serve as an interface between ocean and land transportation, 2) receive outbound (export) containers from shippers for loading into vessels, 3) unload inbound (import) containers from vessels to be picked up by consignees and, finally 4) provide a temporary storage of containers between ocean and land transportation.

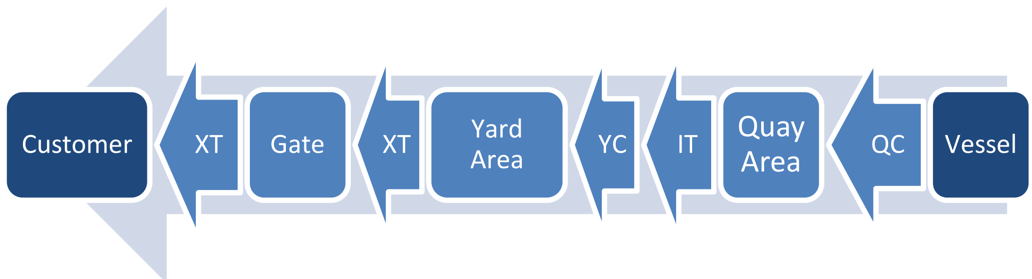
2.2 FLOW OF CONTAINERS IN A MARINE TERMINAL

To further explain the operations in a marine container terminal, it is necessary to illustrate the flow of inbound and outbound containers. The loading/unloading operation starts when a vessel berths along the quay area. The unloading operation for the inbound containers consists of three stages: 1) the QCs collect the containers from the vessel and load them onto the YTs or internal trucks (ITs), 2) the YTs transfer the containers to the YCs, and 3) the YCs store the containers in the designated blocks of yard area. The loading operations consist of the same stages in reverse order. Outbound containers brought in by customer external trucks (XTs) enter the container terminal through terminal gates. At the gates, the container and its documentation are checked. The XTs will then proceed to the storage area where the container will be stored by YCs. The container will remain in the storage area until the vessel on which it will be loaded arrives. When the vessel arrives, the YC removes the container from the stored position, puts it on an IT, and the IT takes the container to a QC for loading into the vessel. The flow of outbound containers is represented in Figure 2.3.

As shown in Figure 2.3, the flow of containers can be seen as a composition of four subsystems: 1) loading/unloading to/from vessel from/to berth (QCs operation), 2) transport to/from quay area to yard area (ITs operation), 3) storage in yard area (YCs operation), and 4) delivery/receipt to/from customer (XTs operation). Each container goes through these subsystems between the vessel and designated customer. Each of these subsystems has a container handling capacity. This capacity is based on the operational strategy in the subsystem and resources deployed. The performance of the container terminal depends on the performance of these subsystems, and a bottleneck in any of these subsystems will decrease the terminal throughput (Henesey, 2006).



(a)



(b)

Figure 2.3 Flow of (a) outbound and (b) inbound containers. (Modified from Rashidi and Tsang, 2006)

2.3 PROCESSES IN CONTAINER TERMINALS

Container terminal operations include several processes in the four subsystems mentioned previously. This section presents the various scheduling decisions, different equipment involved in each scheduling decision, and the related existing literature. The literature section starts with a comprehensive survey of the studies on container terminal operations. The work by Vis and de Koster (2003) reviewed the literature of decision problems in a marine container terminal. The problems include arrival of a vessel, unloading and loading of vessel, transport of containers from vessel to stack yard, and stacking and retrieving of containers. Another work by Steenken et al. (2004) addressed the logistic processes of storage, stacking, and transport optimization. Murty et al. (2005) studied optimization decisions, such as: berth allocation, QC allocation, XT appointment allocation, truck routing, terminal gate dispatching policies, storage space assignment, YC deployment, IT allocation to QCs, and IT hiring plans. Henesey (2006) reviewed studies on strategic, tactical, and operational level problems in a container terminal. Vacca et al. (2007) addressed five types of decision problems which include berth allocation, QC scheduling, yard operations, transfer operation, and vessel stowage planning. This dissertation studies decision problems concerned with the three primary container handling equipment types in a marine terminal: QCs, YCs, and YTs. The operations of the container handling equipment and the available literature that study their operations individually or jointly are presented in the following sections.

2.4 QUAY CRANE SCHEDULING PROBLEM

QCs load and unload containers to/from the container vessel and could be moved from one berth to another. QCs share a rail track and therefore, they cannot cross over one another. For safety, QCs are kept at a safe distance from each other. The safety distance is called safety margin and is typically one ship bay long. Figure 2.4 shows a QC which is unloading the containers from the vessel on the trucks to be moved to the storage area.

Here some studies on QC scheduling problem (QCSP) are reviewed. Daganzo (1989) addressed static QC scheduling problem assuming that only one crane can work on hold of a vessel at a time. The author aims on minimizing the vessel's aggregate cost of delay and develops exact and approximate solution techniques to solve the scheduling problem. Peterkofsky and Daganzo (1990), in a related study, proposed a branch and bound solution approach for the QCSP. Both of these studies, assumed one task per ship-bay and the interference between the QCs is not considered.

Figure 2.4 A typical QC. (Murty et al, 2005)

Bose et al. (2000) proposed evolutionary algorithms for optimizing the productivity of cranes. Kim and Park (2004) studied the QCSP with multiple tasks in a ship-bay and took into account crane interference and precedence relations among tasks. They proposed and compared a branch and bound based method and a heuristic algorithm for solving the model. Ng and Mak (2006) proposed a method that decomposes the QCSP into smaller sub-problems and partitions the vessel into a set of non-overlapping zones. Moccia et al. (2006) formulated the QCSP as a vehicle routing problem considering the precedence relationships between vertices. They proposed a branch and cut algorithm for tackling large size problems. Sammarra et al. (2007) developed a model considering the precedence and non-simultaneity constraints between tasks. They proposed a tabu search heuristic for solving the problem and compared it with a branch and cut algorithm and a greedy randomized adaptive search procedure. Lee et al. (2006) illustrated that the QCSP with crane interference is NP-hard and therefore proposed a genetic algorithm to solve the mixed-integer program. Tavakoli et al. (2009) extended Kim and Park's model to solve the QC assignment problem and scheduling problem jointly. Bierwirth and Meisel (2009) revised the interference constraints and proposed a heuristic based on branch and bound algorithm to solve the QC scheduling problem. Based on the growing number of studies on the QCSP with different models and solution methods, Meisel and Bierwirth (2011) proposed a unified approach for evaluating the performance of these studies.

As shown by Bierwirth and Meisel (2009) solution approaches that only search for unidirectional schedules cannot guarantee an optimal solution for problems with container groups. Studies on the QCSP have traditionally focused on addressing constraints such as non-crossing of QCs (e.g. Kim and Park 2004 and Lee et al, 2008), safety margin between adjacent QCs (e.g. Bierwirth and Meisel, 2009), precedence relationships among tasks (e.g. Bierwirth and Meisel, 2009 and Kim and Park, 2004), and QC travel time between bays (e.g. Bierwirth and Meisel, 2009; Kim and Park, 2004; Ng and Mak, 2006). Consideration of QC ready times and withdrawal times has only been addressed recently. Many papers have examined the berth allocation problem (e.g. Golias et al, 2009, Golias 2011, Golias et al, 2011) or berth allocation and QC assignment problem (Meisel and Bierwirth, 2006 and Chang et al, 2010) but the effect of the berth and QC assignment on QC scheduling (QC time window) was first presented by Meisel. He developed a QCSPTW mathematical model and solution approach (Meisel, 2011). His solution methodology relied on unidirectional scheduling approach previously proposed for solving the QCSP. In the work by Legato et al. (2011), the authors solved the QCSPTW that also considered the non-uniform QC productivity rate. They proposed a solution approach based on the unidirectional scheduling and Time Petri Net. In another work by Monaco and Sammarra (2011), the authors considered soft time windows along with spatial limits on cranes (i.e., some QCs have limited range within the berth). Their proposed tabu search metaheuristic also relied on unidirectional movement of the quay cranes that was evaluated with a maximum of 4 quay cranes on a single vessel.

Although the non-unidirectionality in crane work schedules (each QC is allowed to move in different directions, independent of one another), is considered in the mathematical formulation of previous studies, the proposed solution approaches do not support this type of QC movement and is restricted to unidirectional schedules. As shown by the relation below, the optimal objective function value of the QCSP is less than or equal to that of the NUQCSP (non-unidirectional QC scheduling problem) and the NUQCSP's objective function value in turn is less than that of the UQCSP (unidirectional quay crane scheduling problem).

$$Z^*(QCSP) <= Z^*(QCSPNU) <= Z^*(QCSPU)$$

2.5 YARD CRANE SCHEDULING PROBLEM

YCs stack and retrieve the import and export containers to/from the yard blocks. YCs also move the containers in the blocks. One important decision in marine terminals is to determine how many YCs need to work in each block and when to move from one block to another. Such decisions would affect the turn time of vessels and the waiting times of QCs and YTs (Rashidi Tsang, 2006). Figure 2.5 shows a rubber tyred gantry crane (RTGC). RTGC sits across the width of a container block with seven rows of container space between its legs. Six rows are used for storing the containers and the seventh will be used for truck passing. Each row consists of 26 stacks of TEUs which are stored lengthwise side by side. For storing FEUs, the number of stacks will be decreased to 13 (Murty et al, 2005).

Figure 2.5 Rubber tyred Gantry crane. (Murty et al, 2005)

Efficiency of a container terminal is highly affected by the yard operations and therefore, YC scheduling has been studied in several researches. Zhang et al. (2002) studied the dynamic deployment of RTGCs in order to minimize the total delayed workload in the yard. They proposed lagrangian relaxation as a solution approach. Kim and Kim (2002) developed a cost model to calculate the optimal storage space and the number of YCs for handling inbound containers. Ng and Mak (2005) studied scheduling of a single YC. They aimed on minimizing the sum of the task waiting times and defined tasks as loading and unloading operations with different ready times. In 2005, Ng extended the problem to multiple YCs and proposed a two-phase heuristic for solving it.

In another work, Jung and Kim (2006) extended the YC scheduling problem of single YC to multiple YCs working on a block, considering non-crossing constraints. They aimed on minimizing the makespan of the loading operations. Lee et al. (2007) studied the scheduling of two YCs serving one QC at two different container blocks, and aimed on minimizing the total transfer time in the stack area. Petering and Murty (2009) developed a discrete simulation model and investigated the effect of YCs deployment among blocks on the overall performance of the terminal. In 2010 Huynh and Vidal developed an agent-based approach to schedule YCs, with focus on assessing the impact of different crane service strategies on drayage operations. They modeled the cranes as utility maximizing agents, and introduced a set of utility functions to determine the order in which individual containers are handled. Wenkai et al. (2012) developed an efficient continuous time mixed integer linear programming model for YC scheduling. Their model considered realistic operational constraints like multiple inter-crane interference, fixed YC separation distances, and simultaneous container storage/retrievals. They proposed a heuristic and a rolling-horizon algorithm for solving the model. Gharehgozli et al. (2014) studied sequencing container storage and retrieval requests in a container terminal. They minimized the total travel time of an YC to handle requests in a block. The authors developed a continuous time integer model and proposed a two-phase solution method to solve the problem.

2.6 YARD TRUCK SCHEDULING PROBLEM

Inbound containers are transported from quay side area to storage area and outbound containers are transported from storage area to quay side area. The equipments that perform the transfer operations include: ITs or YT, straddle carriers (SCs), automated guided vehicles (AGVs) and automated lifting vehicles (ALVs). SCs are able to lift containers from the storage yard without YC assistance. Thus SCs can both transfer and stack containers. AGVs and YTs cannot lift the containers. AGVs and ALVs are usually deployed at automated container terminals, and can travel along a predefined route without a driver. ALVs in comparison with AGVs are capable of lifting a container from a buffer area without crane assistance. Though AGVs and ALVs offer high mobility, and lower labor cost, due to higher initial capital investment they are not preferable in marine terminals with low labor costs (Vis and de Koster, 2003). Current dissertation studies the scheduling of YTs.

YTs transport the containers between the QCs and YCs. In a marine terminal, there will be several vehicles to carry containers between the yard area and quay area or vice versa. The scheduling and routing of these vehicles is important for minimizing the container transportation costs and the waiting times of the QCs and YCs.

The YT operations have been studied extensively in the recent years. Meer (2000) studied the control of guided vehicles in container terminals and examined different dispatching rules under different environments. Bish et al. (2001) studied the problem of dispatching vehicles in combination with space allocation for containers in storage area. Their objective was to minimize the total time to unload all containers from the vessel. They showed that the problem is NP-hard and a heuristic method is proposed to solve it. In 2002, Huang and Hsu developed two integer programs for optimizing the dispatching of yard vehicles. They proposed two heuristic algorithms and a lagrangean relaxation to solve the models. Bish (2003) studied the truck dispatching problem in order to minimize the vessel turn time for a set of vessels. In a more recent work, Bish et al. (2005) proposed a heuristic for solving the vehicle dispatching problem in terminals for one and multiple QCs. Their proposed solution approach was able to find the optimal solution for one QC and near-optimal solution for multiple QCs. Ng et al. (2007) proposed the scheduling of YTs with sequence-dependent processing times. Some papers have focused on scheduling the handling equipment in automated container terminals and have studied the automated transporters like AVGs and ALVs. Kim and Bae (2004) developed a look-ahead dispatching method to minimize the AGV's travel time and QC's waiting time. Nguyen and Kim (2009) proposed a heuristic algorithm considering the ALV dispatching problem.

2.7 INTEGRATED QUAY CRANE, YARD TRUCK AND YARD CRANE SCHEDULING PROBLEM

So far we have reviewed the available literature on QC, YT and YC operations in a marine container terminal. This section covers the available literatures that consider QC, YT, and YC operations jointly. Studying two or more terminal operations jointly has been the focus of recent year's papers.

Meersmans (2002) developed models for integrated scheduling of QCs, AGVs, and automated stacking crane (ASCs) in an automated container terminal. The author proposed a branch and bound based algorithm and a heuristic beam search algorithm to solve the problem. Vidovic and Kim (2006) proposed two approaches to estimate the productivity of three-stage material handling systems: one continuous Markov chain model and two approximated mathematical models. Their proposed approximated models are based on probability theory. Chen et al. (2007) proposed a model for integrated QC, YT and YC scheduling in a container terminal. They proposed a tabu search algorithm for solving the model and formulated the problem as a hybrid flow shop scheduling problem with precedence and blocking constraints (HFSS-B). Zeng and Yang (2009) developed an HFSS-based model for scheduling QCs, YTs and YCs jointly. They proposed an integrated simulation and optimization method for solving the problem. Jinxin et al. (2010) modeled the operations of a QC and YTs jointly for unloading containers. Their mathematical model treated this problem as a two-step flow shop problem with a single QC working on the vessel. No precedence relationship between the tasks was considered in their problem. The authors used a ruled based heuristic and a GA to solve the problem. Lau and Zhao (2008) developed a mixed integer programming model for the integrated scheduling of QCs, AGVs and YCs. They proposed a multi-layer genetic algorithm to solve the model. Chen et al. (2013) formulated an integrated model for scheduling all three equipments (QCs, YTs, and YCs) as a constraint programming model. They proposed a three-stage algorithm to solve the model.

2.8. CONTRIBUTION TO LITERATURE

The container terminal operations are attracting more attention and the number of publications appearing in the literature is growing. Researchers develop mathematical models and propose different solution approaches to tackle the marine container terminal decision problems. It is anticipated that this research contributes to the area of container terminal operations on three critical problems: the QC, YT and YC scheduling. Optimizations of resource allocation, minimization of vessel turn time, and enhancement of the container terminal productivity are the main objectives of these problems. The contributions of current dissertation are discussed in the following sections.

QUAY CRANE SCHEDULING PROBLEM

The contributions of this study to literature are: enhancing the efficiency of GA available in MATLAB by 1) using an initial solution based on the S-LOAD rule developed by Sammarra et al. (2007) 2) presenting a new approach for defining the chromosomes representation to reduce the number of decision variables 3) using new procedures for calculating tighter lower and upper bounds for the decision variables. The benefits of using MATLAB GA in this study include reducing development time, eliminating GA coding errors, and using a platform that others could easily reproduce results and extend methodologies.

A GENETIC ALGORITHM HEURISTIC FOR SOLVING THE QUAY CRANE SCHEDULING PROBLEM WITH TIME WINDOWS

The contributions of this study to literature are: 1) allowing non-unidirectional movements of the QCs (i.e. QCs can move in different directions), 2) Allowing QCs to change their directions in certain situations to yield more realistic and flexible schedules, 3) developing a GA that can provide high quality solutions in a faster time for medium and large-sized instances, and 4) improving the solution quality for instances with fragmented time windows.

INTEGRATED QUAY CRANE AND YARD TRUCK SCHEDULING FOR UNLOADING INBOUND CONTAINERS

The contributions of this study to literature are: 1) developing a new hybrid flow shop model for scheduling multiple QCs and YTs jointly, 2) defining task (a group of containers) as decision variables for the QC scheduling stage and reducing the computation time, 3) considering QC interference and safety margin constraints in the model, 4) developing and comparing an integrated solution against a sequential approach, and 5) developing a GA that can solve the integrated model within a reasonable time.

ROBUST SCHEDULING OF TERMINAL CONTAINER HANDLING EQUIPMENT

The contributions of this study to literature are: 1) developing a robust integrated model to schedule QCs, YTs and YCs jointly while considering non-deterministic nature of container processing times, 2) formulating the robust integrated model based on a recent robust optimization approach: *p*-robust, 3) considering both assignment of inbound containers to the equipment and the processing sequence of the containers in the integrated model, and 4) developing a GA for solving the integrated robust problem which is capable of minimizing the makespan of the nominal scenario while bounding the makespan of all possible scenarios.

CHAPTER 3

AN EFFICIENT GENETIC ALGORITHM FOR SOLVING THE QUAY CRANE SCHEDULING PROBLEM[1]

ABSTRACT

This study addresses the quay crane scheduling problem (QCSP), which has been shown to be NP-complete (Lee et al. 2008). For this reason, a number of studies have proposed the use of genetic algorithm (GA) as the means to obtain the solution in reasonable time. This study extends the research in this area by utilizing the GA that is available in the latest version of Global Optimization Toolbox in MATLAB 7.13 to facilitate development. It aims to improve the efficiency of the GA search by (1) using an initial solution based on the S-LOAD rule developed by Sammarra at al. (2007), (2) using a new approach for defining the chromosomes (i.e. solution representation) to reduce the number of decision variables, and (3) using new procedures for calculating tighter lower and upper bounds for the decision variables. The effectiveness of the developed GA is tested using several benchmark instances proposed by Meisel and Bierwirth (2011). Compared to the current best known solutions, experimental results show that the proposed GA is capable of finding the optimal or near-optimal solution in significantly shorter time for larger problems.

1 Kaveshgar N., N. Huynh and S. Khaleghi Rahimian. 2012. *Expert Systems with Applications*. 39:13108–13117. Reprinted here with permission of publisher.

3.1 INTRODUCTION

Container terminals operate under several performance goals. The primary objective is to achieve rapid flow of containers at a minimum cost. As such, the time to load/unload a vessel has generally been the terminal's highest priority; the time spent by a vessel at berth is known as vessel turn time. This study deals with the quay crane scheduling problem (QCSP). Figure 3.1 shows a picture of quay cranes in position to load and unload containers to and from the vessel. For the QCSP, it is assumed that the assignment of cranes to the vessel has already been made; that is, how many cranes should be allocated to each vessel.

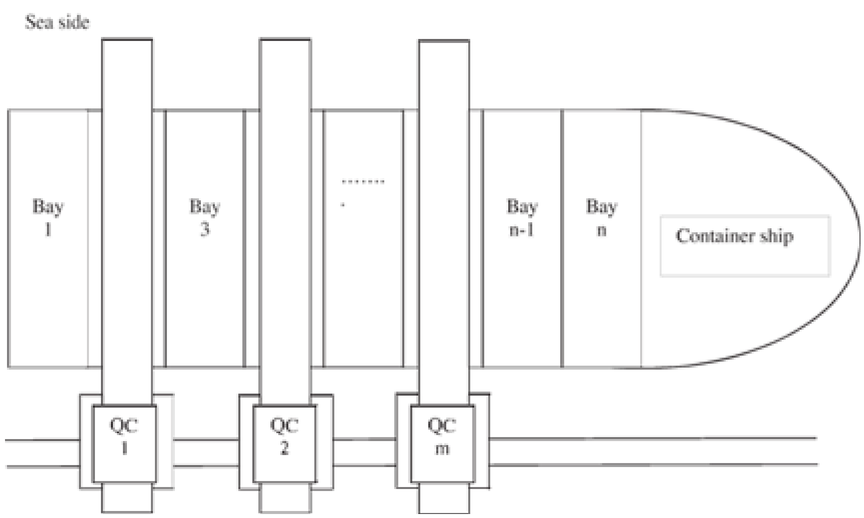


Figure 3.1 Illustration of quay crane scheduling problem.

Thus, the objective of the QCSP is to determine the task sequence for each quay crane so that the vessel turn time is minimized. As shown in Figure 3.1, a vessel is divided longitudinally into multiple bays in which the containers are stored. The vessel could have 10–50 bays. Containers stored in these bays are typically grouped together by their criteria (e.g., size, weight, origin port, destination port). A cluster is a collection of adjacent slots on the vessel which containers of the same group are stored in. As discussed in Meisel and Bierwirth (2011), there are three different classes of the QCSP problem: QCSP with container groups (highest complexity), QCSP with complete bays, and QCSP with bay areas (lowest complexity). In this study, a ‘‘task’’ is either a loading or unloading operation for a cluster, and therefore, the problem discussed in this study is from the class of QCSP with container groups which has the highest complexity among three classes.

Loading and unloading operations of containers follow a logical precedence relationship. Specifically, the unloading operation precedes the loading for the tasks located in the same ship bay. Also, for the tasks in the same ship bay the unloading operation must start first with tasks on the deck before preceding to the tasks in the hold (below deck). Conversely, for the loading operation of tasks in the same ship bay, the tasks in the hold must be loaded first before those tasks on the deck. For the QCSP, it is typically assumed that once a quay crane starts to load/unload containers to/ from the cluster, it will continue to do so until all slots in the cluster are loaded/emptied. For safety reasons, quay cranes are typically kept at a safe distance from one another. In this study, a one bay safety margin is considered. Quay cranes cannot cross over one another since they share the same track. Crane interference is accounted for in this study.

The objective of this study is to apply GA to solve the QCSP which has been demonstrated to be effective in solving the QCSP (Lee et al. 2006) as well as berth scheduling (e.g., Golias et al. 2009), yard crane scheduling (e.g., He et al. 2010), and yard truck scheduling (e.g., Ng et al. 2007). The specific aim is to obtain solutions faster than currently known approaches for larger problems using established bench mark data.

It is accomplished using MATLAB GA combined with (1) a heuristic for generating an initial solution based on the S- LOAD rule developed by Sammarra et al. (2007), (2) a new method for defining the chromosome to reduce the number of decision variables, and (3) new procedures for generating tighter lower bounds and upper bounds for the decision variables.

3.2 MATHEMATICAL FORMULATION OF THE QCSP

The mathematical formulation used for the QCSP in this study are based on the one developed by Kim and Park (2004). Modified versions of this formulation are presented in Moccia et al. (2006), Sammarra et al. (2007) and Bierwirth and Meisel (2009).

Notations

Indices:

i, j = Indices of tasks to be performed which is an increasing order according to their locations on the ship-bay

k = QC number which are in an increasing order according to their locations on the ship-bay

Problem data:

p_i = i^{th} Task processing time

n = Total number of tasks

r_k = Earliest available time of processor k (QC here)

l_i = i^{th} Task location (relative ship-bay number)

lc_0^k = Initial location of quay crane k (relative ship-bay number)

lc_T^k = final location of quay crane k (relative ship-bay number)

t = Travelling time between two adjacent bays

M = A sufficiently large positive constant

α_1 and α_2 = weight for makespan and total completion time respectively

Set of indices

Ω = Set of all tasks

Ψ= When (i, j) e W then (i, j) cannot be performed simultaneously.

Φ = Set of task with precedence relationship. When *(i, j)* $(i,j) \in \Phi$ then *i* must precede *j*.

Decision variables:

$$x_{ij}^{k} = \begin{cases} 1 & \text{Iff QC } k \text{ performs task } j \text{ immediately after task } i \\ 0 & \text{Otherwise} \end{cases}$$

$$x_{0j}^{k} = \begin{cases} 1 & \text{Iff QC } k \text{ performs task } j \text{ as its first task} \\ 0 & \text{Otherwise} \end{cases}$$

$$x_{iT}^{k} = \begin{cases} 1 & \text{Iff QC } k \text{ performs task } j \text{ as its final task} \\ 0 & \text{Otherwise} \end{cases}$$

$Y_k =$ Completion time QC k

$D_i =$ Completion time of task i

$$z_{ij} = \begin{cases} 1 & \text{Iff task } j \text{ performance starts later than the completion of task } i \\ 0 & \text{Otherwise} \end{cases}$$

W=Makespan

$|l_j - l_i|$= travel time of QC k from i*th* task to j*th* is assumed to be relative to

the ship bay numbers.

