Coordinated optimization of equipment operations on a container terminal

Tim Jonker

Technische Universiteit Delft

TU Delft Delft University of Technology

Challenge the future

Coordinated optimization of equipment operations on a container terminal

by

Tim Jonker

Master of Science
in Mechanical Engineering

at the Delft University of Technology,
to be defended publicly on Wednesday August 1, 2018 at 14:00.

Graduation thesis:	ME54035
Student number:	4281004
Report number:	2018.TEL.8250
Supervisor:	ir. M.B. Duinkerken
Company supervisor:	A. de Waal

TU Delft Delft University of Technology

Abstract

Increased international marine transport has increased the use of containers. This goes hand in hand with the increased demand for well-functioning container terminals. The speed at which containers can be handled on the container terminal is an important performance indicator for container terminals. In particular, the productivity of the Quay crane (QC)s is used to determine the performance of a container terminal. To achieve a high performance, a well designed schedule of operations is required. This research investigates to what extent a coordinated planning can contribute to increasing the QC productivity. Currently, an uncoordinated scheduling heuristic is used to dispatch the equipment operating on a container terminal. With a coordinated schedule, operations of all equipment working on the terminal can be considered at once to achieve optimal QC productivity. A Hybrid Flow Shop (HFS) model with revolutionary features is used to model the waterside of a container terminal in a coordinated way. The revolutionary features are called: bi-directional and job-pairs. The former enables jobs to move through the HFS in both directions, the latter constrains certain jobs to be performed simultaneously by a single machine. Subsequently, a tailored Simulated Annealing (SA) algorithm is used to obtain a heuristic solution to this mathematical model. The performance of the HFS model in combination with the SA algorithm is evaluated with a sophisticated and validated container terminal simulation model developed by TBA. Therefore, the SA algorithm should balance between both quality and computational time. Computational time should not be excessive since the schedule has to be adjusted frequently since operations on a container terminal vary quickly and predictions with regards to processing times are unreliable. To evaluate the performance of the coordinated schedule, it will be compared with the uncoordinated scheduling heuristic of TBA. Based on the results obtained during this research, the QC productivity whilst using TBA's uncoordinated scheduling heuristic cannot be increased with a coordinated schedule that uses a HFS to model the container terminal and a SA algorithm to solve that model (p-value 0.2731). On the other hand, it can neither be concluded that TBA's uncoordinated scheduling heuristic results in higher productivity compared to the coordinated schedule. Therefore, the coordinated schedule is a solid alternative to TBA's uncoordinated scheduling heuristic. However, it is expected that with certain changes to the coordinated schedule the QC productivity could be increased.

Abstract

Door toegenomen internationaal zeetransport is het gebruik van containers toegenomen. Dit gaat hand in hand met de toegenomen vraag naar goed functionerende containerterminals. De snelheid waarmee containers kunnen worden verwerkt op een containerterminal is een belangrijke prestatie-indicator voor containerterminals. In het bijzonder wordt de productiviteit van de kadekranen gebruikt om de prestaties van een containerterminal te bepalen. Om een hoge prestatie te bereiken, is een goed ontworpen planning vereist. Dit onderzoek onderzoekt in hoeverre een gecoördineerde planning kan bijdragen aan het verhogen van de kadekraan productiviteit. Momenteel wordt een ongecoördineerde planningsheuristiek gebruikt om de machines op een containerterminal aan te sturen. Met een gecoördineerde planning kunnen operaties van verschillende machines tegelijkertijd worden beschouwd om een optimale productiviteit van de kadekranen te bereiken. Een Hybrid Flow Shop (HFS) model met revolutionaire kenmerken wordt gebruikt om de waterkant van een containerterminal op een gecoördineerde manier te modelleren. De revolutionaire functies worden genoemd: bi-directional en jobpairs. Met de eerste kunnen taken in beide richtingen door de HFS worden verplaatst, de laatste zorgt ervoor dat taken tegelijkertijd door één machine moeten worden uitgevoerd. Vervolgens wordt een Simulated Annealing (SA) algoritme aangepast op dit specifieke probleem gebruikt om een heuristische oplossing voor dit wiskundige model te verkrijgen. De prestaties van het HFS - model in combinatie met het SA algoritme worden geëvalueerd met een geavanceerd en gevalideerd simulatiemodel voor containerterminals ontwikkeld door TBA. Daarom moet het SA algoritme een balans vinden tussen kwaliteit en rekentijd. De rekentijd mag niet buitensporig lang zijn, omdat de planning vaak moet worden aangepast. Dit is omdat de bewerkingen op een containerterminal snel variëren en voorspellingen met betrekking tot behandeltijden onbetrouwbaar zijn. Om de prestaties van de gecoördineerde planning te evalueren, wordt het vergeleken met de ongecoördineerde planningsheuristiek van TBA. Op basis van de resultaten die tijdens dit onderzoek zijn verkregen, kan de kadekraan productiviteit tijdens het gebruik van de ongecoördineerde planningsheuristiek van TBA niet worden verhoogd met een gecoördineerde planning dat een HFS gebruikt om de containerterminal te modelleren en een SA algoritme om dat model op te lossen (p-waarde 0.2731). Aan de andere kant kan evenmin worden geconcludeerd dat de ongecoördineerde planningsheuristiek van TBA resulteert in een hogere productiviteit in vergelijking met de gecoördineerde planning. Daarom is de gecoördineerde planning een solide alternatief voor de ongecoördineerde planningsheuristiek van TBA. Het is echter te verwachten dat met bepaalde wijzigingen in de gecoördineerde planning de productiviteit van de kadekranen kan worden verhoogd.

Acknowledgements

I would like to thank everyone who had unconditional confidence that this thesis would be successful. During times of personal despair you all held faith. This gave me both stress and strength. I pushed myself to the limits of my capacities to satisfy your expectations. In the end this is what lead to a thesis of which I am proud.
A few people deserve some special acknowledgements. These are my supervisors: M.B. Duinkerken, R.R. Negenborn, A. de Waal and N. Yorke-Smith. I want to thank them for their support and criticism.
All colleagues at TBA who helped me solving problems and gave me a wonderful graduation period.
Chris Langhout who has helped me countless times with fixing the layout of my report and for being a sounding board to many of the problems I encountered.
Arent Jonker and Marielle Schmitz for their weekly interested in my report without actually understanding what it was about.
Sebastiaan van Bruggen en Bo Langeveld who have helped me each weekend to escape the grind of graduation. Without them I would have been consumed by stress and been unable to achieve what I have achieved now. You both helped me to burst my university bubble and enter another world. I am very glad that I have such amazing friends.
Finally, I would like to thank Yvonne Schouten, without her I would have never been able to graduate university. This is the last piece of my life that you will ever know of. From now on my life will be completely separated from the one that you knew. You stimulated my curious and critical attitude which had brought me to this point. For this and many other things you deserve eternal gratitude.

Tim Jonker
Delft, July 2017

Preface

This report contains a graduation thesis as part of the Master Mechanical Engineering with the track Transport Engineering and Logistics. Within this track the report topic is related to theme three: Real-time Coordination for Operational Logistics. The graduation assignment is based on a agreement between TBA and the TU Delft. The assignment is provided by TBA, the main working environment will be in their office. The TU Delft is responsible for the grading and to guarantee a sufficient technical and scientific level.

Tim Jonker
Delft, July 2017

Contents

Acronyms

AGV Automated Guided Vehicle. 3, 11, 31, 46, 62, 67, 70–72, 75, 76, 81, 86, 90

ASC Automated Stacking Crane. 10, 12, 23, 31

B&B Branch and Bound. 36–38, 56, 57, 65–68

CNT Containers. 67

CP Constraint Programming. 31, 36

ECT European Container Terminal. 11

FIFO First In First Out. 54, 55

GA Genetic Algorithm. 40–42

HFS Hybrid Flow Shop. iii, v, 25–33, 35, 36, 42–51, 53, 56, 57, 59, 66–68, 93, 94, 96

HTV Horizontal Transport Vehicle. 2–4, 7, 9, 10, 12–17, 20, 24, 26, 29–32, 93

JIT Just in time. 86

KPI Key Performance Indicator. 3, 11, 21

Lift-AGV Lift Automated Guided Vehicle. 3, 4, 9–11, 20, 22–24, 31, 45–48, 50–53, 55, 56, 58–61, 67, 71, 73, 74, 76–81, 83–86, 88–91, 94

LP Linear Programming. 63, 65, 68, 94

MCT Minimum Completion Time. 54

MET Minimum Extra Time. 54

MIP Mixed Integer Programming. 31, 32, 36, 56, 92, 96

MPS Model Predictive Scheduling. 25, 32, 33, 43, 94

ND Non-Delay. 54–56

QC Quay crane. iii, 1, 3, 4, 7, 10–17, 20–25, 29–32, 45–48, 50–56, 58, 60, 61, 65–67, 71, 73–90, 93, 94, 96

RMG Rail Mounted Gantry crane. 2–4, 10, 23

RTG Rubber Tyred Gantry crane. 2, 3, 10

SA Simulated Annealing. iii, v, 39, 40, 42, 44, 56, 57, 59, 63–67, 80, 81, 93, 94, 96

SC Straddle carrier. 3, 10

ShC Shuttle carrier. 3, 17

SMPL Switching Max-Plus Linear. 32, 33, 43

TBA Technisch Bestuurskundig Adviesbureau. iii, v, vii, ix, 4, 5, 7, 11, 18–21, 25, 45, 53, 55–57, 69, 71–73, 75, 78, 81–86, 88–90, 92–96

TEU Twenty-Feet Equivalent units. 7, 11, 20

TT Terminal Truck. 3

TU Technische Universiteit. ix, 19, 32

VRP Vehicle Routing Problem. 59, 60, 62, 63

YC Yard Crane. 2–4, 7–15, 18, 20, 24, 29–32, 45–48, 50–54, 58, 60–62, 67, 75, 78–81, 93, 94

1

Introduction

Today, 60% of the total transport volume is transported in standardized boxes called containers [1]. These containers have universal dimensions to allow quick transshipment. Increased international marine shipping has increased the use of containers. This goes hand in hand with the increased demand for well-functioning container terminals. The function of a container terminal is to transship containers. Deep-sea vessels, barges, trains and trucks all arrive to deliver and pick-up containers. The container terminal ensures that the correct containers are loaded and unloaded to and from their respective carriers. The speed at which containers can be handled on the container terminal is an important performance indicator. In particular, the turnaround time of deep-sea vessels is used to determine the performance of a container terminal [2]. This is because the berth of deep-sea vessels is expensive. Furthermore, container terminals wish to decrease the turnaround time of deep-sea vessels since the container terminal becomes more attractive for shipping lines to move their wares through. Moreover, the shorter the turnaround time of deep-sea vessels, the higher the container terminal capacity.

1.1. Problem statement

The turnaround time of a deep-sea vessel is influenced by the number of Quay crane (QC)s handling the deep-sea vessel and their (un)loading rate. Therefore, container terminals wish to increase the number of QCs and their (un)loading rate. However, the maximum number of QCs that can handle a single deep-sea vessel simultaneously is limited by their size and safety regulations. Besides that, QCs are rarely able to work as fast as their technical (un)loading rate specifies [3]. This can partly be attributed to insufficient feed and discharge of containers to and from the QC [3]. In other words, once the QC is stuck with a container without the ability to discharge. Or once there is no new container available to be loaded onto the deep-sea vessel whenever the QC becomes available for the next operation, the QC remains idle for a certain time. Idle times of the QC decrease the (un)loading rate and thus increase the turnaround time of the deep-sea vessel that is being handled.
The problem of insufficient feed and discharge of containers to the QC could be reduced by either using more vehicles to serve the QC or by using the fleet of vehicles in a smarter way. The former option works only to a certain limit and requires huge investment costs. The latter option also works to a certain limit but, is much cheaper. This is because, smart usage of QC serving vehicles could be re-used for multiple terminals and is easily scaled. One method to use the fleet of vehicles in a smarter way is by creating sophisticated schedules that minimize the idle times of the QC.
Schedules could be subdivided into three different categories which are strongly connected. These categories are scheduling, dispatching and routing. Scheduling signifies the process of assigning operating times to the individual containers. Subsequently, dispatching is a process during which equipment is assigned to handle the individual containers. Thereafter, routing describes the process of determining the routes of equipment required to handle the containers. Coordinated schedules are schedules that recognize the connection between the different categories and adapt their schedule accordingly. Furthermore, coordinated schedules integrate these categories while considering all equipment simultaneously.

On a container terminal, containers are being moved between different carriers via a stack. In the stack, containers are temporarily stored awaiting to be loaded on the next carrier. A schematic drawing of a typical container terminal is presented in Figure 1.1.

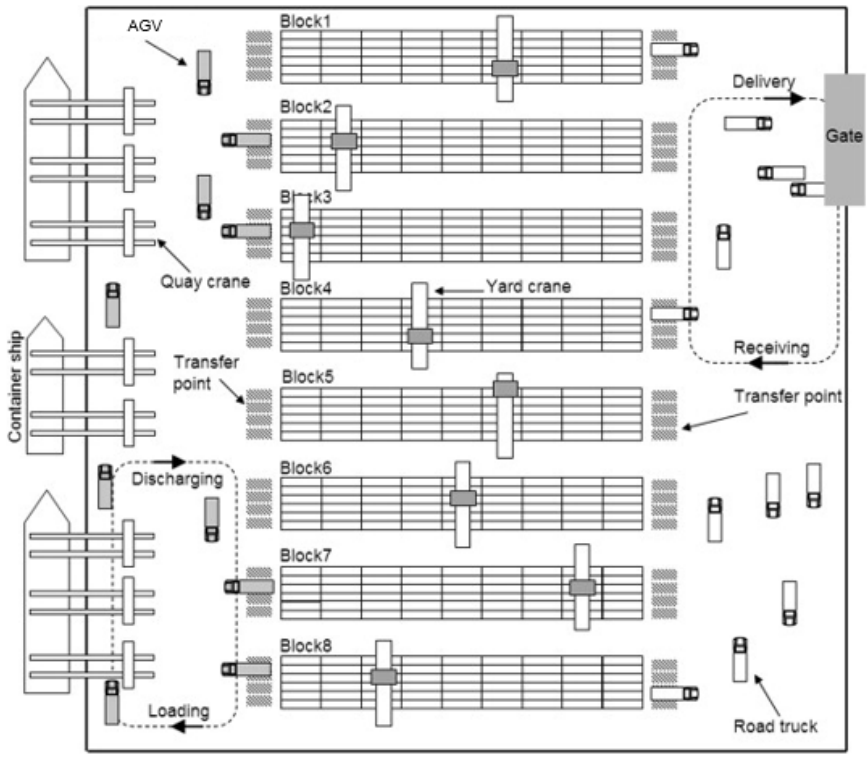


Figure 1.1: Schematic drawing of a container terminal [4].

This report focuses on the interchange of containers between the stack and the deep-sea vessel, called the seaside of a container terminal. This interchange requires three different types of vehicles. These types of vehicles will further be addressed as machine sets.

1. **Machine set 1- Yard cranes:** are responsible for handling containers within the stacking yard. Yard Crane (YC)s connect Horizontal Transport Vehicle (HTV)s and road trucks. In case of linked interchange, containers are loaded and discharged directly on/from the HTV. In case of unlinked transport, containers can be placed somewhere where both the Yard Crane (YC) and the HTV can access the container. Considering road trucks, the interchange is always linked. However, road trucks are handled on the landside whereas HTVs are handled at the seaside.

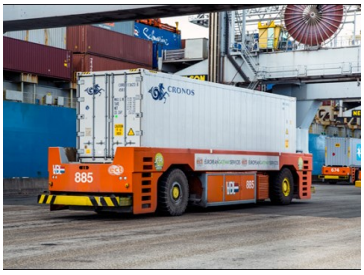
2. **Machine set 2- Horizontal transport vehicles:** are responsible for delivering and picking-up containers at the quay cranes. Containers are moved to and from the yard where the HTVs discharge and obtain their containers. This can either be linked or unlinked.

3. **Machine set 3- Quay cranes:** are responsible for loading and unloading containers to and from the deep-sea vessel. In nearly all cases containers are obtained and delivered directly from and to a HTV.

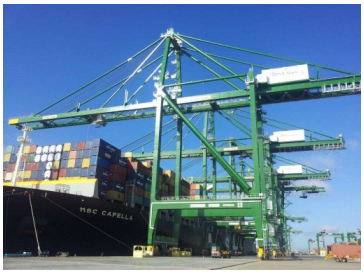
Each machine set is responsible for one and only one move within an order. On a container terminal, there are multiple types of orders: loading, discharging, housekeeping, custom inspection and repairs. This research focuses on loading orders and discharging orders. Each order type requires its own sequence of moves to be completed. For loading orders the move sequence is: Machine set 1 followed by Machine set 2 followed by Machine set 3. For unloading orders the sequence is reversed.
The responsibilities of each machine set are fixed but, the design of the machine is still variable. YCs are either Rubber Tyred Gantry crane (RTG)s or Rail Mounted Gantry crane (RMG)s. These are both

overhead cranes that can transport containers over a certain lane in the stacking yard. RTGs are capable of switching lanes whereas RMGs should remain in the same lane.
HTVs have the most variable designs. These are: Automated Guided Vehicle (AGV)s, Lift Automated Guided Vehicle (Lift-AGV)s, Straddle carrier (SC)s, Shuttle carrier (ShC)s and Terminal Truck (TT)s. The advantage of Lift-AGVs, SCs and ShCs is that they are unlinked. This means that they can pickup a container on their own. However, Lift-AGVs can only pick-up a container independently at the stacking yard. On the contrary, AGVs and TTs require containers to be loaded onto them by the QCs and YCs and are thus considered linked. Moreover, a special feature of SCs is that they access the stacking yard, which means that no YCs are required. A few designs for different equipment are shown in Figure 1.2. This figure could be used to get a feeling for the sizes of the different machine sets.

(a) RTG



(b) AGV



(c) QC

Figure 1.2: Possible machine designs on a container terminal [5–7].

Machine set 3 is leading upon the creation of a schedule for the different machines. This is for two reasons. The first is that QCs determine the turnaround time of a deep-sea vessel directly, so they should have a schedule which is close to optimal. Furthermore, QCs can only load or discharge containers in a pre-specified sequence. This sequence is constructed together with the captain of the deep-sea vessel, who is responsible for the stability of the deep-sea vessel. This means that once the loading and unloading sequence of containers is determined, QCs should obey this sequence. Therefore, it seems logical to schedule the different machine operations such that QCs can maintain their ideal schedule most closely.
However, the problem that the scheduling software currently encounters is that there is no coordination between the different machine sets and between the different machines within a set. This could mean that two different QCs wish to use the same YC at the same time, which is impossible. Therefore, the YC should give priority to one of the two moves. A similar type of conflict could occur at the HTVs. Therefore, the problem that container terminals experience is to construct a schedule which minimizes the idle times of the QC.

1.2. Research goal and scope

The goal of this research is to contribute to the improvement of container terminal performance. This research focuses on an important Key Performance Indicator (KPI), namely the productivity of QCs. More specifically, this research studies the effect of scheduling machine operations in a coordinated way to increase QC productivity.
A coordinated schedule applied to container terminals is a schedule that considers the operations of all equipment operating on the container terminal simultaneously to achieve overall optimal performance. Contrary to an uncoordinated schedule that considers operations of all equipment separately and tries to find optimal solutions for all equipment individually. Since different equipment operating on a container terminal could have conflicting objectives an uncoordinated planning does not lead to the optimal holistic solution. Therefore, a richer coordinated scheduling model than currently exists is desired to find the optimal holistic solution.
The research is bounded by a scope. This is to isolate a certain aspect of the container terminal with the purpose to study the effects of a coordinated schedule on that specific aspect. Furthermore, the scope is used to set limitations which are required to ensure that realistic solutions are found. Finally, the scope is used to narrow the size of the research.

- This research focuses on scheduling and dispatching of the three aforementioned machine sets

to handle containers. We ignore routing of individual machines. In other words, the developed schedule stops once a specific machine is assigned to a move with a specific start time. Individual machine instructions are not considered.

- Both the landside and the waterside of the container terminal are considered. The waterside defines the area where YCs, HTVs and QCs operate. On the landside, containers arrive and depart by road trucks which are directly handled by the YCs. Trains are not considered in this study.
- The research focuses on RMG cranes, Lift-AGVs and QCs. All other designs for equipment are not considered. This is because these equipment designs are most frequently automated and show good value for money [8].

Considering the scope of the research, the goal of the schedule is to determine the starting times of the individual YCs, HTVs and QCs for individual containers. Furthermore, the schedule should assign equipment to a container move.
Since the operations at the waterside of a container terminal are highly interactive, the research focuses on schedules that promote coordination among the different machine sets that are active at the waterside of a container terminal.

1.3. Research questions

The main research topic is to investigate how a coordinated schedule could lead to increased productivity of Quay cranes on a container terminal. The main research question of this report is:

Can Quay crane productivity be increased compared to an uncoordinated schedule by developing a coordinated schedule considering Quay cranes, Yard Cranes and Horizontal Transport Vehicles?

To answer this research question, several sub questions need to be answered.

1. How is the current uncoordinated schedule constructed for the different machine sets?
2. What is the current productivity of Quay cranes? under different to-be-specified operating conditions.
3. Which coordinated scheduling approaches are available for operations involving multiple stages?
4. How could coordinated scheduling approaches be applied to reality?
5. What is the theoretical performance of the coordinated scheduling approach?
6. What is the Quay crane productivity whenever the coordinated schedule is used?