Objective Function:

$$Minimize\ \alpha_1 W + \alpha_2 \sum_{k=1}^{m} Y_k \tag{3.1}$$

Subject to:

$$Y_k \le W \qquad k = 1,\dots,m \tag{3.2}$$

$$\sum_{j=1}^{n} X_{0j}^{k} = 1 \qquad k = 1,\dots,m \tag{3.3}$$

$$\sum_{j=1}^{n} X_{jT}^{k} = 1 \qquad k = 1,\dots,m \tag{3.4}$$

$$\sum_{k=1}^{m}\sum_{i=1}^{n} X_{ij}^{k} + \sum_{k=1}^{m} X_{0j}^{k} = 1 \qquad j = 1,\dots,n \tag{3.5}$$

$$\sum_{j}^{n} X_{ji}^{k} + X_{0i}^{k} - \sum_{j}^{n} X_{ij}^{k} - X_{Ti}^{k} = 0 \qquad i = 1,\dots,n; \qquad k = 1,\dots,m \tag{3.6}$$

$$D_i + p_j + t\left|l_j - l_i\right| - D_j \le M(1 - X_{ij}^{k}) \quad i, j = 1,\dots,n,\ i \ne j \qquad k = 1,\dots,m \tag{3.7}$$

$$D_i + p_j \le D_j \qquad \forall (i, j) \in \Phi \tag{3.8}$$

$$D_i - D_j + p_j \le M(1 - Z_{ij}) \qquad i, j = 1,\dots,n,\ i \ne j \tag{3.9}$$

$$Z_{ij} + Z_{ji} = 1 \qquad \forall (i, j) \in \Psi \tag{3.10}$$

$$\sum_{v=1}^{k}\sum_{u=1}^{n} X_{uj}^{v} + \sum_{v=1}^{k} X_{0j}^{v} - \sum_{v=1}^{k}\sum_{u=1}^{n} X_{ui}^{v} + \sum_{v=1}^{k} X_{0i}^{v} \le M(Z_{ij} + Z_{ji})\ i, j = 1,\dots,n,\ i \ne j,\ l_i < l_j;\ k = 1,\dots,m \tag{3.11}$$

$$D_j + t(\left|l_j - l_{/T}\right|) - Y_k \le M(1 - X_{jT}^{k}) \qquad \forall j = 1,\dots,n;\ \forall k = 1,\dots,m \tag{3.12}$$

$$r_k - D_i + t(\left|lc_0^k - l_i\right|) + p_i \le M(1 - X_{0i}^{k})\ i = 1,\dots,n;\ k = 1,\dots m \tag{3.13}$$

$$X_{ij}^{k}, X_{0j}^{k}, X_{jT}^{k}, Z_{ij} = 0 \text{ or } 1 \qquad i,j = 1,\dots,n,\ i \ne j;\ k = 1,\dots,m \tag{3.14}$$

$$Y_k,\ D_i,\ W \ge 0 \quad i = 1,\dots,n;\ k = 1,\dots m \tag{3.15}$$

The objective function (3.1) minimizes the summation of makespan and completion time of each quay crane. Since the goal is to minimize the vessel turn time, α_1 is set to be much greater than α_2. Constraint (3.2) calculates the makespan (W). Constraint (3.3) forces each QC to choose one task as its first task after its initial state. Constraint (3.4) forces each QC to choose one task as its last and final state. Constraint (3.5) makes sure that each task is handled exactly by one QC. Constraint (3.6) is for guaranteeing a well-defined sequence for tasks. Constraint (3.7) defines the property of the completion time of each task and also eliminates the sub-tours. Assuming task j being immediately performed by QC k after task i, constraint (3.7) would avoid a sub-tour by forcing the completion time of task j to be at most equal to the completion time of task i plus the performance time of task j and the time required to travel from task i to task j. Constraint (3.8) forces task i to be completed before task j. Constraint (3.9) defines the property of Z_{ij} when needed, such that $Z_{ij} = 1$ when the performance of task j starts after the performance of task i is completed; and 0 otherwise. Constraint (3.10) ensures that the pair of jobs that are members of the set Ψ will not be handled simultaneously. Constraint (3.11) eliminates the interference among QCs. Constraint (3.12) defines the property of Y_k, the completion time of each QC. Constraint (3.13) controls the quay cranes operation starting time. It ensures that completion time of the first task by each QC would be at least equal to the time required to perform the task plus the time required to travel from the initial location of the QC to that task.

3.3 DEVELOPED GA APPROACH

Genetic algorithm (GA) is a search heuristic from the class of evolutionary algorithms that simulates the process of natural evolution such as inheritance, mutation, selection, and crossover for finding the solution for optimization problems. In this study, the GA provided in the MATLAB 2011b Global Optimization Toolbox is used which is capable of solving the mixed integer nonlinear programming. The attractiveness of using the commercial software is that it facilitates development. MATLAB provides a convenient programming environment that aids algorithm development.

The developed methodology using GA is illustrated in Figure 3.2. The procedure starts with getting the input data which consists of task processing time, location of each task, and quay cranes' initial positions. Then the upper bound and lower bound for the number of tasks that can be assigned to each quay crane and the bounds for the task number that can be processed by each QC is calculated (this procedure is explained in detail in section 3.3.1). To reduce the computation time an initial solution is utilized based on the S-LOAD rule proposed by Sammarra et al. (2007). The S-Load rule would divide the workload almost equally among the cranes. The initial solution would be considered as one of the individuals of the initial population and the remaining ones are generated randomly based on the specified lower and upper bounds (step 1 of Figure 3.2). Next, step 2 of Figure 3.2, the objective function value for every individual is calculated and stored for creating the next generation. At each iteration, until the stopping criteria are met (step 3), GA would create the next generation (step 4) and repeat the process (starting at step 2).

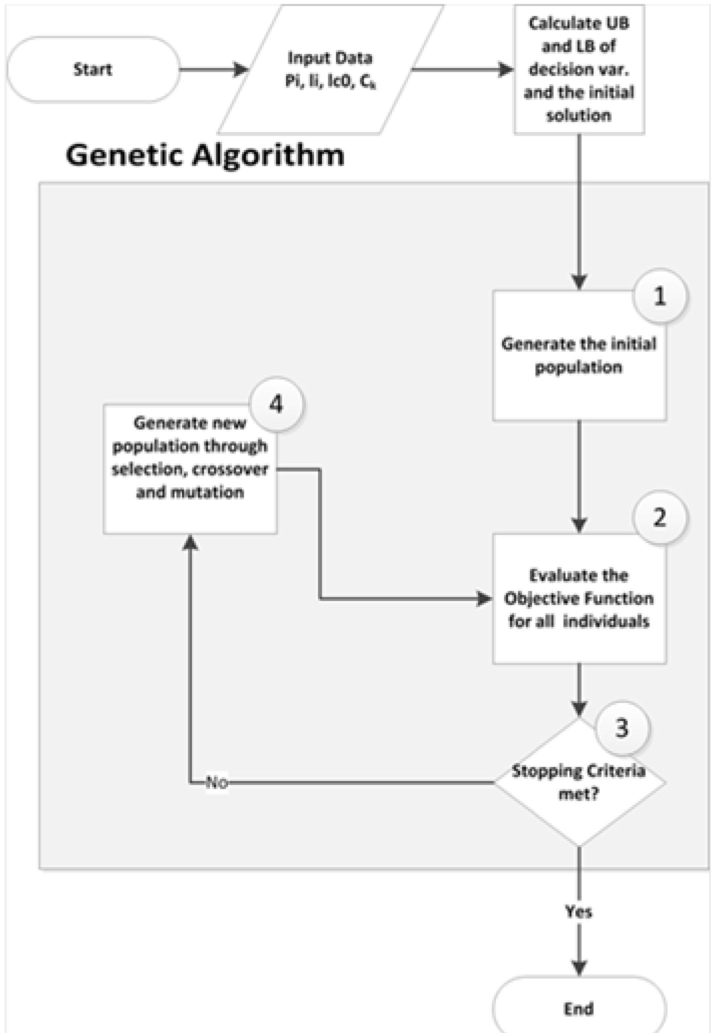


Figure 3.2 Flowchart of methodology using GA.

The four key GA steps are explained in further detail below.

1. Create a random initial population (Step 1 in Figure 3.2)

The GA procedure starts with the creation of a random initial population. The population size is a value that could be set in the Population options. In addition, this study uses the S-LOAD rule to generate an individual as part of the initial population to facilitate the search.

2. Evaluation of the objective function (Step 2 in Figure 3.2)

In order to use GA to obtain the optimal or near-optimal solution, the objective function must be defined in MATLAB. This function would accept input data and return a scalar which is the objective function value. The chromosome representation and evaluation of the objective function play a significant role in our developed method. These two aspects are explained in detail in section 3.4.1.

3. Stopping criteria (Step 3 in Figure 3.2)

A number of options could be used for the stopping criteria: Generations, Stall generations, Time limit, Fitness limit, Stall time limit and Function Tolerance. They can be specified in the Optimization Tool or in the GA options. In this study the following stopping criteria are used:

1- Generations — the number of generations reaches the value of Generations specified by user.
2- Stall generations — the weighted average change in the fitness function value over Stall generations is less than Function tolerance specified by user.
3- Function Tolerance — the algorithm will stop if the cumulative change in the objective function is less than a predefined tolerance (1e-6) over the stall generations limit. .

4. Create the Next Generation (Step 4 in Figure 3.2)

At each step, GA uses the current population as a set of parents for creating the children for the next generation. GA would select the parents that have better fitness values (lower objective function value in comparison with other individuals). At each step, three types of children are created for the next generation:

a) Elite children (best fitness values and will survive to the next generation)

b) Crossover children (created by combining the genes of a pair of parents). Other than elite children a fraction of the population in each generation, indicated by the crossover fraction, is put together as the crossover children. With a crossover fraction of 1 all children except for the elite individuals will be considered as the crossover children. On the contrary a crossover fraction of 0 would take all children as mutation children. When using the default crossover function in MATLAB (i.e. Scattered) a binary vector will be produced randomly and the genes of the first parent matching 1 in the binary vector and the genes form the second parent matching 0 in the binary vector will be selected to combine and create the child. The following example from MATLAB illustrates the crossover operation:

   If p1 = [a b c d e f g h] is parent 1 and p2 = [1 2 3 4 5 6 7 8] is parent two and v = [1 1 0 0 1 0 0 0] is the random binary vector, the created child will be:

   child = [a b 3 4 e 6 7 8].

c) Mutation children (random changes to a single parent). The mutation operation increases the genetic diversity and as a result GA searches a broader area. Mutation helps to prevent the search from stalling at a local optimum. The default mutation option in MATLAB is Gaussian. A random number is selected from a Gaussian distribution with mean 0 and is added to each parent chromosome. The amount of mutation is proportional to the standard deviation of the Gaussian distribution and will reduce at each new generation (MATLAB user's guide).

The following line of code in MATLAB will call GA and return two outputs, namely: x (i.e. quay cranes' work sequence) and fga (i.e. the objective function value). Parameters like equality or inequality constraints that are not used in this study are set to [].

```
[x fga] = ga(objfunc,N,[],[],[],[],LB,UB,[],intcon,options);
```

In the input arguments above "objfunc" is the objective function which needs to get minimized, "N" represents the number of decision variables, "LB" and "UB" are the lower bound and upper bounds for the decision variables which are explained in section 3.3.1. Lastly, "options" specify the parameters, such as population size, mutation and crossover operations, that the GA algorithm should employ.

3.3.1 CHROMOSOME REPRESENTATION

As explained previously, GA starts with a population of individuals or solutions. Each solution is called a chromosome consisting of genes that represent the decision variables. The total number of decision variables plays a significant role in the computational time; problems with more variables will require longer computation time. In the mathematical formulation presented in Section 3, the total number of decision variables for a problem size of $n \times m$ is as follows:

Decision variables in QCSP:

$X_{i,j}^{k}$, $X_{0,j}^{k}$, $X_{j,T}^{k}$, $Z_{i,j}$, Y_k, D_i, W.

$$\delta_{total} = n \times n \times m + n \times m + n \times m + n \times n + n + m + 1$$

where δ is the number of different combinations of indices for each decision variable. Equation (3.16) shows an example calculation of δ for the first decision variable $X_{i,j}^{k}$.

$$X_{i,j}^{k} \qquad \delta = n \times n \times m \tag{3.16}$$

For example, the total number of variables for a problem with 9 tasks and 2 quay cranes is 291 (162+18+18+81+9+2+1), consisting of 279 binary and 12 continuous decision variables.

In previous work by Lee et al. (2006), the authors defined a chromosome as a sequence of all possible tasks to be completed, without explicit assignment of tasks to quay cranes. In another study by Chung and Choy (2012), the authors defined a chromosome to have two parts. In the first part, from left to right, genes represent the tasks and their performance sequence whereas in the second part, they represent the quay cranes assigned to each task of the first part. Therefore, with their method, the total number of decision variables is *2n* where *n* is the total number of tasks.

Before introducing our chromosome definition, it is necessary to point out that in this study a quay crane is allowed to move in any direction, independent of the direction of other quay cranes. However, once it moves in one direction it cannot reverse direction. This concept is referred to as independent unidirectional in the literature in regard to the quay crane movement. It should be noted that this approach is more general and robust than the one proposed by Meisel and Birwirth (2009) where all cranes are constrained to move in only one direction (unidirectional).

In this study, as shown in equation (3.17), we defined a chromosome to have three parts. The first part represents the sequence of tasks assigned to the quay cranes, the second part represents the number of tasks assigned to each quay crane, and the last part represents the direction of each quay crane which is essential for creating solutions with independent unidirectional movements of the QCs.

$$S = \left[X_1 \quad X_2 \; \ldots \; X_n \quad K_1 \quad K_2 \quad K_m \quad H_1 \quad H_2 \; \ldots \; H_m \right] \tag{3.18}$$

The first n elements (X) show the sequence of tasks to be done. The elements denoted by K represent the total number of tasks that are assigned to each quay crane, and the last m elements denote the direction of each quay crane. H can only take 1 or 2 as its value. H equals to 1 means ascending (left to right) direction of the QC, and H equals to 2 means descending direction. Since m is much less than n this method of defining the chromosome reduces the total number of decision variables significantly. The total number of decision variables will be: $n+2\times m$

Figure 3.3 shows an example of a chromosome encoding. This chromosome represents a QCSP with 6 tasks and 2 quay cranes. As indicated in the first part of the chromosome, there are six tasks (numbered 1 to 6). Tasks 1, 2 and 3 are assigned to the first QC and the remaining tasks (4, 5, and 6) are assigned to the second QC. The second part of the chromosome indicates that each QC is assigned three tasks, and the third part of the chromosome indicates that both QC will move in the same direction (left to right).

3	2	1	5	6	4	3	3	1	1

Part one | Part Two | Part Three

⇩

QC number	Task sequence
1	1,2,3
2	4,5,6

Figure 3.3 Chromosome Representations.

In addition to using the above approach to reduce the number of decision variables, a lower and upper bound is defined for the number of tasks assigned to a quay crane (second part of the chromosome). The lower bound is set to one task for each quay crane and the upper bound is calculated according to the following procedure.

Step 1. Arrange the performance time of tasks in an ascending order and name the set Q

Step 2. Set $i = n - m + 1$,

Step 3. Set $a = \sum_{1}^{i} Q$ (3.18)

Step 4. Arrange tasks in descending order (set G)

Step 5. Set $b = \sum_{1}^{n-i-(k-2)} G$ (3.19)

If $b > a$ stop and set upper limit=i; otherwise set i=i-1 and go to step 3.

The above procedure will guarantee that the completion time of the quay crane assigned with the maximum number of tasks (with lowest processing time among all the tasks) will not exceed the completion time of the quay crane assigned with the highest work load.

Moreover, a lower limit and an upper limit are developed for the task numbers (first part of the chromosome) which would confine the tasks that could be done by a specific quay crane. This could be done from knowing the locations of the tasks and quay cranes and also the fact that QCs share the same track and cannot cross one another. For example, when two or more QCs are working on a ship, the tasks located on the first ship bay (left most tasks) could only be performed by the first quay crane (the one positioned on the left). The lower limit is set as follows:

For the first *(n-(m-1))* genes of the chromosome the lower bound is 1. For the next *(m-1)* remaining genes the lower limit is 2, 4, 6,…, *K*. The skip in number is due to the one bay safety margin between the quay cranes.

For the first *(m-1)* genes, the upper limit is set to *(n-K)*, *(n-K+2)*, *(n-K+4)*,…and for the remaining genes the upper limit is set to the total number of tasks.

3.3.2 CHROMOSOME VALIDATION

Before evaluating the objective function, the two important criteria that need to be satisfied are crane interference and precedence relationship of tasks. The tasks that are close to each other (i.e. violating the one bay safety margin between the quay cranes) are treated as crane interference. In Kim and Park (2004), during the first phase of the proposed heuristic the tasks that violate these constraints are excluded from the set of feasible tasks for the next operation. In Lee et al. (2006), which used a GA, if the solution does not satisfy the non-interference constraint then the fitness value of the chromosome is set to zero.

In this study, chromosomes violating the interference constraints are discarded by setting a high fitness value to them. In the cases that the precedence relationship between the tasks is violated, the tasks are swapped. For example, if task 1 should precede task 2 but in the created chromosome task 2 is performed before task 1, the tasks will be swapped and the chromosome will be adjusted according to the precedence relationship between the tasks.

The solutions obtained from the GA could be invalid for QCSP. That is, there could be two genes in the first part of the chromosome with identical values. Thus, a validation procedure is needed to ensure that the genes representing the task number in the first part of the chromosome have unique values and that the sum of the task numbers assigned to each quay crane is equal to the total number of tasks. This validation procedure was coded by the authors in MATLAB.

3.3.3 EVALUATING THE GA OBJECTIVE FUNCTION

The objective function value of a solution is dependent on the set of tasks assigned to each quay crane and the number of tasks that are going to be performed by each crane as well as the crane direction (based on the task location). For each chromosome, the objective function will be evaluated as follows:

When starting a new task a QC would check the adjacent tasks on the left and right. If there is a starting time for that task and another QC has started performing it, current QC would check the finishing time of that task. If the task is not completed yet, it has to wait; otherwise it will start that task right away. The quay cranes' destinations and completion time after performing each task is calculated as follows.

Determination of the QC's destinations

The destination is determined for each quay crane after it performs a task in its set. There are four possibilities for each quay crane: (1) quay crane travels to its assigned task and completes that task, (2) quay crane waits to avoid a collision and then traverses to its next task, (3) quay crane remains idle and will stay at its current position, and (4) quay crane remains idle will move to avoid collision with other quay cranes. Each of these possibilities would determine a quay crane's displacement and as a result the required travel time.

Evaluating each task's completion time (C_k)

This procedure is critical since there could be a displacement of a quay crane according to four possibilities noted above. Therefore, the required time for a quay crane to travel has to be added to its completion time. Consider a problem illustrated in Figure 3.4.

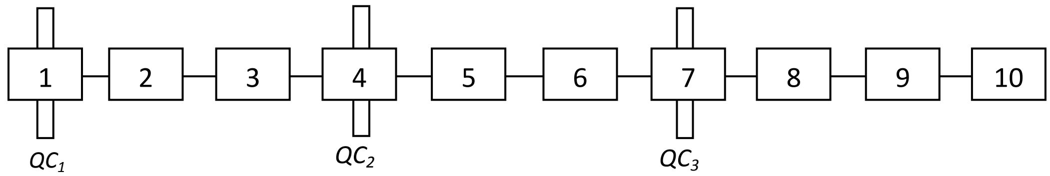


Figure 3.4 Illustration of tasks, bays and QC location for evaluating the QC completion time.

Three quay cranes and 10 tasks are shown in this figure. Each task is located on a bay with the same number and there are 10 bays. Consider a situation that tasks *1, 3* and *6* are performed by QCs *1*, *2* and *3* respectively and QCs *1* and *3* next tasks are going to be tasks *2*, and *7* and no task is assigned to QC_2. But if C_1 is less than C_2 it is not possible to perform task *2* until task *3* is finished. Therefore, QC_1 would start task *2* after C_2 time units have elapsed. Moreover, after performing task *3*, QC_2 has to be displaced and therefore, its final location is not the same as its last position (l_4 instead of l_3). The completion time of the first QC is presented in Equation (3.20):

$$C_1 = C_2 + P_2 + |l_2 - l_1| \tag{3.20}$$

Note that the objective function algorithm would consider the location of all the QCs to check for the possible interference between adjacent ones. Therefore, if there is a QC further ahead which blocks QC_2 from moving to the right (QC_3), and $C_2 < C_3$, both

QC_2 and QC_1 have to wait. The last step in the algorithm is to compute the objective function (Equation 3.1) which is the maximum completion time among all the QCs.

3.4 EXPERIMENTAL DESIGN

To evaluate the effectiveness of the developed GA methodology, the experiments used benchmark data provided by Meisel & Bierwirth (2011). In these benchmark sets, the processing time, the initial location of each quay crane and the location of each task are provided, as well as the precedence relationships between the tasks. No non-simultaneous tasks are specified in these benchmark instances. The travel time between each ship bay is 1 time unit and the Table 3.1 Input Data for Set A: Instance 1quay crane ready time is zero. In this study, sets A and F (as named by Meisel and Bierwirth) are used, with a safety margin of one ship bay. Table 3.1 shows an example of the data instance provided by Meisel and Bierwirth.

Table 3.1 Input Data for Set A: Instance 1 (Meisel & Bierwirth, 2011).

Task number	1	2	3	4	5	6	7	8	9	10
processing times (P_i)	131	190	8	69	8	2	200	192	99	101
bay locations (l_i)	1	2	3	4	4	6	7	8	10	10
precedence relations	$\Phi = [(4,\ 5), (9, 10)]$									
Number of bays	10									
QC ready times	$r_k = [0,0]$									
Initial QC location	$lc^0_k = [(1,3)]$									

In set A, all the instances have two quay cranes, and the number of tasks is varied from 10 to 40 which are located across 10 bays. In set F, all the instances have 50 tasks, located across 15 bays, and the number of quay cranes is varied from 2 to 6.

As mentioned in Section 3.4, the "options" specify the parameters for the GA algorithm, such as population size, mutation and crossover operations. The following shows the options settings used in this study.

```
options = gaoptimset('display','iter','Generations',nG,'PopulationSize',nP,'StallGenLimit',nS,'TolFun',eps,'CrossoverFraction',.9,'InitialPopulation',P0);
```

In all experiments, the stall generations (nS) are set to 200, and the Function Tolerance (eps) is set to 2.22x10e-16. The population size (nP) is set to 50 individuals and number of generations (nG) is set to 100 for set A and 200 for set F. The initial guess is an individual created using the S-LOAD rule from the initial population (P0). The experiments were conducted on a PC with 8 GB of RAM and 3.40 GHz processor.

3.5 RESULTS AND DISSCUSSION

Tables 3.2 to 3.4 compare the performance of the developed GA with the best-known solutions as reported in Meisel & Bierwirth (2011). The average gap over all instances is equal to 0.6%. Table 3.2 shows the maximum observed gap and the computation time for small to medium size problems (10 to 25 tasks) whereas Table 3.3 shows the results for medium to large size problems (30 to 40 tasks). As shown in Tables 3.2 and 3.3, the gap ranges from 0 to 1.6% for set A and the developed GA is able to find near optimal solutions in a reasonable amount of time. Table 3.4 presents the GA performance for set F. The gap is less than 2% except for the group of instances with 50 tasks 4 quay cranes. Set F shows that the gap grows as the number of quay cranes increases which is consistent with the results obtained by Lee et al. (2006). This is due to the higher probability of crane interference, which makes it harder for the GA to find the best solution. The computation time of the GA for larger problems is very promising. As it can be seen in Table 3.4, the GA computation time is about 20 times shorter than those reported in Meisel & Bierwirth (2011). It should be noted that the computation time comparison is not entirely accurate because of different computing hardware and software used in these studies. However, their relative difference and trend do tend to indicate that GA is more efficient in solving larger problems.

Figures 3.5 and 3.6 provide a graphical representation of the average difference in the computation time and average gap between the current best solution and the developed GA solution. The average gap for set A is 0.57% and except for the last set with 40 tasks there is no noticeable trend in the gap. The average gap reaches 2.84% for set F and the gap is monotonically increasing. GA offers no improvement in computation time for set A and the difference increases as the number of tasks increases. However, for set F (with more than two quay cranes working on the vessel) the improvement is noticeable (1214 seconds for the largest instance). Overall, the developed GA approach is capable of solving larger problems in a reasonable time in comparison with Kim and Park (2004) and Bierwirth and Meisel (2009) studies. The quality of the results is excellent for small to medium size problems and for larger problems the quality is acceptable. These results further corroborate that GA is suitable for solving the QCSP. Also, it validates the suitability of MATLAB GA for future studies.

Table 3.2 Comparison of GA Performance with current best solution (Set A 10 to 25 tasks).

Experiment no.	Size (cranes×tasks)	Bierwirth & Meisel (2009)		Developed GA		Gap* (%)
		Obj. value	Average Time(s)	Obj. value	Time(s)	
1		520	< 1	520	4	0.0
2		508	< 1	508	4	0.0
3		513	< 1	513	4	0.0
4		510	< 1	510	4	0.0
5	2×10	515	< 1	515	4	0.0
6		513	< 1	513	4	0.0
7		511	< 1	511	4	0.0
8		513	< 1	513	4	0.0
9		512	< 1	512	4	0.0
10		549	< 1	549	4	0.0
1		514	< 1	514	5	0.0
2		507	< 1	507	5	0.0
3		515	< 1	515	5	0.0
4		513	< 1	516	6	0.6
5	2×15	507	< 1	507	5	0.0
6		508	< 1	513	5	1.0
7		507	< 1	508	6	0.2
8		508	< 1	513	6	1.0
9		507	< 1	507	7	0.0
10		513	< 1	514	6	0.2
1		508	< 1	509	7	0.2
2		509	< 1	514	7	1.0
3		509	< 1	509	6	0.0
4		509	< 1	513	7	0.8
5	2×20	506	< 1	507	7	0.2
6		508	< 1	508	7	0.0
7		507	< 1	507	7	0.0
8		510	< 1	510	6	0.0
9		508	< 1	508	7	0.0
10		507	< 1	511	6	0.8
1		508	< 1	513	9	1.0
2		507	< 1	513	8	1.2
3		507	< 1	507	8	0.0
4		507	< 1	507	9	0.0
5	2×25	507	< 1	507	9	0.0
6		507	< 1	507	8	0.0
7		508	< 1	508	8	0.0
8		507	< 1	507	8	0.0
9		506	< 1	507	9	0.2
10		506	< 1	513	8	1.4

* Difference between the lower bound and GA objective function value in percent (GA–Lower bound)/lowerbound×100.

Table 3.3 Comparison of GA Performance with current best solution (Set A 30 to 40 tasks).