1.4. Approach

TBA uses a sophisticated and validated model [9] of a container terminal to represent a real container terminal. The model allows users to evaluate the effects of different strategies in a controlled environment. The model has numerous advantages compared to reality. Since the environment is controlled, comparison between different strategies is fair. Furthermore, implementation and start-up is much cheaper compared to testing on a physical terminal. The model of TBA is considered valid by an external party. Therefore, from now on, the model will be used as reality. This means that scheduling approaches will be evaluated within a model and not on a physical terminal.
The sub questions defined in the previous section are answered in subsequent order and will eventually lead to an answer to the main research question. The approach to answer the sub questions is explained in more detail in this section.
The first two sub questions will mainly be answered by analysis of TBA's internal documentation on the current system. Furthermore, the model of the current system will be studied carefully to get a detailed insight of the current operation. This information will be complemented with a small literature study. Furthermore, simulations can be used to acquire the current productivity of QCs under the to-be-specified operating conditions.

To find available coordinated scheduling approaches, an extensive literature study will be performed. This literature study will not restrict itself to container terminals alone but, will be focused on coordinated scheduling with multiple stages in general. Subsequently, a synergy between the current system and the literature should be formed. This will, most likely, require a combination and modification of multiple coordinated scheduling approaches which were found during the literature study. Probably, the coordinated scheduling approach consists of a model and a solution technique. First, the model should be defined mathematically. Afterwards, the model should be implemented and possibly modified to be usable by the selected solution technique. The theoretical performance of the combined and modified scheduling approach will be evaluated. The theoretical performance of the coordinated scheduling approach will determine whether the coordinated scheduling approach can be solved, within reasonable time, to optimality or whether it requires approximations. It could be that multiple new coordinated scheduling approaches can be constructed from the literature. In that case, the theoretical performance of the coordinated scheduling approach will determine which of the coordinated scheduling approaches will be selected to proceed to the next stage of the research.
The selected coordinated scheduling approach, should then be implemented in the simulation model of TBA. It is not yet clear if this will be done directly in the software of TBA or with intervention of an external software package.
Once the coordinated scheduling approach is implemented in the simulation model, simulations can be used to determine the performance of the coordinated scheduling approach. This performance can thereafter be compared with the current performance of the system in a simulation environment considering the to-be-determined different scenarios.
It is expected that the above stated approach will lead to sufficient information to answer the main research question.

1.5. Structure of the report

In this section we present the global structure of the report. The structure of the report is as follows: First, some general information with regards to container terminals is presented in Chapter 2. Subsequently, in the same chapter, the current scheduling technique of TBA is discussed together with its performance. In Chapter 3 an extensive literature study is performed to find coordinated scheduling techniques whether or not applied to container terminals. Furthermore, previous efforts to coordinated scheduling on container terminals is categorized and their relevance is discussed. First, different mathematical models for coordinated scheduling are discussed. Second, available solution techniques to these mathematical models are introduced with their advantages and disadvantages. Thereafter, in Chapter 4 the mathematical model that is used to represent the container terminal of this study is presented. Subsequently, the solution technique that is used to solve that mathematical model is presented. The performance of the model and the solution technique is theoretically evaluated in this chapter. Thereafter, in Chapter 5, the coordinated schedule is integrated with TBA's software. Experimental results are published and compared with the realised performance of the current uncoordinated schedule as used by TBA. Finally, we present our conclusions and suggest future research directions Chapter 6.

2

Introduction to container terminals and analysis of the current scheduling and dispatching strategy

In this chapter we will start with an introduction to container terminals. This introduction will discuss the layout of a container terminal and the equipment that is deployed. Furthermore, some specifics of containers and the stacking yard will be discussed to identify containers and their location. Throughout the chapter, several terms used on container terminals will be explained. Subsequently, the current scheduling and dispatching strategy on container terminals used by TBA will be investigated.

2.1. Introduction to container terminals

In this section we introduce several terms used on container terminals. Furthermore, it will give some general information with regards to a container, the container terminal layout, deep-sea vessels, YCs, HTVs and QCs.

2.1.1. Container

A container is a standardized box which allows for easy intermodal transport without the need to reload upon switching carrier [10]. The standardization of containers results in flexibility, low transport cost and rapid transshipment [11]. However, there are different sizes of containers. These are expressed as Twenty-Feet Equivalent units (TEU). The most common sizes are 20-foot (6m x 2.6m) and 40-foot (12m x 2.6m/2.9m) containers, respectively one and two TEU. These account for 84% of all transported containers worldwide [12]. Two 20-foot containers make one 40-foot container, this feature is commonly used to handle multiple containers simultaneously. To fix and secure a container, twistlocks are used. These are positioned at each of the eight corners of a container. In contrast to dry bulk containers, there are reefers. Reefers use external power to control the temperature inside the container. This allows them to store perishable goods. Reefers account for another 6% of the total amount of transported containers [12].
Whenever a group of containers is considered, there are some important terms to classify the composition. First is the TEU-factor, which expresses the average size of containers within a group of containers. Once a group of containers is composed of solely 20-foot containers, the TEU-factor is 1. Once there are only 40-foot containers, the TEU-factor is equal to 2. Second is the dwell time, which expresses the average time that containers remain in the stack of the container terminal before they are transported. Thirdly, there is the modal split of a container stack. The modal split defines, in percentages, how many containers arrive and depart by what modality [13, 14].
Furthermore, there are certain characteristics which determine the location of the container within a stack. Containers are grouped together based on their shared attributes. These attributes are: container type, container size, container weight (divided into weight classes) and container destination. Especially containers which are to depart the container terminal by vessel need to be organized very

neatly. This is because these containers need to be loaded quickly onto the vessel. The purpose of organizing containers within a stack is to reduce the number of shuffle moves. Shuffle moves are moves that are required once a container cannot be directly handled since other containers are on top. However, sometimes shuffle moves are unavoidable because a container switches destination while it is already in the yard. In that case, this container needs to be removed from the pile. These containers are called rollings [13, 14].

2.1.2. Container terminal layout

A container terminal is primarily a material handling system; it is a link in the intercontinental transportation chain. Within this chain, the container terminal fulfills a buffer function in between sea and land transportation. Furthermore, the container terminal provides secondary services like inspection, washing, repairs and cargo consolidation [13]. The layout of a container terminal was already shown in Figure 1.1. Additionally, a section view of a similar terminal can be found in Figure 2.1.

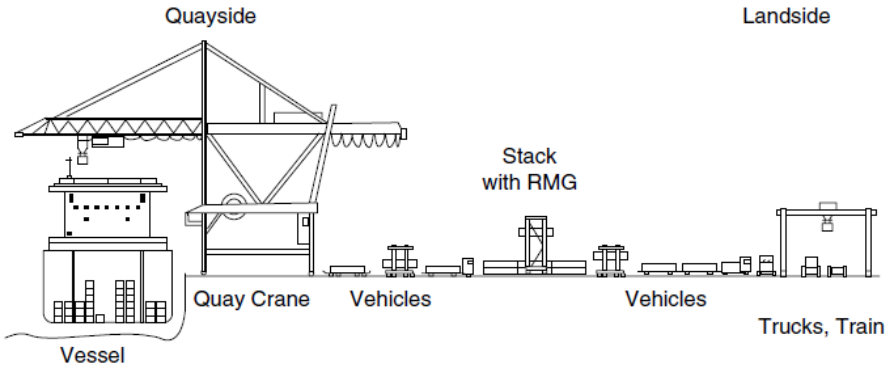


Figure 2.1: Section view of a container terminal [15].

The quayside is sometimes also referred to as the waterside. At the quay, both deep-sea vessels and barges are served. Deep-sea vessels sail on seas and oceans and move containers between large container terminals. Whereas barges use rivers and canals for inland transport of containers over water. At the landside of a container terminal, trucks and trains are served from the same stack. Trucks all enter and leave the terminal via a gate [15].
The stack on a container terminal is organized into certain blocks which are just as wide as the span of the YC. Within a block, container terminals use a specific coordinate system to describe the location of individual containers. This coordinate system is shown in Figure 2.2. Within the stack, certain locations are reserved for special types of containers. These are: reefers which require electricity, dangerous goods and alternative sized containers [15].

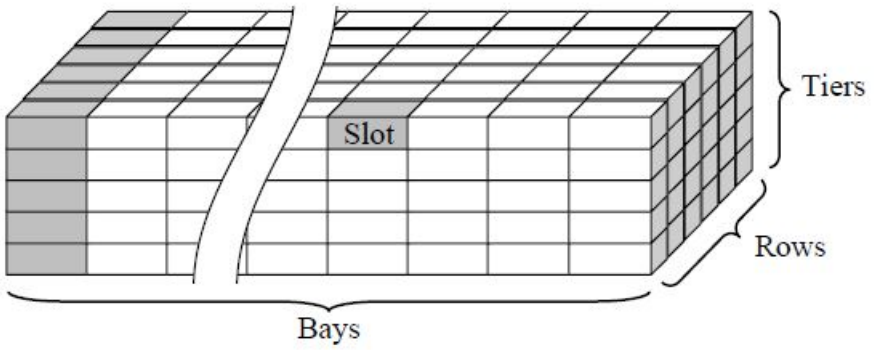


Figure 2.2: Coordinate system to describe the position of a container within a block [16].

Most yards have their blocks either parallel or perpendicular to the quay. Both stacking arrangements have their advantages and there is no general consensus which stacking arrangement is superior [17]. As can be seen in Figure 2.3, the transfer point in parallel stacks is along the entire block. This means that the width of the YC is one container wider then the block itself to allow for interaction with the HTVs. Once perpendicular stacks are used, the transfer point of containers is located at the head of each block. The perpendicular layout is most common once Lift-AGVs are used. In that case, racks are placed at the transfer point. Both Lift-AGVs and YCs can interact with this rack by themselves [18].

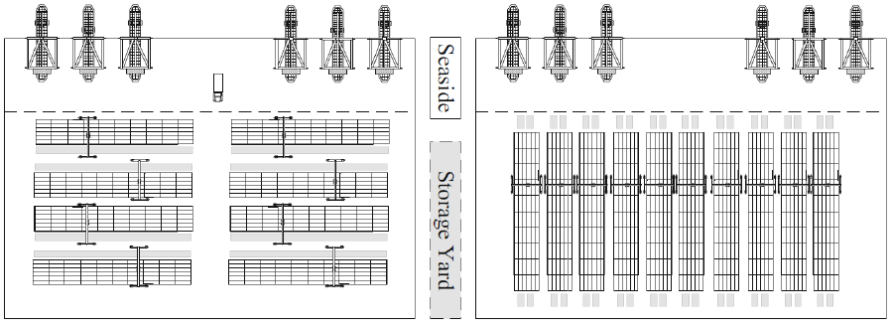


Figure 2.3: Parallel stack (left) arrangement and perpendicular stack (right) arrangement [18].

2.1.3. Deep-sea vessels

With increasing intercontinental freight transport, also the deep-sea vessels have increased in size significantly. This effect is shown in Figure 2.4. On the right side of the Figure, views of individual bays are shown. With the grey scale, the number of tiers in the hold and on the deck of the ship is indicated. Upon loading, first the hold needs the be completely filled before the deck can be loaded. Upon discharge, this is in reverse order [19]. Furthermore, containers are always handled in such a way that the container can be seen from the quay. For loading this means that the containers are handled from waterside to landside and in reverse for discharging [20]. Bigger vessels also lead to larger call sizes [3, 21]. Larger call sizes are not necessarily negative since less time is wasted with mooring vessels to the quay [21].

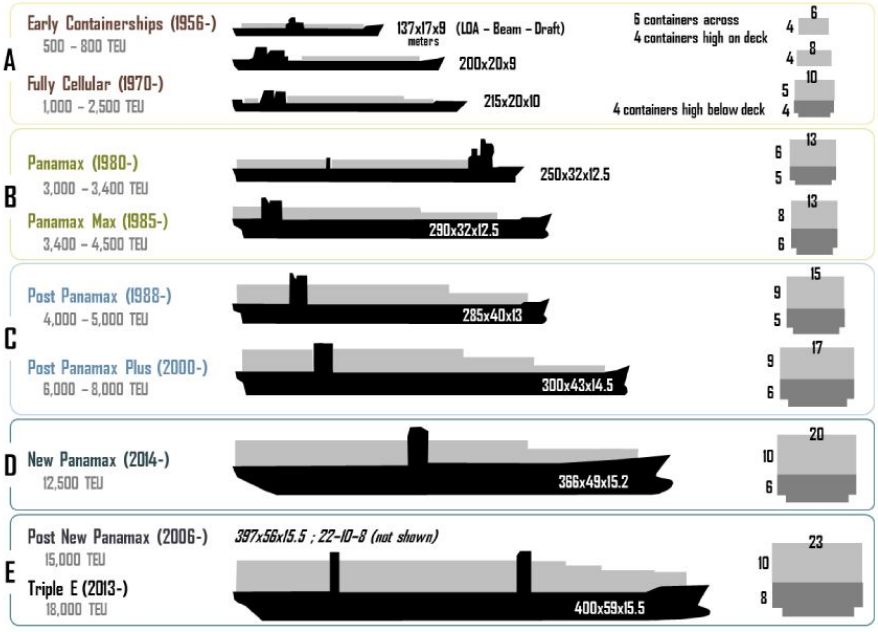


Figure 2.4: Container ship size development over the years [22].

Once a ship arrives at the terminal, a berthing location needs to be assigned to this ship. This is a complex assignment problem in itself; since container terminals typically have multiple quays available and not all quays have the same characteristics [23, 24]. Subsequently, quay cranes need to be assigned to specific bays on the container ship. This is also a though scheduling problem since there are multiple quay cranes and variable bay sizes [23–26]. Furthermore, it is complicated to assign quay cranes to bays since the QCs cannot pass each other. Besides that, it is important to note that the container terminal proposes a loading or discharging sequence but, that the captain of the ship is responsible for the stability of the ship [20]. Therefore, the captain of the container ship is the person who approves the schedule and without his approval, operations cannot start.

2.1.4. Yard crane

A YC is a crane that can pick and place containers from and in the container yard. There are typically three different types of stacking cranes which all define a specific yard layout [27]. This is illustrated with Figure 2.5. The most important difference is the location of the transferzone. This research will not consider the usage of SCs since once SCs are used, there are only two handling stages (QCs and SCs). That is why no transfer point is indicated for the SC-stack. Both RMG cranes and RTG cranes have three translational degrees of freedom: gantry, trolley and spreader movement. Gantry drive is used for switching bays, trolley ride is used to switch rows and the spreader is used to change the tier. Sometimes the spreader is also able to rotate if required. Typically, both RMG and RTG cranes can span 8-12 rows and stack up to 10 tiers high [15]. However, high stacks are preferably avoided since the chance for shuffles is large and other containers need to be lifted high to pass high placed containers.

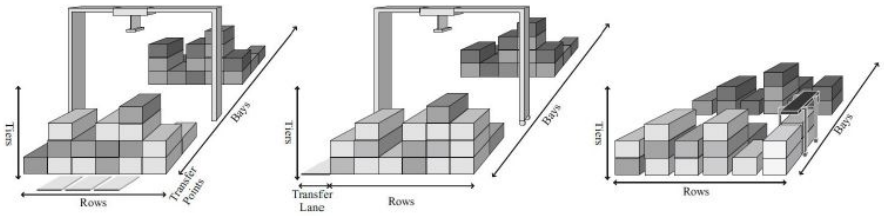


Figure 2.5: Examples of yard layouts and their respective YC. Form left to right: RMG-stack, RTG-stack and a SC-stack [28].

Rail mounted gantry crane

RMG cranes are attached to a rail on the sides of a stack. Therefore, they are unable to switch blocks within the stack. Often, RMG cranes are fully automated. Therefore, sometimes they are referred to as Automated Stacking Crane (ASC)s. Since on both the head and tail of the stack transfer points are located, trucks can be served directly from the stack. Whenever the stack is long, multiple RMG cranes can be used on a single block [29]. Considering movement, the spreader is always moved to a specified position before the gantry and trolley can move, these are allowed to move simultaneously [14]. This research will mainly focus on RMG cranes since these are automated.

Rubber tired gantry crane

Contrary to RMG cranes, RTG cranes drive freely on the container terminal since they have rubber tires. This allows them to switch lanes, for instance if one stacking module is more heavily used. Furthermore, RTG cranes are operated manually. Therefore, they are popular in countries where labor costs are low [27]. Often stacks are placed parallel to the quay once RTG cranes are used. Considering movement, gantry travel of RTG cranes is only allowed once the trolley and spreader are stationary. However, the trolley and spreader are allowed to move simultaneously to speed up the process [14].

2.1.5. Horizontal transport vehicle

As was discussed in Section 1.1, a HTV is responsible for moving containers between the stack and the quay cranes. This research will focus on Lift-AGVs. Lift-AGVs can pick-up and deliver a container from/to a rack which is also accessible by the YC. The advantage is that handling speed is increased since the Lift-AGV does not have to wait for a YC or vice versa. Lift-AGVs are bidirectional, which means that they do not need to turn, upon delivery at either of the two crane types. Furthermore, Lift-AGVs have

space for two TEU. This means that they can also carry two 20-foot containers simultaneously. With regards to driving and collision avoidance, a central planning system is used to claim space for specific Lift-AGVs. Usually, Lift-AGVs are given priority on a First Come First Serve policy which means that a Lift-AGV tries to minimize its own travelling time without considering other Lift-AGVs [14]. Furthermore, the central planning system determines the directions and the speed of the individual Lift-AGVs [18].

2.1.6. Quay crane

The most critical pieces of equipment on a container terminal are the QCs. They are most expensive and directly influence the turnaround time of vessels. Furthermore, together with the water depth and the quay size, the size of the QC determines which vessel sizes can be handled. Similar to a YC, QCs can move their gantry, trolley and spreader. To save time, the QC only moves the gantry once a complete bay is handled. QCs have the ability to load single containers or multiple containers simultaneously. The distribution between the different moves is strongly related to the considered terminal. Twinlift moves means that two 20-foot containers are hoisted as one 40-foot container. This requires one AGV since they can carry two 20-foot containers simultaneously. Whereas in tandem moves, two containers are loaded side by side. This is mostly done for two 40-foot containers. Tandem moves require two AGVs since containers are loaded side by side.

The performance of a container terminal is often expressed as the average turnaround time of deep-sea vessels. An important KPI which influences the turnaround time is the productivity of the QCs. The QC productivity is expressed as the number of boxes per hour. An extensive study into QC productivity is performed by Schim van der Loeff [3]. Within this study, the operation time of QCs is divided into inactive losses and cycle time. Inactive losses occur once a QC should be moving but, stands still. Cycle time represents the time during which the crane is in operation. Cycle time was further subdivided in inactive cycle intervals and active cycle intervals. Inactive cycle intervals contain delays such as; positioning of the spreader. Active cycle intervals are the actions which add value to the operation.
Schim van der Loeff [3] found that on the European Container Terminal (ECT) the achieved productivity was 22.5 boxes/hour. When all inactive losses could be avoided, the productivity could increase to 30.47 boxes/hour. Almost 10% of the total production time of the QC is lost because the QC is waiting for an AGV. With a more advanced schedule, AGVs are expected to be more punctual and thus the QC productivity increases.

2.2. Analysis of the current scheduling and dispatching strategy

The analysis of the current operation will be based on the simulation model of the current system constructed by TBA [14]. This simulation model is a representation of reality. However, it shows close resemblance with reality. Therefore, the simulation model will be used to evaluate the performance of the different schedules instead of a real terminal.
A simulation study is initiated by a scenario. The scenario contains the vessel, truck and train arrivals and their respective cargo. Furthermore, it defines the type of equipment on the terminal and its specifics. On top of that, the scenario distributes the work over the different QCs. The scenario only releases certain information to the terminal at specified times which resembles reality. For vessels, 18 to 24 hours in advance of arrival; the load list is shared. This allows stacking cranes to start 'housekeeping'. This means that containers are moved to favourable locations in the stack. 60 to 120 minutes before arrival of the deep-sea vessel, the QC determines the load/unload sequence. The sequence is mainly based on the weight class and port-of-discharge of the containers. This means that containers can be shuffled in the yard to promote quick handling. Approximately 20 minutes prior to arrival, containers are allowed to be moved from the stack to the waterside buffer.

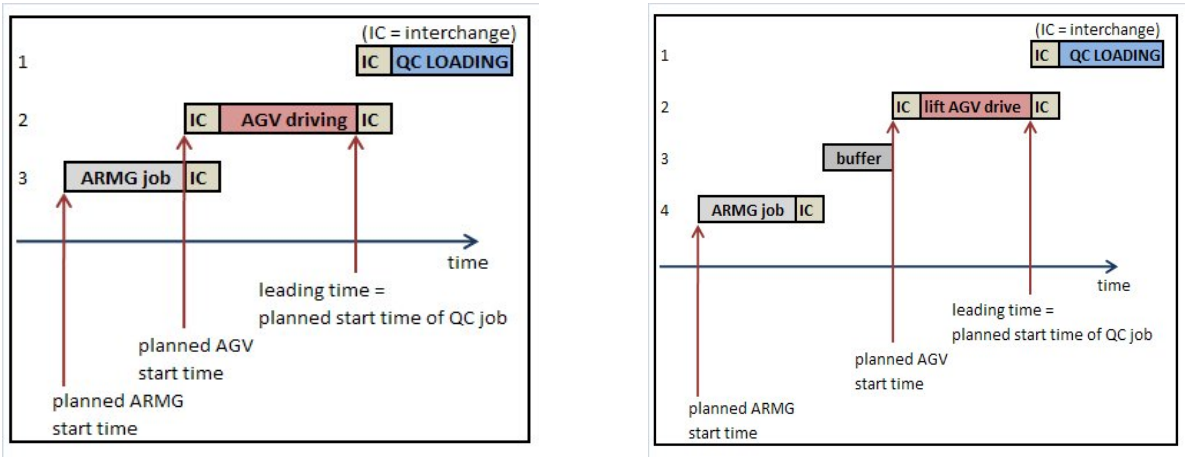
TBA uses a complicated classification method for different parts of the planning on the terminal. The different classes are:

- **Work:** Contains a list of orders, multiple containers must be (un)loaded by one specific QC.
- **Order:** Is a list of moves. An example of an order is: Container A must be moved from stack module 1 to QC 4.