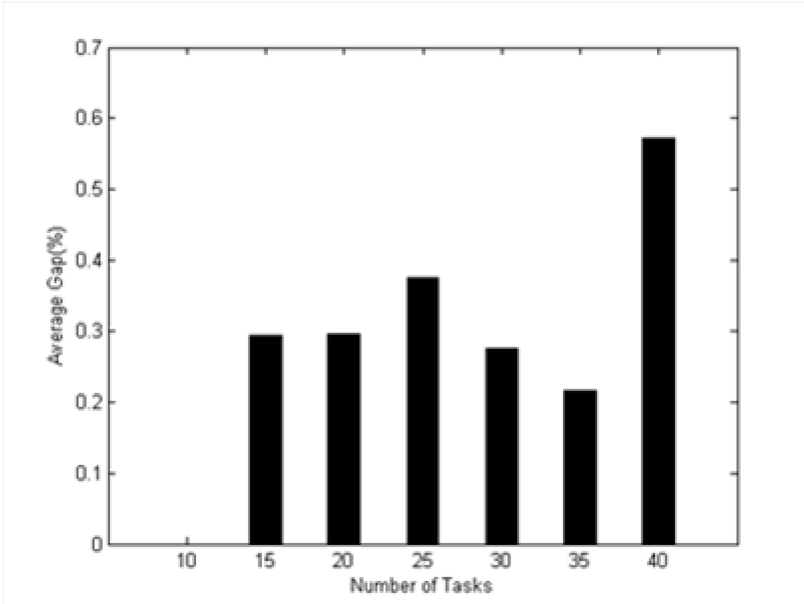
Experiment no.	Size (cranes×tasks)	Bierwirth& Meisel (2009)		Developed GA		Gap* (%)
		Obj. value	Average Time(s)	Obj. value	Time(s)	
1	2×30	506	< 1	507	10	0.2
2		508	< 1	508	10	0.0
3		507	< 1	507	10	0.0
4		507	< 1	507	9	0.0
5		506	< 1	506	10	0.0
6		506	< 1	513	9	1.4
7		508	< 1	514	9	1.2
8		508	< 1	508	9	0.0
9		506	< 1	506	10	0.0
10		506	< 1	506	10	0.0
1	2×35	506	< 1	506	11	0.0
2		507	< 1	507	11	0.0
3		506	< 1	506	11	0.0
4		507	< 1	507	10	0.0
5		507	< 1	508	11	0.2
6		511	< 1	511	11	0.0
7		507	< 1	512	10	1.0
8		506	< 1	506	11	0.0
9		506	< 1	506	11	0.0
10		508	< 1	513	11	1.0
1	2×40	506	< 1	506	12	0.0
2		506	< 1	506	12	0.0
3		505	< 1	512	12	1.4
4		507	< 1	507	11	0.0
5		506	< 1	507	12	0.2
6		507	< 1	513	11	1.2
7		507	< 1	507	15	0.0
8		506	< 1	514	12	1.6
9		506	< 1	506	12	0.0
10		507	< 1	514	11	1.4

* Difference between the lower bound and GA objective function value in percent (GA–Lower bound)/lowerbound×100.

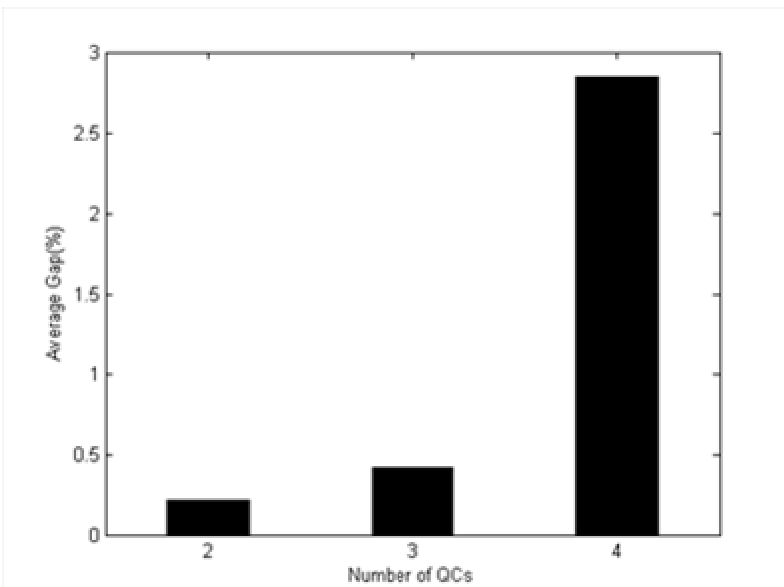
Table 3.4 Comparison of GA Performance with current best solution Set F (2 to 4 QCs).

Experiment no.	Size (cranes×tasks)	Bierwirth& Meisel (2009)		Developed GA		Gap* (%)
		Obj. value	Average Time(s)	Obj. value	Time(s)	
1	2×50	1509	11	1509	15	0.0
2		1510	11	1524	14	0.9
3		1510	11	1511	14	0.1
4		1510	11	1511	15	0.1
5		1509	11	1521	15	0.7
6		1509	11	1510	13	0.1
7		1511	11	1512	15	0.1
8		1509	11	1511	14	0.1
9		1510	11	1511	15	0.1
10		1510	11	1510	13	0.0
1	3×50	1007	199	1014	31	0.7
2		1008	199	1013	31	0.5
3		1008	199	1013	27	0.5
4		1009	199	1015	27	0.6
5		1007	199	1011	35	0.4
6		1008	199	1009	30	0.1
7		1009	199	1011	28	0.2
8		1008	199	1009	29	0.1
9		1012	199	1019	30	0.0
10		1008	199	1012	29	0.4
1	4×50	774	1248	784	33	1.3
2		771	1248	782	34	1.4
3		772	1248	784	28	1.6
4		765	1248	803	32	5.0
5		762	1248	792	34	3.9
6		765	1248	769	35	0.5
7		782	1248	782	28	0.0
8		761	1248	781	31	2.6
9		798	1248	860	35	7.8
10		759	1248	792	47	4.3

* Difference between the lower bound and GA objective function value in percent (GA–Lower bound)/lowerbound×100.

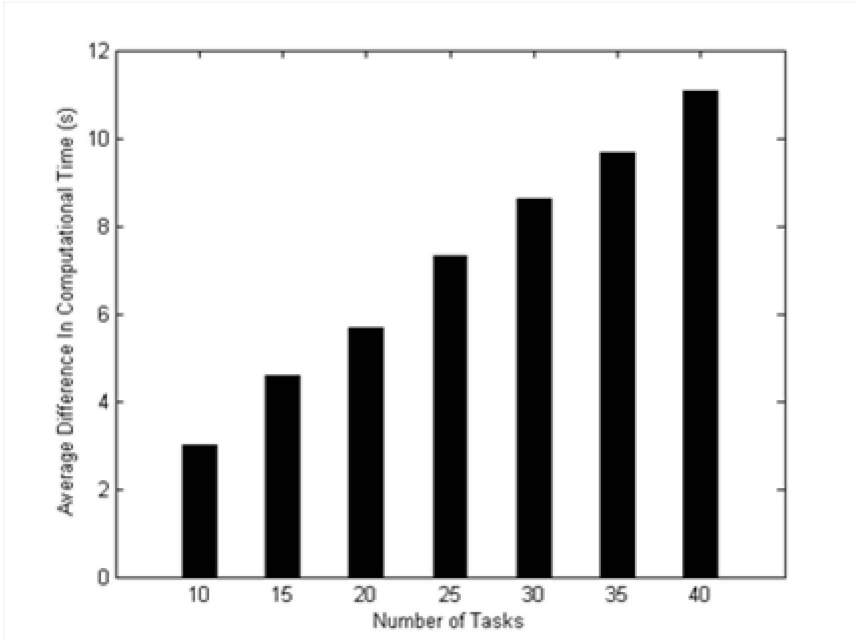


(a)

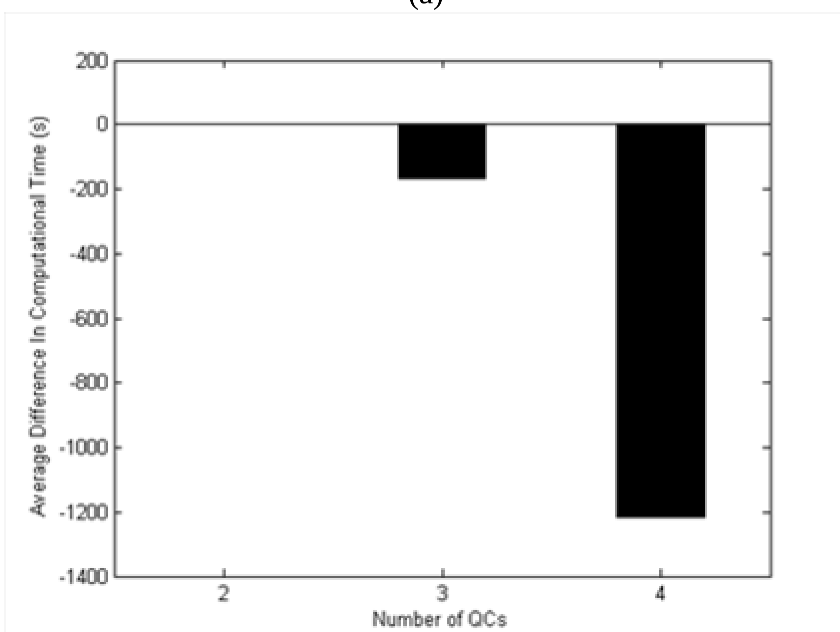


(b)

Figure 3.5 Comparison of GA performance against current best solution: a) Set A instances and b) Set F instances.



(a)



(b)

Figure 3.6 Comparison of GA computational time against current best solution: (a) Set A instances and (b) Set F instances.

3.6 CONCLUSION

The growing importance of container terminals has stimulated many studies to apply operations research models to optimize processes within a seaport container terminal. Among the processes within a container terminal, the QCSP has received considerable attention. A number of studies have proposed the use of genetic algorithm (GA) to solve the QCSP. This study contributed to this research area by utilizing the GA that is available in the latest version of Global Optimization Toolbox in MATLAB 7.13 to facilitate development. It improved the efficiency of the GA search by 1) using an initial solution based on the S-LOAD rule developed by Sammarra et al. (2007), 2) using a new approach for defining the chromosomes (i.e. solution representation) to reduce the number of decision variables, and 3) using new procedures for calculating tighter lower and upper bounds for the decision variables. Experimental results using benchmark instances showed that the developed GA provide solutions in faster time for larger problems compared to the current best-known solutions. An advantage of the developed GA methodology is that the quay cranes are not limited to unidirectional movement. This study validated the suitability of MATLAB GA for future studies. In the same spirit taken by Meisel and Birwirth to propose benchmark instances to compare QCSP solution methodologies, the same can be said about the use of MATLAB GA. That is, we would advocate the use of MATLAB GA in future studies so that it would be easier to compare the effectiveness of different proposed solution methodologies. Other advantages of using MATLAB GA include reducing development time, eliminating GA coding errors, and using a platform that others could easily reproduce results and extend methodologies.

CHAPTER 4

A GENETIC ALGORITHM HEURISTIC FOR SOLVING THE QUAY CRANE SCHEDULING PROBLEM WITH TIME WINDOWS[1]

ABSTRACT

One of the most important operations in marine container terminals is quay crane scheduling. The quay crane scheduling problem (QCSP) involves scheduling groups of containers to be loaded and unloaded by each quay crane. It also requires addressing practical issues such as minimum spacing between quay cranes and precedence relationships between container groups. This study addresses the QCSP with one additional consideration: time availability of quay cranes. This problem is referred to as QCSP with time windows (QCSPTW) in the literature. This study discusses the genetic algorithm (GA) developed to solve the QCSPTW. It builds on a previously developed GA to solve the QCSP by the authors. The results of a large set of numerical experiments using benchmark instances highlight several key characteristics of the proposed solution approach: (1) the developed GA can provide near optimal solutions in a faster time for medium and large-sized instances (overall average gap is less than 3%), and (2) the

[1] Kaveshgar N. and N. Huynh. Accepted by *Maritime Economics and Logistics*. Reprinted here with permission of publisher, 11/04/2014.

developed GA leads to an improvement in the solution quality (lower vessel turn time) for instances with fragmented time windows.

4.1 INTRODUCTION

This study deals with the quay crane scheduling problem with time windows (QCSPTW). The key difference between the QCSPTW and the more commonly known quay crane scheduling problem (QCSP) is that the QCSPTW deals with an additional constraint: time availability of quay cranes. The assignment of the quay cranes to vessels may result in time windows for quay cranes consisting of different ready times and withdrawal times. This is because higher priority vessels may require additional quay cranes at certain times in order to expedite operations, which will result in the temporary removal of quay cranes from lower priority vessels. Quay cranes' time windows are an important practical issue that must be considered in quay crane scheduling.

The objective of the QCSPTW is to determine the task sequence for each quay crane to minimize the vessel turn time or the latest time that all quay cranes are done with their assigned tasks while satisfying (1) the container groups' precedence relationships, (2) crane interference and safety requirement, and (3) time availability of the cranes. Operationally, there exists a precedence relationship among container groups.

Temporally, cranes have different time windows of availability for each vessel. In this study, up to two time windows are considered for each quay crane. That is, a quay crane may leave a vessel it is currently serving, serve another vessel and then return to resume service for the original vessel. In such a situation, the quay crane has two windows because it serves the first vessel at two different time windows.

The objective of this study is to develop a genetic algorithm (GA) solution approach for solving the QCSPTW which is known to be NP-hard. It builds on the GA previously developed by the authors (Kaveshgar et al, 2012). The modified GA has two key extensions: (1) the algorithm is modified in order to meet the time window constraints, and (2) a new procedure is developed to reassign tasks to quay cranes when they have to take on tasks left by a neighboring crane that is scheduled to leave and not return to the vessel. The key differences between this study's developed GA and other QCSPTW solution approaches are: (1) quay cranes are allowed to move in directions independent of one another, and (2) in certain situations, the quay cranes are allowed to change their directions.

4.2 MATHEMATICAL FORMULATION OF THE QCSPTW

The key parameters in the QCSPTW are the processing time of the tasks, tasks' locations in terms of bay number, quay crane location, quay cranes' assignment and precedence relationships between tasks. In this study, it is assumed that the quay cranes have uniform productivity rate and are located along the quay in an increasing sequence from left to right. The objective is to find the assignment of tasks to the quay cranes and the processing order of the tasks such that the latest completion time of the tasks (i.e. makespan) is minimized. The solution should also satisfy the aforementioned constraints. The time window method used in this study is based on the definition discussed by Meisel (2011). If k is a quay crane then TW^k is the set of time windows of that crane and u is a Time window from set TW^k which represents the ready time and withdrawal time of the quay crane. The ready time is denoted by r^{ku}, the initial position of the quay crane at the beginning of the time window is denoted as l_I^{ku}, withdrawal time as d^{ku} and the position of the quay crane at the end of the time window is denoted as l_F^{ku}. This information is assumed to be provided by the quay crane assignment plan. The following example illustrates the quay crane assignment with time windows.

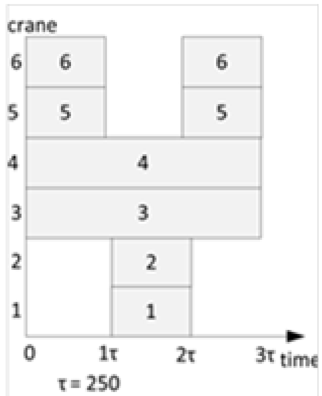


Figure 4.1 Illustration of a quay crane assignment (Meisel 2011).

The ready and withdrawal times (i.e. beginning and end times of time windows) are extracted directly from the crane assignment. For example, in Figure 4.1, cranes 3 and 4 are available throughout the vessel service time, but cranes 5 and 6 need to leave the vessel at time 250 and return at time 500. An open ended time window is assumed for the cranes that will be working to the end of the vessel's service interval. Cranes 3, 4, 5 and 6 have open ended time windows according to this definition.

In Figure 4.1, there are two groups of cranes based on their initial locations. The first group is the quay cranes that are ready when the vessel begins operation. These cranes will be arranged along the vessel, with the left most crane starting at bay 1. The second group consists of cranes that start sometime after the vessel operation has begun.

This group will be positioned according to their approaching direction. If they are going to approach from the right of a vessel (i.e. bow of vessel that has b bays) bays they have to be positioned starting at bay b + 1+δ, b + 2(1+δ), and so on where δ is the safety margin between the quay cranes. On the other hand if the quay cranes are approaching from the left (aft of vessel) then they will be positioned from 1 – (δ + 1), 1 – 2(δ + 1), and so on. The same logic is true for the cranes' final positions. For the cranes leaving the vessel before its completion time they will be positioned in accordance to the direction of their movement, at bay b + 1+δ, b + 2(1+δ) and so on if they go to the right and at bay 1 – (δ + 1), 1 – 2(δ + 1) and so forth if they go to the left. The group of quay cranes with an open ended time window will be arranged along the vessel, with the left most crane positioned at bay 1, and all other cranes positioned to its right (with each separated by the safety margin). It should be noted that these final repositioning are not necessary. They are simply done for completeness; the travel times involved in the repositioning do not affect the solution.

The following presents the mathematical formulation of the QCSPTW developed by Meisel (2011).

Notations:

Indices

i, j Indices of tasks to be performed which are in an increasing order according to their locations on the ship-bay

k Quay crane number which is in an increasing order according to their locations on the ship-bay

Problem data

p_i = i^{th} Task processing time

n = Total number of tasks

δ = Safety margin

u = A time window and $u \in \mathrm{TW}^k$

r^{ku} = Earliest available time of quay crane k (beginning of the time window)

d^{ku} = Withdrawal time of quay crane k (the end of the time window)

l_i = i^{th} Task location (relative ship-bay number)

l_I^{ku} = Initial location of quay crane k at time window u (relative ship-bay number)

l_F^{ku} = Final location of quay crane k at time window u (relative ship-bay number)

t = Travelling time between two adjacent bays

t_{Ii}^{ku} = Quay crane traveling time from its initial position at the beginning of the time window to the location of task i

t_{iF}^{ku} = Quay crane traveling time from the location of task i to its final position at the end of the time window.

M = A sufficiently large positive constant

Set of indices

Ω = Set of all tasks

Q= set of quay cranes

Φ Set of tasks with precedence relationship. When (i, j) ϵ Φ then processing of i must finish before the processing of j starts.

$\Theta = \{(i, j, v, w) \in \Omega^2 \times Q^2 \mid (i < j) \wedge (\Delta_{ij}^{vw} \geq 0)\}$ Set of all possible combinations of a pair of tasks and a pair of quay cranes which need a temporal separation of processing Δ_{ij}^{vw}.

TW^k The set of time windows for quay crane k, $u \in TW^k$

Decision variables:

$$x_i^{ku} = \begin{cases} 1 & \text{If QC } k \text{ performs task } i \text{ in time window } u \\ 0 & \text{Otherwise} \end{cases}$$

C_i = Completion time of task i

$$y_{ij} = \begin{cases} 1 & \text{If task } j \text{ processing starts later than the completion time of task } i \\ 0 & \text{Otherwise} \end{cases}$$

$C^{\max}$ = Makespan (latest completion time of all tasks)

Travel time of quay crane k from i^{th} task to j^{th} is assumed to be proportional to the ship bay numbers: $|l_j - l_i|$

Objective Function:

$$\text{Minimize } C^{\max} \tag{4.1}$$

Subject to:

$$C^{\max} \geq C_i \qquad \forall i \in \Omega \tag{4.2}$$

$$\sum_{k \in Q} \sum_{u \in TW^k} x_i^{ku} = 1 \quad \forall i \in \Omega \tag{4.3}$$

$$C_i - p_i \geq \sum_{k \in Q} \sum_{u \in TW^k} x_i^{ku} (r^{ku} + t_{Ii}^{ku}) \ \forall i \in \Omega \tag{4.4}$$

$$C_i \leq \sum_{k \in Q} \sum_{u \in TW^k} x_i^{ku} (d^{ku} + t_{iF}^{ku}) \qquad \forall i \in \Omega \tag{4.5}$$

$$\sum_{u \in TW^v} x_i^{vu} + \sum_{u \in TW^w} x_j^{wu} \leq 1 + y_{ij} + y_{ji} \qquad \forall (i, j, v, w) \in \Theta \tag{4.6}$$

$$C_i + \Delta_{ij}^{vw} - C_j + p_j \leq M(3 - y_{ij} - \sum_{u \in TW^v} x_i^{vu} - \sum_{u \in TW^w} x_j^{wu}) \quad \forall (i, j, v, w) \in \Theta \tag{4.7}$$

$$C_j + \Delta_{ij}^{vw} - C_i + p_i \leq M(3 - y_{ji} - \sum_{u \in TW^v} x_i^{vu} - \sum_{u \in TW^w} x_j^{wu}) \quad \forall (i, j, v, w) \in \Theta \tag{4.8}$$

$$C_i - C_j + p_j \le 0 \qquad \forall (i,j) \in \Phi \tag{4.9}$$

$$C_i \ge 0 \qquad \forall (i,j) \in \Omega \tag{4.10}$$

$$x_i^{ku}, y_{ij} \in \{0,1\} \qquad \forall i,j \in \Omega, k \in Q, u \in TW^k \tag{4.11}$$

In this formulation, Δ_{ij}^{vw} is defined as the minimum time span between the processing of tasks *i* and *j* assigned to quay cranes *v* and *w*.

According to Bierwirth and Meisel (2009) and Meisel (2011), Δ_{ij}^{vw} will be calculated as follows. Additional information about Δ_{ij}^{vw} could be found in the aforementioned papers.

$$\Delta_{ij}^{vw} =$$

$$\begin{cases} (l_j - l_i + \delta_{vw}) \cdot t, & \text{if } v > w \text{ and } i \ne j \text{ and } l_i < l_j + \delta_{vw} \\ (l_i - l_j + \delta_{vw}) \cdot t, & \text{if } v < w \text{ and } i \ne j \text{ and } l_i > l_j - \delta_{vw} \\ |\, l_i - l_j \,| \cdot t & \text{if } v = w \text{ and } i \ne j \\ -M & \text{otherwise} \end{cases} \tag{4.12}$$

δ_{vw} is the smallest acceptable distance between the position of the two quay cranes v and w.

$$\delta_{vw} = (\delta + 1) \cdot |\, v - w \,|$$

The objective function (4.1) minimizes the vessel handling time, which is the latest completion time of all tasks. Constraint (4.2) calculates the vessel handling time ($C^{\max}$). Constraint (4.3) forces each task to be chosen by exactly one quay crane during one of its time windows. Constraint (4.4) ensures that the task is not started earlier than the assigned quay crane time window considering the travel time between the initial position of the quay crane and the task location. Constraint (4.5) guarantees that each crane will be at its final position before the withdrawal time. Constraints (4.6) through (4.8) ensure no conflict between the quay cranes. In particular, constraint 6 ensures that no two tasks belonged to the set *Θ* can be processed concurrently, and constraint 4.7 allows sufficient time between the completion of one task and the starting of the next task. In constraint (4.8), if both tasks are done by the same crane then a duration equal to the processing of the tasks are inserted as well as the time required for the crane to move from bay location li to bay location l_j.. Finally, constraint (4.9) ensures that precedence relationships between the tasks are respected. The remaining constraints specify the range of the decision variables.

In the work by Meisel (2011), three additional constraints were considered to reduce the computation time.

$$C^{\max} \geq x_i^{ku} \cdot (r^{ku} + t_{Ii}^{ku}) + \sum_{j \in \Omega^v} X_j^{ku} \cdot P_j \qquad \forall i \in \Omega, \forall k \in Q, \forall u \in TW^k \tag{4.13}$$

$$\sum_{j \in \Omega^v} x_j^{ku} \cdot P_j \leq d^{ku} - r^{ku} - \left| l_I^{ku} - l_F^{ku} \right| \cdot t \qquad \forall k \in Q, \forall u \in TW^k \tag{4.14}$$

$$\sum_{j \in \Omega^v} x_j^{ku} \cdot P_j \leq d^{ku} - r^{ku} - x_i^{ku} (t_{Ii}^{ku} - t_{iF}^{ku}) \cdot t \ \forall i \in \Omega, \forall k \in Q, \forall u \in TW^k \tag{4.15}$$

Constraints (4.13) and (4.14) guarantee that the total processing time of the tasks will not exceed the capacity of the assigned time window. Constraint (4.15) relates a crane's travel time with its capacity within the time window (i.e. the capacity decreases each time the crane moves to a different bay).

4. 3 PROPOSED GENETIC ALGORITHM

In this study, the GA provided in the MATLAB 2011b Global Optimization Toolbox is modified and used. Figure 4.2 shows the proposed GA framework. The details of the GA steps are presented in Chapter 3. The key modifications made to the GA in Chapter 3 to solve the QCSPTW are:

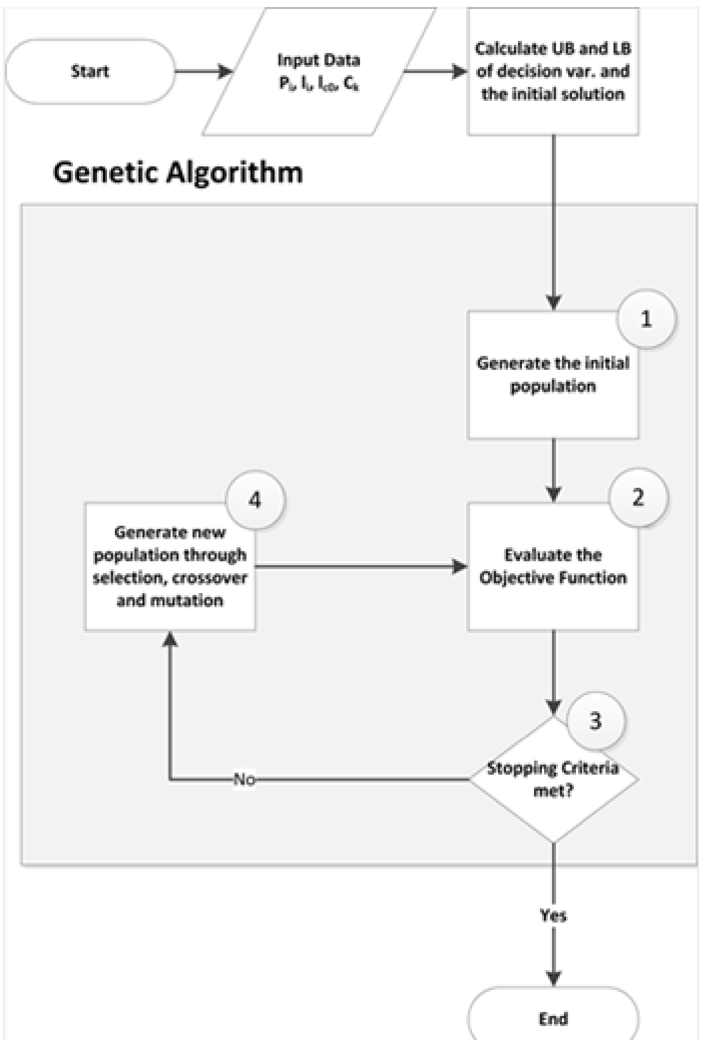


Figure 4.2 Flowchart of methodology using GA (Kaveshgar et al, 2012).