- **Move:** Contains a list of jobs. Moves are operations for different machine sets. For example: Container A must be moved from stack module 1 to the waterside transferpoint by a YC.
- **Job:** Specifies a list of instructions. A job itself could be: Transport container A from position [bay,row,tier] in stack module 1 to waterside transferpoint B by ASC 1.
- **Instruction:** An instruction is the lowest level of actions these actions specify the different operations very detailed. One example could be: ASC 1 should gantry travel 2.6 meters.

2.2.1. Planning and scheduling

Once the work is distributed to the QCs, the planning and scheduling of the moves on the other machines starts. Based on the operation times of the QC and the cycle time estimates, the desired start times of the other machines could be determined.
The cycle time estimates of loading orders are more accurate compared to those of unloading orders. This is because the exact location of a container in the yard is known far in advance. This allows for an accurate cycle time estimate of the YC and the HTVs. Both the YC and the HTV determine their cycle time based on the distance between the origin and destination of the move. However, possible congestion on the route is ignored. This is a routing problem, which is out of scope of this research. Therefore, during this research, the estimate is considered accurate. Finally, the QC cycle time is estimated based on the type of hoisting (single, twinlift or tandem). In Figure 2.6 an example of the schedule for different moves is shown. Moreover, this Figure illustrates the difference in planning strategy between linked and unlinked HTVs. The capacity of the buffer determines the robustness of the schedule and the filling rate of the racks.

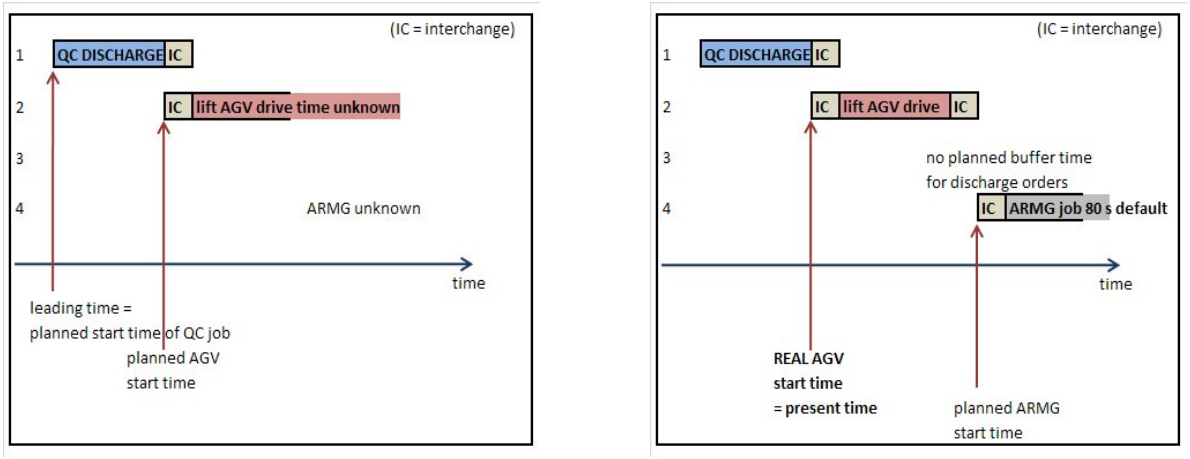


(a) Cycle time estimates for linked interchange

(b) Cycle time estimates for unlinked interchange

Figure 2.6: Cycle time estimates for loading orders based on the start time of the QC operation [14].

Unloading orders are slightly more complicated to schedule since the destination of the container is unknown. Similar to loading orders, the QC start time is known so the interchange with the HTV could be scheduled directly. However, the stack module where the container should be placed in the yard is only determined once the container is being loaded on the HTV. At that moment, the HTV can estimate, with the same accuracy as for loading orders, at what time it will arrive at the stack. For unloading orders, no buffer is scheduled since unloading containers have a lower priority compared to loading orders (see Section 2.2.3). Naturally, in case of linked interchange, a HTV will be discharged with a higher priority such that it becomes available for the next move. This procedure is illustrated with Figure 2.7. The final end time of the ASC move is only determined once the ASC is assigned to the container.



(a) Schedule 60-120 minutes prior to ship arrival (b) Schedule at the first interchange

Figure 2.7: Cycle time estimates for discharging orders based on the start time of the QC operation [14].

It should be noted that all of these start times indicated in Figures 2.6 and 2.7 are desired start times from the perspective of the QC. The start times are not necessarily realistic nor do they have any relation with the capacity of the system. For instance, the number of available HTVs is not included in these desired start times. Therefore, this initial schedule is referred to as planning. Based on this planning the individual machine (sets) will try to meet the planning by developing their own schedule without further considering the other equipment.

2.2.2. Stack crane scheduling

A possible outcome of the constructed plan for a single YC could be the one as shown in Figure 2.8. Naturally, this plan is not feasible, multiple moves need to be performed simultaneously. Such unfeasible plans occur once the QC is quicker then the YC and requires multiple operations within the same stack module. Or once multiple QCs require operations within the same stack module. When the YC cannot follow the plan, it will construct a new schedule which most closely resembles the plan.

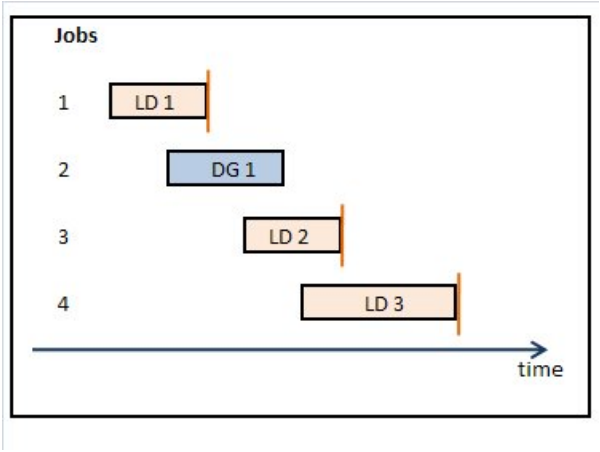
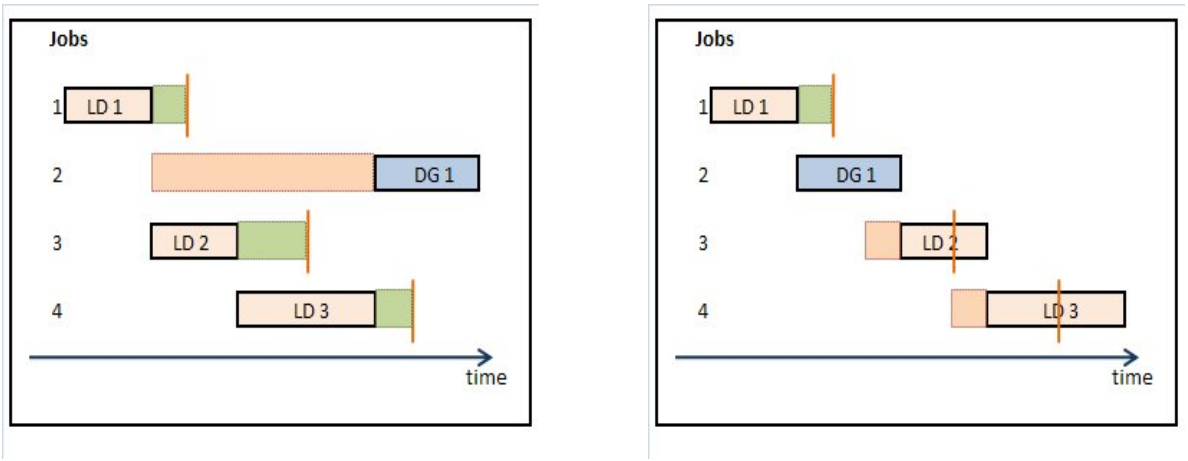


Figure 2.8: Conflict of multiple container moves (three loading moves, one discharge move) planned at the same time [14].

In case of unlinked interchange, the YC prioritizes loading moves over discharging moves. Because delayed load orders delay multiple machines in contrast to delayed unloading orders which only cause delays at the YC. Furthermore, the YC will try not to postpone any orders to minimize the effect on the QC operations. The adjusted feasible schedule is indicated in Figure 2.9a. Note that moves can only be scheduled forward to a certain degree. Because of previous orders which are not displayed in this Figure. For linked interchange discharge, moves cannot be postponed for so long since this would

cause large delays. This is because, the HTV will be unavailable for a long time. Therefore, the plan is adjusted to the schedule shown in Figure 2.9b. Delays for loading moves 2 and 3 are unavoidable in this case.



(a) Adjusted schedule for unlinked interchange

(b) Adjusted schedule for linked interchange

Figure 2.9: Adjusted schedule for an individual YC based on the unfeasible plan presented in Figure 2.8 [14].

2.2.3. Stack crane dispatching

Once the schedule is adjusted, the YC will not simply follow the schedule. This is partly because there could be multiple YCs working in one stacking module. And the realised schedule may have deviated from the constructed schedule. Therefore, a dispatching algorithm has been developed to assign specific YCs to specific containers. Each move which is to be performed by a YC receives a score based on the following parameters:

- Origin of the move (bay, row, tier or the specific transferpoint);
- Destination of the move (bay, row, tier or the specific transferpoint);
- Latest delivery time of the move;
- Sequence position of the move at the QC;
- The buffer occupancy, in case of unlinked interchange.

The dispatching strategy that is currently applied to dispatch the YCs is shown in Figure 2.10. The penalties in the latest block are introduced to prevent the buffer from getting full. This is because a high penalty is given to every container besides the ones currently in the buffer awaiting to be placed in the stack. Furthermore, penalties are given to containers which are likely to lead to inference of multiple YCs. The urgency multiplier reflects the nature of the order (productive move, shuffle order, housekeeping or pre-positioning) and the desired time in the schedule. Furthermore, the urgency multiplier is used to prioritize discharging a loaded HTV in case of linked interchange.

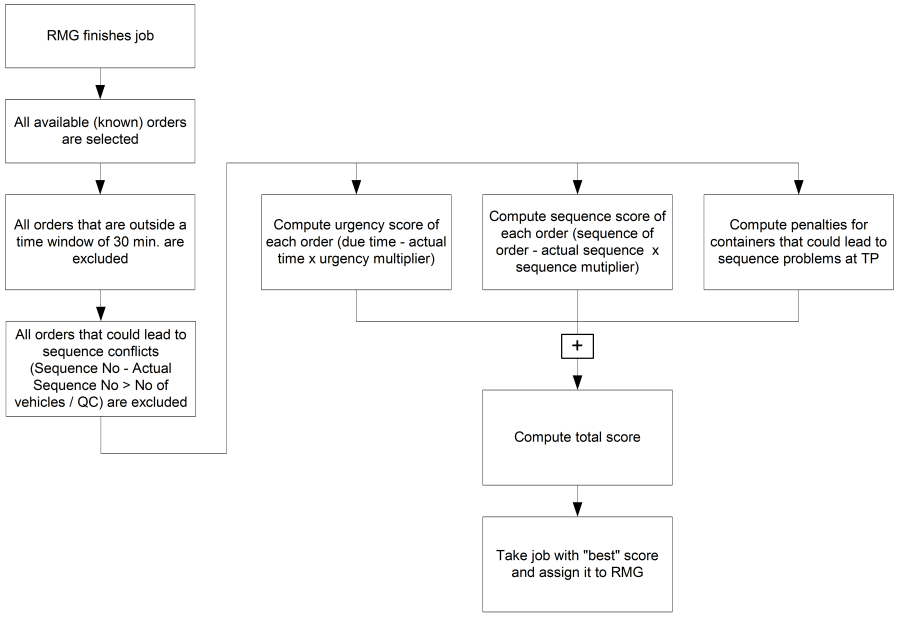


Figure 2.10: Dispatching strategy of a YC [14].

In other words, the YC schedule is used as a guideline since it directly influences the urgency score. However, it is not a strict sequence, the YC could still deviate from this sequence.

2.2.4. Horizontal transport vehicle dispatching

Not related to linked or unlinked interchange, there are two types of dispatching strategies for HTVs:

1. **Dedicated assignment strategy:** Each HTV is assigned to one specific destination. However, one destination might have multiple HTVs assigned. For instance, a HTV is merely allowed to handle moves for one specific QC. Dedicated assignment strategy is not very efficient since whenever the destination has no moves available, the HTVs are idle as well. On the other hand, dispatching with the dedicated assignment strategy is rather simple.

2. **Pooling strategy:** A group of HTVs is assigned to a group of destinations. This strategy is much more efficient at the cost of complicating the dispatching strategy. The pooling strategy is more efficient since the chance that none of the equipment in the destination group has a move available is smaller compared to when there is only one equipment in the destination group. The complexity of dispatching lies in the decision making and prioritization of certain moves.

Similar to the YC dispatching strategy, the HTV dispatching strategy relies on a scoring mechanism. From the total orderlist, certain orders are excluded on an individual basis for each HTV. This is based on the destination of the move, the due time and moves leading to deadlocks. Furthermore, each crane, QC or YC, has a maximum number of HTVs that could be assigned to it to prevent congestions. Now that each HTV has its own set of possible moves, scores are assigned to each move individually based on: the travel time, the urgency of the move, the relative handling sequence of the move at the QC and the number of moves which are already assigned to a particular destination to avoid congestion. The difference between linked and unlinked interchange is that with linked interchange, the focus is more on arriving on time instead of short driving distances. This is promoted by giving a high priority to moves which are started by the same equipment as was involved in the interchange. The exclusion and dispatching strategy applied to HTVs is shown in Figure 2.11.

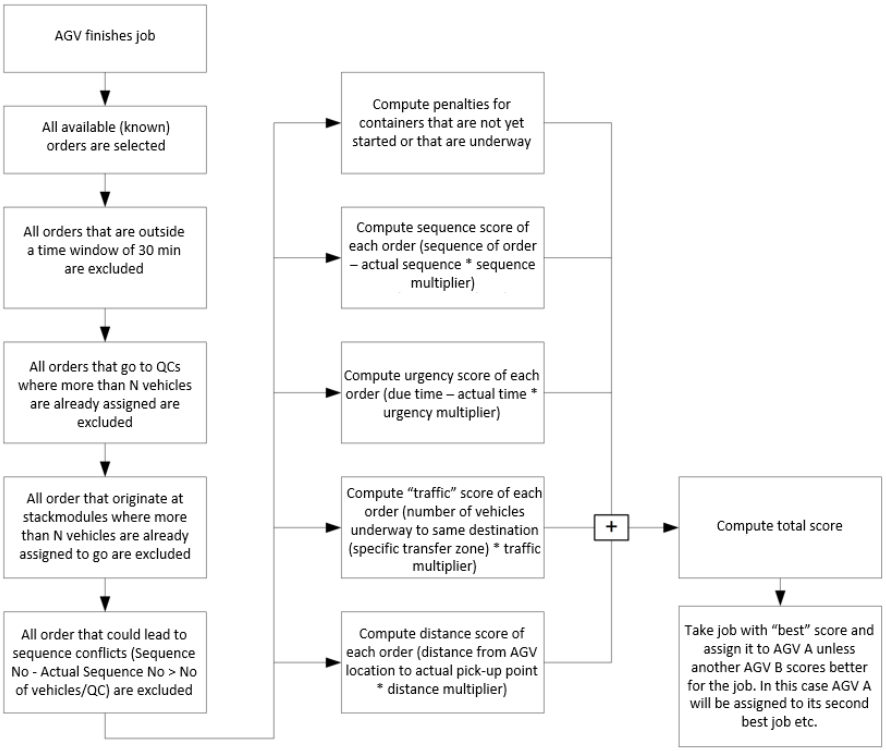


Figure 2.11: Dispatching strategy of a HTV [14].

2.2.5. Quay crane bay plans

For each bay, the QC determines the sequence of containers to be handled. The sequence strategy is different for discharging and loading orders. Note that in the real world, the load and discharge plan is constructed together with the captain of the vessel who is responsible for the stability of the vessel and has the final say.

- **Discharging:** the model generates a batch of discharge containers to fill up a vessel bay. The number, the type and the size of the containers is determined randomly based on user specified distributions. Subsequently, the sequence of discharging containers is determined solely based on the weight class. Such that the containers in the heaviest weight class are placed in the lower tiers and the containers in a lighter weight class are placed on the higher tiers.
- **Loading:** the model generates a batch of loading containers with a random port of discharge, type and size. The QC randomly selects containers from the yard that match the specifics. Subsequently, the sequence of container loading is determined intelligently based on the weight; the heaviest containers are placed on the lower tiers. Furthermore, yard operations are optimized; top tier containers are selected first and the sequence of containers is spread over multiple stack modules to reduce the workload. In case of twin-lifting operations, containers from the twin-loads either originate from the same stack to reduce travel distance of the HTV or from different stacks to reduce workload. The selected strategy is pre-specified before the simulation starts and is based on the expected workload on the different machine sets.

Once the sequence of container handles is determined, the start times of the different moves are also determined. This process also initiates the scheduling of the other machine sets (see Section 2.2.1). Currently the schedule remains fixed and is not coordinated with the other machine sets. However, once a move is delayed more then two minutes, the QC updates its schedule to reflect the new status.

To avoid deadlocks, the QCs make sure that the sequence at which HTVs are directed to the QC transferzone corresponds with the sequence of operations. The sequence number of the container on

the HTV arriving in the QC waiting area is compared with the actual sequence number. Once these are equal there is no sequence conflict and the HTV is allowed to proceed to the transferzone. Sometimes the transferzone at the QC can hold N HTVs. In that case, the first N containers from the sequence may proceed to the transferzone. Whenever this condition is not satisfied, HTVs are usually directed to the QC waiting area. However, sometimes the model is given permission to handle out of sequence moves. This can be done according to three different strategies:

1. **Fixed sequence gap:** All orders with a sequence gap below a user-specified threshold are allowed to be handled by the QC.
2. **Vessel-bay-dependent sequence gap:** All containers that can be loaded on their originally planned slot are allowed to be loaded. This means that all containers directly on top of the current stored containers are allowed to be loaded.
3. **Category loading:** Containers are allowed to switch order given that they have the same port of discharge, type and weight class.

However, the handling time for out of sequence orders is increased since these are slightly harder for the QC. The process of ensuring correct sequences is visualized in Figure 2.12. A ShC is a type of HTV (see Section 1.1).

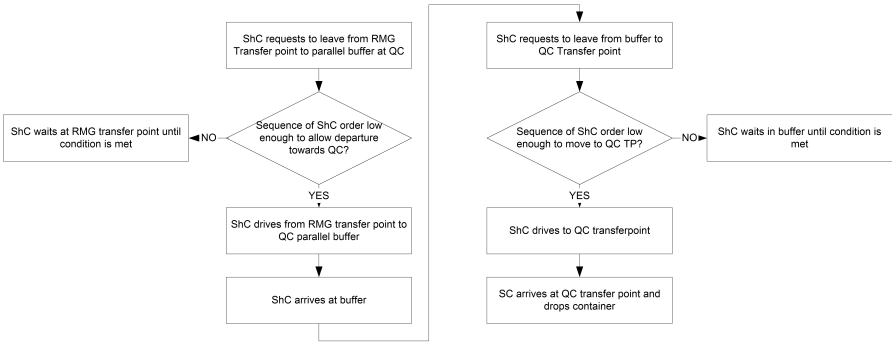


Figure 2.12: Sequence manager scheme [14].

2.2.6. Grounding

Grounding is a definition used to specify the logic and strategy which is used to determine the locations of inbound containers in the yard. Generally, grounding is a two step procedure. First, the stack module is determined. Thereafter, the specific slot within the stack module is determined. A rule of thumb with regards to grounding is: the later, the better. Since the later the decision to assign a grounding location is made, the more accurate the yard information is.

Stack module assignment

A stack module is assigned to a container once it is unloaded from a deep-sea vessel or train and placed on a HTV or once a road truck arrives at the gate. The stack module is selected based on a scoring mechanism incorporating the following criteria:

- Distance from the origin to the stack module transfer point;
- The expected next destination, preferably a container is brought close to the next destination;
- Presence of similar containers (modality, container type, container size, container weight and port of discharge), preferably containers with similar properties are stacked on top of each other;
- The occupancy of the transferzone;
- The workload at a stack module, stack modules with a low workload are preferred;
- The availability of HTVs, once availability is low; short trips are preferred.

The result of this scoring mechanism is that gradually each stacking module fills to an equal filling rate. The strategy is also visualized with a scheme; shown in Figure 2.13

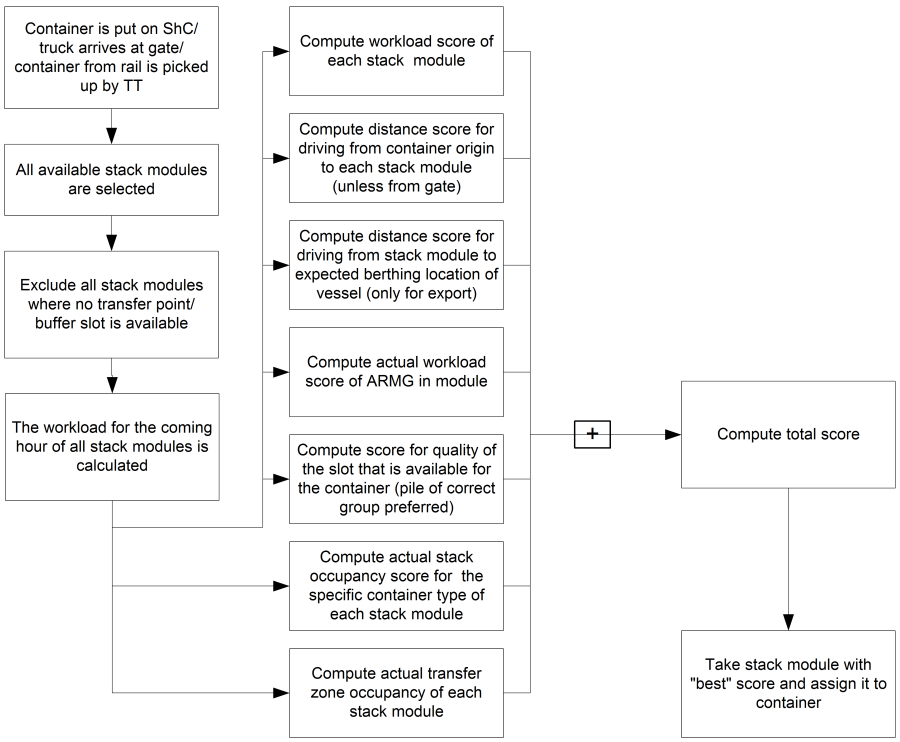


Figure 2.13: Scheme illustrating the strategy for selecting a stack module [14].

Slot assignment

A slot within a stack module is assigned to a container once a YC is assigned to a container. An ideal slot is determined based on the workload, travel distance and the possibility of placing containers with similar properties together. When the workload is very high, containers can be placed in a location close to the transferzone. Such that the YC is available quickly again for a new move. However, the container will need to be moved to the ideal slot later.

2.3. Current performance of the scheduling & dispatching strategy

The scheduling algorithm used by TBA is reactive and uncoordinated. Reactive means that decisions are made following discrete events. The scheduling algorithm thus reacts to the current situation on the terminal. Uncoordinated means that decisions are made which seem best for the specific machine that requires a new schedule. In other words, other machines are not considered when making scheduling decisions. This strategy offers a lot of flexibility with regards to scheduling and dispatching. The advantage of the current scheduling and dispatching strategy is that there is always a solution available without long computation times. The disadvantage is that the constructed schedule is not optimal when considering the container terminal as a whole. Furthermore, there is no theoretical time available at which a move is completed. This is why it is not possible to compare the current scheduling strategy with another strategy purely from a theoretical point of view. Therefore, simulations are used to estimate the effects of other strategies and only afterwards compared to the current strategy.

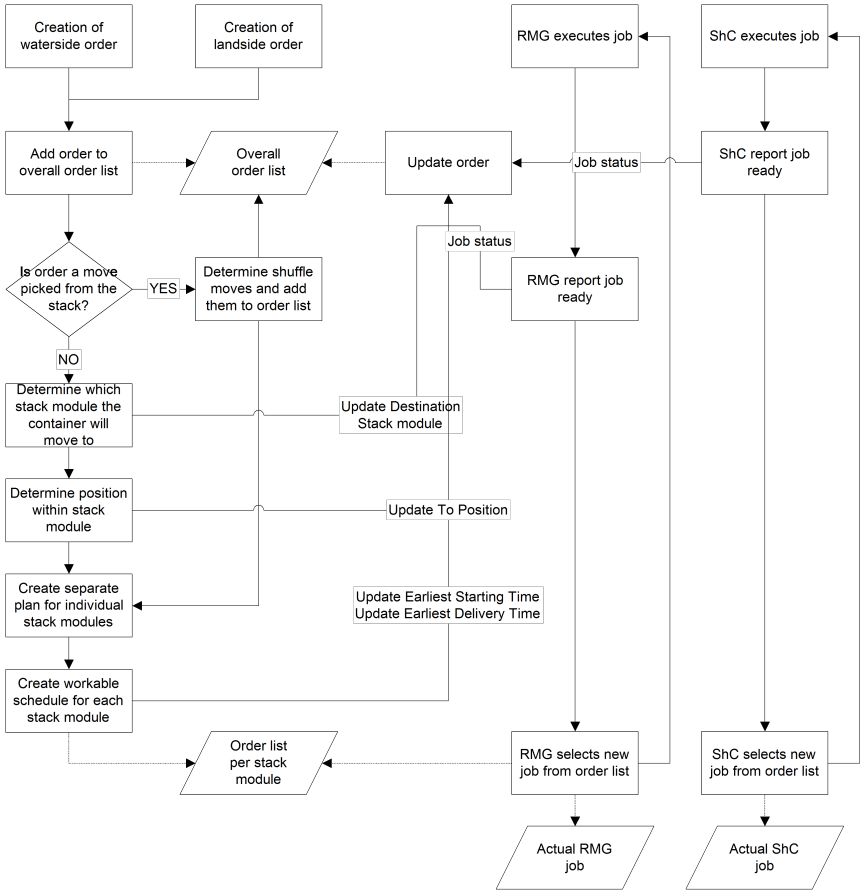


Figure 2.14: Overview of the planning with regards to waterside moves [14].

Formally, the scheduling and dispatching strategy currently in use by TBA is a heuristic control. This means that decisions with regards to scheduling and dispatching are based on a set of pre-defined rules. Based on the current situation, the system knows what to do by following the set of pre-defined rules. The advantage is that computational times are low. The disadvantage is that optimality cannot be guaranteed whenever heuristics are used. Therefore, there is a major difference between heuristics and optimized planning. Optimized planning uses a model to optimally plan container moves over a fixed horizon. The obtained solution guarantees optimality for that fixed horizon but, requires longer computation times. Furthermore, optimized plannings deliver a theoretical performance that can be used for comparison. Contrary to heuristic control that does not have a theoretical performance.

2.3.1. Simulation model validation

Most optimization models require certain assumptions and simplifications to limit the problem size. Therefore, the practical performance could be different from the theoretical performance. The practical performance of a planning cannot be evaluated on real systems since this is far too expensive. Therefore, simulations are extensively used to mimic a real system and allow quick evaluations of different strategies. However, an important question is whether the model is a valid representation of reality. First, the definition of a simulation model is given to establish a framework. Simulation is *"An experimental and applied methodology which seeks to describe the behavior of systems, in order to to understand the observed behavior and to predict future behavior."* [30]. Second, the definition of validation: *"The relation between a model, a source system and an experimental frame"* [31]. A team of TU Delft experts and TBA employees evaluated the validity of the simulation model. This was done by comparison of the simulation results and experimental performance of APM terminals. The validation team considered the models to be valid. The models behave as intended based on face validity

tests and input-output tests [9]. Therefore, the simulation model can be used to evaluate the effects of decisions on container terminal performance.

2.3.2. Benchmark terminal

To answer the research question it is important to establish a benchmark to compare different approaches [32]. The benchmark terminal is created together with TBA to represent real terminal sizes. It was decided to start with one small sized benchmark terminal to allow proper comparison. The benchmark terminal size is shown in Table 2.1. In was chosen to use Lift-AGVs as HTVs since Lift-AGVs offer the best value for money [8]. Therefore, it is expected that most people are specifically interested in performance with Lift-AGVs. The number of QCs typically determines the number of other equipment on the terminal as well. This is reflected by the factors under the different machines. The exact number of required Lift-AGVs is typically unknown and is part of the study. Logically, more Lift-AGVs leads to more boxes/hour. However, this is a decreasingly increasing function, at a certain point adding extra Lift-AGVs does not result in much more productivity. Therefore, four different factors are used to determine the number of Lift-AGVs. These factors are based on experience of TBA. The number of YCs is directly related to the number of QCs. At the landside of the yard, approximately ten containers arrive each hour by truck. However, the landside schedule is not optimized, it cannot be neglected since it introduces uncertainties in the handling times. In other words, the boxes arriving at the gate introduce disturbances. The baysize is not a function of the terminal size but, is somehow related to the terminal size. This is because at larger terminals typically larger vessels are being served. Larger vessels come with larger baysizes.

	QC	Lift AGV				YC	Baysize	Truck arrival [box/hr]
Factor	**1**	**3**	**4**	**5**	**6**	**2.5**		**25**
Academic example	5	15	20	25	30	13	50	130

Table 2.1: Benchmark problem sizes in case Lift-AGVs are used.

Besides the terminal size, other parameters have to be specified as well to generate a scenario. These parameters are introduced in Table 2.2. The TEU-factor and twinlift % are important parameters which partially determine the productivity of the QCs. Since 40 ft. containers are heavier, their processing time is longer thus the productivity decreases. Whereas twin moves increase the productivity since two boxes are simultaneously handled. Furthermore, the initial yard density is required to create a realistic scenario.

TEU-factor	1.5
Twinlift %	40
Initial yard density	0.75

Table 2.2: Parameters to support the benchmark terminals.

Additionally, the modal split of container arrivals and departures should be specified. The modal split defines the fraction of containers arrived by one modality that departs by another modality. This is useful information for the container terminal since it supports the yard allocation of containers. The location of containers in the yard directly determines the processing times of the YCs. A matrix showing the modal split at the considered benchmark terminal is presented in Table 2.3.

Arrival / Departure	Vessel	Truck	Volume Delivery Share	Yard Share
Vessel	0.3	1	0.65	0.545
Truck	0.7	0	0.35	0.455

Table 2.3: The modal split for each benchmark terminal. The first two columns indicate the fractions of transshipment. The last two columns indicate the share of containers in the yard for each modality.

2.3.3. Key performance indicators

To quantify the performance of the current system, KPIs can be used [33]. It is important to recognize that different actors value different aspects of a container terminal. However, some KPIs will be recognized by multiple actors. The most dominant one; the turnaround times of deep-sea vessels, is top priority of both shippers and container terminal operators [1, 15, 24, 34]. The turnaround time is directly influenced by the productivity of the equipment on the container terminal [19, 35]. Other relevant aspects of container terminal performance are energy consumption [36, 37] and driving distance [38]. Additionally, KPIs could be used to define the quality and practical applicability of the developed scheduling approach. The specific KPIs that are used in this research are:

- The QC productivity [boxes/hour];
- Computation time;

These KPIs are sufficient to quantify the performance of the current system with regards to scheduling and dispatching. The QC productivity directly reflects the performance of the terminal. Whereas the computation time could be perceived as the costs required to achieve the desired performance.
Other relevant performance indicators could be used to explain the QC productivity. Primarily, this is the distribution of actions of all equipment on the terminal.

2.3.4. Benchmark results

We provide results of the simulation runs on the benchmark terminal within this section.

Small terminal 5 Quay Cranes and Lift-AGVs

To acquire valid results 15 replications of each simulation are performed. This is to mitigate the effects of randomness. 15 replications was chosen as an initial guess based on experience of TBA personnel. Later, the number of replications is considered sufficient once the confidence interval of the acquired productivity is smaller than, or equal to 1 box/hr. Subsequently, the average productivity over all QCs for every hour is used as the QC productivity. The results of the simulation runs are presented in Figure 2.15. Sometimes a simulation experiment is not valid, for instance when a deadlock manifests. Deadlocks manifest when two vehicles claim the same area and are unable to move anywhere. Whenever a simulation experiment is not valid, the hours of failure are deleted. Furthermore, the first hour of each replication is also deleted to remove the start-up effects of the simulation experiment. Deleting failed simulation hours is common practice and is valid since the causes are unrelated to the investigated effects. Whenever the failed data is deleted, the number of simulation hours and the performance during the remaining hours is used to determine the average QC productivity and the 95% confidence interval. The confidence interval is determined with a two-tailed student t-test with $n-1$ degrees of freedom, where n is the number of successful simulation hours (samples). The number of replications is accepted whenever the confidence interval is smaller than, or equal to 1 box/hr. The average QC productivity and the confidence intervals are presented in Table 2.4. The number of samples is not equal for every experiment since a different number of failed hours manifested.

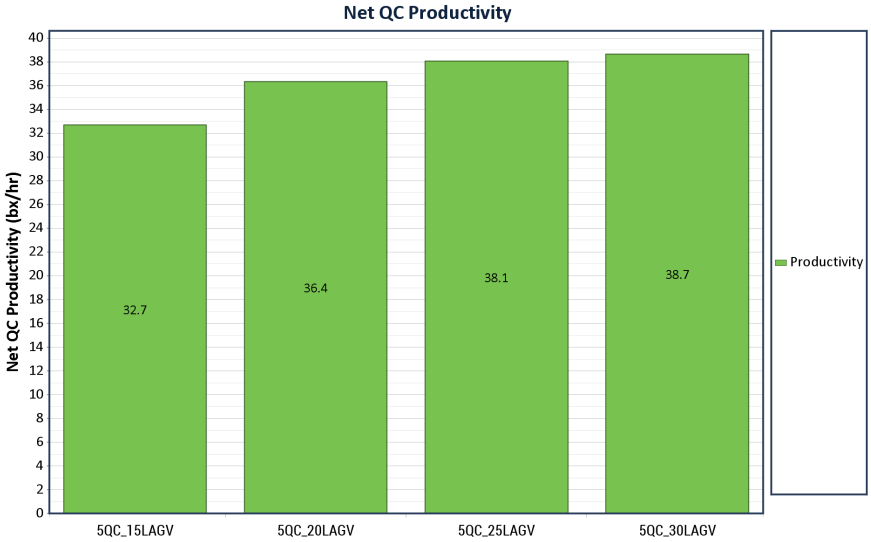


Figure 2.15: Bar-chart indicating the QC productivity on the small terminal whenever 15, 20, 25 and 30 Lift-AGVs are deployed.

LAGV		**15**	**20**	**25**	**30**
Average QC productivity	[box/hr]	32.71	36.36	38.07	38.67
Standard deviation	[box/hr]	3.68	4.03	3.95	4.05
Samples		98	99	98	105
$t_{.975}$		1.98	1.98	1.98	1.98
Confidence interval	[box/hr]	±0.74	±0.80	±0.79	±0.78

Table 2.4: Confidence intervals for the benchmark terminal with 5 QCs

As can be seen in Table 2.4 the confidence interval is smaller than, or equal to 1 box/hr for every number of Lift-AGVs. This means that the number of samples is considered sufficient. Subsequently, the status of every equipment type is used to extract more detailed information regarding the productivity. The status reports are shown in Figures 2.16 to 2.18.

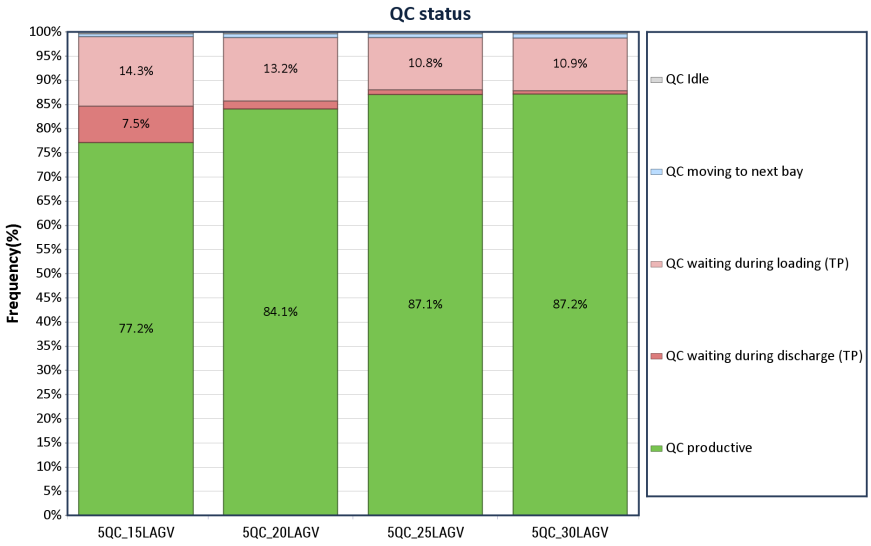


Figure 2.16: Bar-chart indicating the QC status on the small terminal whenever 15, 20, 25 and 30 Lift-AGVs are deployed.

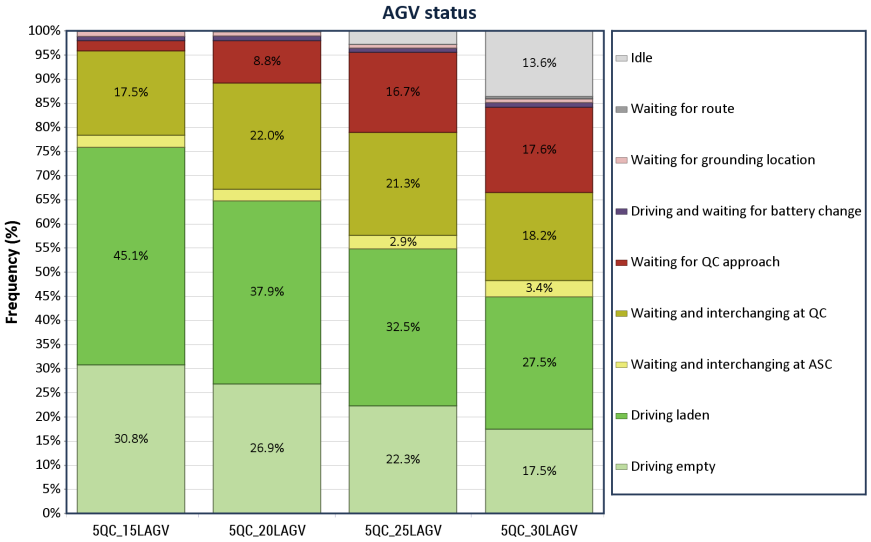


Figure 2.17: Bar-chart indicating the Lift-AGV status on the small terminal whenever 15, 20, 25 and 30 Lift-AGVs are deployed.

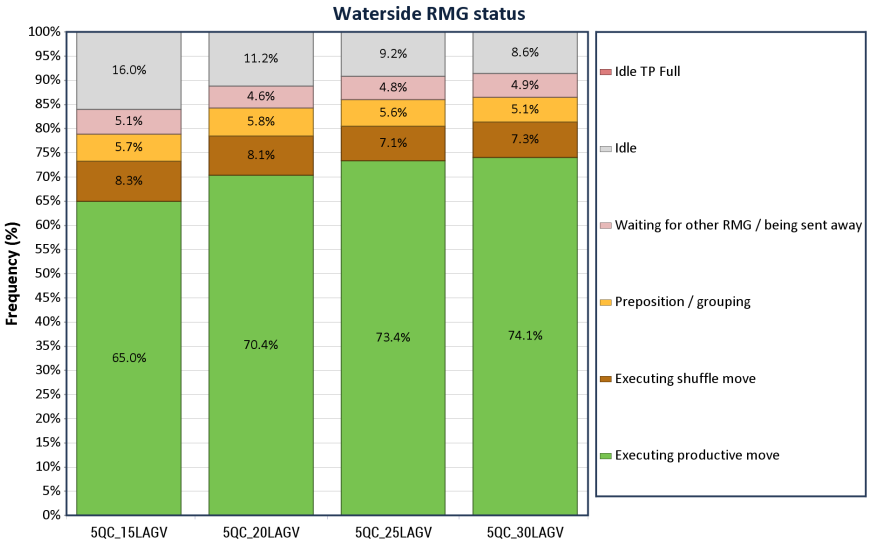


Figure 2.18: Bar-chart indicating the RMG crane status on the small terminal whenever 15, 20, 25 and 30 Lift-AGVs are deployed.

Analysis

Additional results of the simulation runs that can be used for verification and validation are presented in Appendix A. As expected, the QC productivity increases with increasing number of Lift-AGVs. From the status report the cause becomes clear, the QC spends less time waiting for Lift-AGVs. However, the increase in productivity of QCs stagnates quickly. Indicating that the number of Lift-AGVs is no longer the bottleneck. Logically, the reverse effect is found for the Lift-AGV productivity. Whenever more Lift-AGVs are deployed, the productivity decreases. Finally, similar to the QC productivity, the waterside RMG productivity increases and stagnates quickly whenever more Lift-AGVs are deployed. Interesting is that the buffer in between the RMG crane and the Lift-AGV is never full. This can be concluded since the percentage of "Idle TP Full" and the percentage of "Waiting and interchanging at ASC" are (almost) zero.

2.4. Concluding remarks

At the waterside of a container terminal are three different machine sets: YCs, HTVs and QCs. Each of the machine sets is responsible for a specific move within an order. Each container has its own specific order. The majority of orders are loading and discharging orders. Since the QC is the most critical machine set with regards to the turnaround time of deep-sea vessels, the QCs are the initiators of the planning on the container terminal. QCs determine the sequence of discharging a vessel based on the weight. During loading the characteristics of the containers as well as the workload at the stack modules is considered upon determining the load sequence. For loading orders, the complete planning for each machine set is immediately scheduled upon the creation of the order. Discharging orders are more flexible, so their planning is updated gradually.

Since there is no coordination among the different machine sets and the machines within a machine set, moves at the stacking crane could overlap. Once this is the case, the YC adjusts the planning to a new schedule. Once the schedule is complete, the dispatching process of the YC begins. A move is assigned to a YC based on various parameters which are all converted to a single score. The HTVs are subsequently assigned to moves based on a similar scoring mechanism. Generally, containers are handled in the sequence specified by the QC. However, sometimes out of sequence containers can also be handled.