4.3.1 INITIAL SOLUTION

The GA involves generating an initial solution which is based on the S-LOAD rule proposed by Sammarra et al. in (2007) is used. The S-Load rule divides the workload equally among the cranes, but to make it work for the QCSPTW a modification is required. This study adopts the modification proposed by Meisel (2011) in which the initial solution only considers quay cranes with an open-ended time window and the workload is divided only among the open-ended quay cranes. This modification ensures that the set of tasks assigned to a quay crane would not violate its capacity.

4.3.2 CHROMOSOME REPRESENTATION

Each solution in GA is called a chromosome and each gene represents a decision variable. The chromosome shown in Equation (4.16) consists of three kinds of genes: *Xs* are the sequence of tasks assigned to the quay cranes, *Ks* represent the number of tasks assigned to the quay cranes, and finally *Hs* represent the movement direction of each quay crane (ascending or descending).

$$s = [x_1\, x_2 \ldots x_{n_1}\; k_1 k_2\, k_{m_1}\; h_1\, h_2\, h_{m_1}] \qquad (4.16)$$

4.3.3 OBJECTIVE FUNCTION EVALUATION

A function is developed to simulate the operations of QCs with time window. This function receives input data (number of QCs and tasks, task locations, QC initial locations, safety margin, precedence relationship between tasks, time windows, etc.) and returns the makespan of each chromosome.

Time window in Objective Function

In the QCSPTW, cranes may have multiple time windows. These time windows enforce temporal restrictions on quay cranes. In this study, if a quay crane schedule involves a time window, two situations may occur. In the first situation the quay crane temporarily leaves the vessel and would come back to resume its work schedule. In this case, the remaining tasks, if any, will be completed by the same quay crane in the next time window. In the second situation, the quay crane will leave the vessel and will not come back. Thus, the remaining tasks will need to be assigned to the nearest quay crane that will continue to work on the vessel. Reassignment of the tasks to another quay crane is a part of the chromosome validation procedure and ensures that a quay crane's schedule would not violate its time window constraints. If a quay crane will process a departing neighboring crane's remaining tasks, these tasks need to be rearranged in order to decrease the travel time. An example is illustrated in Figure 4.3.

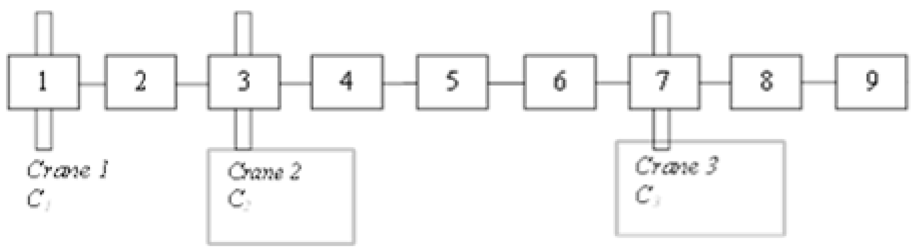


Figure 4.3 Example illustrating how reassignment of tasks is performed.

Consider a situation in Figure 4.3 in which the set of tasks that should be done by crane 2 is 3, 4, 5, and 6 and after processing tasks 3 it has reached the end of its time window and along with crane 1 leaves the vessel. Tasks 4, 5 and 6 are the set of remaining tasks that will be assigned to crane 3. The original set of tasks for crane 3 is 7, 8 and 9. In order to reduce the travel time of crane 3, the new set of tasks needs to be rearranged. That is, we want to arrange the tasks such that crane 3 would continue to travel in the ascending direction to finish tasks 8 and 9, and then move to task 6 and move in the descending direction to finish tasks 5 and 4. In situations when a quay crane takes on a neighboring crane's tasks, our GA heuristic will rearrange tasks such that it will result in the crane traveling fewer numbers of bays to finish all of its assigned tasks and, if necessary, change movement directions.

4.3.4 STOPPING CRITERIA

Different stopping criteria could be used for terminating the GA. In this study, the following stopping criteria are used:

- Generations — the algorithm stops after reaching a certain number of generations.
- Stall generations — the algorithm stops if the weighted average change in the objective function value over Stall generations is less than Function tolerance.
- Function Tolerance — the algorithm stops when the cumulative change in the objective function value over Stall generations is less than or equal to the Function tolerance (1e-6) over the stall generations limit.

4.4 ILLUSTRATION OF DIFFERENCES BETWEEN SOLUTIONS OF PROPOSED APPROACH AND OTHER APPROACHES

As mentioned, a key difference in this study's developed solution approach is that the quay cranes are allowed to move in different directions, but they are not allowed to change their directions, except when they need to complete tasks of a neighboring quay crane. Figures 4.4a and 4.4b show the difference between a unidirectional solution generated by others studies' method and the non-unidirectional constraint solution generated by this study's method. The example problem used set A benchmark data provided by Meisel (2011) which is presented in Table 4.1.

Table 4.1 Input data for set A: instance 1 (Meisel 2011).

Task number	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
processing times (p_i)	20	33	52	64	7	69	40	139	7	8
bay locations (l_i)	1	2	2	2	2	3	4	5	5	6
Task number	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
processing times (p_i)	67	2	4	30	52	8	59	125	24	190
bay locations (l_i)	6	6	6	7	7	8	8	9	9	10
precedence relations	Φ = [(2, 3), (2, 4), (3, 4), (2, 5), (3, 5), (4, 5), (8, 9), (10, 11), (10, 12), (11, 12), (10, 13), (11, 13), (12, 13), (14, 15), (16, 17), (18, 19)]									
Number of bays = 10; safety margin = 1, crane travel time = 1										

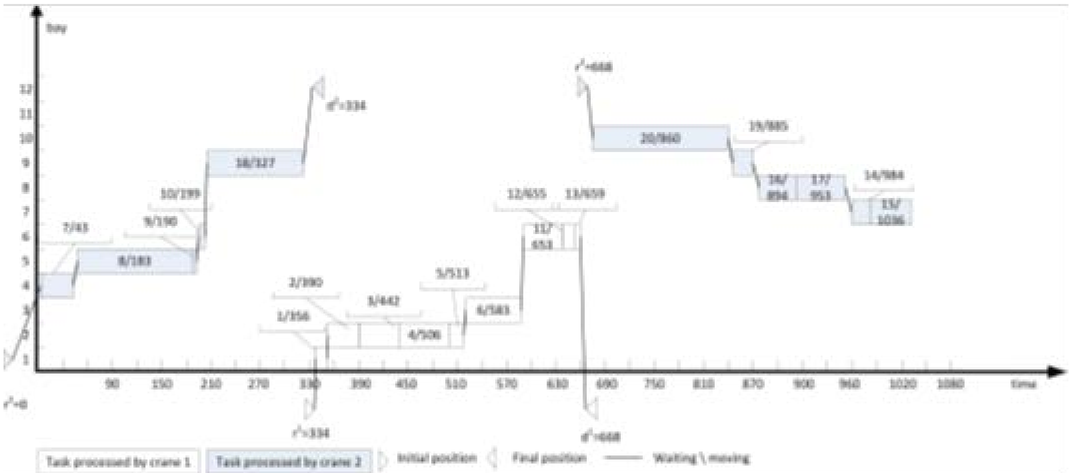
Crane *k*	**Time window *u***	**Ready time r^{ku}**	**Withdrawal time d^{ku}**	**Initila position l_I^{ku}**	**Final position l_F^{ku}**
1	1	334	668	-1	-1
2	1	0	334	1	12
2	2	668	M	12	1

The example problem consists of 2 quay cranes and 20 tasks. The quay cranes follow pattern IV illustrated in Figure 4.5. In this pattern, crane 2 is available at the beginning, but it would leave the vessel after some time and then resume its work and will be available until the end of service (open ended time window). Crane 1 is only available during one time window and is not available at the beginning of the service.

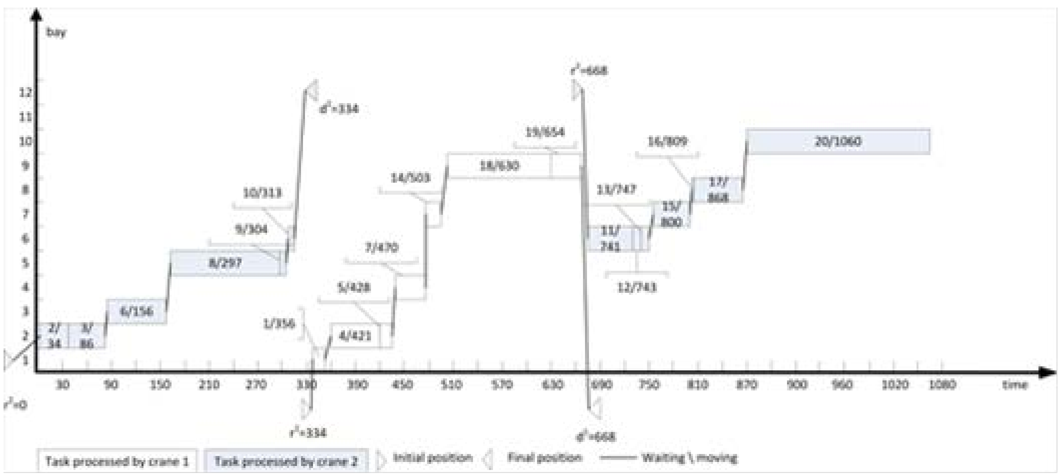
Both solutions shown in Figures 4.4a and 4.4b are obtained using GA with the same parameters. Since the schedules are obtained from GA, it provides a good reference for comparing the difference in solution characteristics. Both solutions are feasible since the precedence relationship, safety margin, the interference of the quay cranes and the time windows are respected. In Figures 4.4a and 4.4b, two numbers are shown in each box. They denote the task number and the time in which that the quay crane is done with that task. In Figure 4.4a, crane 2 resumes the assigned tasks in the second time window. Because there has been a reassignment of the tasks from crane 1 to crane 2, though its previous direction is ascending, it is allowed to change direction in the second time window. This means that it can perform tasks that are located on lower bay numbers than its final task and it can start from a task that is closer to its initial location in the second time window (bay 12).

This method results in less travel time between the bays. Moreover, due to the fact that the quay crane can resume its jobs in a different direction in a new time window, it can more efficiently use the capacity of the time window.

As shown in Figure 4.4a, crane 2 would travel up to bay 9 to process task 18 and therefore can use 327 time units of the first time window. For the solution shown in Figure 4.4b using the unidirectional approach, because of the constraint to maintain the same direction throughout the service crane 2 has limited choice and can only use 313 time units of the first time window. That is, the unidirectional constraint forces a crane moving in the ascending direction to start its work from a bay higher than the one it has left at the end of its first time window. For this reason, it is less effective in certain scenarios.



(a)



(b)

Figure 4.4 Solution for QCSPTW: (a) using proposed approach; (b) using unidirectional approach.

4.5 EXPERIMENTAL DESIGN

The proposed modified GA is evaluated using the set of benchmark instances proposed by Meisel (2011). These instances provide information such as processing time, bay location of the tasks, total number of bays, ready times and initial location of the quay cranes, and the precedence relationships between the tasks. These instances can be generated using QCSPgen which is available online at http://prodlog.wiwi.uni-halle.de/qcspgen.

This study uses three different sets of benchmark instances in which the travel time between each ship bay is 1 time unit and the safety margin is 1 ship bay. These are the same data used in the study by Meisel (2011) from which this study seeks to compare the results against. These three sets correspond to different sizes of vessels: small, medium and large. Each set includes 10 different instances. A summary of the characteristics of these sets is presented in Table 4.2.

Table 4.2 Parameters of the set of instances (Meisel 2011).

Set	**Description**	**Number of instances**	**Number of bays b**	**Number of tasks n**	$\sum P_i$	**Bay capacity**
A	Vessel of small size	10	10	20	1000	200
B	Vessel of medium size	10	15	50	3000	400
C	Vessel of large size	10	20	80	6000	600

Figure 4.5 provides the time window patterns used in the experiments, which were proposed by Meisel (2011). As shown in Figure 4.5, there are four different time windows and three different number of quay cranes (2, 4 and 6). Each time window specifies the ready times, withdrawal times, and the initial and final positions of the cranes. In pattern I, all the cranes are available at the beginning of the service and then half of the cranes are removed. In pattern II, one set of cranes start the service and they are replaced by another set of cranes at a later time. Cranes in patterns III and IV have more than one time window. In pattern III, all the cranes are available to start the service but after a while a subset of the cranes leave the vessel temporarily (a vessel with a higher priority must be served by these cranes). In pattern IV, the average number of the

cranes serving a vessel is kept constant while some of them are temporarily removed and replaced with new cranes. All patterns, except for pattern III, have quay cranes that leave a vessel permanently, and consequently, if there are any remaining tasks left by these cranes, they will be reassigned to the neighboring cranes.   τ in Figure 4.5 represents a point in time (ready times and withdrawal times of the cranes). The value τ is selected according to the total processing time of the tasks in a QCSPTW instance. It will guarantee that the capacity of the cranes assigned to a vessel meets the total processing time of all the tasks in a set of instances. Thus, three different values τA, τB and τC are provided for three set of instances A, B and C. For example, τB is set to 1000 for pattern IV with two cranes, which means that the total capacity of the two cranes becomes equal to 3000 time units. This is equal to the ∑Pi (task processing time) for set B provided in Table 4.2 (Meisel 2011).

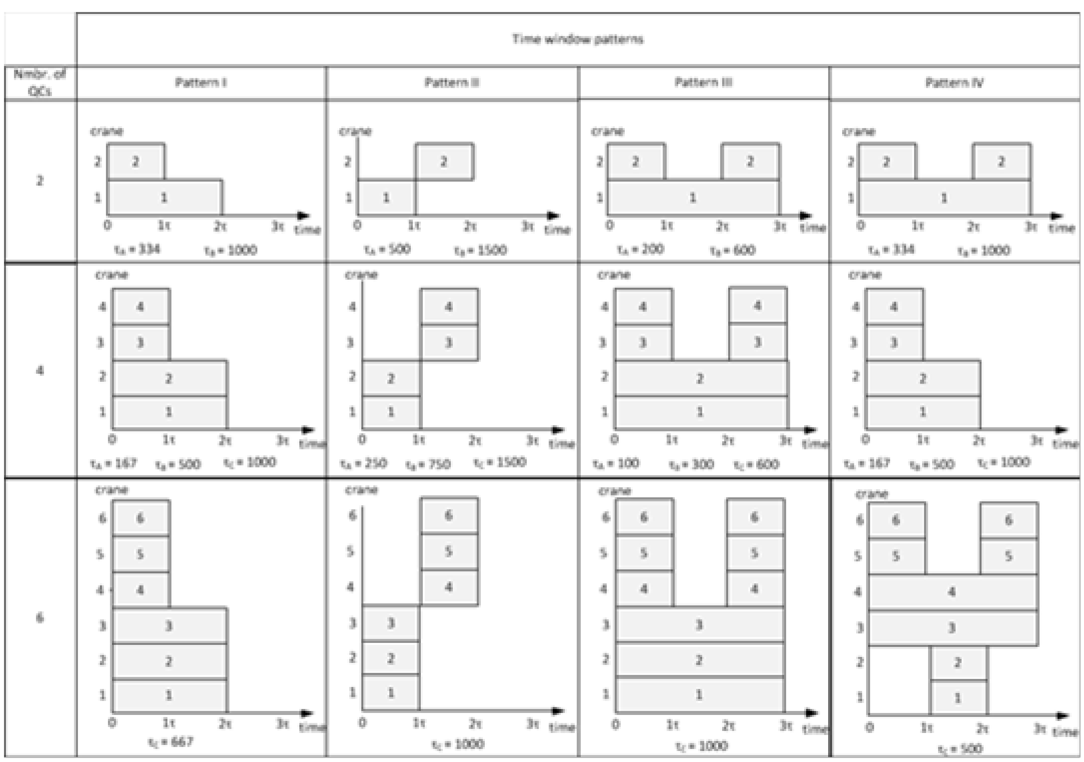


Figure 4.5 Different quay crane assignments used in experiments (Meisel 2011)

The “options” specify the parameters for the GA algorithm. These parameters include population size, mutation and crossover operations. The following shows the parameter values used in the experiments.

Options=gaoptimset('display','iter','Generations',nG,'PopulationSize',nP,'StallGenLimit',nS,'TolFun',eps,'CrossoverFraction',.9,'InitialPopulation',P0);

The stall generation (nS) is set to 150, the Function Tolerance (eps) is set to 2.22x10e-16, and the population size (nP) is set to 50 individuals in all experiments. Number of generations (nG) is set to 150. The experiments were conducted on a PC with 4 GB of RAM and 2.80 GHz processor. In total, 240 QCSPTW instances were tested and the results were compared with the ones obtained by Meisel (2011).

4.6 RESULTS AND DISCUSSION

The results obtained from the developed GA are presented in Table 4.3. The objective function values and the computation times are compared against the results reported in Meisel’s work (2011) which used UDSTW (the unidirectional search heuristic for the QCSPTW). The first two columns show the time window pattern used in the problem and the number of quay cranes. The reported gap measures the difference between the non-unidirectional method solution and the ones obtained by Meisel (2011) and are averaged over ten instances in a set. In Meisel’s work (2011), two different computational times are reported. The second one is achieved after activating the lower bounds proposed in their approach which yield considerably faster computation times than the one without these lower bounds. The computation time achieved by this study’s non-unidirectional method is compared with the lower computation time reported in

Meisel's work (2011). The solution method used in Meisel's work (2011) has a runtime limit of ten minutes for each instance.

For set A with 20 tasks and with 2 and 4 quay cranes the gap ranges from 0.1% to 4.9% and the average gap is 2.7%; a negative gap indicates that this study's approach yields a lower objective function value. The developed non-unidirectional method improved the solution quality in the set of instances using time window pattern IV with 2 quay cranes. The main reason that contributed to better solutions (i.e. lower objective function value) is the cranes' ability to change directions as illustrated in Figures 4.4a and 4.4b.

For set B with 50 tasks and with 2 and 4 quay cranes the average gap is 1.05%. The proposed non-unidirectional method yields better solutions (lower objective function values) in comparison to the best available solution for instances with two quay cranes and time window patterns III and IV. The minimum gap is -0.4%, for instances with 2 quay cranes and time window pattern IV. Similar to the previous set (set A), the greatest gap occurred in instances involving a high number of quay cranes (4 quay cranes for set B). The maximum gap is 5.0%. Except for one instance (in pattern II with 2 cranes) the proposed non-unidirectional method's computation times are lower than the UDSTW's obtained by Meisel. The maximum improvement in the computation time is achieved for an instance with 4 quay cranes and pattern II in which the computation time of the proposed non-unidirectional method is 16 times faster than the UDSTW. The average computation time is about 10 times faster than the UDSTW.

For set C with 80 tasks and with 4 and 6 quay cranes the average gap is 3.1%. The higher average gap is due to instances involving 6 quay cranes. In all sets, the gap grows as the number of quay crane increases. These results are consistent with the results reported by Lee et al. (2008) and Kaveshgar et al. (2012). It could be concluded that when the number of tasks are constant, increasing the number of cranes highly affects the solution quality obtained by GA. It should be noted that in reality it is rare to have 6 quay cranes on a vessel, but it was done here for comparison purposes. For set C, the maximum gap is 6.9 %. In all instances in set C the UDSTW run time limit is reached (10 minutes) and the proposed non-unidirectional method achieves lower computation time. On average, the proposed non-unidirectional method is about 9 times faster than the UDSTW for set C.

Time window patterns III and IV are more fragmented, meaning that these patterns have more than one time windows. Pattern III and IV represent the situations in which a couple of QCs leave the current vessel for a certain amount of time to serve a vessel with higher priority and after they finish working on the vessel with higher priority, they would resume their work on the current vessel. It can be concluded from the results reported in Table 4.3 that the proposed non-unidirectional method is more successful in finding higher quality solutions (i.e. lower objective function values) in comparison to the unidirectional method for time window patterns III and IV. Generally speaking, the proposed non-unidirectional method is promising in situations when QCs' time windows are more fragmented. More specifically, the proposed non-unidirectional method works better in situations where a QC has two time windows because it does not require the QC to resume its work on the same bay where it left to service another vessel and therefore result in more flexible schedules with less QC travel times.

Table 4.3 Comparison of non-unidirectional method performance against current best solution.

	Pattern	Number of QCs	UDSTW (Meisel 2011)		Developed GA		Gap* (%)
			$C_{UDSTW}^{\max}$	Average Time(s)	$C_{GA}^{\max}$	Time(s)	
Set A	I	2	684.1	< 1	684.4	14	0.0
		4	392.5	1	399.9	19	1.9
	II	2	1020.9	< 1	1029.7	13	0.9
		4	525.1	2	548.4	18	4.4
	III	2	625.1	< 1	634.2	15	1.5
		4	381.8	21	412.2	21	8
	IV	2	1045.9	< 1	1036.5	14	-0.1
		4	396.4	3	415.7	20	4.9
	Average		**634.0**	**4**	**645.1**	**17**	**2.7**
Set B	I	2	2025.2	277	2025.1	35	0
		4	1021.8	587	1034.8	40	1.3
	II	2	3027.3	6	3035.6	30	0.3
		4	1528.8	600	1553	37	1.6
	III	2	1828.6	84	1826.7	34	-0.1
		4	953.3	577	1001	40	5.0
	IV	2	3052.1	72	3040.2	31	-0.4
		4	1048.0	600	1055.6	38	0.7
	Average		**1810.6**	**350**	**1821.5**	**36**	**1.05**
Set C	I	4	2031.5	600	2040	57	0.4
		6	1461.2	600	1548.7	68	6.0
	II	4	3036.6	600	3054.7	56	0.6
		6	2063.6	600	2126.1	69	3.0
	III	4	1841.2	600	1901.1	60	3.2
		6	1348.9	600	1441.8	76	6.9
	IV	4	2034.7	600	2064.6	57	1.5
		6	1641.1	600	1691.5	71	3.1
	Average		**1932.4**	**600**	**1983.6**	**64**	**3.1**

$^*(C_{GA}^{\max} - C_{UDSTW}^{\max}) / C_{UDSTW}^{\max} \times 100$

4.7 CONCLUSION

This study developed an efficient GA for solving the QCSPTW. The proposed approach differs from the work of Meisel (2011) and Legato et al. (2011) in that cranes are allowed to move in different directions independently and that cranes are allowed to change their directions in specific situations. The results of a large set of numerical experiments using benchmark instances highlight several key characteristics of the proposed solution approach: (1) the developed GA can provide near optimal solutions in a faster time for medium and large-sized instances (overall average gap is less than 3%), and (2) the developed GA leads to an improvement in the solution quality (lower vessel turn time) for instances with fragmented time windows.

From a practical point of view, unidirectional schedules may require all the QCs to work at one end of the vessel. This may make them inapplicable in actual practice. Based on this study's findings, additional research is needed to investigate the effectiveness of the non-unidirectional method on fragmented time windows. Since the quay crane assignment enforces temporal and spatial restrictions on the quay crane, it can be inferred from this study's findings that the quay crane allocation problem directly affects the quay crane scheduling problem. Thus, future research should consider these two problems jointly. Additionally, the potential of using the non-unidirectional method for solving integrated problems (considers two or more terminal operational problems jointly) should be explored. Integrated models are much more complicated to solve, and thus, researchers will need to explore distributed and parallel computing techniques to reduce the computation time.

CHAPTER 5

Integrated Quay Crane and Yard Truck Scheduling for Unloading Inbound Containers[1]

ABSTRACT

To lower vessel turn time, it is crucial that the operations of quay cranes, yard trucks, and yard cranes are well coordinated. Most studies have sought to optimize each of these processes independently. Since the operations by quay cranes and yard trucks are highly interrelated it is necessary to develop and solve these operations in an integrated manner that reflects the characteristics of the marine container terminals. This study developed a mixed integer programming model for scheduling quay cranes and yard trucks jointly. The integrated model explicitly considered real-world operational constraints such as precedence relationships between tasks, blocking, quay crane interference, and quay crane safety margin. To solve the integrated optimization model, a genetic algorithm (GA) combined with a greedy algorithm was developed. The results indicated that the solutions obtained from the integrated model are superior to those obtained from the non-integrated approach. The GA solutions demonstrated that the

[1] Kaveshgar N. and N. Huynh. Accepted by *International Journal of Production Economics*, Reprinted here with permission of publisher, 09/17/2014.

developed integrated model is solvable within reasonable time for an operational problem.

5.1. INTRODUCTION

Containerization has grown dramatically in the last decade. UNCTAD (United Nations 2012) states that the world container trade expressed in TEUs (twenty-foot equivalent units) has grown 7.1% in 2011 and the world container terminal throughput has increased by 5.9% to its highest level ever (572.8 million TEUs) in 2011. Marine container terminals respond to this increase in container trade by improving their level of service. The most critical objective for terminal operators is to lower the vessel turn time (i.e. the total time a vessel spends at the terminal).

Figure 5.1 shows the typical layout of a marine container terminal. It consists of a quay side area with berths for vessels to dock and a container yard to store containers. The containers are stored in yard blocks that are six wide, four high, and 40 TEU long.

As illustrated in Figure 5.1, there are three primary container handling equipment in a terminal: quay cranes, yard cranes, and yard trucks. Quay cranes (QCs) are responsible for loading and unloading containers to and from the vessel. Yard cranes (YCs) are responsible for stacking and retrieving the export and import containers to and from the yard blocks. Yard trucks (YTs) are responsible for transporting the containers between the quay cranes and yard cranes. Containers on a vessel are typically segregated into groups based on port of discharge, container size, and container weight. These container groups (i.e. inbound/import containers, referred to as *tasks*) are usually located on adjacent bays in the vessel.