For discharge orders the planning updates gradually since the destination of a container is not yet known in advance. This is because the stack module is only determined once the container is loaded on the HTV. The slot within the stack module is determined once a stack crane is assigned to the container. Both the stack module and the slot within the stack module are determined based on a scoring mechanism. A partial overview of the planning and scheduling procedure is shown in Figure 2.14.

Based on the results obtained with the benchmark terminal it was found that 4 Lift-AGVs per QC is efficient. Therefore, most experiments within this report will be based on terminals with 4 Lift-AGVs per QC.

3

Literature study

In this chapter we present the literature published on integrated scheduling of container terminal operations. Supported by general literature published on integrated scheduling. First, a brief overview of other research in the field of container terminals is given to establish a framework and to put this research into perspective. Subsequently, three scheduling techniques are presented which could be used for integrated multi-stage scheduling. These are Hybrid Flow Shop (HFS), Model Predictive Scheduling (MPS) and Markov chains. Each of these techniques are explained first in a broader sense and later converge towards container terminals specifically. Thereafter, a brief reflection of the applicability of the scheduling technique on the problem at hand is presented. Next, different solution techniques are presented which could be used to solve the scheduling problems. Finally, we conclude with a summary of the available scheduling and solution techniques and the most promising scheduling and solution technique will be selected.

3.1. Literature review

In this section we highlight the relation of this research with other research topics regarding container terminals. As was indicated in Section 2.1.3, two scheduling problems precede the scheduling problem of individual containers. These scheduling problems were berth allocation and bay scheduling. The solutions to these scheduling problems define the starting point of the scheduling problem considered in this research [23–26]. However, since peak simulations (Section 1.2) are considered and no actual ships are being served, within the simulation program of TBA, these scheduling problems are ignored in the simulation program. Remember that a QC receives a bay containing a certain number of containers which it should handle based on the specified scenario, not based on an arriving ship. Since the scenario is determined prior to the start of the simulation, berth allocation and bay scheduling are not required.

Another problem which should be solved prior to the individual container scheduling problem is the problem of determining the yard location of containers [15]. This problem is addressed in [39–41] all using different assumptions. Contrary to berth allocation and bay scheduling, yard allocation is considered in the simulation model. Individual container schedules and yard allocation are closely related. Therefore, the quality of the yard allocation solution partially determines the performance of the individual container handling schedule. This is why certain researchers integrated yard allocation with individual container scheduling [42].

Research towards constructing schedules for individual containers can be divided into two different categories. The first is category is optimization. Within this category, research is dedicated towards finding optimal solutions for the individual container scheduling problem. This is also the category of this research. The second category concerns the construction of simulation models for container terminals. These simulation models are used to evaluate the performance of different policies and strategies. TBA already possesses a well designed simulation model. Therefore, the focus of this research is not the construction of a simulation model. Interested readers are referred to [43, 44] for more information on simulation models for container terminals.

Another problem present on container terminals is routing [15]. Routing considers the problem of

determining the route of HTVs on the container terminal. Routing is often used to avoid congestions and collisions between different automated HTVs while guaranteeing short driving times. Routing involves path planning, area claiming, energy consumption and prioritization. Typically, routing problems are solved after scheduling and dispatching problems are solved. Since this research considers both scheduling and dispatching, routing is the first encountered problem after the individual container schedule is constructed. Routing problems are frequently studied and interested readers are referred to [18, 45, 46]. However, since scheduling, dispatching and routing are closely related; some researchers aim to integrate these functions [47, 48].

3.2. Hybrid Flow Shop Problems

The problem of assigning jobs to different machines is widely studied since it is relevant to many different industries. A special case of this problem, which shows promising similarities with the problem of this research, is the Hybrid Flow Shop (HFS) problem. A HFS problem is a combination of the parallel machine scheduling problem and the ordinary flow shop scheduling problem. The former assigns jobs to machines whereas the latter determines the handling sequence of the different jobs [49]. HFS problems are problems where n different jobs should be processed in a series of m stages while optimizing an objective function [49, 50]. A job contains multiple operations which should be performed on the different stages. Problems classify as HFS problems once the following characteristics apply [19, 50]:

1. The number of processing stages m is at least 2.
2. Each stage k has $M^k \geq 1$ machines in parallel and at least one stage has $M^k > 1$.
3. All jobs follow the same production flow through the stages. However, a job is allowed to skip one or more stages.
4. Each job j requires a processing time p_{jk} on stage k.

Within the standard form of the HFS problem, each job is available from the start and can be processed on any of the different parallel machines within a stage. Furthermore, each machine is equal and has the same deterministic processing time which is known in advance. Moreover, machines can only serve one job at a time. Additionally, sequence dependent setup times are negligible, buffers between stages are unlimited and preemption of operations is not allowed. However the standard form of the HFS problem is not always applicable, the basis is the same for many different industries. Usually, only a few aspects of the basis problem need to be modified to represent the problem at hand. A visualization of a HFS problem is shown in Figure 3.1.

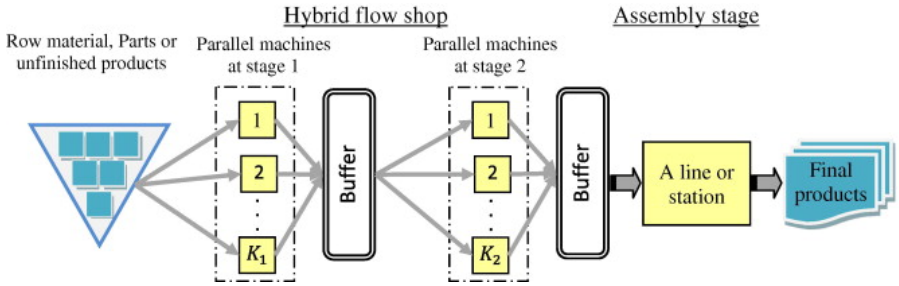


Figure 3.1: Schematic illustration of a classical 2-stage HFS [51].

To distinguish between different scheduling problems, a classification system was developed which is described with the triplet $\alpha|\beta|\gamma$ [49, 50, 52]. α represents the structure of the shop and is composed of four parameters $\alpha_1, \alpha_2, \alpha_3$ and α_4. α_1 describes the configuration of the shop, FH stands for HFS [50]. α_2 gives the number of stages in the shop. For flow shops, $\alpha_2 = 2 \ldots k$. α_3 and α_4 give information about the machines. α_3 represents the type of machines. The base case is that each machine on a stage has the same processing speed for each job. This is called identical machines in parallel, indicated with P. More advanced is the case in which there are multiple machines in parallel on a stage with different processing speeds, independent of the job. This is indicated with the letter Q. An even more

generic version is called unrelated parallel machines which is indicated with R. This case states that the processing speed of an operation at a stage is dependent both on the job and the machine at which it is being processed. In other words, within one stage each job could have a different processing time which is also dependent on the machine at which it is being handled [49, 53]. The final possibility is $\alpha_3 = \emptyset$ which means that there is only one machine. Aforementioned, α_3 is used in combination with α_4 in the following notation: $(\alpha_3\alpha_4)^k$. This means that there are α_4 machines of type α_3 at stage k. β, the second letter of the triplet, describes additional constraints and features of the HFS. Common modifications to the standard HFS problem are given below [49, 50]. The letter j is used to indicate jobs, k to indicate the stage.

- Individual jobs have individual release dates, indicated by r_j;
- Jobs must be processed in the same order on every stage, indicated by $prmu$;
- Certain jobs must precede other jobs, indicated by $prec$;
- Certain jobs can only be performed on certain machines; called machine eligibility, indicated by M_j;
- The setup time between different jobs is dependent on the job preceding it. The setup time is sequence dependent indicated by S_{sd};
- Preemption of jobs is allowed, indicated by $prmp$
- There is a limited buffer size between different stages. This means that jobs cannot be released by a machine until the upstream buffer opens up, indicated by $block$;
- Jobs are allowed to or must be handled multiple times on the same stage, indicated by $recrc$;
- Jobs cannot wait between successive stages meaning that a First In First Out strategy is used, indicated by $no - wait$;
- All processing times are equal, indicated by $p_j = p$;
- Machines have the possibility of breaking down or requiring maintenance, indicated by $unavail$;
- Operations could require processing on multiple machines simultaneously, indicated by $size_{jk}$.

Finally, the letter γ is used to indicate which objective function is being used. The completion time of a job is given by C_j, the flowtime by $F_j = C_j - r_j$, the lateness by $L_j = C_j - d_j$, the tardiness by $max(C_j - d_j, 0)$ and the earliness by $max(d_j - C_j, 0)$. In these equations d_j represents the due date of job j. Besides different abilities and constraints on the model, there are also multiple common objectives [50]:

- Maximum/Total/Average (weighted) completion time, indicated by $C_{max}/\bar{C}$;
- Maximum/Total/Average (weighted) flow time, indicated by $F_{max}/\bar{F}$;
- Maximum/Total/Average (weighted) lateness, indicated by $L_{max}/\bar{L}$;
- Maximum/Total/Average (weighted) tardiness, indicated by $T_{max}/\bar{T}$;
- Maximum/Total/Average (weighted) earliness, indicated by $E_{max}/\bar{E}$;
- Maximum/Total/Average (weighted) number of late jobs, indicated by $U_{max}/\bar{U}$;

Once a weighted number should be used, a w is used in the power. For instance, $\bar{C}^w$ indicates that the weighted total/average completion time is used. With this classification system the basic HFS problem is classified as $FHm, ((PM^{(k)})_{k=1}^{m}||C_{max}$. From left to right these mean: a HFS with m stages, M identical parallel machines on stages $k = 1$ to m, no additional features and the maximum completion is used as an objective.

3.2.1. Hybrid flow shops with uncertainties

The majority of research into HFS problems is dedicated to deterministic problems. This means that the processing times, set-up times, number of available machines, buffer sizes etc. are all fixed numbers which are known in advance. More advanced, is the use of stochastic distributions to all of these values. For instance, processing times are no longer equal but, follow a user-defined random distribution [54, 55]. This is done to better resemble real-world situations [53, 56–58].
To include uncertain parameters in the previously defined classification system, an extra option is added to the features part (β) of the classification triplet. Once a parameter is uncertain, it will be specified as $parameter \sim PF$, where PF is the probability function. Another option to specify uncertainty, is by fuzzy numbers. Once this is the case, the notation $Fuz(parameter)$ is used.
The techniques that are used to solve HFS problems with uncertainties are slightly different than for deterministic HFS problems. Basically there are three different approaches to solve HFS problems. These techniques are exact solutions, dispatching rules and metaheuristics. More on this topic can be found in Section 3.5. The distribution of used solution techniques for both deterministic and stochastic approaches can be seen in Figure 3.2. Interesting is that 8% of the papers which considered deterministic HFS problems used a genetic algorithm whereas this was 24% for stochastic HFS problems.

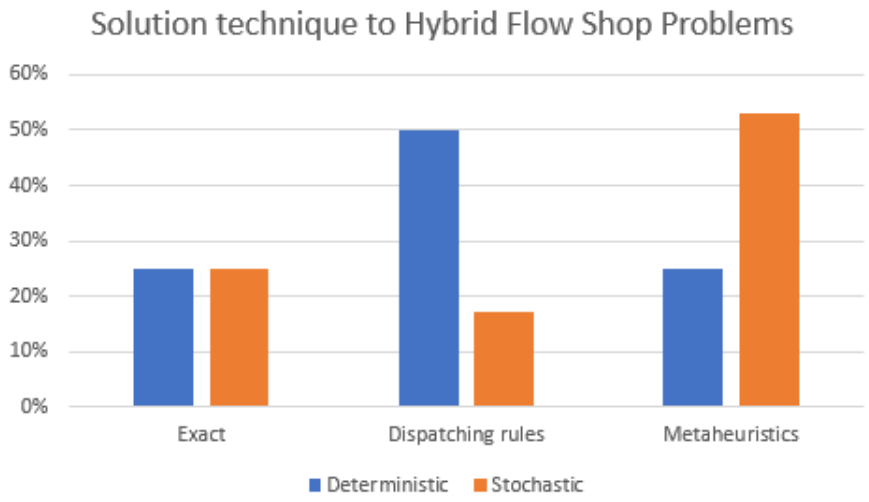


Figure 3.2: Distribution of used solution techniques to deterministic [50] and stochastic [54] HFS problems.

3.2.2. Hybrid flow shops applied to integrated container terminal scheduling

The similarity between the container terminal scheduling problem and a HFS problem has not gone unnoticed by other researchers. Therefore, in this section we compare multiple previous efforts towards integrated container terminal scheduling approaches applying HFS models. However not all research explicitly mentions hybrid/flexible flowshops, most mathematical models can be categorized as such. Therefore, the classification described in [49, 50, 52] will be used to describe the different models used to solve the integrated container terminal scheduling problem. Besides that, the used solution technique will also be given since it is a major part of the research. The results of this analysis can be found in Table 3.1.

Article		α	β	γ	Cont	QC	HTV	YC	Solution technique
Chen	[19, 59]	$FH3, ((RM)^k)_{k=1}^3$	$M_j, S_{sd}, block$	C_{max}	500	5	30	20	Tabu Search, Constraint programming
Lau	[60]	$FH3, ((RM)^k)_{k=1}^3$	$prec, S_{sd}, M_j$	$\bar{C}^w$	16	2	4	2	Multi Layer Genetic Algorithm
Bish	[2]	$FH2, ((PM)^k)_{k=1}^2$	$M_j, S_{sd}, block$	$\bar{C}^w$	2500	4	40	-	Transshipment Based Heuristic
Xin	[61, 62]	$FH3, ((RM)^k)_{k=1}^3$	$block$	C_{max}	40	5	10	8	Exact, Variable Neighbourhood Search
Lu	[56]	$FH2, ((PM)^k)_{k=1}^2$	$p_{ij} \sim N(\mu, \sigma), S_{sd}, block$	T_{max}	30	-	4	1	Particle Swarm Optimization
Kim	[63]	$FH2, ((PM)^1, (RM)^k)_{k=2}^3$	S_{sd}	$\bar{L} + \bar{C}$	100	2	7	5	Dispatching method
He	[64]	$FH3, ((RM)^k)_{k=1,3}, (PM)^2$	$prec, S_{sd}$	$\bar{L}^w + energy$	500	10	50	20	Genetic Algorithm
Kaveshgar	[65]	$FH2, ((RM)^k)_{k=1}^2$	$block, prec$	C_{max}	1050	2	15	-	Genetic Algorithm
Cao	[35]	$FH2, ((PM)^k)_{k=2}^3$	$S_{sd}, M_j, block$	C_{max}	500	-	60	40	MIP+Benders Cut
Cao	[66]	$FH2, ((PM)^k)_{k=2}^3$	$S_{sd}, M_j, block$	C_{max}	15	1	4	-	Genetic Algorithm + Heuristic
Kizilay	[67]	$FH2, ((PM)^k)_{k=1,3}$	$prec, S_{sd}, M_j$	C_{max}	550	4	∞	6	MIP, Constraint programming

Table 3.1: Classification scheme of other papers considering integrated container terminal scheduling. Extended with the maximum managed problem size (# of containers, QCs, HTVs and YCs) and the applied solution technique.

As can be seen in Table 3.1, the variation of HFS approaches is quite large. This holds for both the structure of the shop as for the features. The structure of the shop is either a two-stage or a three-stage approach. Additionally, the type of machines is different; either identical machines in parallel or unrelated machines in parallel are used in the flow shop.

The additional features added to the flow shop are different for most researches. Most research considered at least sequence dependent set-up times. The research that did not use unrelated parallel machines used the set-up times to compensate for this.
Another common feature is machine eligibility. This feature is used to represent the fact that containers are positioned within a stack or bay of a ship which can only be handled by a limited number of cranes. Moreover, precedence is commonly used. However, most research only considers one ship which first requires loading and afterwards unloading or in reverse order. This could be classified as precedence but it does not account for precedence among different containers.
Finally, blocking is used multiple times to ensure that a HTV cannot proceed to its next operation without being discharged of its current container.

The objective function is often either the maximum or total completion time. Only Lu et al. [56] and He et al. [64] used a different objective function. The objective function of [64] is particularly special since it also tries to minimize energy consumption of the machines operating on the container terminal.

Regarding the problem size, the research of [19, 59, 64] most closely represents real terminal sizes. Other research is either too small with regards to the number of containers or the number of machines. The minus signs indicate that a certain stage is not considered in that research paper. However, the problem size gives an indication towards the size of the problem, it does not necessarily mean that the problem is also solved within reasonable time. For instance, in [61] the maximum sized problem required more than 10 hours to solve. These computational times are unacceptable when applied to an operational terminal.

Interesting is that many research applied a different solution technique. Only the genetic algorithm is used multiple times. However, even among those there is great variation.

Similarities and differences of hybrid flow shops in container terminal scheduling

In this section, the similarities and differences between the different efforts of other researchers to apply HFS models to integrated container terminal scheduling are explored. The first similarity between all previous efforts is the definition of the jobs, the machines and the operations.

1. **Jobs:** correspond to orders; since an order is composed of multiple operations which require operations sequentially by multiple machines.
2. **Machines:** correspond to the QCs, HTVs and YCs. Each order should be handled by each of these machines in a sequential order. However, the sequence through the shop is dependent on the order type; either loading or unloading.
3. **Operations:** Correspond to the moves. Each order is composed of several moves: a transfer between the quay and the ship, a transfer between the stack transfer point and the quay and a transfer between the stack and the stack transfer point. These moves are respectively handled by the QC, the HTV and the YC.

A few common assumptions that were made in almost every research to both represent reality and to reduce the problem size are [19]:

1. Loading operations on a ship can only start after all unloading operations are completed.
2. Within the storage yard, inbound and outbound containers are not mixed within one block. Which means that a YC only performs either pick-up or retrieval moves.
3. All vehicles on the terminal can only handle one container at a time.
4. Operation and travel times of vehicles are deterministic and independent of the load.
5. All routes are considered available at all times meaning that possible congestions are ignored.
6. Container transfer time between different machines is included within the processing time.
7. There is no buffer space between the different stages.

Together with the assumptions, the integrated container terminal scheduling problem could formally be defined as a HFS problem. Four modifications are required to modify the classical HFS problem to be applicable to a container terminal [19]:

1. **Unrelated parallel machines:** Chen et al. [19] uses a different definition than the one found in [49, 50, 53]. Chen et al. [19] describes machine eligibility with unrelated parallel machines. Meaning that certain operations can only be handled by certain machines. This occurs at the QCs and the YCs since the location of the container defines the equipment that will handle it. However, the common description of unrelated parallel machines is also applicable since HTVs have different processing speeds dependent on their current position;
2. **Job precedence constraints:** This additional feature is required to ensure that containers on deck are unloaded first before containers in the hold. The reverse order should be applied for loading operations;
3. **Sequence dependent setup times:** are used to represent empty trips of HTVs. Ideally the HTV alternates between loading and unloading moves since the empty travel time is reduced.
4. **Blocking:** Each machine has a finite buffer size meaning that once the buffer is full, a machine cannot release a container. Therefore, the machine cannot proceed to the next order and has to wait. Most research did not consider any buffer at all but, simply considered buffer sizes of zero or infinity.

The aim of the problem is then to determine the sequence and the starting times of machine moves for each specific container. With the overall goal to minimize the ship turnaround time. In other words, find the optimal sequence of moves for which the QC is finished as early as possible. Alternatively, also the reduction of delays on QC operations could be used as an objective.

The major part of every research mentioned above is not dedicated to describing a container terminal system as a HFS problem. The major part of the research is often dedicated either to additional features [59, 61, 62, 64, 68, 69] or to the solution technique [2, 19, 35, 56, 59, 60, 63–66]. The aim of the solution technique is to reduce the computational time. Benchmarks that are used to compare different solution techniques are: the size of the problem, the CPU-time and the optimality gap. An optimality gap is usually present whenever an approximate solution technique is used. Most researchers claim their uniqueness by either adding special features (β) or by developing a new solution technique which has a smaller optimality gap or is applicable to larger sized problems.
No different is the research performed by Lu et al. [56] who used a stochastic HFS problem to model the behaviour of a container terminal. However, since this research is so unique (it is the only found stochastic model) it deserves some special attention. Three different uncertainties are distinguished by Lu et al. [56] of which only the second is implemented in the model.

1. **External uncertainties:** these are uncertainties that originate from external sources. These are; the quantity and arrival time of vessels, the bay size and the stowage plan.

2. **Uncertainties within normal operations:** these are the uncertainties that originate from performance fluctuation of equipment and human operators.

3. **Discrete abnormities:** these are the uncertainties that occur rarely. Such as, accidents and machine breakdowns.