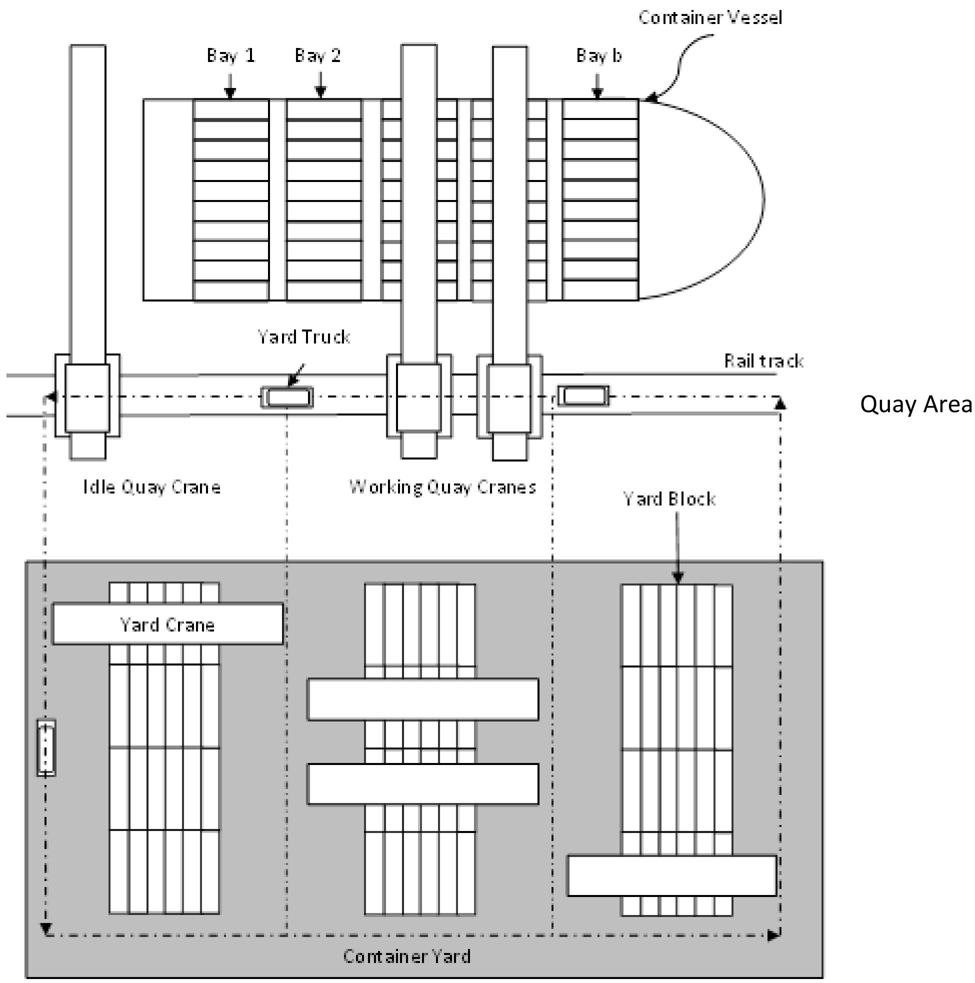


Figure 5.1 Typical layout of a marine container terminal.

Once a vessel berths along the quay area and is secured, vessel operations commences. The unloading process involves three stages: (1) the QCs pick up containers from the vessel and loads them onto the YTs, (2) the YTs transfer the containers to the YCs, and (3) the YCs stack the containers in the designated yard blocks. Any delay in these three stages would increase the overall vessel turn time. Hence, it is essential that the scheduling of tasks between these processes are coordinated.

Most studies have addressed the optimization of QCs, YCs, and YTs operations independently of one another. Recently, researchers have examined different combinations of the QCs, YTs, and YCs processes jointly, in part due to the advances in computing technology. In particular, a few studies have examined the QCs and YTs operations jointly (Jinxin et al. 2010), and QCs, YCs and YTs jointly (Chen et al. 2007 and 2013). Due to the complexity in modeling and solving the integrated models, these papers simplified the scope of the problem by considering just one QC and ignored real-world operational constraints such as precedence relationships between tasks, quay crane interference, and safety margin.

This study addressed these limitations for the integrated QC-YT problem. To this end, it developed a new mathematical formulation and solution method to solve the integrated model. The developed model is more comprehensive than existing integrated models. Specifically, it considered multiple quay cranes and also the operational constraints such as precedence relationships between tasks, blocking, quay crane interference, and quay crane safety margin. The developed solution approach utilized a greedy algorithm to find initial solutions and a genetic algorithm to find optimal solutions.

5. 2. MATHEMATICAL MODELING

The objective of the integrated QC-YT problem is to find the assignment of tasks to the equipment (QCs and YTs) and the processing order of the tasks in such a way that the latest completion time of the tasks (i.e. makespan) is minimized while satisfying precedence relationships between tasks, blocking, quay crane interference, and safety margin. Precedence relationships between tasks require that some tasks be completed before others. For example, unloading operation must start with tasks on the deck before proceeding to the tasks in the hold (below deck) when the tasks share the same bay. There is no buffer area available for quay cranes and trucks. Therefore, they have to finish the current task in order for the equipment in the next stage to start processing it; if not, it is considered blocking. As illustrated in Figure 5.1, QCs travel on the same rail track and thus cannot cross one another. This is an important operational characteristic that limits the movement of the QCs and presents situations where cranes interfere with one another. For safety reasons, quay cranes are typically kept at a safe distance from one another, called safety margin. In this study, the number of quay cranes assigned to a vessel, processing time of the tasks, locations of tasks in terms of bay number, locations of quay cranes, precedence relationships between tasks, and travel time of trucks are assumed to be known in advance. Also, quay cranes are assumed to have a uniform productivity rate and are located along the quay in an increasing sequence from left to right.

In practice, the loading and unloading of containers onto and off the vessel are done separately and in stages. That is, some import containers are unloaded and then the empty spaces are filled with export containers. This process is repeated until all import containers are discharged and all export containers are loaded. In this study, only the unloading procedure is considered (for import containers); however, the developed model could also be used for loading procedure.

The integrated QC-YT problem is formulated based on the hybrid flow shop scheduling (HFSS) problem. In a HFSS, there are *n* tasks in a set called Ω that need to be processed in *t* consecutive stages. Each stage has *K* identical machines $k_t = |K(t)|,\ t = 1, 2, \ldots, T$. Each stage *t*, has $k_t \geq 1$ parallel identical machines and for at least one stage $k_t \geq 2$. The processing time of task *i* at stage *t* is denoted as p_{it} and each machine can only process one task at a time. The tasks have a unidirectional flow through the shop. The objective is to assign the tasks to the machines and sequence the assigned tasks in order to minimize the makespan. In practice, a QC operates on one group of containers (i.e. tasks) at a time until all containers in the group are all unloaded. Thus, it is only necessary to solve the problem at the task level in the QC stage. This fact is used in this study to reduce the number of decision variables in the first stage (quay crane scheduling). However, in the second stage (yard truck scheduling) each individual container is considered a task.

Problem Parameters

Task = group of containers (first stage) or individual container (second stage)

$t =$ Stage index

$\Omega_t =$ Set of tasks at stage t

$B =$ Set of first container number in a task $\{b_i\}$

$H =$ Set of last container number in a task $\{h_i\}$

$R =$ Set of the containers in a group of containers, other than the first container in that group

$b_i =$ First container number in a task *i*

$h_i =$ Last container number in a task *i*

$n_t =$ Total number of tasks in stage *t*

$i, j =$ Task index

$K_{it} =$ Set of machines at stage t (QCs in stage 1 and YTs in stage 2)

k = Machine index

k_t = The number of machines at stage t

Φ = The set of precedence constrained tasks

l_i = Bay position

l_0^k = Initial position of quay crane (k)

f_{kt} = Earliest available time of machine k at stage (t)

p_{it} = Processing time of task i at stage t (1, 2). At stage 2 it is the required time for transporting container i to its storage location in yard.

g = QC travel time

δ = Safety margin

M = Large positive number

$\Theta = \{(i, j, v, w) \in \Omega^2 \times Q^2 \mid (i < j) \wedge (\Delta_{ij}^{vw} \geq 0)\}$ Set of all possible combinations of a pair of tasks and a pair of quay cranes which need a temporal separation of processing Δ_{ij}^{vw}.

d = The required time for unloading a container from yard truck by a yard crane

Travel time of quay crane k from i^{th} task to j^{th} task is assumed to be relative to the ship bay numbers. $|l_j - l_i|$

Dummy containers and tasks are denoted by *0* and *T*

Decision variables

$$x_{ijt}^k = \begin{cases} 1 & \text{Iff machine } k \text{ performs task } i \text{ and task } j \text{ consecutively at stage } t \\ 0 & \text{Otherwise} \end{cases}$$

$$z_{ij} = \begin{cases} 1 & \text{Iff task } i \text{ starts after the completion of task } i \text{ at satge } t \\ 0 & \text{Otherwise} \end{cases}$$

$c_{it} =$ Completion time of task i at stage t

$s_i =$ Starting time of task i at stage t

$c_T =$ Makespan (latest completion time of all tasks)

Objective Function

$$\text{Minimize } C_t \tag{5.1}$$

Constraints

$$\sum_{j \in \Omega_t^T} x_{0jt}^k = 1 \qquad \forall t \in \{1,2\}, \forall k \in K_t \tag{5.2}$$

$$\sum_{i \in \Omega_t^0} x_{iTt}^k = 1 \qquad \forall t \in \{1,2\}, \forall k \in K_t \tag{5.3}$$

$$\sum_{k \in K_t} \sum_{j \in \Omega_t^T} x_{ijt}^k = 1 \qquad \forall i \in \Omega_t, \forall t \in \{1,2\} \tag{5.4}$$

$$\sum_{j \in \Omega_t^0} x_{ji}{}^k - \sum_{j \in \Omega_t^T} x_{ij}^k = 0 \qquad \forall i \in \Omega_t, \forall k \in K_t, \forall t \in \{1,2\} \tag{5.5}$$

$$g \times \left| l_{b_i} - l_{b_j} \right| + p_{b_j t} - c_{b_j t} + s_{h_i} \leq M \times (1 - x_{ijt}^k) \, \forall i, j \in \Omega_t, k \in K_t, t = 1 \tag{5.6}$$

$$s_{h_i t} + p_{jt} - c_{h_j t} \leq 0 \qquad \forall (i, j) \in \Phi, t = 1 \tag{5.7}$$

$$s_{h_i t} + p_{jt} - c_{h_j t} \leq M(1 - z_{ij}) \qquad \forall i, j \in \Omega_t, t = 1 \tag{5.8}$$

$$s_{h_j t} - p_{jt} - c_{h_i t} \leq M(z_{ij}) \qquad \forall i, j \in \Omega_t, t = 1 \tag{5.9}$$

$$\sum_{u \in \Omega_t^0} x_{uit}^v + \sum_{u \in \Omega_t^0} x_{ujt}^w \leq 1 + z_{ij} + z_{ji} \qquad ((i, j, v, w) \in \Theta), t = 1 \tag{5.10}$$

$$s_{h_i t} + \Delta_{ij}^{vw} + p_{jt} - c_{bjt} \leq M(3 - z_{ij} - \sum_{u \in \Omega_t^0} x_{uit}^v - \sum_{u \in \Omega_t^0} x_{ujt}^w)((i, j, v, w) \in \Theta), t = 1 \tag{5.11}$$

$$s_{h_j t} + \Delta_{ji}^{vw} + p_{it} - c_{b_i t} \le M(3 - z_{ji} - \sum_{u \in \Omega_t^0} x_{uit}^{v} - \sum_{u \in \Omega_t^0} x_{ujt}^{w}) \; ((i, j, v, w) \in \Theta), t = 1 \quad (5.12)$$

$$f_{kt} - c_{b_j t} + g \times \left| l_0^k - l_{b_j} \right| + p_{b_j t} \le M \times (1 - x_{0jt}^k) \quad \forall j \in \Omega_t, \forall k \in K_t, t = 1 \quad (5.13)$$

$$c_{it} - c_{Tt} \le M(1 - x_{iTt}^k) \qquad (\forall i \in \Omega_t, t = 2, k \in K_t) \quad (5.14)$$

$$p_{ri} - c_{rit} + s_{(ri)-1} \le 0 \qquad (\forall i \in \Omega_t, t = 1) \quad (5.15)$$

$$s_i \ge c_{it} \qquad (\forall i \in \Omega_{t+1},\; t = 1) \quad (5.16)$$

$$s_j + (1 - x_{ijt}^k)M \ge c_{it} + p_{it} \qquad (\forall i, j \in \Omega_t,\; t = 2, k \in K_t) \quad (5.17)$$

$$s_j + (1 - x_{0jt}^k)M \ge 0 \qquad (\forall j \in \Omega_t,\; t = 2, \forall k \in K_t) \quad (5.18)$$

$$s_i + p_{it} + d \le c_{it} \qquad (\forall i \in \Omega_t,\; t = 2) \quad (5.19)$$

Model Explanation

Equation (5.1) is the objective function which seeks to minimize the makespan. In constraints (5.2) each machine chooses one dummy task (*0*) as its first task at each stage. Constraints (5.3) force each machine to choose one dummy task (*T*) as its last task at each stage. Constraints (5.4) ensure that each task is done by one and only one machine at each stage. Constraints (5.5) define the sequence of the tasks. Each task has only one task before and after itself at each stage. Constraints (5.6) determine the completion time for each task (container group) and eliminate sub-tours. This is done by relating the completion time of the first container in a group (stage 1) to the time that the last container of previous group is delivered to a yard truck plus the quay crane transport time between these two tasks and processing time of the first task of the current group by QC. Constraints (5.7) require that task *i* be completed before task *j* if they belong to set

Φ. Constraints (5.8) define z_{ij} such that z_{ij} is equal to 1 in the case that the operation of task *j* starts after the operation of task *i* is complete, and 0 otherwise. Constraints (5.9) define z_{ij}, meaning that if z_{ij} is 0, then task *j* can start before the completion time of the task *i*. Constraints (5.10) guarantee that tasks *i* and *j* are not processed simultaneously by *v* and *w*. If $z_{ij} + z_{ji} = 0$ (meaning that these two tasks are done simultaneously) the other side of the equation needs to be less than or equal to 1, which means the two tasks are not done by quay cranes *v* and *w*. In the case $z_{ij} = 1$, constraints (5.11) would insert the necessary time computed by Equation (5.20) that has to be elapsed before two adjacent quay cranes can process two tasks close to each other. Constraints (5.12) work in the same manner as (5.11) but for the case $z_{ji} = 1$. Constraints (5.13) restrict the earliest starting time of operations by each QC (defines the completion time of the first container of the task). Constraints (5.14) make sure that the dummy task *T* is selected as the last task. Constraints (5.15) compute the completion time of the containers which are in a group. The completion time of the first container in a group is calculated by constraints (5.16). Constraints (5.17) state that a container can be transported (stage 2) only after it has been processed by the QC (stage 1). Constraints (5.18) work in the same manner as Constraints (5.17), but for dummy task *0*. Constraints (5.19) state that a container incurs travel time to the yard (stage 2) and stacking time.

In this formulation Δ_{ij}^{vw} is defined as the minimum time span between the processing of tasks *i* and *j* assigned to quay cranes *v* and *w*. Constraints (5.10), (5.11) and (5.12) are based on constraints A.11 to A.13 from (Bierwirth and Meisle 2009).

According to (*14*), Δ_{ij}^{vw} is calculated as follows:

$$\begin{cases} (l_i - l_j + \delta_{vw})\,.\,t, & \text{if } v < w \text{ and } i \neq j \text{ and } l_i > l_j - \delta_{vw} \\ (l_j - l_i + \delta_{vw})\,.\,t, & \text{if } v > w \text{ and } i \neq j \text{ and } l_i < l_j + \delta_{vw} \\ 0 & \text{otherwise} \end{cases} \quad (5.20)$$

δ_{vw} is the smallest acceptable distance between the position of the two quay cranes v and w.

$\delta_{vw} = (\delta + 1).|v - w|$

Additional information about Δ_{ij}^{vw} could be found in the aforementioned paper.

5.3. PROPOSED GENETIC ALGORITHM

In this study, the GA provided in the MATLAB 2012b Global Optimization Toolbox is used but modified for the current problem. This version of GA has the capability to solve mixed integer nonlinear programming problems and has been used in Kaveshgar et al. (2012).

The overall GA framework is shown in Figure 5.2. Before GA starts, two lower and upper bounds are calculated for the task numbers and the number of tasks that can be processed by each equipment (see Section 3.3 for more details). In order to reduce the computation time, a two-stage heuristic is used to create a high quality initial solution for the GA. The initial solution has two parts. The first part is used for the first stage of the problem (quay crane scheduling) and is based on the S-LOAD rule proposed by Sammarra et al. (2007). The second part is used for the second stage of the problem (yard truck scheduling) and is based on a heuristic proposed by Bish et al. (2005) and used in (Lee et al. 2008). Bish et al. (2005) showed that their heuristic finds the optimal solution for problems with one QC, but they also showed that due to its "myopic" nature it could only find near optimal solutions for multiple QCs. We extended Bish's look-ahead rule to work for multiple QCs. It is essentially a greedy algorithm. Below we first explain how the look-ahead rule work and then explain the improvement we made to it.

Let $i_{q,j}, q = 1,2 \; and \; j = 1,2,...,n$ denote the *j*th task in the task sequence of quay crane *q*. Each task is represented in terms of its traveling time between the ship and its location in the storage yard area. Each vehicle that arrives at the ship area looks for the earliest task (container) that is ready to be picked up. When there are multiple QCs, it may happen that multiple tasks are ready at the same time. In Bish et al. (2005), a weight is assigned to each task and the truck picks the task with the highest weight. If each QC *q* has l_q tasks, let $p \le l_q$ be a fixed number and a weight is assigned to each task $i_{q,j}, \; for \; k = 1,...,l_q$ as follows:

$$w_{q,j} = \sum_{k=j}^{\min\{j+p,l_q\}} i_{q,k} \tag{5.21}$$

The weight represents the minimum required time to finish the remaining tasks on QC *q*'s list and depends only on the travelling time between the ship and the containers' location in the storage yard area. The calculated weight in Equation (5.21) does not consider the QC completion time in the first stage. In the case of multiple QCs, it would be better if the YTs were to serve the QC that has a longer makespan and that makespan is the completion time for both stages of the problem. Thus, we have modified Equation (5.21) to give a higher priority to the tasks that belong to the QC that has a longer makespan. Specifically, *i* is modified to include the processing time for both stages. The makespan of each QC is approximated by combining the QC completion time (obtained from the initial solution for the first stage) and YT traveling time.

The initial solutions were used as one of the individuals in the GA initial population. The remaining ones were generated randomly but the specified lower and upper bounds were applied when generating them. In Step 2, the objective function value for every individual was calculated. These values were stored for creating the next generation. Until the stopping criteria are met (step 3), the GA algorithm continued with the creation of new generations (step 4) and repeated the process starting at step 2. The process for creating the next generation is explained in Section 3.3 of this dissertation.

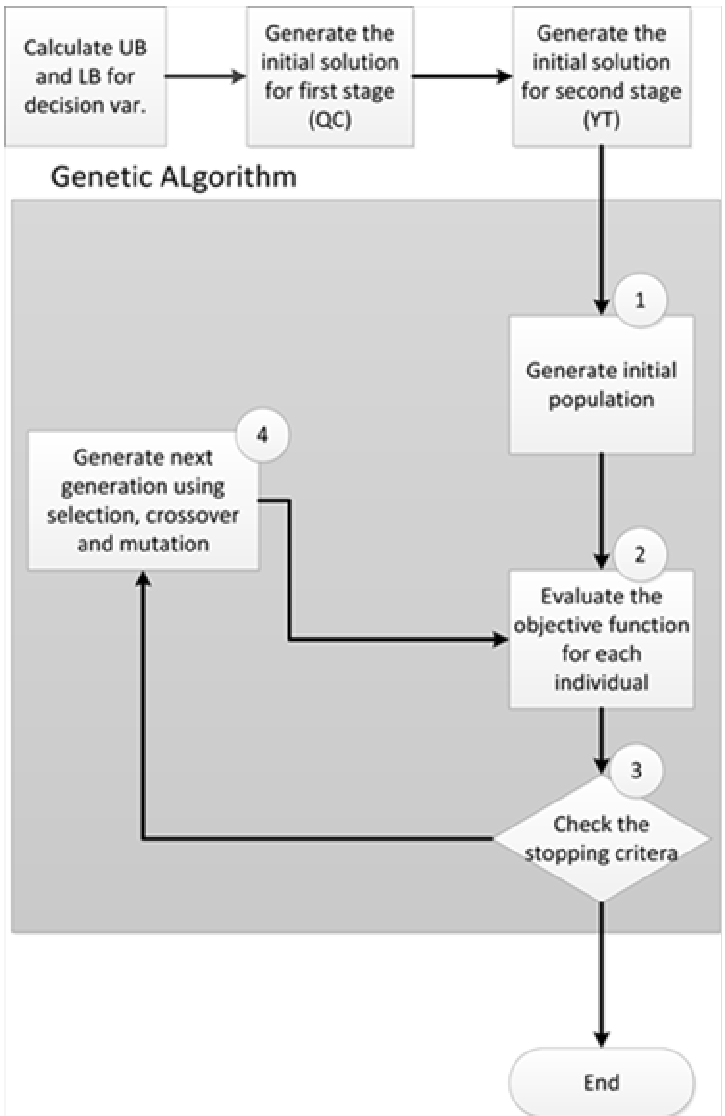


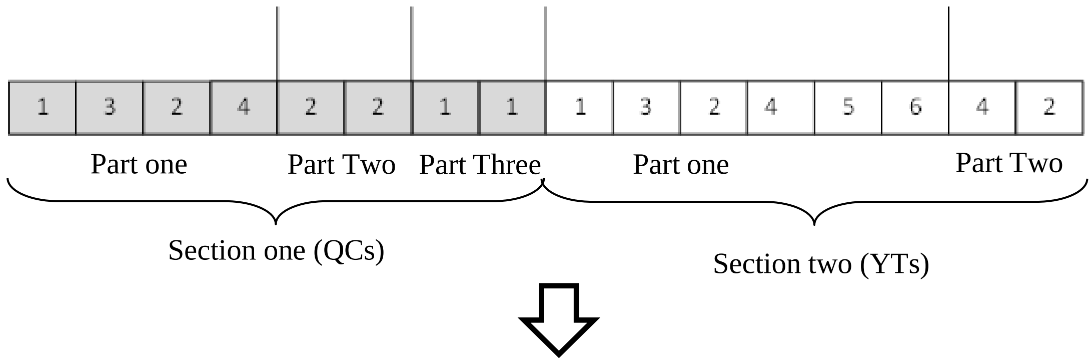
Figure 5.2 Flowchart of developed GA algorithm. (Modified from Kaveshgar et al. 2012)

5.3.1 CHROMOSOME REPRESENTATION

In each generation, GA creates a population of solutions. Each solution is called a chromosome and each gene represents a decision variable. The chromosome used for representing the solutions in this study is composed of two sections as shown in Equation (5.22). The first section represents the solution for the first stage (QC scheduling). It consists of three kinds of genes: *Xs* are the sequence of tasks assigned to the quay cranes, *Ks* represent the number of tasks assigned to the quay cranes, and finally *Hs* represent the movement direction of each quay crane (ascending or descending). As mentioned previously, solutions with unidirectional movement of the quay cranes are optimal or near optimal in most of cases and highly reduces the computational time of the solution method. Thus, here, cranes are limited to unidirectional movement. The second section of the chromosome represents the solution for the second stage of the problem (YT scheduling). It has the same elements as the first section, except for the *Hs*. *Xs* represent the sequence of tasks assigned to the YTs and *Ks* represent the number of tasks assigned to each YT.

$$s = [\underbrace{X_1 X_2 \ldots X_{n_1} K_1 K_2 K_{m_1} H_1 H_2 H_{m_1}}_{\text{Section 1 (QCs)}} \underbrace{X_1 X_2 \ldots X_{n_2} K_1 K_2 K_{m_{21}}}_{\text{Section 2 (YTs)}}] \tag{5.22}$$

Figure 5.3 shows a sample chromosome of a problem with 4 tasks, 2 quay cranes, 6 containers and 2 trucks. Tasks (container groups) 1 and 3 are assigned to the first quay crane and tasks 2 and 4 are assigned to the second quay crane. The third part of the chromosome indicates that both quay cranes will move from left to right (in ascending order). Section two indicates that tasks (containers) 1, 3, 2 and 4 will be transported by the first YT, and tasks 5 and 6 will be transported by the second YT.



QC No.	Task sequence	YT No.	Task sequence
1	1, 3	1	1, 3, 2, 4
2	2, 4	2	5, 6

Figure 5.3 Chromosome representation.

By defining the lower and upper bound on the decision variables we can further reduce the number of decision variables. The first stage has three different kinds of genes and each kind has a unique lower and upper bound. The first part of the lower bound for section one (QCs) is defined based on the locations of the tasks and quay cranes. Since quay cranes share the same track and when two or more quay cranes are assigned to a vessel, the first quay crane on the left is the only one that can process the tasks located on the first ship bay (left most tasks).

The lower bound of the first *(n-(m-1))* genes of the section one of chromosome is 1. The next *(m-1)* remaining genes' lower bound is 2, 4, 6,… , *m*. The skip in number represents of safety margin between the quay cranes (one bay is used in this study). The lower bound for all the genes in section two is equal to 1.

The upper bound of the first *(n-(m-1))* genes of section one, is set to *n-2(m-1), n-2(m-1) + 2, n-2(m-1) + 4, ...;* for the remaining genes of section one and all genes in section two the upper bound is the total number of tasks.

The lower bound of the second part of the chromosome for both sections is set to one task for each equipment (QC or YT) and the upper bound is calculated using the procedure described below:

Step 1. Start by arranging the processing time of tasks in an ascending order (set *Q)*

Step 2. Set $i = n - m + 1$,

Step 3. Set $a = \sum_{1}^{i} Q$ (5.23)

Step 4. Rearrange the tasks in a descending order (set *G*)

Step 5. Set $b = \sum_{1}^{n-i-(k-2)} G$ (5.24)

If $b > a$ stop and set upper limit=*i*; otherwise, set *i*=*i*-1 and go to step 3.

The above procedure would set a limit on the maximum number of tasks assigned to equipment. The tasks are arranged in an ascending order according to their processing time. The algorithm finds the maximum number of tasks that could be grouped together and be assigned to a QC or YT such that its total processing time is not greater than the processing time of the longest task. The number of tasks identified for the group is the maximum number of tasks that can be assigned to a QC or YT.

If the precedence relationship between tasks is not satisfied, then the tasks will be swapped. The GA in MATLAB could create identical values for two genes in the first section of the chromosome. To overcome this shortcoming, we coded a function to validate the chromosomes.

5.3.2 EVALUATING THE GA OBJECTIVE FUNCTION

The objective function is a unique part of the GA and must be developed based on the particular characteristics of each problem. The developed objective function in this study simulates the container unloading operation and calculates the makespan. It consists of a set of procedures that are repeated until all the tasks are processed. These procedures involve selecting the equipment (QC or YT) with the earliest ready time. The equipment would then process task in its schedule, taking into account operational constraints like interference (if it is a QC). If the equipment needs to wait (e.g. QC has to wait for the YT to deliver the container) then a waiting time is added to that equipment ready time. The equipment with the minimum ready time will be the one selected next to process a task. These steps are repeated until all the tasks (containers) are stored in the container yard. The following sections explain how the operational constraints are accounted for in the determination of the objective function value.

Avoiding interference

The location of each quay crane, the starting and finishing time of each task, and where each quay crane is working is tracked by the objective function in GA and stored in a matrix data structure. Each time a quay crane needs to start processing a new task on a bay different from the crane's current bay location, it would check the tasks on adjacent bays (to the left and right of the new task's bay). If there are tasks being performed by a different quay crane that interferes with its movement, then the finishing time of those tasks is checked. If those tasks are not complete, then the current quay crane needs to wait; otherwise, it may move to the next task's location and start the unloading or loading operation.

QCs' positions

Each quay crane has to maintain its initial and final position, and each time a quay crane performs a task it has to reevaluate its position and set a destination according to its work schedule. The travel time is determined according to the positions and destinations of the quay crane. Based on the current location of the quay crane, the location of the task it will perform next, and the location of other quay cranes, four different destinations are possible: (1) quay crane travels to its assigned task and processes that task, (2) quay crane needs to wait to avoid a collision with another crane and then traverses to its next task location, (3) quay crane remains idle and will stay at its current position, and (4) quay crane remains idle, but needs to move in order to avoid a collision with adjacent quay cranes.

YTs' travel time

YTs have to get the containers from the QCs and transport them to the YCs in the yard. Every time a truck transfers a container from a QC to the yard area, a value equal to the travel time is multiplied by two and the yard crane operation time is added to its current ready time. If the truck has already reached to its final task in its schedule then only one travel time and yard crane operation time is added to its current ready time.

Makespan time and blocking

The quay crane's completion time is calculated based on the processing time of the tasks (P_i), the travel time according to one of the four different possibilities stated above, and the delay caused by the YT serving that quay crane. The following example illustrates how a crane's completion time is calculated.

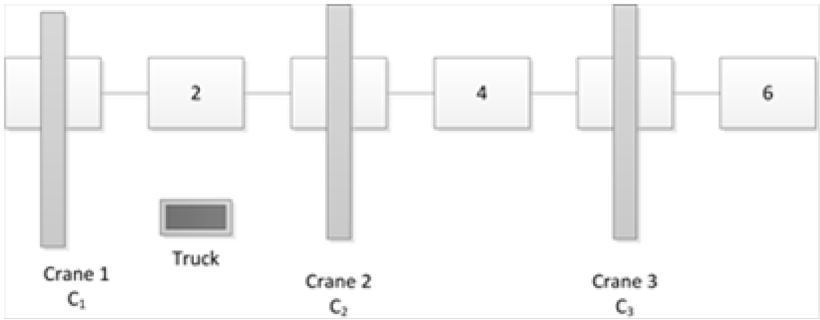


Figure 5.4 Illustration of how quay crane's completion time is determined.

Consider the scenario shown in Figure 5.4 that has 3 quay cranes and 6 tasks located on bays 1 to 6. Tasks 1, 3 and 5 are currently being processed by cranes 1, 2 and 3, respectively. Cranes 1 and 3 next tasks are tasks 2 and 6 according to their schedule and the second quay crane is done with its assigned tasks after completing task 3. If crane 1 finishes its task earlier than cranes 2 or 3, or crane 2 or 3 need to wait for a truck to deliver the container, due to the one bay safety margin, it is not possible for the crane 1 to perform task 2 until task 3 is finished as well as task 5 because crane 2 needs to move over to bay 4 in order for crane 1 to be in bay 2. After crane 2 finished processing task 3 and relocated to bay 4 then crane 1 can start task 2 and the total completion time of crane 1 after processing task 2 is computed as follows:

$$C_1 = \max\{C_2, C_3\} + P_2 + |l_2 - l_1| \tag{5.25}$$

As shown in the above example, any possible interference between the quay cranes and delay caused by YTs is checked and considered in the objective function. The makespan is set as the maximum completion time among all quay cranes.

5.3.3 STOPPING CRITERIA

Different stopping criteria could be specified in MATLAB GA. In this study the following stopping criteria are used:

- Generations — the algorithm stops after reaching a certain number of generations.
- Stall generations — the algorithm stops if the weighted average change in the objective function value over Stall generations is less than Function tolerance.
- Function Tolerance — the algorithm stops when the cumulative change in the objective function value over Stall generations is less than or equal to the Function tolerance (1e-6) over the stall generations limit.

5.4 NUMERICAL EXPERIMENTS AND DISCUSSION

To demonstrate the solvability of the proposed integrated QC-YT model and to gain insight into the integrated solution, the developed GA was evaluated using a total of 32 instances, ranging from small to large-sized problems. Data for the first stage of the problem (quay crane scheduling) was obtained from benchmark instances developed by Meisel and Bierwirth (2011). These instances provide information such as processing time of the tasks, bay location of the tasks, total number of bays, ready times and initial location of the quay cranes, and the precedence relationships between the tasks. These instances can be generated using QCSPgen, which is available online at http://prodlog.wiwi.uni-halle.de/qcspgen. For the second stage, the time that it takes a YC to perform a loading or unloading operation was generated from a uniform distribution (min = 60, max= 130), and the time it takes a YT to transport a container from a QC to YT and vice versa was also generated from a uniform distribution (min =

38, max = 70). These parameter values were taken from Chen et al. (2007). For all instances, the QC travel time is set to one time unit per bay and the safety margin is set to one bay.

The following lists the GA parameter values used in the experiments.

- Stall generation (nS) = 1000
- Function Tolerance (eps) = 2.22x10e-16
- Population size (nP) = 25 individuals
- Number of generations (nG) = 1000

The experiments were conducted on a PC with 4 GB of RAM and 2.80 GHz processor. The results obtained from the developed GA for small size problems are presented in Table 5.1. The first column shows the experiment number. The second column shows the problem size consisting of the number of quay cranes, trucks, tasks and containers. The third column shows the objective function values (makespan) obtained from CPLEX. The fourth and fifth columns show the non-integrated solution and its gap, respectively. The non-integrated solution is obtained by using the sequential method. That is, the solution from the first stage (QC scheduling) is fed into the second stage where a greedy heuristic was used to solve for the YT scheduling. The sixth and seventh columns show the objective function value obtained from the integrated solution method (using the developed GA) and its gap, respectively. It can be seen that the integrated solutions match that of CPLEX; that is, the developed GA found the optimal solutions, and thus the gap to CPLEX is 0. Among the seven instances, the sequential method found the optimal solution only once. For the remaining six instances, the gap ranges from 6.17 to 41.77%. These results highlight the effectiveness of the integrated solution.

Table 5.1 Comparison of Integrated solution (GA) with sequential (GA-Greedy) and CPLEX.

No	*Problem size*[1]	*CPLEX*	*Non-Integrated Solution*		*Integrated Solution*	
		Makespan	*Makespan*	*Gap*[2] *(%)*	*Makespan*	*Gap*[2] *(%)*
1	4×5×2×2	81	86	6.17	81	0
2	5×5×3×3	79	112	41.77	79	0
3	5×8×2×4	57	57	0.00	57	0
4	6×6×2×2	65	72	10.77	65	0
5	6×6×3×4	72	82	13.89	72	0
6	6×8×2×2	75	83	10.67	75	0
7	6×8×2×4	104	114	9.62	104	0

[1] No. of tasks × No. of containers × No. of QCs × No. of trucks

[2] Gap = (makespan− CPLEX makespan)/ CPLEX makespan ×100

Table 5.2 shows the results of larger instances. The first and second columns show the experiment number and problem size. CPLEX runs were limited to two hours and the best solution obtained by it is reported in the third column. An “N/A” in the third columns indicates that CPLEX was not able to obtain a solution. The fourth columns show the CPLEX computation time if a solution was obtained within two hours (7200 seconds). The fifth and sixth columns show the objective function value and the computation time of the integrated solution method (using the developed GA). The last column shows the gap which measures the difference between the GA solution and the CPLEX solution. A negative gap means that the GA solution was lower (better) than the CPLEX solution, in part due to the time limit. For very large problems (instances 30 to 32), CPLEX could not find the optimal solution due to either time limit or memory limit. Out of the 32 instances, GA obtained the solution much faster than CPLEX, except for two instances (1 and 4). It was observed that GA’s computation time is not as affected as CPLEX by the increase in number of containers and trucks. However, CPLEX can solve instances with strong precedence relationships more effectively. The maximum computation time for GA is 108 seconds, which is within acceptable range for operational planning problems.

Table 5.2 Comparison of GA Performance against CPLEX

No	*Problem size*[1]	*CPLEX*		*Integrated Solution*		
		Makespan	*Time(s)*	*Makespan*	*Time(s)*	*Gap*[2] *(%)*
8	10×10×2×4	810	1554	805	41	-0.62
9	10×15×2×4	1071	3702.76	998	65.57	-7.31
10	10×15×3×4	1180	859.86	955	69.40	-23.56
11	10×20×2×4	1627	1929	1271	86.10	-28.01
12	10×20×2×6	1212	3577.52	1023	87.56	-18.48
13	10×20×2×8	1199	2994	799	86.75	-50.06
14	12×12×2×4	1008	928.08	843	56.91	-19.57
15	12×12×3×4	1132	921.00	832	83.54	-36.06
16	12×15×2×4	1273	3168.49	1038	67.58	-22.64
17	12×15×2×6	816	7200	780	68.46	-4.62
18	12×20×2×4	1574	1052.34	1332	84.64	-18.17
19	12×20×2×6	1044	7200	996	107.62	-4.82
20	12×20×2×8	1092	3012.11	780	89.00	-40.00
21	14×14×2×4	959	4003.97	951	65.63	-0.84
22	14×14×3×4	880	7200.21	866	66.95	-1.62
23	14×14×2×6	738	5660.36	658	64.09	-12.16
24	14×14×2×8	852	3482.89	636	65.63	-33.96
25	14×20×2×4	1473	4770.31	1240	69.65	-18.79
26	15×15×2×4	1302	2160	1062	67.97	-22.60
27	15×15×2×6	1000	1195.90	825	72.26	-21.21
28	15×15×2×8	742	7200	726	69.62	-2.20
29	15×20×2×4	1545	7200	1283	87.31	-20.42
30	15×20×2×6	*N/A*	*N/A*	908	84.37	*N/A*
31	20×25×2×6	*N/A*	*N/A*	1123	106.37	*N/A*
32	30×35×2×15	*N/A*	*N/A*	897	151.43	*N/A*

[1] No. of tasks × No. of containers × No. of QCs × No. of trucks

[2] Gap = (worst solution−Best solution)/ Best solution ×100

To further investigate the performance of the developed GA solution method, the effect of the number of yard trucks on the objective function value was evaluated. In total, 16 experiments were generated with 10, 15, 20 and 30 tasks (corresponding to 15, 20, 25 and 35 containers) and 6, 9, 12 and 15 yard trucks. In Figure 5.5, the objective function value (makespan) obtained from GA (integrated solution) is shown against different number of yard trucks. Each line in the figure corresponds to a problem size (10, 15, 20, or 30 tasks). The results indicated that increasing the number of YTs does not result in significant improvement in makespan for small problems (as depicted by the

10-task line); however, it does result in significant improvement for large problems (as depicted by the 30-task line). For all problem sizes, the experiment results indicated that increasing the number of yard trucks beyond 12 will yield little to no improvement. This finding is intuitive because at some point, the number of QCs will become the bottleneck instead of the YTs.

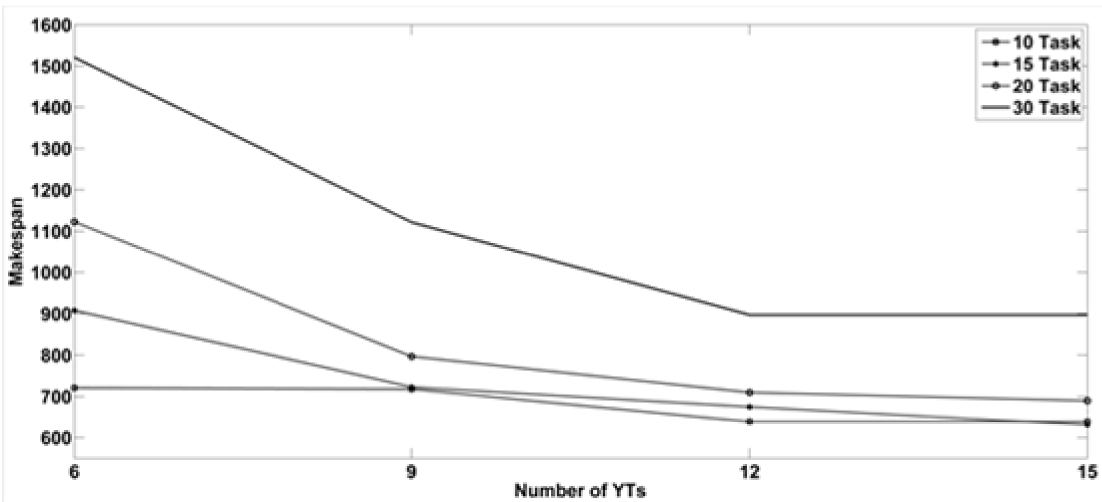


Figure 5.5 Effect of number of YTs on makespan.

Figure 5.6 shows the relationship between the GA computation time and number of tasks. Each line in the Figure corresponds to an operational problem involving 6, 9, 12, or 15 YTs. As expected, increasing the problem size will increase the computation time. However, the GA computation time is not greatly affected by the problem size. Increasing the problem size three fold (from 10 to 30) does not increase the computation three fold. Higher number of YTs also does not have an effect on the GA computation time. These results suggested that the developed GA approach has the potential to solve the integrated QC-YT model for much larger problems within reasonable time.

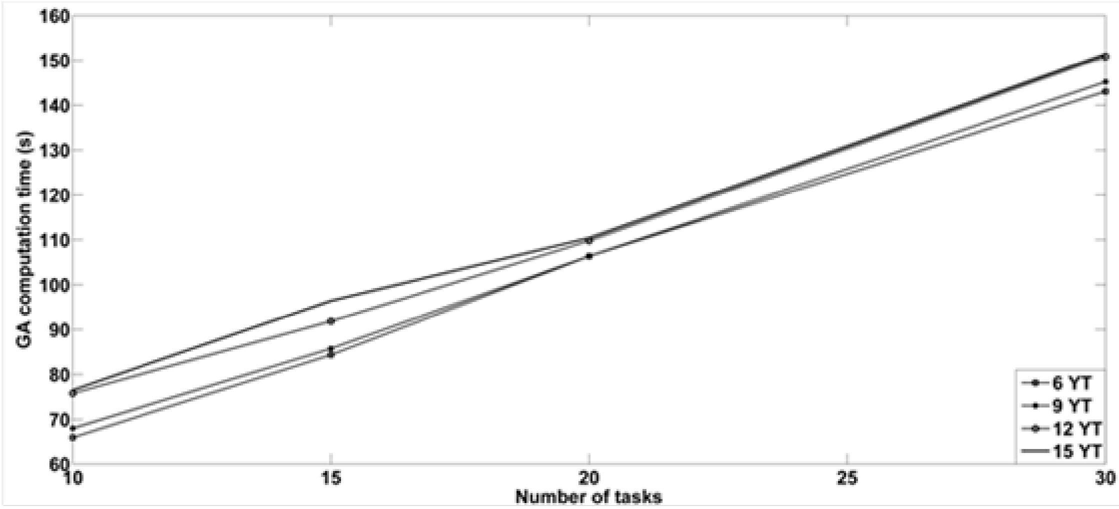


Figure 5.6 Effect of number of tasks on GA computation time.

5.5. CONCLUSION

This study developed a mixed integer programming model for scheduling QCs and YTs jointly using the hybrid flow shop scheduling technique. It extended the existing body of work by considering multiple QCs, as well as non-crossing constraints and safety margins between QCs. The formulation is also unique in that the decision variables in the first stage of the problem (QC scheduling) used groups of containers instead of individual containers. This technique reduced the number of decision variables significantly and consequently the computation time. To solve the integrated optimization model, a genetic algorithm (GA) combined with a greedy algorithm was developed. The experimental results indicated that the solutions obtained from the proposed integrated GA algorithm are superior to the sequential approach. The GA solutions demonstrated that the developed integrated model is solvable within reasonable time for an operational problem.

Key limitations should be considered when reviewing the study results. These include (1) CPLEX results were limited to two hours of runtime, and (2) problem sizes were limited to a maximum of 3 QCs and 35 containers. This research could be extended to cover all three vessel operation processes: QC, YT and YC scheduling. The integrated model could be further enhanced to represent reality by considering (1) stochastic task processing times for all three equipment rather than deterministic values, (2) different productivity rates for QCs and YCs, and (3) different time windows for QCs and YCs.

CHAPTER 6

ROBUST SCHEDULING OF TERMINAL CONTAINER HANDLING EQUIPMENT[1]

ABSTRACT

To lower the vessel turn time, the operations of quay cranes, yard cranes and yard trucks need to be coordinated. The majority of the terminal operation studies have sought to optimize these operations individually. This study develops a robust optimization model that schedules all three operations jointly in a holistic manner. The unique difference between this study and previous works is that it accounts for the non-deterministic nature of container processing times by the quay cranes, yard cranes, and yard trucks. Due to the complexity of the terminal equipment scheduling problems, previous works have simplified the problems by assuming deterministic processing times. To the best of our knowledge, this is the first study to consider this additional layer of complexity. To deal with the uncertainty in processing times, a model is formulated based on a recent robust optimization approach (p-robust). The objective function of the proposed model seeks to minimize the nominal scenario makespan, while bounding the makespan of all possible scenarios using the p-robustness constraints. To solve the

[1] Kaveshgar N. and N. Huynh. Submitted to *Computers and Operations Research*, 09/17/2014.

robust integrated optimization model, a genetic algorithm (GA) is developed. Several numerical examples are considered and the GA solutions are compared against those obtained from CPLEX for smaller problems and a lower bound for larger problems. The experimental results demonstrate that the developed robust integrated model is solvable within reasonable time for an operational problem. The comparative analysis between the proposed p-robust method and the conventional minimax method indicates that solutions produced by p-robust are less conservative.

6.1 INTRODUCTION

Marine container terminals are an important link in the supply chain because they are the interface between land and sea transportation. To maximize its efficiency, terminal operators need to manage their available container-handling equipment effectively to minimize the ship turn time to retain or gain a competitive advantage.

Loading/unloading operation of a vessel starts when a berth is assigned to it. Afterwards, quay cranes (QCs) are allocated to unload inbound (import) containers and load outbound (export) containers. Yard trucks (YTs) are then used to transport containers between the quay and container yard. In the container yard, yard cranes (YCs) are used to move containers from the YTs to the yard blocks and vice-versa. Figure 6.1 illustrates the typical layout of a marine container terminal: a quay with berths for vessels to dock and a container yard to store containers. As illustrated, the operations of the QCs, YTs, and YCs are highly connected and they need to work in unison in order to minimize the vessel turn time.

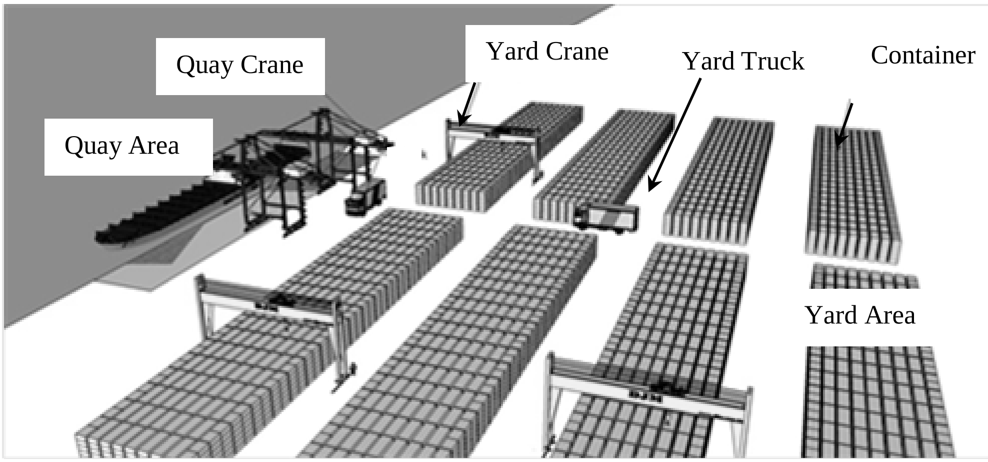


Figure. 6.1. Layout of a marine container terminal.

This study focuses on developing a model for the integrated scheduling of QCs, YTs, and YCs for the unloading operations. The model seeks to capture many of the operational characteristics and practical constraints as observed in actual terminal operations. A unique contribution of the proposed model is the explicit consideration of the variation in the container processing times. That is, the time for a QC to unload a container from a vessel, YT to transport a container from the quay to the container yard, and YC to move the container from the YT to the block is non-deterministic. To the best of our knowledge, this study is the first to consider this additional layer of complexity. To deal with the uncertainty in processing times, a model is formulated based on a recent robust optimization approach, *p*-robust (Peng et al., 2011). The objective function of the proposed model seeks to minimize the nominal scenario makespan, while bounding the makespan of all possible scenarios using the *p*-robustness constraints. Other contributions made within the proposed integrated model include: 1) improving upon the works by Chen at al. (2007), Lau and Zhao (2008), and Chen at al. (2013) by assuming that the container to QCs, YCs and YTs assignment is unknown, 2) enhancing the works done by Meersmans (2002), Chen et al. (2007), Lau and Zhao (2008), Zeng and Yang (2009), Jinxin et al. (2010), and Chen et al. (2013) by adding the non-interference constraints and safety margin for the QCs, and 3) reducing the number of decision variables and consequently the computation time by defining tasks as group of containers in the first stage of the problem (QC scheduling). To solve the robust integrated optimization model, a GA based solution approach is developed. To enhance the probability of finding the optimal solution and reduce the computation time, heuristics are developed to create high quality initial solutions for each stage. Several numerical experiments are conducted and the GA solutions are compared against those obtained from CPLEX for smaller problems and a lower bound for larger problems.

6.2 MATHEMATICAL MODELING

The objective of the integrated model is to find the assignment of inbound containers to the equipment and the processing sequence of the containers (vessel → QCs → YTs → YCs) in a way that the latest completion time of containers (makespan) is minimized. The solution needs to satisfy a number of practical constraints: precedence relationships between containers, blocking (which occurs when the equipment of the subsequent stage is not ready to accept the task/container), QC non-interference, and safety margin.

QCs are mounted on a single rail track alongside the quay and cannot cross each other. Also, a safety margin (a prescribed space between adjacent QCs), has to be kept at all times. In this study, a one ship-bay safety margin is considered between the QCs. The assignment of YCs to yard blocks is typically predetermined in practice and is assumed to be given as an input and that each YC can process containers within the assigned adjacent yard blocks. The time that it takes a QC, YC or YT to move between one job and the next is referred to as setup time. Additional assumptions made in this study include: 1) task processing times are uncertain, 2) all QCs, YTs and YCs have unit capacity, 3) YC or YT setup times are the empty travel times between two consecutive container, 4) QCs setup times are based on the number of ship-bays they travel between two consecutive tasks, 5) there is precedence relationships between containers, 6) task/container to equipment assignment is unknown, 7) there is potential for blocking between operations, and 8) the number of quay cranes assigned to a vessel and locations of containers and QCs (expressed in terms of bay number) are assumed to be known.

To address uncertainty in container processing times, the robust approach is employed. A robust solution is one that performs well for a wide range of scenarios. In other words, a robust solution is less sensitive to uncertainty in data (e.g. List et al., 2003, Huynh and Walton, 2008 and Peng et al., 2011). There are different robustness measures. The most commonly used robustness measures are: minimax cost and minimax regret. The minimax cost approach minimizes the maximum cost among all scenarios and the minimax regret approach minimizes the maximum regret across the scenarios. As defined in Snyder and Daskin (2006) "the regret in a given scenario is the difference between the cost of the solution in that scenario and the cost of the optimal solution for that scenario". In other words, regret is the difference between the cost of a feasible solution for a scenario and the optimal solution for that scenario. The *p*-robustness approach sets an upper bound on the maximum allowable relative regret for each scenario (1). Assuming that there are a set of scenarios E, p_e is the deterministic minimization problem for scenario *e*, *x* is the vector of a feasible solution to p_e for all

$e \in E$, $c_e(x)$ is the objective function value of problem p_e under solution x and c^*_e is the optimal objective function value (makespan) for p_e, the solution is called p-robust if for all $e \in E$:

$$\frac{c_e(x) - c_e^*}{c_e^*} \leq p \tag{6.1}$$

or

$$c_e(x) \leq (1 + p)c_e^* \tag{6.2}$$

The p-robustness approach requires that each scenario's makespan may not be more than $100(1+p)\%$ of the optimal scenario makespan. Different p values indicate the desired robustness. In Equation (6.2), the right hand side is the relative regret of scenario *e* (Peng et al., 2011 and Hatefi and Jolai, 2014). The optimal makespan for each scenario has to be calculated separately. By solving the following MIP formulation, the c_e^* can be calculated for each scenario e. The proposed model is based on the (HFSS) technique and is extended to a p-robust model.