The constructed mathematical model is somewhat less extensive compared to models developed by other researchers, such as [19]. Furthermore, even though the problem is small-scale, it requires an approximation method to be solvable within reasonable time. A Particle Swarm Optimization based algorithm was used as an approximation method and that proved to work well [56].
Other special research is performed by Xin et al. [68] and He et al. [64] who have integrated energy consumption into their scheduling algorithm. However, both of them used a different approach. Xin et al. used a multi-stage approach. On the first stage, a schedule was created. Subsequently, operations are smeared out over a longer period to reduce accelerations and thus to use equipment in a more environmentally friendly way. In [64] the energy consumption is directly integrated in the HFS problem. This is done by reducing travel distances of the different equipment in addition to optimizing QC operations.
Xin et al. continued with their previously developed model in [62]. However, the model is now extended with a layer to avoid collisions of AGVs.
In the most extended version of Xin et al. [61] 5 QCs, 10 AGVs and 8 ASCs are considered. However, this research did not centrally focus on scheduling. It is more dedicated towards the integration of discrete-event dynamics (the schedule) and continuous-time dynamics (energy consumption and collision avoidance). With regards to the scheduling, Xin et al. recommends to investigate how the computational time can be reduced. Furthermore, Xin et al. recommended to consider simultaneous loading and unloading of containers. Xin et al. focused on the case where there are merely unloading containers and each QC serves 8 containers. Besides that, an extra degree of freedom was used since it was assumed that each container can be directly handled; no precedence relations exist. Therefore, the research is not similar to the problem of this research. However, the research also considers a receding horizon, which is used to reschedule events based on new information. A receding horizon means that a time window is used. Within that time window, the optimization is performed. As time progresses, the time window shifts along the time line as well. This is an interesting feature since the complete horizon of moves is generally not known in advance. Besides that, once the complete horizon is used, the problem would become huge. Huge problems are hard to solve because of the computational effort.
Moreover, the research performed by Kizilay et al. [67] is rather special. This is because infinite HTVs were considered in their study. It was chosen to use an infinite number of HTVs because a survey among small ports within Turkey showed that there are always sufficient HTVs. Another novelty is that bay scheduling is included. Furthermore, this research compares MIP with CP. The final remarkable research is performed by Chen et al. [59]. Within this research the scheduling and dispatching problem is extended and integrated with the routing problem. This is achieved by adding multiple layers to the problem which are subsequently solved.

An interesting addition to the container terminal scheduling problem is proposed by Jung et al. [70], who proposed that multiple YCs might be working within the same block. Since YCs are typically overhead cranes; they cannot pass each other. This means that YCs should coordinate their operations to reach every location in the yard. Jung et al. developed a model that ensures optimal coordination between the different YCs.

3.2.3. Reflection on the problem

To solve the problem of this research, a combination of the features: M_j, S_{sd}, $blocking$ and $prec$ is required. This is not earlier performed, to the best of my knowledge, in any other research considering container terminal scheduling. On top of that, the most recent review paper (2010) regarding HFS problems, indicates that there is no research that considers all of the above features simultaneously in any HFS problem.
Furthermore, the blocking constraint should be adjusted once Lift-AGVs are used since up to a certain number of containers there is no blocking. This is because the racks have some limited capacity to hold containers and thus acts as a buffer. Nguyen et al. [71] consider Lift-AGV dispatching with a

limited buffer capacity. Within this research an integrated scheduling approach using limited buffers is suggested by the authors but, has not yet been followed up upon.
Another novel required feature compared to other integrated container terminal scheduling research is that loading and unloading moves are considered simultaneously. Currently, research has focused on handling of vessels that require either unloading or loading and in some cases these two operations sequentially. This research considers the case where loading and unloading jobs can be performed simultaneously by different QCs. This feature makes the HFS bi-directional which is against the nature of a flow shop. To the best of my knowledge this feature was not earlier considered in HFSs. For normal flow shops this additional feature is not so special since the direction of the flow shop can be reversed whenever required. However, within a HFS different jobs require different flows using the same machines and buffers.
Additionally, there are twinlift moves on modern container terminals that require simultaneous handling by the QC and the HTV. In other words, two jobs require simultaneous processing by a single machine on specific stages. This additional feature was not earlier described in other research considering integrated terminal scheduling.

3.3. Model Predictive Scheduling

Besides the commonly used HFS approach to scheduling problems, there are other approaches. One of them is Model Predictive Scheduling (MPS), which is applied to integrated container terminal scheduling in [72]. MPS uses a radically different approach compared to HFSs. The scheduling problem is perceived more as a control problem that requires optimal input. This makes this technique ideally suitable for operations in a receding horizon way. As was said before, receding horizons are often used for container terminal scheduling since it reduces the problem size and the complete horizon is unknown from the start. Furthermore, since MPS is predictive, predictions of future behaviour can be used to reschedule routes of container handling equipment and thus optimize the system's performance [72].
Not required for MPS, but used in [72] are Switching Max-Plus Linear (SMPL) techniques to model the system [73]. SMPL systems are used since they are capable of analyzing the evolution of the system's states. Max-plus eigenvalues and eigenvectors play an important role towards finding bottlenecks in the scheduling process [73].
However, the approach of MPS is radically different from HFS scheduling there are also many similarities. First, the max-plus model is recasted into a MIP problem since there are many efficient solvers available. Second, additional constraints are added to the MIP problem to represent the special features on a container terminal. These are: the number of containers on a vehicle is constrained, the sequence of container operations at the QC is known in advance and blocking constraints are introduced to create buffers. On top of that, the objective function is, just as in most HFS problems, the minimization of the makespan at the QCs.
The method proposed in [72], has been positively received by the authors themselves. However, the authors comment on the computational time required to solve the problem since it is too excessive. The authors suggest that the computational time could be reduced by a priori elimination of certain combinations of decision variables leading to infeasible solutions. This would greatly reduce the solution space. Interesting is that nobody has picked up on the proposal for future research and that articles describing either MPS and SMPL algebra are only published by the same TU Delft research group DCSC.

3.3.1. Greedy scheduling algorithm

MPS is compared with a greedy scheduling algorithm which is doomed from the start to be less efficient than algorithms that seek optimality. However, for the sake of completeness the greedy algorithm is briefly described here as well. The central idea of a greedy scheduling algorithm is that only one container is scheduled at a time. This is also why the schedule can never be optimal. The steps of the greedy algorithm are as follows [72, 74]:

1. Find a QC with the lowest ready time from the set of QCs which still have unscheduled cycles.
2. Find the corresponding YC to the unscheduled cycle.
3. Find the set of vehicles which can arrive on time at the loading crane, either the QC or the YC, depending on the cycle type.

4. If this set is not empty; select the vehicle which will arrive closest to the end time of the cycle.
5. Else; select the vehicle that is able to arrive the earliest at the loading crane.
6. Return to step one.

Because only one container is considered at once, each container is scheduled optimally in a recursive way. However, this does not guarantee an overall optimal schedule since an optimal schedule for one container could drastically reduce the optimal schedule of another container. This effect is disregarded with a greedy algorithm.

3.3.2. Reflection on the problem

MPS supported by SMPL modelling has proven to be a valuable scheduling technique. However, since this technique is not commonly used, the chances that other researchers will continue with future research are limited. Since HFSs are widely applied, the chances that other researchers will continue with the to be performed research are higher. Therefore, the scientific isolation of MPS supported by SMPL modelling is the major reason for not proceeding with this research. The aim is to develop a model which can easily be extended and modified to container terminal's individual desires. This requires broad scientific support which is not the case for MPS supported by SMPL modelling.

3.4. Markov chains

Yet another technique towards scheduling is the usage of Markov chains. A system is perceived as a system of queues. Queues are connected to each other and together they form the container terminal. A container enters the system via one queue and subsequently transfers along different queues until it is finally removed from the system [75, 76]. The machines on the container terminal are modelled as resources which determine the processing rate of the queues. By assigning more equipment to a queue, the queuing process accelerates. The allocation of resources to queues is the optimization problem and can be based on multiple strategies. Furthermore, queues have finite sizes which means that not an infinite number of containers is allowed to be moved to a certain queue. Two types of queues are distinguished by Alessandri et al. [76], storage queues and handshake queues. The former is used to describe the movement of containers to a new area on the container terminal. The latter is used to describe an interchange between different storage queues. A simple queuing system that illustrates the position of storage and handshake queues is shown in Figure 3.3.

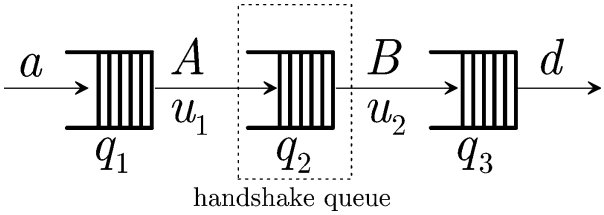


Figure 3.3: A visualization of two storage queues connected by a handshake queue [76]. a indicates arrival, d is used for departure, A and B are resources allocated to serve queues q_1 and q_2 respectively with processing rates u_1 and u_2 respectively.

Ideally, the handshake queue remains empty which indicates that processing rate $u_2 \geq u_1$. Once q_2 fills up either too many resources are allocated to A, which means a waste. Or too few resources are allocated to B, which indicates a delay. Based on the concept of Figure 3.3, the entire container terminal is constructed (see Figure 3.4).

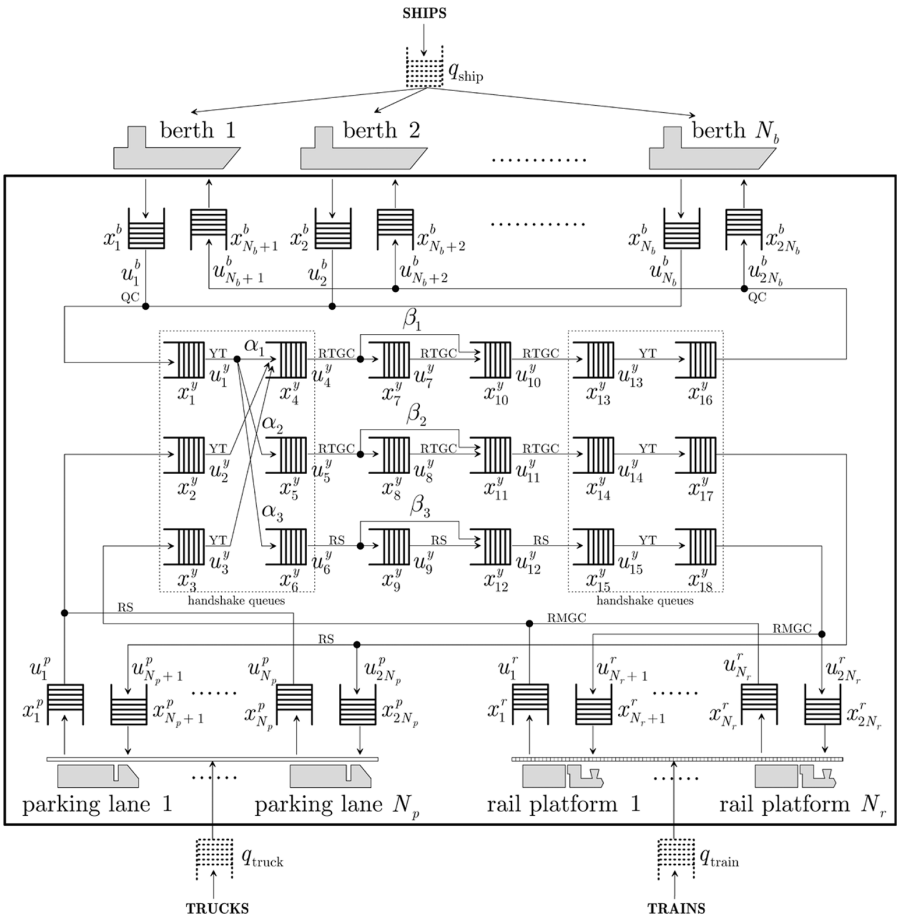


Figure 3.4: Queuing model of a complete container terminal [76].

By the time that the container terminal is described with queues. The constraints need to be added to guarantee that the queuing model represents reality. These constraints are used to constrain working rates, to ensure balance of input and output, to limit the maximum number of equipment assigned to a queue and to specify the maximum number of containers in a queue. Additionally, an objective function should be specified. Commonly, the objective function is minimizing the ship turnaround time [75, 76].

With the model, the constraints and the objective function defined, the aim is to optimally allocate resources to different queues. The first approach is to assign the most resources to the longest queues. Another approach is to assign the most resources to the queues which have containers in them for the longest period. Implicitly this strategy leads to serving queues based on a First in First out basis. However, different carriers could be given a different weight factor to promote, for instance, quick handling of deep-sea vessels over trucks [76]. Both solution techniques could be considered as heuristics since they allocate resources based on dispatching rules. These dispatching rules were pre-defined with knowledge of the structure and the characteristics of the problem at hand [77, 78].

3.4.1. Reflection on the problem

The disadvantage of using Markov chains to address scheduling problems, is that it is more focused on container flows instead of individual container schedules. This makes the Markov chain approach more suitable for strategical decisions compared to operational decisions. However, this information is valuable, it is not the aim of this research. Therefore, the information will not be used. Once the Markov chains should be applied to operational levels, the number of queues would hugely increase which also increases the computational effort. Besides that, it is questioned whether the proposed solution techniques and the approach would lead to optimal results once operational scheduling problems are

considered.

3.5. Solution techniques

In this section more attention is paid towards the solution techniques used to solve scheduling problems. In particular, solution techniques applicable to HFS problems will be discussed. First, an exact method will be presented followed by a tailored heuristic strategy. However, the main part of this section will be dedicated towards metaheuristics. This is because metaheuristics are suitable alternatives to exact solutions whenever the problem is large and complex while remaining generic.

3.5.1. Problem complexity

The best suitable solution technique depends on both the complexity and the size of the HFS problem. With regards to complexity the HFS problem is NP-hard even if there is only one stage with with multiple parallel machines [79]. NP-hard problems are problems which cannot be solved in polynomial time as a function of the problem size [53]. Furthermore, NP-hard problems do not necessarily need to be verifiable in polynomial time either [80]. An Euler diagram of problem complexities is shown in Figure 3.5. This diagram shows that NP-hard problems are the most complex problems [81]. Since P versus NP is not proven, there exist problems which can be verified in polynomial time but, cannot be solved in polynomial time. This is the (current) definition of a NP-problem [82]. However, since NP-hard problems do not necessarily possess polynomial time verifying techniques they are still excluded from P even if P=NP can be proven. A formal classification of NP-hard problems is: "A decision problem H is NP-hard when for every problem L in NP, there is a polynomial-time reduction from L to H." [82].

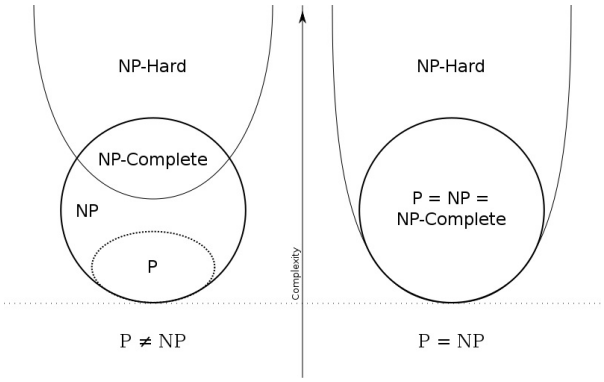


Figure 3.5: Euler diagram for sets of P, NP, NP-complete and NP-hard problems. The left side is valid under the assumption that P≠NP, while the right side is valid under the assumption that P=NP [81].

However, NP or NP-hard problems are not necessarily "hard problems" [83]. This is also the case for the HFS problem [53]. Therefore, many larger HFS problems use (meta)heuristics to approximate the exact solution within reasonable time. However, if the problem is small enough, a branch and bound technique is typically used to arrive at an exact solution.
Tailored heuristics is a collective name for solution strategies that use the structure and the characteristics of a problem to quickly find an approximation to the solution [77, 78]. The most common heuristic strategy in HFS problems is the use of dispatching rules. The use of dispatching rules means that the assignment of jobs to equipment is based on simple rules such as, First in First out. An illustrative example of more advanced dispatching rules is presented in [84]. Another popular heuristic method is divide-and-conquer. This strategy relies on dividing the initial problem into multiple easier solvable sub-problems. Afterwards, the solutions to the sub-problems can be combined to an overall solution. A particular well working technique is the shifting bottleneck procedure which focuses on the bottleneck stage. The general idea is that by optimizing the operations at the bottleneck stage the overall flow shop is optimized as well [50, 85].
General metaheuristics could also be applied to HFS problems. Using metaheuristics means that a general concept to many different problems is applied to find a solution. However the concept is similar for different problems, the implementation could be different. Often metaheuristics require multiple itera-

tions and exploit randomness; hoping that it will lead to a global optimum quicker than by deterministic approaches. This is generally the case once efficient metaheuristics are applied to large and complex problems. The most efficient metaheuristic techniques for HFS problems are: Tabu search, Simulated annealing and Genetic Algorithms. Examples of Tabu Search metaheuristics applied to HFS problems are [19, 86, 87]. Simulated annealing was applied to a HFS problem with unrelated parallel machines in [88]. Moreover, simulated annealing was also used to solve HFS problems with unrelated parallel machines, sequence dependent setup times and transportation constraints in [89, 90]. Finally, also genetic algorithms are widely applied to the HFS problem and its different variations such as sequence dependent setup times [91], unrelated parallel machines and machine eligibility [92] and limited buffers [93]. In [94, 95], the performances of different (meta)heuristics applied to HFS problems are compared with each other. Generally, tailored heuristics work quicker than metaheuristics at the cost of losing genericity. This is because tailored heuristics use problem specific information to quicker find a solution [96]. An important remark towards both tailored heuristics and metaheuristics is that these methods approximate the exact solution in contrast to exact solution techniques that provably find the optimal solution.

3.5.2. Mixed Integer Programming

To solve HFS problems, first a mathematical model should be constructed. The mathematical model contains decision variables and fixed parameters. Typically, a Mixed Integer Programming (MIP) model is used to represent the HFS problem. The MIP uses integer decision variables to define the sequence of operations and to assign certain operations to certain machines. Usually, these integer decision variables are binary. Furthermore, the MIP uses non-integer variables to represent the starting times of different operations. As a function of decision variables and fixed parameters, the objective function and the constraints could be mathematically expressed. The number and type of constraints that are used depends on the specifics of the HFS problem (see β). Different techniques were developed to efficiently find a solution to the MIP model that satisfies all constraints while minimizing an objective value.

3.5.3. Constraint programming

A radically different approach to solve HFS problems is using Constraint Programming (CP) instead of MIP. CP combines features of logic programming, graph theory and artificial intelligence to solve combinatorial optimization problems such as, HFS problems. It is particularly effective since these problems are often described as constraint satisfaction problems [97]. Combinatorial optimization problems are described by certain variables which can attain certain values within a domain. The domain is bounded by either implicit or explicit constraints. Implicit constraints are constraints which constrain one variable as a function of other variables. Whereas explicit constraints are constraints that constrain a variable with external values. The aim of combinatorial problems is to find a combination of values assigned to the variables which satisfies all the constraints and optimizes an objective function. CP does not only check whether a combination of assigned values violates a constraint, it makes active use of these constraints to reduce the problem size [97]. A typical solution technique of CP is to construct a tree of attainable values for each variable. Initially, all values can be assigned to all variables. However, by subsequently adding constraints, the number of attainable values reduces. Sometimes, the solution method also requires to go back to retry to find a solution from a different angle. Similarly to ordinary HFS solution techniques, there are different heuristics available to determine the order in which constraints should be added to the tree [97]. Recently, CP proved to be a powerful and easy-to-use optimization tool especially suited for scheduling problems. This is because of the available data-types supported by CP languages. A decision variable could be specified as an interval, which means that it has a start and an end time. Besides that, variables could be sequences which define a sequence of operations. These variable types support a major part of a schedule [59]. Furthermore, these datatypes allow CP to model a problem at a higher level compared to MIP. CP combines multiple techniques to find solutions that satisfy all constraints and if specified, optimize an objective value. Among these techniques are backtracking, B&B and several forms of local search [98]. Moreover, CP is not restricted to linear models; it has access to a much wider range of models [98]. For container terminal scheduling, CP is rather new but, it has been successfully applied to other HFS problems [59, 67]. Moreover, hybrid approaches of CP and MIP were used to solve though and large scheduling problems [99].

3.5.4. Branch and bound

The first solution technique that is highlighted is Branch and Bound (B&B). B&B is an algorithm used to find exact solutions to discrete combinatorial problems. More specifically, B&B is a solution method designed to exactly solve linear programming models containing variables that are restricted to be integer. The algorithm was introduced first in [100] and currently it is the most widely used solution technique to NP-hard problems [101]. B&B could be visualized as a tree which starts from a root and is systematically branched from there. At each node where a branch is split into new branches the candidate solution is evaluated. The crux of the algorithm is that since the tree is systematically branched not all candidates need to be evaluated but, only a subset of branches. The branches are not further explored once the solution of the branch cannot yield any solutions better than the best one found so far by the algorithm. In other words, the complete search space of feasible solutions is split into smaller spaces. Each split adds a new branch to the problem. A branch is disregarded or "fathomed" once it can be proved that branching further will not lead to a better solution compared to the best found solution thus far. Bounding defines the process to determine the possibilities of obtaining a solution better than the current best solution [78]. An illustration of B&B is presented in Figure 3.6.

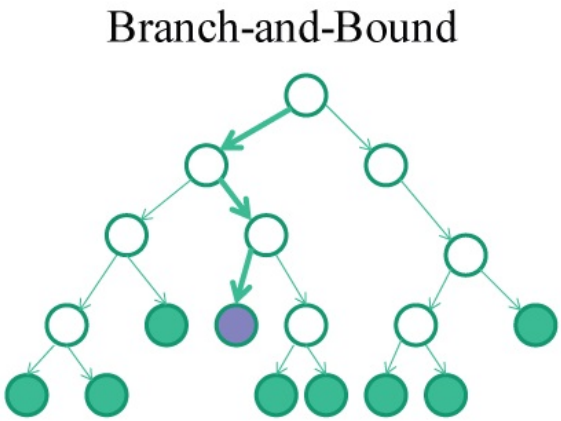


Figure 3.6: Illustration of a tree being explored by subsequently branching, bounding and fathoming [102].