Problem Parameters

$t =$ Stage index

$\Omega_t =$ Set of tasks in QC scheduling stage (t=1) and containers at YT and YC scheduling stage (t=2, 3).

$B =$ Set of first container number in a task $\{b_i\}$

H = Set of last container number in a task $\{h_i\}$

R = Set of the containers in a task, other than the first container in that task

b_i = First container number in a task i

h_i = Last container number in a task i

n_t = Total number of tasks/containers in stage t

i, j = Task/container index

K_t = Set of machines at stage t (QCs in stage 1, YTs in stage 2 and YC in stage 3)

k = Machine index

k_t = The number of machines at stage t

Φ = The set of precedence constrained tasks

Ψ^k = The set of containers that cannot be processed by YC k

l_i = Bay position

l_0^k = Initial position of quay crane (k)

f_{kt} = Earliest available time of machine k at stage t

p_{it} = Processing time of task/container i at stage t

g = QC travel time

δ = Safety margin

M = Large positive number

$\Theta = \{(i, j, v, w) \in \Omega^2 \times Q^2 \mid (i < j) \wedge (\Delta_{ij}^{vw} \geq 0)\}$ Set of all possible combinations of a pair of tasks and a pair of QCs which need a temporal separation of processing Δ_{ij}^{vw} for safety requirements.

d_{ij} = The setup time of yard crane from container *i* to container j. It is defined as the YC empty movement when it moves from container *i* to container *j*.

Travel time of quay crane k from i^{th} task to j^{th} task is assumed to be relative to the ship bay numbers $|l_j - l_i|$.

Dummy containers and tasks are denoted by *0* and *T*

Decision variables

$$x_{ijt}^{k} = \begin{cases} 1 & \text{Iff machine } k \text{ performs task } i \text{ and task } j \text{ consecutively at stage } t \\ 0 & \text{Otherwise} \end{cases}$$

$$z_{ij} = \begin{cases} 1 & \text{Iff task } j \text{ starts after the completion of task } i \\ 0 & \text{Otherwise} \end{cases}$$

c_{it} = Completion time of task/container *i* at stage *t*

s_{it} = Starting time of task *i* at stage *t*

c_{T3} = Makespan (latest completion time of all containers at stage 3)

Objective Function

Minimize c_{T3} (6.3)

Subject to:

$$\sum_{j \in \Omega_t^T} x_{0jt}^{k} = 1 \qquad \forall k \in K_t, \forall t \in \{1, 2, 3\} \tag{6.4}$$

$$\sum_{i\in\Omega_t^{0}} x_{iTt}^{k} = 1 \qquad \forall k\in K_t, \forall t\in\{1,2,3\} \tag{6.5}$$

$$\sum_{k\in K_t}\sum_{j\in\Omega_t^{T}} x_{ijt}^{k} = 1 \qquad \forall i\in\Omega_t, \forall t\in\{1,2,3\} \tag{6.6}$$

$$\sum_{j\in\Omega_t^{0}} x_{ji}{}^{k} - \sum_{j\in\Omega_t^{T}} x_{ij}^{k} = 0 \qquad \forall i\in\Omega_t, \forall k\in K_t, \forall t\in\{1,2,3\} \tag{6.7}$$

$$g\times\left|l_{b_i}-l_{b_j}\right| + p_{jt} - c_{b_jt} + s_{h_it} \le M\times(1-x_{ijt}^{k}) \qquad \forall i,j\in\Omega_t, k\in K_t, t=1, b\in B, h\in H \tag{6.8}$$

$$s_{h_it} + p_{jt} - c_{h_jt} \le 0 \qquad \forall (i,j)\in\Phi, t=1, h\in H \tag{6.9}$$

$$s_{h_it} + p_{jt} - c_{h_jt} \le M(1-z_{ij}) \qquad \forall i,j\in\Omega_t, t=1,\ h\in H \tag{6.10}$$

$$s_{h_jt} - p_{jt} - c_{h_it} \le M(z_{ij}) \qquad \forall i,j\in\Omega_t, t=1, h\in H \tag{6.11}$$

$$\sum_{u\in\Omega_t^{0}} x_{uit}^{v} + \sum_{u\in\Omega_t^{0}} x_{ujt}^{w} \le 1 + z_{ij} + z_{ji} \qquad ((i,j,v,w)\in\Theta), t=1 \tag{6.12}$$

$$s_{h_it} + \Delta_{ij}^{vw} + p_{jt} - c_{b_jt} \le M(3 - z_{ij} - \sum_{u\in\Omega_t^{0}} x_{uit}^{v} - \sum_{u\in\Omega_t^{0}} x_{ujt}^{w})(i,j,v,w)\in\Theta, t=1, b\in B, h\in H \tag{6.13}$$

$$s_{h_jt} + \Delta_{ji}^{vw} + p_{it} - c_{b_it} \le M(3 - z_{ji} - \sum_{u\in\Omega_t^{0}} x_{uit}^{v} - \sum_{u\in\Omega_t^{0}} x_{ujt}^{w})((i,j,v,w)\in\Theta), t=1, b\in B, h\in H \tag{6.14}$$

$$f_{kt} - c_{b_jt} + g\times\left|l_0^{k} - l_{b_j}\right| + p_{b_jt} \le M(1-x_{0jt}^{k}) \qquad \forall j\in\Omega_t, \forall k\in K_t, t=1, b\in B \tag{6.15}$$

$$c_{it} - c_{Tt} \le M(1-x_{iTt}^{k}) \qquad \forall i\in\Omega_t, k\in K_t, t=3 \tag{6.16}$$

$$p_{rit} - c_{rit} + s_{(ri)-1t} \le 0 \qquad \forall i\in\Omega_t, t=1,\ r\in R, b\in B \tag{6.17}$$

$$s_{i+1t} \ge c_{it} \qquad \forall i\in\Omega_{t+1},\ t=1,2 \tag{6.18}$$

$$s_{jt} + (1-x_{ijt}^{k})M \ge c_{it} + p_{it} \qquad \forall i,j\in\Omega_t, k\in K_t, t=2 \tag{6.19}$$

$$s_{jt} + (1-x_{ijt}^{k})M \ge s_{it+1} + p_{it} \qquad \forall i,j\in\Omega_t, k\in K_t, t=2 \tag{6.20}$$

$$s_{jt} + (1-x_{ijt}^{k})M \ge c_{it} + d_{ij} \qquad \forall i,j\in\Omega_t, k\in K_t, t=3 \tag{6.21}$$

$$s_{jt} + (1-x_{0jt}^{k})M \ge c_{jt-1} \qquad \forall j\in\Omega_t,\ \forall k\in K_t, t=2,3, \tag{6.22}$$

$$s_{it} + p_{it} \le c_{it} \qquad \forall i \in \Omega_t, \forall k \in K_t, t = 2,3 \tag{6.23}$$

$$\sum_{j \in \Omega_t^T} x_{ijt}^k = 0 \qquad \forall i \in \Omega_t, \forall k \in K_t, t = 3, i \in \Psi^k \tag{6.24}$$

$$x_{ijt}^k, z_{ij} \in \{0,1\} \qquad (\forall i, j \in \Omega_t, \forall k \in K_t, \forall t \in \{1,2\}) \tag{6.25}$$

$$c_{it}, s_{it} \ge 0 \qquad (\forall i \in \Omega_t, \forall t \in \{1,2\}) \tag{6.26}$$

Model Explanation

Equation (6.3) is the objective function that minimizes the makespan. Constraints (6.4) and (6.5) ensure that each machine chooses one dummy task/container (*0*) as its first task/container and chooses one dummy task/container (*T*) as its last task/container at each stage. Constraints (6.6) ensure that each task/container is processed by one and only one machine at each stage. Constraints (6.7) define the sequence of the tasks/containers: each task has only one task before and after itself at each stage. Constraints (6.8) calculate the completion time for each task (container group) in stage 1 and prevent sub-tours. Constraints (6.9) ensure that task *i* be completed before task j if they belong to set Φ. Constraints (6.10) assign values (0 or 1) to z_{ij} depending on whether task *i* preceded task *j*. Constraints (6.11) assign values (0 or 1) to z_{ij} depending on whether task *j* preceded task *i*. Constraints (6.12) ensure that tasks *i* and *j* are not processed simultaneously by *v* and *w*. If $z_{ij} = 1$, constraints (6.13) would insert the necessary time computed by Equation (6.20) that has to elapse before two adjacent QCs can process two tasks located close to each other. Constraints (6.14) work in the same manner as (6.13) but for the case $z_{ji} = 1$. Constraints (6.15) restrict the earliest starting time of operations for each QC and determine the completion time of the first container in the task. Constraints (6.16) guarantee that the dummy container *T* is selected as the last container. Constraints (6.17) compute the completion time of the all containers grouped together as a task. The completion time of the first container in a task is calculated by constraints (6.18). Constraints (6.19) guarantee that a container can only be transported to the container yard (stage 2) if it has been processed by the QC (stage 1) and the YT has returned to the quay area. Constraints (6.20) guarantee that container j is processed consecutively after container *i* by an YT only after container *i* has been delivered to an YC and the YT is back to the quay area. Constraints (6.21) guarantee that a container can be processed by a YC (stage 3) only after it has been transported to the container yard by the YT (stage 2) and after the YC has made it back to the receiving point in the yard block. Constraints (6.22) work in the same manner as Constraints (6.21), but for dummy task *0*. Constraints (6.23) state that a container incurs YT/YC travel time/stacking time. Constraints (6.24) ensure that containers are assigned to the correct YC. Constraints (6.25) and (6.26) restrict the domains of the decision variables.

Constraints (6.12), (6.13) and (6.14) are based on constraints A.11 to A.13 from Bierwirth and Meisel (2009). In this formulation, Δ_{ij}^{vw} is defined as the minimum time span between the processing of tasks *i* and *j* assigned to adjacent quay cranes v and w. According to Bierwirth and Meisel (2009), Δ_{ij}^{vw} can be calculated as follows:

$$\begin{cases} (l_i - l_j + \delta_{vw})\,.\,t, & \text{if } v < w \text{ and } i \neq j \text{ and } l_i > l_j - \delta_{vw} \\ (l_j - l_i + \delta_{vw})\,.\,t, & \text{if } v > w \text{ and } i \neq j \text{ and } l_i < l_j + \delta_{vw} \\ 0 & \text{otherwise} \end{cases} \tag{6.27}$$

$$\delta_{vw} = (\delta + 1).|v - w|$$

δ_{vw} is defined as the smallest acceptable distance between the position of the two quay cranes v and w. Additional information about the calculation of Δ_{ij}^{vw} could be found in Bierwirth and Meisel (2009).

To obtain the robust solution, the objective function, Equation (6.3), is calculated for only the nominal scenario (the most likely scenario to happen) and constraints (6.28) are added to the formulation. Moreover, constraints (6.4) to (6.26) have to be satisfied for all possible scenarios. It is assumed that all scenarios, except for the nominal scenario, are equally likely to occur.

$$c_e \leq (1+p)\times c_e^* \qquad \forall e \in E \setminus \{\text{nominal scenario}\} \tag{6.28}$$

The objective function in the p-robust approach shows that the decision makers are more interested in finding a solution that performs well under normal conditions. The p-robustness constraints indicate that they are also interested in making extra investment to protect against uncertainty in processing times. The additional investment depends on their desired robustness level (the value for *p*).

6.3 PROPOSED GENETIC ALGORITHM

In this study we have developed a genetic algorithm (GA) to solve the integrated robust problem. The key components of the proposed GA are shown in Figure 6.2 and are explained in the following subsections.

Initial population and chromosome representation

In our proposed GA, each chromosome (solution) is composed of a number of genes, representing the tasks/containers to be processed. To limit the value range of decision variables, lower and upper bounds are calculated for the task/container number and the number of tasks/containers that can be processed by each stage's equipment. The technique used for calculating the lower and upper bound is explained in Kaveshgar et al. (2012) and Section 3.3 of this dissertation. In order to reduce the computation time, a heuristic is used to create a high quality initial solution for each stage. The initial solution for the QC scheduling stage is based on the S-LOAD rule proposed by Sammarra et al. (2007).

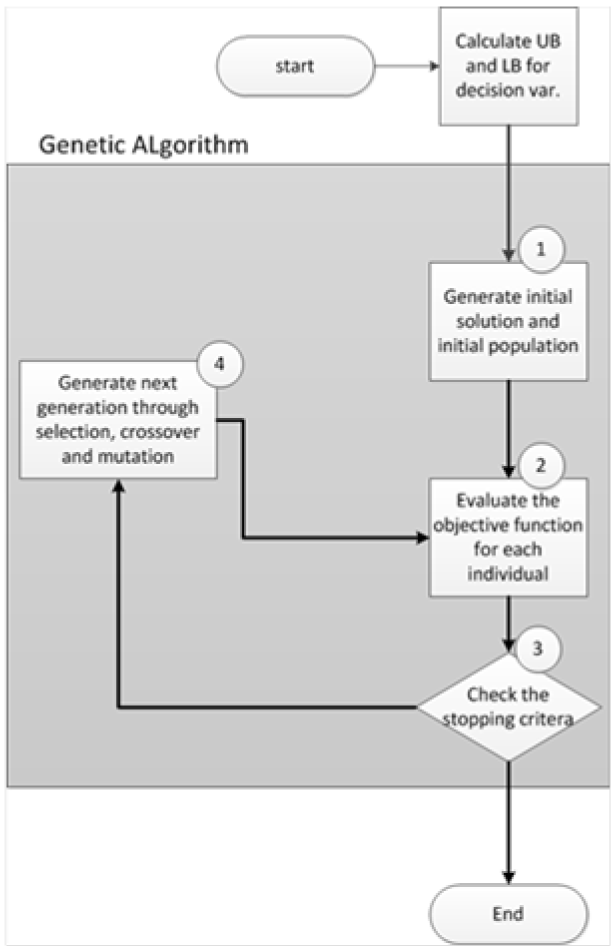
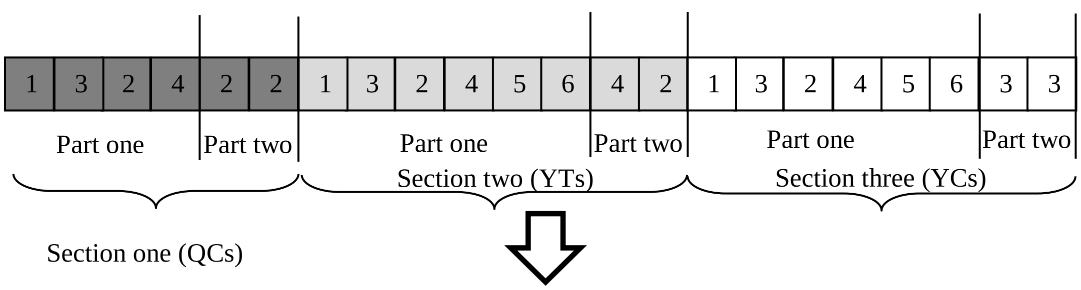


Figure 6.2 Flowchart of proposed GA algorithm. (Modified from Kaveshgar et al. 2012)

The initial solution for the YT scheduling is based on a heuristic proposed by Bish et al. (2005) and used in Lee et al. (2008). Bish et al. (2005) proved that their heuristic finds the optimal solution for problems with single QC and near optimal solutions for multiple QCs. Kaveshgar and Huynh (2014) improved upon Bish et al.'s work for situations that involve multiple QCs. In Bish et al. (2005), a weight based on the second stage (YT) processing times is calculated and assigned to each container and the YTs select containers based on their weight. In Kaveshgar and Huynh (2014), the processing time for both QC and YT stages are considered together to determine the weight. In this study, we have used the improved heuristic by Kaveshgar and Huynh (2014). The initial solution for YC scheduling is based on the first-come first-served principle. The initial solution generated by the aforementioned heuristics is used as one of the individuals in the GA initial population. The remaining individuals are generated randomly with values bounded by the determined lower and upper bounds. Figure 6.3 shows the chromosome representation of a solution for a problem consisting of 4 tasks, 6 containers, 2 QCs, 2YTs and 2YCs. Tasks 1 to 3 have 1 container each and task 4 has three containers (4, 5, and 6). Each chromosome consists of three sections which represent the QC, YT and YC schedules, respectively. Each section has two parts: sequence of tasks/containers and number of tasks/containers assigned to each equipment. For the chromosome shown in Figure 6.2, the results in section two indicates that YT 1 will process 4 containers (1, 3, 2, and 4) and YT 2 will process 2 containers (5 and 6).

After a chromosome is created, it is checked for feasibility and repaired, if necessary. Three procedures are coded as functions to validate and repair the chromosomes: (1) if the precedence relationship between tasks/containers is not satisfied, then the corresponding genes will be swapped, (2) identical values in the chromosomes are eliminated, and (3) if some containers of a QC are assigned to a YT, or some containers of a YT are assigned to a YC, then the container sequence of the two equipment must match. For example, if the container sequence for the QC is 1→2→3→4→6→5, and the container sequence for the YT is 1→2→3→4→5→6, then the YT container sequences have to be changed to 1→2→3→4→6→5. The QC interference is treated differently. The location of each QC, and the starting and finishing times of each task are stored in a matrix data structure. This information makes it possible to check for potential interference between the current QC and neighboring QCs. If so, the current QC is forced to wait until the other QCs have moved to their next locations.



QC No.	Task sequence	YT No.	Container sequence	YC No.	Container sequence
1	1, 3	1	1, 3, 2, 4	1	1, 3, 2
2	2, 4	2	5, 6	2	4, 5, 6

Figure 6.3 Chromosome representation.

Objective Function Evaluation

Evaluating the objective function or the makespan of the solutions by simulating the operations has been proposed and used in several studies (e.g. Zeng and Yang (2009), April et al. (2003), Allaoui and Artiba (2004) and Guo et al. (2006)). Following this approach, a function is developed to simulate the operations of QCs, YTs and YCs. This function receives input data (number of cranes and tasks/containers, task/container locations, QC initial locations, safety margin, precedence relationship between tasks, etc.) and returns the makespan of each chromosome.

Stopping criteria

Different stopping criteria could be used for terminating the GA. In this study, the following stopping criteria are used:

- Generations — the GA stops after reaching a certain number of generations.
- Stall generations — the GA stops if the change in the fitness value over a certain number of generations (stall generations) is less than a predefined value (1e-6).

Next generation, crossover, mutation and selection

In each generation, GA uses the current population to create children for the next generation. A group of individuals in the current population with better fitness values (parents) are selected which will contribute their genes to the next generation. Three types of children are created: (1) elite children, the individuals with the best fitness values in current generation automatically survive to the next generation, (2) crossover children, created by exchanging genes between the vectors of a pair of parents, and (3) mutation children, created by making random changes, or mutations, to a single parent's genes. Crossover fraction indicates the fraction of population that will be created by the crossover operation. It ranges from 0 to 1. After preliminary tests with the "scattered crossover", "ordered crossover" and "two point crossover" functions, the "ordered crossover" function was found to be the most successful, and thus, was used in the proposed GA. This function was also used in the work done by Jinxin et al. (2010).

To prevent the search algorithm from stalling at a local optimum, some children are created by mutation. In this study, based on the mutation probability, the mutation operator randomly selects a chromosome and then randomly selects two genes in one section of that chromosome and exchanges them. This procedure has also been used in the work done by Jinxin et al. (23). For the selection procedure, "roulette wheel selection" is used. The probability of selecting an individual is proportional to its fitness value. For this study, the fitness value is equal to the inverse of the makespan of the solution (calculated in step 2). The lower the makespan, the higher the fitness value and the probability of that individual being selected.

GA for p-robust model

The integrated model is solved for each scenario to find c_e^*. From the solution (x), the objective function value $c_0(x)$ for the nominal scenario, as well as the scenario's makespan $c_e(x)$ can be calculated. Instead of using the makespan to evaluate the quality of the solution, the following equation is proposed to evaluate the solution of the p-robust model:

Minimize $c_0(x) + \omega \times \sum_{e \in E} (c_e(x) - (1+p) \times c_s^*)^+$ (6.29)

In Equation (6.29), the parameter ω is the penalty for violating the *p*-robust constraints. Based on the results reported by Peng et al. (2011) and findings from preliminary experiments, 50 was found to be an appropriate value for ω.

6.4. NUMERICAL EXPERIMENTS AND DISCUSSION

A series of numerical experiments are performed to evaluate the performance of the proposed model and solution approach. The GA is coded in MATLAB 2011a, and the experiments are run on a desktop computer with a 2.80 GHz processor and 4 GB of RAM. Two sets of problems are tested: small with a maximum of 8 containers (instances 1 to 4 in Table 6.1) and large with the number of containers ranging from 15 to 90 (instance 5 in Table 6.1 and Table 6.2 and instances 6 to 17 in Table 6.2). CPLEX runs are limited to two hours and instance 5 cannot be solved optimally by CPLEX within the two hour time limit. Therefore, instance 5 is repeated in Table 6.2 with the remaining large instances. The task/container processing times are generated based on the uniform distribution and parameter values (in seconds) reported in (Chen et al., 2007):

- U(105, 161) for a QC to complete a task
- U(60, 130) for a YT to transport a container to the yard
- U(38, 70) for a YC to transfer a container from the YT to the yard block

Precedence relationship is assumed between tasks. Setup times are considered for QC, YC and YT empty movements. The setup time for QCs is defined as the number of ship bays that they need to travel between two consecutive tasks. The YTs setup time is the same as the container transport time. YCs setup times are generated randomly from U(10, 50).

Uncertainty in processing times is created following the approach used by Lau and Zhao (2008) where 9 different variations of the distributions are considered. That is, given $U(mean - \Delta \times mean, mean + \Delta \times mean)$, 8 additional variations are generated using different values for Δ (0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9). The value of p is selected by the decision maker. Based on Peng et al. (2011) and findings from preliminary experiments, we chose to run all instances with p = 0.15. The GA parameters and their values are set as follows for the experiments.

- Population size = 100 individuals
- Stall generation = 40
- Function Tolerance = 1e-6
- Number of generations = 70
- Crossover probability = 0.9
- Mutation probability = 0.9

Performance of integrated model

The results for the smaller set are benchmarked using the branch and cut algorithm in CPLEX. CPLEX runs are limited to two hours. CPLEX cannot find an optimal solution for larger instances within the time limit. Thus, in order to evaluate the GA performance for larger instances, the lower bound developed by Nahavandi and Gangraj (2014) is used. Their lower bound has been proposed for flexible flow shop problems with unrelated parallel machines. It takes into account the machines' waiting times at each stage and each machine's workload to estimate the time needed by the last job (i.e. task/container) to pass through all the stages.

The results for the integrated model (Equations (6.3) to (6.26)) are reported in Table 6.1. The first column shows the experiment number, the second column shows the problem size consisting of the number of tasks, containers, QCs, YTs and YCs. The third column shows the makespan values obtained from the lower bound. The fourth and fifth columns show the makespan obtained from CPLEX and associated computation time. The sixth column shows the gap between the CPLEX results and the lower bound. The seventh column shows the initial solution and its gap, respectively. The eighth and ninth columns show the objective function value obtained from GA and its gap in comparison with lower bound, respectively. It can be seen that the proposed GA results match that of CPLEX for the first four instances. Note that CPLEX could not find the optimal solution for instance 5 within two hours. The solution (makespan) obtained at the end of the two-hour run (1207) is higher than the GA solution (1175). For smaller instances (instances 1 to 4), CPLEX is faster than developed GA; however, GA's computation time is not as affected as CPLEX by the increase in problem size.

Table 6.1 Comparison of GA Performance against CPLEX for Small Problems

No	Problem size[1]	LB	CPLEX			Initial Solution		Integrated GA		
		Makespan	Makespan	Time (s)	Gap[2] (%)	Makespan	Gap[2] (%)	Makespan	Time (s)	Gap[3] (%)
1	4×5×2×3×2	592	609	1	2.87	654	10.47	609	86	2.87
2	6×8×2×4×2	705	709	26	0.57	1091	54.75	709	210	0.57
3	5×5×3×3×3	434	451	3	3.92	686	58.06	451	143	3.92
4	5×5×3×3×3	505	519	186	2.77	714	41.38	519	211	2.77
5	10×15×2×4×5	1159	1207	>7200	4.14	1239	6.90	1175	402.37	1.38

1 No. of tasks × No. of containers × No. of QCs × No. of trucks × No. of yard cranes
2 Gap = (makespan obtained by CPLEX−LB)/ LB×100
3 Gap = (makespan obtained by the solution algorithm−LB)/ LB×100

Table 6.2 shows the results for larger instances. The first and second columns show the experiment number and problem size. The third column shows the lower bound value. The fourth and fifth columns show the objective function value and the computation time of the GA. The last column shows the gap which measures the difference between the GA solution and the lower bound value. The gap is less than 5% up to instance 14 with 60 containers and 3 QCs. For larger problems with 4 QCs and 70 to 90 containers (instances 16 and 17), the gap ranges from 6% to 8%. This finding is consistent with the results reported in the work by Chen et al. (2007) and Jinxin et al. (2010); both studies also dealt with the integrated scheduling of QCs, YTs and YCs in container terminals. As expected, the GA's computation time increases with problem size, but its computation time remains within reasonable range for fairly large problems.

Table 6.2 Comparison of GA Performance against Lower Bound for Large Problems

No	Problem size[1]	LB	Integrated GA		
		Makespan	Makespan	Time(s)	Gap[2] (%)
5	10×15×2×4×5	1159	1175	402.37	1.38
6	30×30×2×5×6	2117	2145	286.28	1.32
7	30×30×2×10×5	2084	2112	348.56	1.34
8	30×30×2×5×5	2149	2247	285.77	4.56
9	30×35×2×10×5	2383	2474	382.61	3.82
10	30×35×2×5×5	2383	2444	333.77	2.56
11	60×70×2×15×10	4775	4962	955.19	3.92
12	50×60×3×10×10	2807	2846	851.03	1.39
13	50×60×3×15×10	2798	2930	955.54	4.72
14	60×60×3×15×10	2674	2752	932.76	2.92
15	60×70×3×15×10	3224	3425	1110.91	6.23
16	60×70×4×15×10	2459	2611	1371.53	6.18
17	70×90×4×15×10	3195	3454	1640.79	8.11

1 No. of tasks × No. of containers × No. of QCs × No. of trucks × No. of yard cranes

2 Gap = (makespan obtained by the solution algorithm−LB)/LB×100

Performance of *p*-robust problem

Table 6.3 shows the results obtained for the *p*-robust problem. The first and second columns show the experiment number and problem size. The third column shows the solution found by CPLEX. The fourth and fifth columns show the objective function value and the computation time of the developed GA. The last column shows the gap between the GA solution and CPLEX solution. As shown, GA's results match that of CPLEX for the first 5 instances. For larger problems (instances 6 to 15), CPLEX is unable to obtain the optimal solution within two hours.