3.5.5. Shifting bottleneck

The shifting bottleneck procedure is a heuristic specially designed to solve job shop problems. In particular flow shop problems are considered. The core thought is that once there are multiple jobs requiring handling by (multiple) machines, called resources, one resource acts as the bottleneck in the system. The heuristic is designed to use the bottleneck resource in the most efficient way. By shifting the bottleneck every time the bottleneck is resolved, the total makespan will approach the optimum. The bottleneck resolving and shifting process is an iterative process of which the steps are briefly discussed here [85, 103]:

1. Create a graph representing all jobs and required resources to establish the initial makespan. The makespan is determined by adding all the processing times required on a job. The job with the largest sum of processing times assuming no conflicts is used as the makespan.

2. Determine and resolve the bottleneck resource. The bottleneck resource is determined by considering the processing time, release time and due time of each job on each machine. The release time is found by adding all the processing times of the respective job preceding the operation on the considered machine. The due time is determined by subtracting the processing times of the jobs succeeding the job on the respective machine from the previously determined makespan. Now, for each resource; calculate the minimum lateness by finding the optimal sequence of jobs for instance with B&B. The resource with the largest minimum lateness is the bottleneck resource. At the bottleneck resource the optimal sequence of jobs is fixed and the minimum lateness is added to the makespan to determine the new makespan. At this moment the first

iteration is finished; the first bottleneck is identified and resolved. The solution to the bottleneck is now fixed by adding constraints for the next iteration.

3. Repeat the previous iteration. However, there is a new makespan and the constraints resulting from the previous iterations should be respected upon determining the release time and due time. Repeat iterations until all resources have been accounted for.

The steps listed above illustrate the iterative shifting process of the bottleneck resource. It should be noted that the shifting bottleneck procedure solves an optimization problem, commonly with B&B, multiple times. However, since the considered optimization problems on a single machine are much smaller than the overall optimization problem; the computation time is reduced. On the other hand, since the overall optimization problem is divided into smaller problems the shifting bottleneck heuristic cannot guarantee to find the optimum. [103].

To use the shifting bottleneck heuristic on problems where there are multiple parallel machines, the algorithm should be modified [85]:

1. Convert the problem to a graph;
2. Calculate lower bounds for each stage assuming only one machine.
3. Solve the machine assignment problem sequentially for the stage with the largest lower bound. It is proposed to use the reversibility of a scheduling problem [85].
4. Identify the critical paths on each stage.
5. Solve the machine assignment problem for the critical stage. If the schedule improves return to step 4.
6. If all stages have been considered or there are no improvements: stop the algorithm.

The advantage is that the computational time could now be expressed as $O(Im^2n^2)$, where I is the number of iterations, m the number of stages and n is the number of arcs connecting the machines [85]. Unfortunately, the shifting bottleneck heuristic is not suitable once blocking constraints are added to the problem. Therefore, the shifting bottleneck heuristic cannot be used on the problem considered in this research [19, 104].

3.5.6. Tabu search

Tabu search is a popular metaheuristic which has been widely used to solve combinatorial optimization problems. The core of tabu search is to employ a local search method. This means that from a given solution, other similar solutions are evaluated. Subsequently, the best one of the alternatives is selected to obtain the new solution. However, tabu search also allows worsening moves whenever there is no better alternative. This is to escape local minimums. Furthermore, prohibited moves are introduced (listed on the tabu list) to prevent the algorithm from moving back and forth between two solutions. Additionally, the tabu list stores solutions which turned out to be infeasible. This is to prevent the algorithm from considering an infeasible solution multiple times [105–107].
A local neighbourhood search defines a search strategy that iteratively moves from potential solution $\mathbf{x}$ to an improved solution $\mathbf{x}'$. This process is repeated until a stopping criterion is encountered. Commonly used stopping criteria are a score threshold or a maximum number of iterations. Unfortunately, local searches often get stuck in local minimums/maximums. This means that other areas in the search space remain unexplored. Therefore, tabu search carefully selects possible solutions added to the local search space $\mathbf{N}^*(\mathbf{x})$ to avoid this problem [78]. $\mathbf{N}^*(\mathbf{x})$ is created with the aid of memory structures. The memory structure is used to create the tabu list. The most simple tabu list contains a list of solutions visited in the previous n iterations. This means that a new solution cannot be one of other recently visited solutions. This is called short-term memory. Intermediate term memory is used to promote promising search areas by favouring solutions with certain features. Long term memory is used to promote diversity among solutions by favouring solutions with new features [108]. However, short-term memory is the core of tabu search, intermediate and long-term memory is required to efficiently solve harder problems [109]. Important is that the tabu list has a finite length which means that elements on the tabu list could also expire. This feature is implemented to prevent the algorithm from using much memory. Furthermore, finite tabu lists prevent the algorithm from entering unpromising search areas [78].

3.5.7. Simulated annealing

Simulated Annealing (SA) is a stochastic metaheuristic involving randomness. The metaheuristic is based on the annealing process of metals. It is particularly effective to locate the global minimum of large problems. The core thought is that the solution iteratively progresses towards the global optimum. Initially, the algorithm quickly explores the search space by accepting a large portion of moves, both improving and worsening. This allows the algorithm to escape local minimums. As time progresses, SA becomes more picky and starts favouring improving moves only [78, 110]. Typically, SA is only used on very complex problems on which other optimization algorithms have failed since it involves many function evaluations and is therefore inefficient [78, 110].

The basic working principle of SA is as follows [110]:

1. Select an initial temperature T and solution $\mathbf{x}$;
2. Randomly generate a new solution $\mathbf{y}$ in the neighbourhood of $\mathbf{x}$ and subsequently evaluate $f(\mathbf{y})$;
3. If $f(\mathbf{y}) < f(\mathbf{x})$, accept $f(\mathbf{y})$. Otherwise, compute the probability $P_{accept}(T)$ (Equation 3.1) of accepting the new solution $f(\mathbf{y})$. Use a random number generator to decide whether to accept or reject $f(\mathbf{y})$;
4. Reduce the temperature and return to step 2.

The probability to accept a worsening move is expressed with Equation 3.1. This chance gradually decreases towards zero with decreasing temperature, meaning that only improving moves are selected [111]. In other words, in the beginning SA is exploratory, whereas later SA becomes more investigative.

$$P_{accept}(T) = e^{\frac{f(\mathbf{x})-f(\mathbf{y})}{T}} \tag{3.1}$$

Commonly, the temperature is decreased gradually following a linear function. However, also more dynamic cooling or even re-heating processes might be used [112]. The probability to accept a worsening move is plotted with respect to temperature for several differences in objective value in Figure 3.7.

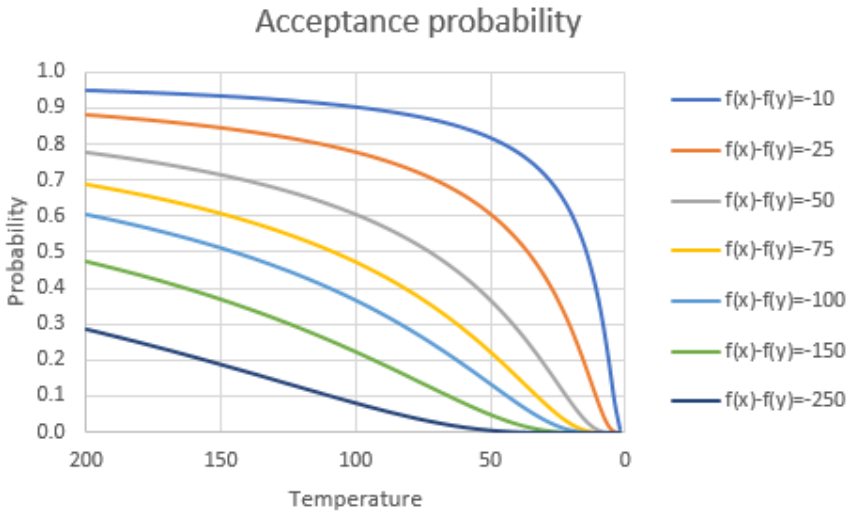


Figure 3.7: The probability to accept a worsening move with respect to temperature for several differences in objective value.

Generally, SA is only used for unconstrained and continuous problems [110]. However, there are techniques available to enable SA to solve constrained and discrete optimization problems. Constrained problems can be solved by SA by modifying the objective function and a reparation method to restore infeasible solutions [113]. The objective function is extended with the positive difference with each constraint. Therefore, whenever a solution is infeasible; the objective function increases. Subsequently,

the infeasible solution should be repaired. This is done by iterative modification of the variable which has the biggest negative effect on the objective value (most helpful when restoring feasibility) with exclusion of the variable that initiated the infeasible solution [96, 113]. To include discrete variables when using SA, a discrete variable is altered to any of its attainable values. Thereafter, a similar reparation algorithm can be used as for continuous constrained optimization [96]. Combinations of discrete and combinatorial problems could be solved in different ways. One way is to first use SA to assign values to the discrete variables. Subsequently, an ordinary continuous linear program remains which can be solved much quicker. However, a condition to this approach is that the remaining continuous linear program remains small since it requires to be solved many times [114].

3.5.8. Genetic algorithms

Another popular metaheuristic algorithm is a Genetic Algorithm (GA). A GA is inspired on evolution and natural selection described in [115]. Many genetic processes such as mutation, crossover and selection are used as operators in GAs [116].
Initially GA generates a set of candidate solutions as the initial population. The fitness (objective function) of each candidate solution is evaluated. Randomly a portion of the "fit" candidates is selected to reproduce. Reproduction involves crossover and mutation and forms a new generation of candidates. Similarly to tabu search and simulated annealing, GA terminates once a pre-specified fitness level is reached or once a certain number of generations is produced [78].

Selection

The process of selecting the most fit candidates from a generation is the first genetic operator used in GA. Numerically the selection process works in the following way [117]:

1. The fitness function of each individual is calculated and thereafter normalized;
2. Sort the population in descending fitness order;
3. The accumulated fitness values are calculated, this means the normalized fitness value of the respective individual and all its predecessors;
4. Draw a random number between 0 and 1;
5. Select the individual whose accumulated normalized fitness is closest but, smaller than the random number.

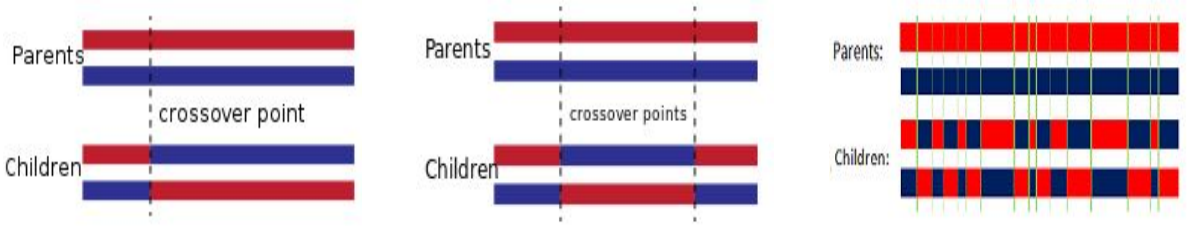
An illustrative example of a selection process is presented in Table 3.2.

Random number=0.38					
Candidate	**Fitness value**	**Normalized**	**Descending order**		**Accumulative**
1	1	0.1	**3**	0.4	0.4
2	3	0.3	**2**	0.3	0.7
3	4	0.4	**4**	0.2	0.9
4	2	0.2	**1**	0.1	1

Table 3.2: An illustration of the selection procedure. Since 0.38 is closest but still smaller than 0.4 that fitness value is selected. This fitness value corresponds to candidate **3** in this example.

Crossover

Crossover is the first genetic operator used to generate new solutions from the set of selected fit parent solutions. Crossover means that from multiple parents, two in this case, an equal number of children is generated with a mixture of the genome of both parents. A solution could be visualized as a string of numbers. One child starts with the string of one parent and the other child with the string of the other parent. Subsequently, a crossover point is randomly determined. At the crossover point the child solution string switches parent. This results in a new solution which is a combination of the parent solutions. The crossover process is visualized with Figure 3.8. As can be seen in Figure 3.8, there are multiple strategies to simulate crossover. With uniform crossover not the number of crossover points is specified but, the mixing ratio is specified. The algorithm subsequently determines the amount and location of each crossover point.



(a) Single-point crossover (b) Two-point crossover (c) Uniform crossover

Figure 3.8: Crossover process for binary problems[118].

Sometimes the generated children violate constraints since constraints are not considered in the crossover process. However, this could be repaired with a repair process such as mutation [119].

Mutation

The final genetic operator is mutation. Mutation is used to maintain genetic diversity upon creation of a new generation. This is an important feature of GA which allows solutions to escape local minimums. Based on a specified probability, the candidate's chromosome might change randomly independent of the parent solutions. Commonly, the mutation probability should be maintained low to prevent GA from disregarding the selection and crossover operators. Mutation has different variations based on the number type (binary, integers and floats) and the desired variation [120]. Mutation could also be used to repair solutions that no longer satisfy the constraints.

All genetic operators come with tuning parameters which can be used to tailor the GA to the considered problem. The tuning parameters for respectively selection, crossover and mutation are population size, crossover probability and mutation probability.

3.5.9. Particle Swarm Optimization

Particle swarm optimization is a metaheuristic strategy that is based on swarm intelligence. A set of particles is released into a search space. Subsequently, the particles start to move based on their objective value and the objective value of other particles. This behaviour simulates swarm behaviour where the group has more knowledge than the individual [121]. The particles do not use gradient information for their movement and only rely on simple kinematic equations often involving randomness. Similar to other algorithms, particle swarm optimization requires several tuning parameters and a stopping criterion.
An important tuning parameter is the swarm topology. This parameter defines the knowledge and communication abilities of the particles. In other words, it defines with which particles a certain particle can exchange information. This parameter is important since it helps the algorithm exchange information about local minimums. Therefore, particle topology determines the converging speed [122, 123].
A common explanation for the performance of particle swarm analysis is that the algorithm balances between exploratory and exploitative behaviour. This means that the algorithm both searches a large space (exploratory) while the search is directed to an optimum (exploitative). The tuning parameters are required to maintain a proper balance between these two properties [124].
Initially, particle swarm optimization was developed for unconstrained problems. However, modifications could be added to include the possibility of dealing with constraints. Furthermore, particle swarm optimization does only work for continuous problems. Once discrete problems should be solved, it often requires a relaxation and a rounding [125, 126].

3.5.10. Ant colony optimization

Yet another metaheuristic is ant colony optimization. This metaheuristic is inspired on the behaviour of an ant colony foraging for food [127]. A group of artificial ants is released into the solution space. Each time an ant finds a solution, it returns to the source leaving a pheromone trail. Other ants can sense the pheromone trail and are attracted by it. This leads to convergence, all ants will follow the same path. However, a pheromone trail also slowly evaporates. This means that distant paths have weaker pheromone trails compared to shorter paths. Therefore, distant paths will be less attractive. Originally, ant colony optimization was developed to solve travelling salesman problems. However, recently ant

colony optimizations find their application on different problems as well. Alaykyran et al. first applied ant colony optimization on HFS problems [128].

3.5.11. Comparison of different algorithms

To compare different solution techniques applicable to integrated container terminal scheduling, computational results which were published for the integrated container terminal scheduling problems are used. The same papers as in Table 3.1 are used to obtain computational results. Unfortunately, comparison is very difficult since each paper considered a different sized terminal and different terminal features. Furthermore, the stopping criterion could be different as well. This is especially the case with metaheuristics. Generally, metaheuristics quickly find a feasible solution and the more time they get, the better the solution will be. This is a major advantage compared to exact methods where there exists no such thing. In Table 3.3 the results are presented. It was aimed to select the computation experiment which corresponds the best to other research. Furthermore, three different sizes were considered; small medium and large. Whenever published, the optimality gap is also presented. Note that since the scheduling problem is NP-hard we cannot determine a polynomial function to interpolate/extrapolate computational time to establish a single benchmark terminal. Finally, research that used dispatching rules or considered less than three stages was excluded from this table since comparison makes no sense.

	Solution technique	CNT	QC	AGV	YC	Computation time	Optimality gap
Xin [61]	Exact	24	3	6	5	13541s	0%
		32	4	8	6	>10h	-
		40	5	10	8	>10h	-
Xin [62]	Variable Neighbourhood Search	24	3	6	5	600s	10.0%
		32	4	8	6	600s	≥12.7%
		40	5	10	8	600s	≥12.1%
Chen [19]	Tabu search	20	2	6	4	34s	0.15%
		50	4	10	6	131.7s	9.39%
		400	5	25	15	3263s	-
Chen [59]	Constraint programming	15	2	4	4	143s	-
		200	4	20	12	846s	-
		400	5	25	15	2954s	-
Lau [60]	Multi Layer Genetic algorithm	16	2	4	2	4s	-
He [64]	Genetic Algorithm	20	2	6	4	146s	4.25%
		200	4	12	8	621s	13.92%
		300	5	20	10	774s	14.74%

Table 3.3: Comparison of computational results of different solution techniques applied to integrated container terminal scheduling.

Table 3.3 and information from [50, 55] shows that a genetic algorithm is the most popular solution technique whenever dispatching rules are excluded. Surprisingly, [95] found that SA is superior to GA and Tabu Search for large scale HFS problems. Whereas it is the least used metaheuristic of the three when considering deterministic cases alone. Additionally, SA is preferred over GA since it is much easier to modify and tailor to the situation at hand. Besides that, SA requires less computational power per iteration compared to GA and Tabu Search. Therefore, SA is selected as the basis of the solution method that is used within this research.

3.6. Concluding remarks

As was described in Section 3.1, the problem considered within this research is strongly intertwined with other problems on container terminals. The relation of the problem at hand and other logistic problems on container terminals is illustrated with Figure 3.9.

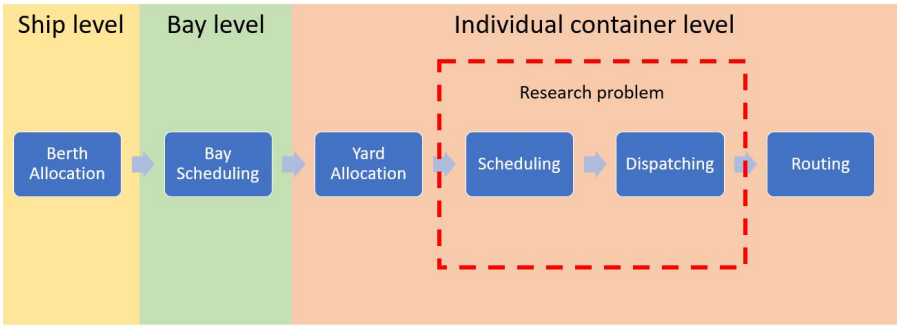


Figure 3.9: Relation of the research problem with other logistic problems encountered on container terminals.

Similar problems to the one considered in this paper have been studied multiple times. Researchers working on those similar problems used a mathematical model to describe the container terminal. Subsequently, a solution technique is selected to optimally schedule and dispatch vehicles according to an objective function. Often, the used objective function is the minimization of the makespan. Different researchers used different approaches to model the container terminal and to optimize the objective function. The most often selected modeling strategy is to describe the individual container scheduling problem as a HFS problem. However, research differed on additional features and the applied solution technique (see Table 3.1). Other techniques which are used to model the individual container scheduling problem are MPS supported by SMPL modelling and Markov chains.
This research will continue on previous work completed by other researchers. The HFS will be selected as the core of the mathematical model. This is because the relation with other research is strong. Therefore, the likelihood that this research will be used by other researchers and industry increases. Furthermore, HFS problems can be solved optimally given sufficient computational time. Whenever computational time is limited, many alternative approximative solution techniques exist which speed-up the calculation process. Moreover, the HFS problem allows many additional features which can easily be turned on and off dependent on the user's wishes. Besides that, the HFS problems appear better structured and organized compared to MPS and Markov chains. Finally, HFS problems can easily be scaled to the requirements of the user. Future research indicated by other researchers, with regards to individual container scheduling, has been grouped and collected in Table 3.4. This research initially concerns loading and unloading, integration of three machine stages, unrelated parallel machines, limited buffer positions and precedence relations. Furthermore, decreasing computational time will be considered as an objective in addition to minimization of the makespan. Therefore, many indicated future research directions are addressed within this report.

	Bish [2]	Xin [61]	Lu [56]	Kim [63]	He [64]	Kavesh-gar [65]	Cao [35]	Cao [66]	Kizilay [67]
Loading and unloading		x	x	x				x	
Decrease computational time		x						x	
Yard allocation	x								x
Uncertainties	x				x	x			
Precedence relations	x								
No buffer				x			x	x	
Initial solution heuristic					x				
Parameter tweaking					x				
Three stage integration						x	x	x	x
Unrelated parallel machines						x			
Due dates							x		
Release date							x		
Larger terminal									x

Table 3.4: Future research directions indicated by other researchers in the field of integrated container terminal scheduling.

Besides constructing the HFS model, it also needs to be optimized. Since HFS problems are NP-hard and the integrated container terminal scheduling problem is large, it is unlikely that exact solutions provide answers within reasonable time. Therefore, approximate solutions are used to approach the optimal solution quickly. The best metaheuristic available to solve HFS problems is SA [95]. Therefore, this research will continue on those conclusions.