Table 6.3 Performance comparison of GA approach with CPLEX for p-robust model

No	Problem size[1]	CPLEX		Integrated GA		
		Makespan	Time(s)	Makespan	Time(s)	Gap[2] (%)
1	4×5×2×2×3×4	609	<1	609	80.87	0
2	5×5×3×3×3×9	519	<1	519	90.2	0
3	6×6×2×4×3×9	633	3.45	633	287.47	0
4	6×8×3×6×2×9	668	141	668	445.48	0
5	7×9×3×6×2×9	687	1040	687	447.82	0
6	8×10×3×6×2×9	n/a	>7200	806	474.48	n/a
7	10×15×2×4×5×9	n/a	>7200	1175	548.64	n/a
8	30×30×2×5×6×9	n/a	>7200	2145	1139.58	n/a
9	30×30×2×10×5×9	n/a	>7200	2108	384.6	n/a
10	30×30×2×5×5×9	n/a	>7200	2247	1128.79	n/a
11	30×35×2×10×5×9	n/a	>7200	2474	1397.28	n/a
12	30×35×2×5×5×9	n/a	>7200	2444	1285.64	n/a
13	60×70×2×15×10×9	n/a	>7200	4962	2470.08	n/a
14	50×60×2×15×10×9	n/a	>7200	2846	2433.06	n/a
15	60×70×3×15×10×9	n/a	>7200	3425	2405.52	n/a

1 No. of tasks × No. of containers × No. of QCs × No. of trucks × No. of yard cranes
2 Gap = (makespan obtained by the solution algorithm−LB)/LB×100

Comparison of *p*-robust against minimax robust criterion

One of the most widely used robustness measure is minimax. From a managerial point of view, the solution of the minimax approach is often considered too conservative because it accounts for the worst-case scenario. Given that the probability of the worst case scenario occurring is very small, a less conservative approach is to find a solution for the nominal scenario (one that is mostly likely to occur). To demonstrate the difference between the two approaches, the proposed model is applied on a test problem (instance 5 of Table 6.3) to compare the *p*-robustness criterion against the minimax cost criterion. The objective function used for minimax cost is:

$$\text{(Minimax Cost)} \quad \textit{Minimize} \ \max_{e \in E} c_e(x) \tag{6.30}$$

Results of the comparison are reported in Table 6.4 and both models are solved by CPLEX. The first column shows the scenario number. The second column shows the optimal scenario makespan. The third and fourth columns show the scenario makespan and the relative regret by the *p*-robust approach. The fifth and sixth columns indicate the scenario makespan and the relative regret by the minimax cost approach. The last column shows the percentage difference between the *p*-robust and minimax methods. A positive value in the last column indicates that the scenario makespan obtained by *p*-robust method is smaller than that obtained by the minimax cost method. The "Diff" is greatest for the nominal scenario (9.17%). This means that the makespan using the *p*-robustness approach is 9.17% lower than minimax approach for the nominal scenario. This is important because the nominal scenario is the most likely scenario to happen and decision makers are more interested in obtaining a solution that performs well under normal conditions (nominal scenario). Although the "Diff" is less than 0 for the remaining scenarios (except for scenario 2) the relative regret is controlled in the p-robustness approach (relative regret is less than 15%) for all scenarios.

Table 6.4 Comparison with other robust criteria for test Problem 5 of Table 6.3

S	c_s^*	*p*-Robust		Minimax		
		Makespan	Relative regret	Makespan	Relative regret	Diff[1] (%)
0	687	687	1.000	750	1.092	9.17
1	726	756	1.041	738	1.017	-2.38
2	730	730	1.000	768	1.052	5.20
3	734	750	1.022	734	1.000	-2.13
4	697	727	1.043	699	1.003	-3.85
5	752	789	1.049	752	1.000	-4.69
6	684	737	1.077	734	1.073	-0.41
7	701	765	1.091	726	0.036	-5.10
8	651	743	1.141	707	1.086	-4.84

[1]Diff = (makespan by Minimax – makespan by p-robust /makespan by p-robust) ×100

6.5 CONCLUSION

The efficient scheduling of QCs, YTs and YCs in container terminals is critical as it contributes to the terminal productivity and throughput. This study presented a robust integrated model to schedule QCs, YTs and YCs jointly. The proposed model is the first to consider the non-deterministic nature of container processing times. The integrated model is formulated based on a recent robust optimization approach: *p*-robust. With the *p*-robust method the makespan of the nominal scenario is minimized while bounding the makespan of all possible scenarios. Several numerical experiments were solved using a proposed GA solution approach and the solutions were compared against those obtained from CPLEX (for smaller problems) and a lower bound (for larger problems). The results demonstrated that the proposed GA can successfully solve the robust integrated model and find high quality solutions within reasonable time for an operational problem. The comparative analysis between the proposed *p*-robust method and the conventional minimax method demonstrated that solutions produced by *p*-robust are less conservative.

This study proposed a model that considered non-deterministic container processing times in an effort to improve the accuracy of the models by incorporating additional operational characteristics as observed in actual terminal operations. Other uncertainties and complexities in operations that need to be addressed include yard congestion, delays due to equipment breakdown or scheduled breaks, and productivity rate of each container handling equipment and operator.

CHAPTER 7

Conclusion and Future Research

In this dissertation, four completed research studies are presented that address critical equipment scheduling problems in container terminals. The efficient solutions to these problems will enhance terminal productivity and competitiveness.

In Chapter 3 of this dissertation a more efficient solution approach for the QC scheduling problem is presented. Several studies have proposed the use of GA to solve the QC scheduling problem. In this dissertation the GA available in MATLAB 7.13 was used for solving the QC scheduling problem. The efficiency of the GA search is improved by 1) using an initial solution, 2) a new approach for defining the chromosomes, and 3) using new procedures for calculating tighter lower and upper bounds for the decision variables. The effectiveness of our proposed method is tested on several experiments. The results show that the developed GA provides solutions in a faster time for larger problems compared to the available best-known solutions.

In Chapter 4, the QC scheduling problem with time windows (QCSPTW) is studied. Time windows add an additional layer of complexity to the QCQP. An efficient GA is developed for solving the QCSPTW. The GA proposed in this research differs from the related work in that 1) QCs are allowed to move in different directions independently, and 2) QCs are allowed to change their directions in specific situations. The proposed solution method is tested on numerous experiments.

Experiments showed that the developed GA can provide high quality solutions in a faster time for medium and large-sized instances. In instances with fragmented time windows the developed GA improved the solution quality.

In Chapter 5, a new mathematical formulation for integrated QC and truck scheduling is presented. The formulation is based on the hybrid flow shop scheduling technique. The existing literature is extended by considering multiple QCs, non-crossing constraints, safety margins between QCs, and using a group of containers instead of individual containers in the QC scheduling. By considering groups of containers it reduces the number of decision variables significantly and consequently the computation time. A GA combined with a greedy algorithm is developed for solving the integrated problem. The experimental results show that the solutions obtained from the proposed integrated GA are superior to the sequential approach and that the developed integrated model is solvable within reasonable time for an operational problem.

In Chapter 6, an integrated model for all three scheduling problems is presented: QC, YT, and YC scheduling problems. The integrated model is further enhanced to represent reality by considering non-deterministic task processing times rather than deterministic values. There are several techniques for dealing with uncertainty, and the most applied robustness criteria in the literature are: minimax cost and minimax regret. More recent studies have recommended and used the *p*-robustness measure. This dissertation proposes the *p*-robustness technique and an efficient GA for solving the robust problem. The results demonstrate that the proposed GA can successfully solve the robust integrated model and find high quality solutions within reasonable time for an operational problem. The comparative analysis between the proposed *p*-robust method and the conventional minimax method shows that the solutions produced by *p*-robust are less conservative.

The following areas are worthwhile for further study: Bierwirth and Meisel (2009) showed that branch and bound can be used to solve the QCSP efficiently by restricting QCs to unidirectional movements. This approach could potentially be extended to solve the integrated scheduling problems. This dissertation considered non-deterministic container processing times to better reflect the actual dynamics of terminal operations. Additional operational characteristics as observed in terminal operations should be considered in future models to improve their accuracies, such as yard congestion, equipment failure or scheduled breaks, and productivity rate of each container handling equipment and operator. Lastly, integrated models could be further extended by considering loading and unloading operations jointly.

REFERENCES

Hatefi, S.M. and Jolai, F. (2014). Robust and reliable forward–reverse logistics network design under demand uncertainty and facility disruptions. *Applied Mathematical Modeling, 38*(9-10): 2630–2647

April, J., Glove, F., Kelly, J. P. and Laguna, M. (2003). *Practical introduction to simulation optimization. Proceedings of the 2003 winter simulation conference. 1*:71–78.

Allaoui, H., and Artiba, A. (2004). Integrating simulation and optimization to schedule a hybrid flow shop with maintenance constraints. *Computers & Industrial Engineering, 47*(4):431–450.

Guo, Y., Liao, W., Cheng, X. and Liu, L. (2006). A new simulation optimization system based virtual simulation for manufacturing system. *Simulation Modeling Practice and Theory, 14*(5): 577–585.

Nahavandi, N. and Gangraj, E. A. (2014). A new lower bound for flexible flow shop problem with unrelated parallel machines. *International journal of Industrial Engineering & Production Research, 25*(1): 65-70

Bierwirth, C., & Meisel, F. (2009). A fast heuristic for quay crane scheduling with interference constraints. *Journal of Scheduling, 12*(4): 345–360.

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

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

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

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

Bose, J., Reiners, T., Steenken, D., & Voß S. (2000). Vehicle dispatching at seaport container terminals using evolutionary algorithms. In *System Sciences, Proceedings of the 33rd Annual Hawaii International Conference on IEEE*: 1-10

Chang, D., Jiang, Z., Yan, W., He, Z. (2010). Integrating berth allocation and quay crane assignments. *Transportation Research Part E, 46*(6): 975–990.

Chen, L., Bostel, N., Dejax, P., Cai, J., Xi, L. (2007). A tabu search algorithm for the integrated scheduling problem of container handling systems in a marine terminal. *European Journal of Operational Research, 181*(1):40–58.

Chen, L., Langevin, A., Lu, Z. (2013). Integrated scheduling of crane handling and truck transportation in a maritime container terminal. *European journal of Operational research, 225*(1): 142–152.

Chung, S. H., & Choy, K. L. (2012). A modified genetic algorithm for quay crane scheduling operations. *Expert Systems with Applications, 39*(4): 4213–4221.

Crainic, T.G. & Kim, K.H. (2007). Intermodal Transportation. In: *C. Barnhart, & G. Laporte, (Eds), Handbooks in Operations Research and Management Science, 14*:467–537).

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

Daganzo, C. F., & Peterkofsky, R. I. (1990). A branch and bound solution method for the crane scheduling problem. *Transportation Research Part B, 24*(3): 159–172.

Gharehgozli A.H., Yu, Y., Koster, R., Udding, J.T. (2014). An exact method for scheduling a yard crane, *European Journal of Operational Research, 235*(2): 431-447.

Golias, M. M., Boile, M., & Theofanis, S. (2009). Berth scheduling by customer service differentiation– A multi-objective approach. *Transportation Research Part E, 45*(6): 878–892.

Golias, M. M., Saharidis, G. K., Boile, M., Theofanis, S., Ierapetritou, M. G. (2009). The berth allocation problem: Optimizing vessel arrival time. Maritime Economics & Logistics, 11(4): 358–377.

Golias M.M. (2011). A bi-objective berth allocation formulation to account for vessel handling time uncertainty. *Journal of Maritime Economics and Logistics. 13*(4): 419-441.

Golias M.M., Haralambides H.E. (2011). Berth scheduling with variable cost functions. *Journal of Maritime Economics and Logistics. 13*(2): 174-189.

He, J., Chang, D., Mi, W., & Yan, W. (2010). A hybrid parallel genetic algorithm for yard crane scheduling. *Transportation Research Part E, 46*(1): 136–155.

Henesey, L. (2006). Multi-agent systems for container terminal management. *Blekinge Institute of Technology.*

Huang, Y., & Hsu, W. (2002). Two equivalent integer programming models for dispatching vehicles at a container terminal. *School of Computer Engineering, Nan yang Technological University, Technical Report, Singapore.*

Huynh, N , & Walton, M. (2008). Robust scheduling of truck arrivals at marine container terminals. *Journal of Transportation Engineering, 134* (8): 347‑353

Huynh, N., & Vidal J. (2010). An agent-based approach to modeling yard cranes at seaport container terminals. In *Proceedings of the 2010 Spring Simulation Multi conference*, pp. 120. ACM.

Ioannou, P. A. (2008). Intelligent Freight Transportation. *CRC Press- Technology & Engineering.*

Jinxin, C., Qixin, S., & Lee, D.H. (2010). Integarted quay crane and yard truck schedule problem in container terminals. *Tsinghua science and technology, 15*(4): 467-474.

Jung, S.H., Kim, K.H. (2006). Load scheduling for multiple quay cranes in port container terminals. *Journal of Intelligent Manufacturing, 17*(4): 479–492.

Kaveshgar, N., Huynh, N., & Khaleghi Rahimian, S. (2012). An efficient genetic algorithm for solving the quay crane scheduling problem. *Expert Systems with Applications, 39*(18): 13108-13117.

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

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

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

Lau, H. Y. K., & Zhao, Y. (2008). Integrated scheduling of handling equipment at automated container terminals. *International Journal of Production Economics, 112*(2): 665-682.

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

Lee, D. H., Wang, H. Q., & Miao, L. (2008). Quay crane scheduling with noninterference constraints in port container terminals. *Transportation Research Part E, 44*(1): 124–135.

Legato, P., Trunfio, R., & Meisel, F. (2011). Modeling and solving rich quay crane scheduling problems. *Computers & Operations Research, 39*(9): 2063–2078.

Li, W., Goh, M., Wu, Y., Petering, M., Souza, R., Wu, Y. (2012). A continuous time model for multiple yard crane scheduling with last minute job arrivals. *International Journal of Production Economics, 136*(2): 332-343.

Lim, A. Rodrigues, B. & Xu, Z. (2007). An m-parallel crane scheduling problem with a non-crossing constraint. *Naval Research Logistics, 54*(2): 115–127.

List, G. F.,Wood, B., Nozick, L. K., Turnquist, M. A., Jones, D. A., Kjeldgaard, E. A. & Lawton, C. R. (2003), Robust optimization for fleet planning under uncertainty, *Transportation Research, Part E, 39*(3), 209–27.

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

MATLAB Users Guide, (2011), Version 7.13.0.564 (R2011b). *The Math Works, Inc*.

Meer, J. (2000). Operational control of internal transport. *Erasmus University Rotterdam.*

Meersmans, P.J.M. (2002). Optimization of Container Handling Systems, *Ph.D. Thesis, Tinbergen Institute 271*, *Erasmus University, Rotterdam.*

Meisel, F.; Bierwirth, C. (2006). Integration of Berth Allocation and Crane Assignment to Improve the Resource Utilization at a Seaport Container Terminal, in: *Haasis, H.-D.; Kopfer, H.; Schönberger, S. (eds.) Operations Research Proceedings 2005, Springer-Verlag*, pp: 105-110.

Meisel, F., & Bierwirth, C. (2011). A unified approach for the evaluation of quay crane scheduling models and algorithms. *Computers & Operations Research, 38* (3): 683–693.

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

Meisel, F. (2011). The quay crane scheduling problem with time windows. *Naval Research Logistics, 58* (7): 619–636.

Monaco, M. F., & Sammarra, M. (2011). Quay crane scheduling with time windows, one-way and spatial constraints. *International Journal of Shipping and Transport Logistics, 3*(4): 454–474.

Mulvey, J, & Vanderbei, R. (1995). Robust optimization of large scale systems. *Operations Research, 43*(2): 264-281.

Murty, K.G., Liu, J., Wan, Y.W., & Linn, R. J. (2005). A Decision Support System for operations in a container terminal. *Decision Support System, 39*(3): 309-332.

Ng, W.C., Mak, K.L. (2005). Yard crane scheduling in port container terminals. *Applied Mathematical Modeling, 29*(3): 263–276.

Ng, W. C., & Mak, K. L. (2006). Quay crane scheduling in container terminals. *Engineering Optimization, 38*(6): 723–737.

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

Nguyen, V.D., & Kim, K. H. (2009). A dispatching method for automated lifting vehicles in automated port container terminals. *Computers and Industrial Engineering, 56*(3): 1002–1020.

Peng, P., L. Snyder, A. Lim, & Zuli L. (2011). Reliable logistics networks design with facility disruptions. *Transportation Research Part B 45*(8): 1190-1112.

Petering, M.E.H., Murty, K.G. (2009). Effect of block length and yard crane deployment systems on overall performance at a seaport container transshipment terminal. *Computers & Operations Research 36*(5): 1711–1725.

Rashidi H., Tsang E.P.K. (2006). Container terminals: Scheduling decisions, their formulation and solutions. *Submitted to the Journal of scheduling*.

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

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

Sammarra, M., Cordeau, J. F., Laporte, G., & Monaco, M. F. (2007). A tabu search heuristic for the quay crane scheduling problem. *Journal of Scheduling, 10* (4-5): 327–336.

Snyder, L., Daskin, M. (2006). Stochastic p-robust location problems. *IIE Transactions 38*(11): 971–985.

Tavakkoli-Moghaddam, R., Makui, A., Salahi, S., Bazzazi, M., & Taheri, F. (2009). An efficient algorithm for solving a new mathematical model for a quay crane scheduling problem in container ports. *Computers & Industrial Engineering, 56*(1), 241–248.

UNCTAD. Review of maritime transport. (2012). *New York and Geneva: United Nations.*

Vacca, I., Bierlaire, M., & Salani M. (2007). Optimization at container terminals: status, trends and perspectives. In *7th Swiss Transport Research Conference, Monte Verita/Ascona,* pp. 12–14.

Vidovic, M., Kim, K.H. (2006). Estimating the cycle time of three-stage material handling systems. *Annals of Operations Research 144*(1): 181–200.

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

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

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

Zeng Q. & Yang Z. (2009). Integrating simulation and optimization to schedule loading operations in container terminals. *Computers and Operations Research, 36*(6): 1935-1944.

APPENDIX A

COPYRIGHT PERMISSIONS TO REPRINT

A.1 CHAPTER 3 COPYRIGHT PERMISSION

ELSEVIER LICENSE
TERMS AND CONDITIONS

Nov 23, 2014

This is a License Agreement between Narges Kaveshgar ("You") and Elsevier ("Elsevier") provided by Copyright Clearance Center ("CCC"). The license consists of your order details, the terms and conditions provided by Elsevier, and the payment terms and conditions.

All payments must be made in full to CCC. For payment instructions, please see information listed at the bottom of this form.

Supplier	Elsevier Limited The Boulevard,Langford Lane Kidlington,Oxford,OX5 1GB,UK
Registered Company Number	1982084
Customer name	Narges Kaveshgar
Customer address	University of South Carolina Columbia, SC 29208
License number	3511420329923
License date	Nov 17, 2014
Licensed content publisher	Elsevier
Licensed content publication	Expert Systems with Applications
Licensed content title	An efficient genetic algorithm for solving the quay crane scheduling problem
Licensed content author	Narges Kaveshgar,Nathan Huynh,Saeed Khaleghi Rahimian
Licensed content date	15 December 2012
Licensed content volume number	39
Licensed content issue number	18
Number of pages	10

Are you the author of this Elsevier article?	Yes
Will you be translating?	No
Order reference number	3454350909242
Title of your thesis/dissertation	Scheduling of Container Handling Equipment in Marine Container Terminals
Expected completion date	Dec 2014
Estimated size (number of pages)	150
Elsevier VAT number	GB 494 6272 12
Price	0.00 USD
VAT/Local Sales Tax	0.00 USD / 0.00 GBP
Total	**0.00 USD**

Terms and Conditions

A.2 CHAPTER 4 COPYRIGHT PERMISSION

Palgrave, Rights <Rights@Palgrave.com>

RE: Permission Request

Wed 11/12/2014 6:20 AM

Inbox

To:KAVESHGAR, NARGES <kaveshga@email.sc.edu>;

Dear Narges,

Thank you for your email. As an author, you retain many rights and may not need to obtain specific permission, although you should provide the usual acknowledgements regarding copyright and give a full bibliographic reference. Please see the following link for full details: http://www.palgrave-journals.com/pal/authors/rights_and_permissions.html#Your-retained-rights-as-an-author

Kind Regards,

Claire Smith

Senior Rights Assistant
Nature Publishing Group & Palgrave Macmillan
The Macmillan Building

-----Original Message-----
From: kaveshga@email.sc.edu [mailto:kaveshga@email.sc.edu]
Sent: 11 November 2014 15:36
To: Palgrave, Rights
Subject: Permission Request

Title: Ms
FName: Narges
Name: Kaveshgar
Organisation: University of South Carolina
Address: 300 Main St., Department of Civil and Environmental Eng.
Columbia, SC 29208
Email: kaveshga@email.sc.edu
Publisher: Palgrave Macmillan
Journal: Maritime Economics & Logistics
Author: Narges Kaveshgar,Nathan Huynh
Article_Title: A Genetic Algorithm Heuristic for Solving the Quay Crane Scheduling Problem with Time Windows
Content_DOI:
Publication_Date: dd/mm/yyyy
Vol_Number: N/A

Issue_Number: N/A
Start_Page: N/A
End_Page: N/A
Message: Dear Sir/Madam,

I am requesting a copyright release to reuse an article going to be published by Palgrave Macmillan in my dissertation. I am the first author of the article and requesting to reuse the article in electronic/print format in my dissertation only.
The title of my dissertation is â€œScheduling of Container Handling Equipment in Marine Container Terminalsâ€ and also the expected publication date is December 2014. Please send me a letter of permission in this regard. I need to receive the permission before November 23rd . I greatly appreciate your assistance.

Regards,
Narges Kaveshgar

Re: Requesting permission for reprinting my articles in dissertation (urgent) - MEL 2014.31

KAVESHGAR, NARGES

Thu 11/13/2014 3:05 PM

Sent Items

To:Palgrave, Rights <Rights@Palgrave.com>;

Dear Claire Smith,

Thank you so much for your timely response. I am copying my adviser Dr. Nathan Huynh (Correspondin author) to see if I can request for a delay in dissertation publication.

Best Regards,
Narges Kaveshgar
PhD Candidate
Department of Civil and Environmental Engineering
University of South Carolina,

From: Palgrave, Rights <Rights@Palgrave.com>
Sent: Thursday, November 13, 2014 9:46 AM
To: Bull, David; KAVESHGAR, NARGES
Cc: Hewins, Sharon
Subject: RE: Requesting permission for reprinting my articles in dissertation (urgent) - MEL 2014.31

Dear Narges Kaveshgar,

Thank you for your request, as an author you will have the right to reuse your paper in your thesis but only after publication in the journal. As David says it is important that the thesis is not published prior to the journal article. Please see the following link for full details on the rights you will retain after publication: http://www.palgrave-journals.com/pal/authors/rights_and_permissions.html#Your-retained-rights-as-an-author

Kind Regards,

Claire Smith

Senior Rights Assistant
Nature Publishing Group & Palgrave Macmillan
The Macmillan Building
4 Crinan Street
London N1 9XW

A.3 CHAPTER 5 COPYRIGHT PERMISSION

RE: Subject: Requesting permission for reprinting my articles in dissertation

Pritchard, Laura (ELS-OXF)

Tue 11/18/2014 7:19 AM

Inbox

To:KAVESHGAR, NARGES <kaveshga@email.sc.edu>;

ELSEVIER

Dear Mr Kaveshgar,

Thank you for your email.

Please note that as one of the Authors of this article, you retain the right to include the journal article, in full or in part, in a thesis or dissertation. You do not require permission to do so.

For full details of your rights as a Journal Author, please visit:

http://www.elsevier.com/wps/find/authorsview.authors/copyright#whatrights

Your retained rights allow you to submit your article in electronic format and to post this Elsevier article online if it is embedded within your thesis. You are also permitted to post your Author Accepted Manuscript online however posting of the final published article is prohibited.

Please refer to Elsevier's Posting Policy for further information:

http://www.elsevier.com/wps/find/authors.authors/postingpolicy

Please feel free to contact me if you have any queries.

Kind regards

Laura

Laura Pritchard

Permissions Supervisor | ELSEVIER | The Boulevard| Langford Lane | Kidlington | Oxford OX5 1GB |

From: KAVESHGAR, NARGES [mailto:kaveshga@email.sc.edu]
Sent: 11 November 2014 14:40
To: Rights and Permissions (ELS)
Subject: Subject: Requesting permission for reprinting my articles in dissertation

Dear Sir/Madam,

I requested a copyright release to reuse an article published by Elsevier in my dissertation. I am the first author of the article and requesting to reuse some parts of the article in electronic/print format in my dissertation only. Some parts of the paper will not be used in my dissertation.

I have already received the permission but since I am changing the title of my dissertation to "Scheduling of Container Handling Equipment in Marine Container Terminals" I wanted to get a new permission with the new dissertation title. Please send me a letter of permission in this regard. I greatly appreciate your assistance.

I am attaching the license I have received from you with the previous dissertation title. The article details are given below-

License number: 3499410897866
License date: Oct 31, 2014
Licensed content publisher: Elsevier
Licensed content publication: International Journal of Production Economics
Licensed content title: Integrated quay crane and yard truck scheduling for unloading inbound containers
Licensed content author: Narges Kaveshgar,Nathan Huynh
Licensed content date: Available online 30 September 2014

Best Regards,
Narges Kaveshgar

Elsevier Limited. Registered Office: The Boulevard, Langford Lane, Kidlington, Oxford, OX5 1GB, United Kingdom, Registration No. 1982084, Registered in England and Wales.