4

Scheduling model applied to a container terminal

In this chapter we will translate the integrated container terminal scheduling problem to a HFS problem. First, the definition of jobs, machines and operations will be presented. Subsequently, a few assumptions are made which are required to represent the current situation at TBA. Thereafter, the required features of the HFS will be identified. Afterwards, a conceptional model will be formulated. Subsequently, the mathematical model will be presented which is used to represent the integrated container terminal scheduling problem as a HFS problem.
Once the model is constructed, a heuristic to generate a feasible initial solution is presented. Subsequently, a solution technique is introduced that uses this initial solution to approximate the optimal solution to the HFS problem. We conclude with computational results obtained with the designed HFS model and solution technique.

4.1. Hybrid Flow Shop Problem

The integrated container terminal scheduling problem could be casted into a HFS without much effort. This is because the structure of both problems is similar. A container must be handled by three different machine sets in a sequential order. Furthermore, each set of machines contains multiple machines. However, since this research considers the case that the QC and the location in the yard are already determined, only the machine set containing the Lift-AGVs has multiple parallel machines. This means that only one Lift-AGV should be selected to transport a container between the quay and the stack. The aim of the integrated container terminal handling problem is to determine the sequence and the starting times of machine moves for each specific container. With the overall goal to minimize the makespan of all operations. In other words, find the optimal sequence of moves for which the QC is finished as early as possible given a predefined loadplan.

4.1.1. Basic notation and assumptions

The HFS can be constructed by translating container handling definitions to HFS definitions [19]:

1. **Jobs:** All actions that need to be performed with one specific container are collected under the term job. Typically, there are two kinds of jobs: loading (stack to ship) and unloading (ship to stack) jobs. This definition corresponds to an order in container terminal terminology.

2. **Machines:** These are the different equipment types: QC, Lift-AGV and YC. Each order requires actions of all three machines. The equipment types correspond directly to the different stages.

3. **Operations:** Each job is composed of several operations: a transfer between the quay and the ship, a transfer between the stack transfer point and the quay and a transfer between the stack and the stack transfer point. These operations are respectively handled by the QC, the Lift-AGV and the YC. This definition corresponds to moves in container terminal terminology.

The following assumptions are introduced to formulate the HFS problem:

- The schedule is created to handle peak loads. This means that all QCs are continuously active. Not necessarily handling vessels. Furthermore, there is a landside peak; which means that it is busy at the road truck gate. Finally, the number of containers in the yard is large.
- The load and unload sequence plans are known and fixed for each QC. QCs are not allowed to deviate from these plans.
- Dual-cycling is not considered; a QC will completely finish a loading or unloading bay before it switches to another loading or unloading bay. However, different QCs could be loading and unloading simultaneously.
- Tandem moves are not considered; only single 20 ft., single 40 ft. and twinlifting two 20 ft. containers is considered.
- The locations of containers in the stack and the deep-sea vessels are known. Which means that the YC and the QC that should handle each container is known beforehand.
- The number of machines in each machine set is fixed and is invulnerable to breakdowns.
- All parallel equipment is equal and can only handle one container simultaneously with the exception of twinlift moves.
- Orders that have an unknown destination receive a destination located at a place with an average distance to all possible destinations for that order.

4.1.2. Problem classification

Since a HFS is used to describe the problem, the problem can be classified based on the classification scheme introduced in [49, 50, 52]. The classification is useful to categorize the problem and compare performance with other HFS problems. First, the problem will be described in detail to highlight all facets of the HFS. Subsequently, the $\alpha|\beta|\gamma$ classification will be provided.

- The number of stages in the HFS is equal to three. This corresponds to the number of equipment types. Each machine within a stage is identical. However, all operations are different. Therefore, the processing time of each operation is different and dependent on the job. The processing time is determined by the position of the container in the yard and on the ship for the YC and QC stage. The processing time on the Lift-AGV stage is determined by the distance between the respective QC and the respective YC. Therefore, the HFS is classified as a three stage unrelated parallel machine problem.
- Precedence relations exist on the QC stage. This is because the stowage plan on the ship is known beforehand; containers can only be (un)loaded in a specific sequence.
- Sequence dependent setup times are used to describe the back route of Lift-AGVs and YCs. A back route is a route that equipment should travel to pickup their next container after delivering a container. The travel time is dependent on the sequence of containers since it defines the origin and the destination of the back route. Additionally, setup times are used at the QC stage whenever a bayswitch is required.
- Machine eligibility is a required feature since certain containers can only be handled by certain machines. This holds for all the YC and QC actions. The location in the yard and the position on the ship directly determines which crane should be selected to handle the container. This is because QC assignment and yard allocation problems are out of scope.
- Blocking is a required feature to constrain the available buffer space between different operations. In case of AGVs there is no buffer at all; which means that equipment can only proceed with its next operation once the next equipment type becomes available. This means that certain containers block a machine for as long as the next machine is not available. In case of Lift-AGVs, there is limited buffer capacity present between the Lift-AGV an YC stage. In a HFS this is

modelled by inserting an extra stage with the number of machines equal to the buffer capacity. Subsequently, the same buffer constraint is used as is used whenever there is no buffer capacity [53]. The inserted buffer stage does not have any processing times so containers do not require to stay in the buffer for a certain time. This means that the HFS deploys unrelated parallel machines on the three stages with machines and parallel identical machines at the buffer stage.

- Bidirectional flow shop is not indicated by [49, 50, 52] but, it is an important aspect on a container terminal. This is because containers should move between the QC and the YC in both directions. Some containers might require loading while others require unloading simultaneously. This a unique aspect of the HFS model considered in this research.
- Release dates are required to couple multiple iterations of the HFS. This is required since a container terminal is in continuous operation and is not capable of collecting the entire container plan for its entire lifetime. This means that gradually containers are added to the planning horizon. Furthermore, the division of the entire container plan into smaller chunks reduces the problem size significantly. Besides that, on a continuously operating terminal there will always be containers on the terminal which are currently in progress.
- Twin lift moves should be carried by a single Lift-AGV, which carries both containers simultaneously. Furthermore, the QC serves both containers of the twin pair simultaneously. Contrary to the YC, which serves both containers individually. It is even possible that both containers should be obtained or delivered from/to different stacks. This feature requires extra constraints which were not earlier described by other research. Different jobs may require simultaneous handling by the same machine only on certain stages.
- The objective of the HFS is to minimize the makespan of the entire known container plan. Primarily, the QC should be finished as early as possible. However, also some value is added to the completion time of other equipment. This is to ensure that containers are removed from the plan whenever possible.

The above mentioned facets of the HFS lead to the following classification triplet:

$$FH4,((RM)^k)_{k=1}^4|r_j,M_j,S_{sd},block,prec,Bidirec,Jobpairs|\bar{C} \tag{4.1}$$

The features Bidirec and Jobpairs are newly introduced in this research. Bidirec is used to describe the relaxation of unidirectional flow in the flow shop. Jobpairs is a feature introduced to specify that some containers require simultaneous handling on a stage by one machine. This removes the assumption that machine capacity is always equal to one. The model developed in this research can be compared to previous research with Table 4.1.

Article		Stages	Machine type	M_j	S_{sd}	block	prec	$p_{ij} \sim N(\mu,\sigma)$	Bidirec	Jobpairs	Objective
Chen	[19, 59]	3	R	x	x	x					C_{max}
Lau	[60]	3	R	x	x		x				$\bar{C}^w$
Bish	[2]	3	P	x	x	x					$\bar{C}^w$
Xin	[61, 62]	3	R			x					C_{max}
Lu	[56]	2	P		x	x		x			T_{max}
Kim	[63]	2	P		x						$\bar{L}+\bar{C}$
He	[64]	3	P,R		x		x				$\bar{L}^w + energy$
Kaveshgar	[65]	2	R			x	x				C_{max}
Cao	[35]	2	P	x	x	x					C_{max}
Cao	[66]	2	P	x	x	x					C_{max}
Kizilay	[67]	2	R	x	x		x				C_{max}
This thesis		**3,4**	**R**	**x**	**x**	**x**	**x**		**x**	**x**	$\mathbf{C^w}$

Table 4.1: A table to compare the model developed in this research with previous efforts, indicating the structure, features and objective value according to the classification scheme of [49, 50, 52]

4.1.3. Conceptual model

Now that the problem is identified, a conceptual model will be constructed which is used to determine which decision variables are required and what constraints should be imposed.

The purpose of the model is to determine which machine should handle which container and at what time. In other words, these are the decision variables. Constraints are required to ensure the following aspects:

- Operations can only start in the future;
- Two sequential operations on different stages should be separated by the processing time of the first operation;
- The next stage can only start once the current stage completes an operation;
- A machine cannot proceed with its next operation once there is no possibility to discharge its current container;
- Setup times between different operations are sequence dependent;
- A machine can only perform one operation simultaneously;
- A job requires one operation on each stage;
- At the QC, the sequence of operations is fixed. Implicitly also specifying which QC handles which container;
- At the YC the containers that should be handled is known but, the sequence is still variable.
- Twin lift containers should be handled simultaneously at the Lift-AGV and the QC stage.
- Preemption of operations is not allowed. More specifically, the initial states of the machines cannot be changed.

A visual description of the HFS indicating the different stages is shown in Figure 4.1. Furthermore, this figure illustrates the difference between the Lift-AGV stage and the other stages. Additionally it indicates the flow pattern and possible machine options from a given machine. Finally, the figure indicates the stage indices.

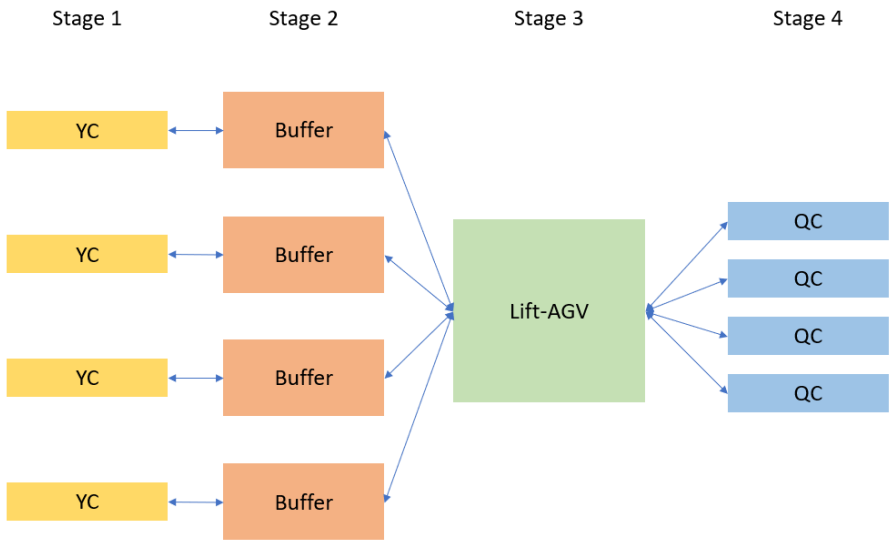


Figure 4.1: Illustration of the HFS applied to a container terminal. The YC and QC stages are represented by multiple small blocks since the machine assignment is fixed. The Lift-AGV stage is represented by a large block since the machine selection is an element of the to be developed schedule. Arrows are used to indicate how the different blocks are connected. Containers can only move along the arrows.

As can be seen in Figure 4.1, each YC has access to its own buffer. This means that the buffer stage contains multiple spots to place containers but that the buffer is not shared between different YCs. Therefore, the buffer stage resembles both the YC and the Lift-AGV stage. Containers can only enter a dedicated pool of buffer locations.

4.1.4. Mathematical formulation

In this section our HFS model is translated into a mathematical formulation. Since the problem size is often a bottleneck for HFS problems it is of paramount importance to reduce the number of variables. Therefore, caution should be taken to add artificial variables. However, in some cases it is impossible to avoid adding artificial variables. For this specific problem, artificial variables are required to mathematically represent the sequence of operations. This is required since setup times are sequence dependent.
Fortunately, the artificial sequence variables relate very well to the earlier defined decision variables. Even so well that the artificial variables can replace the decision variables used to describe which machine handles which container. Since each machine possesses only one sequence of operations, the handled containers per machine could be extracted from the sequence variables. However, the number of sequence variables is larger than the number of machine assignment variables. This is because the artificial sequence variables scale quadratically with the number of containers whereas machine assignment variables scale linearly with the number of containers. Additionally, two artificial jobs are created which define the first and the last job of a sequence [35]. These jobs are excluded from the constraint that they can only be selected once.

The first step towards creating a model is to declare certain parameters and sets.

i, k job index $i, k \in \phi_1 \cap \phi_2$

j stage index $j \in 1, 2, 3, 4$

m machine index $m \in 1, 2, ...n_j$

n_j number of machines on stage j

N total number of containers

ϕ Set of all jobs excluding $i = 0, i = N + 1$

ϕ_1 Set of all jobs including the first dummy job $i = 0$

ϕ_2 Set of all jobs including the last dummy job $i = N + 1$

P set of job pairs with precedence relations

O_{ij} operation i at stage j

M_{ij} set of machines which can process O_{ij}

E_m set of jobs that can be performed on machine m, includes jobs $i = 0$ and $i = N + 1$.

K_{ij} set of jobs that can form a pair with job i on stage j, includes jobs $k = 0$ and $k = N + 1$.

T set of twinlift operations.

T_i the twin pair of container i.

A set of containers that is already assigned to equipment.

Pr set of containers that is already assigned to equipment but not in current progress.

L set of all loading containers.

J_i set of stages that a container i already past or is being currently processes job i.

Q_{jm} number of jobs that must be performed on machine m

r_{ij} release date of job i on stage j.

p_{ij} processing time of operation O_{ij}

p_{ij}^{ik} processing time of operation O_{ij} whenever operation i directly precedes operation k.

p_{ij}^{ki} processing time of operation O_{ij} whenever operation k directly precedes operation i.

s_{ikj} setup time between job i and job k at stage j

s_{0kjm} setup time between the dummy job and job k on stage j for machine m.

H a sufficiently large constant.

β a constant indicating the weight of completion time of operations on the YC stage, $\beta \ll 1$.

To get a better feeling of the relative position of sets with respect to other sets, Figure 4.2 is created. It should be noted that sets ϕ_1 and ϕ_2 are enclosed by sets ϕ and E_m.

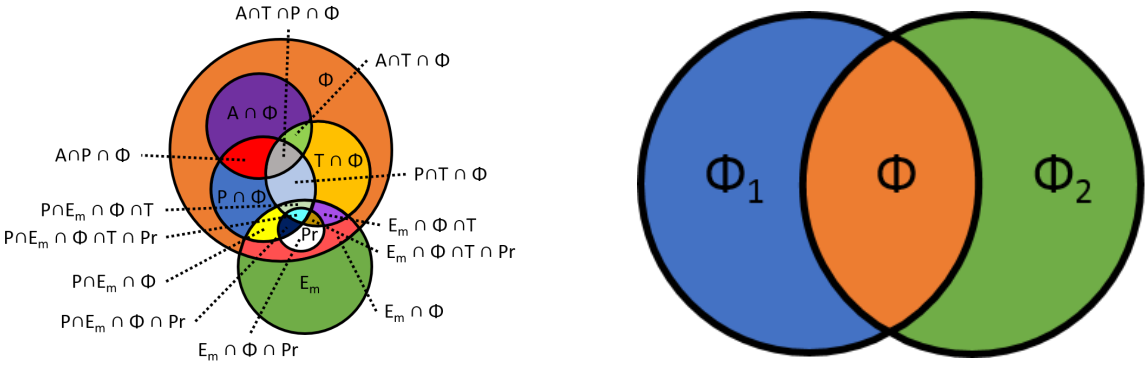


Figure 4.2: Euler diagrams displaying the enclosement of different sets.

Mathematical Lift-AGV model

Lift-AGV models require a finite sized buffer stage between the YC stage and the Lift-AGV stage. Mathematically, finite sized buffers can be modeled as an extra stage with the number of machines equal to the number of buffer spots. The processing time on this inserted stage should be equal to zero but, the blocking constraint should remain active. Together this ensures that containers can directly pass the buffer stage but, that containers cannot enter the buffer stage whenever the buffer spot is claimed by another container or leave the buffer whenever the next machine is occupied. The impact on the HFS model is that an extra stage is inserted such that the Lift-AGV stage becomes stage three and the QC stage becomes stage four. The buffer stage will be referred to as rack stage since containers are placed on a rack. The rack stage shows similarities with both the Lift-AGV and the YC stage in terms of processable containers. There is a pool of buffer machines which can accept all containers assigned to a specific YC. Similar to the YC stage, twin moves are separated whenever they are at the rack stage. An overview of the stage indices can be found in Table 4.2.

	stage index [j]
YC stage	1
Rack stage	2
Lift-AGV stage	3
QC stage	4

Table 4.2: Overview of stage indices for the mathematical Lift-AGV model.

- t_{ij} The start time of job i on stage j
- z_{ikj} 1, if O_{ij} immediately precedes O_{kj}; 0, otherwise $\forall i \in \phi, \forall k \in \phi_2, \forall j \in \{1,2,3,4\}$.
- z_{0kjm} 1, if O_{0jm} immediately precedes O_{kj}; 0, otherwise $\forall k \in \phi_2, \forall j \in \{1,2,3,4\}, \forall m \in M_{ij}$.

The parameters p_{ij}^{ik} and p_{ij}^{ki} are only required for twin moves ($\forall i,k \in T$) since twin moves are represented by two sequential single moves with different processing times. Whenever i,k are pairs of a twin move p_{ij} is replaced by $p_{ij}^{ik} z_{ikj} + p_{ij}^{ki} z_{kij}$. For example, at the QC stage considering an unloading move, $p_{ij}^{ik} = p_{ij}$ seconds and $p_{ij}^{ki} = 0$ seconds. This ensures that the second container of the twin pair does not require additional processing time at the QC stage. A similar approach is used at the Lift-AGV stage to reflect the different driving distances. The twin containers are separated at the YC stage. Therefore, the processing time of these containers does not change whenever they were/later become twin pairs. Subsequently, the integrated scheduling problem can be formulated in a HFS style.

Minimize
$$Z = \beta \sum_{i=1}^{N} t_{i1} + \sum_{i=1}^{N} t_{i4} - \beta \sum_{i=1}^{m} \sum_{k=1}^{N+1} z_{0kjm} \tag{4.2}$$

Subject to
$$t_{ij} \geq r_{ij} \qquad \forall i \in \phi, \forall j \in \{1,4\} \tag{4.3}$$
$$H(1 - z_0 kjm) + t_k j \geq s_0 kjm \qquad \forall k \in \phi \cap A^C, j \in \{3\}, \forall m \in n_j \tag{4.4}$$
$$t_{ij} + p_{ij} \leq t_{i(j+1)} \qquad \forall i \in \phi \cap T^C, \forall j \in \{3,4\} \tag{4.5}$$
$$t_{ij} + p_{ij} \leq t_{i(j+1)} \qquad \forall i \in \phi, \forall j \in \{1,2\} \tag{4.6}$$
$$t_{ij} + p_{ij}^{ik} z_{ikj} + p_{ij}^{ki} z_{kij} \leq t_{i(j+1)} \qquad \forall i \in T, \forall j \in \{3,4\} \tag{4.7}$$
$$\sum_{i=1}^{N} \sum_{k=1}^{N} z_{ikj} = Q_{jm} + 1 \qquad \forall i,k \in E_m, \forall j \in \{1,2,3,4\}, \forall m \in M_{ij} \tag{4.8}$$
$$\sum_{k=1}^{N} z_{0kj} = n_j \qquad \forall k \in E_m, \forall j \in \{1,2,3,4\} \tag{4.9}$$
$$\sum_{i=1}^{N} z_{i(N+1)j} = n_j \qquad \forall i \in E_m, \forall j \in \{1,2,3,4\} \tag{4.10}$$
$$\sum_{k=1}^{N+1} z_{ikj} = 1 \qquad \forall i \in \phi, \forall k \in \phi_2 \cap K_{ij}, \forall j \in \{1,2,3,4\} \tag{4.11}$$
$$\sum_{i=0}^{N} z_{ikj} = 1 \qquad \forall i \in \phi_1 \cap K_{kj}, \forall k \in \phi, \forall j \in \{1,2,3,4\} \tag{4.12}$$
$$z_{ikj} + z_{kij} = 1 \qquad \forall i,k \in T, j \in \{3,4\} \tag{4.13}$$
$$t_{i(j+1)} + s_{ikj} \leq t_{kj} + H(1 - z_{ikj}) \qquad \forall i,k \in \phi \cap T^C, \forall j \in \{1,2,3,4\} \tag{4.14}$$
$$t_{ij} + p_{ij} + s_{ikj} \leq t_{kj} + H(1 - z_{ikj}) \qquad \forall i,k \in \phi \cap T^C, \forall j \in \{4\} \tag{4.15}$$
$$t_{ij} + p_{ij} + s_{ikj} \leq t_{kj} + H(1 - z_{ikj}) \qquad \forall i,k \in \phi, \forall j \in \{1,2\} \tag{4.16}$$
$$t_{ij} + p_{ij}^{ik} z_{ikj} + p_{ij}^{ki} z_{kij} + s_{ikj} \leq t_{kj} + H(1 - z_{ikj}) \qquad \forall i,k \in T, \forall j \in \{3,4\} \tag{4.17}$$
$$\sum_{i=1}^{N} t_{ij} \leq 0 \qquad \forall i \in A, \forall j \in J_i \tag{4.18}$$
$$\sum_{i=1}^{N} t_{ij} \leq H(1 - z_{ik3}) \qquad \forall i,k \in A \cap T, \forall j \in J_i \tag{4.19}$$
$$\sum_{i=1}^{N} z_{0kjm} + z_{ikj} = 1 \qquad \forall i \in A \cup T_k, \forall k \in Pr, \forall j \in \{1,2,3\} \tag{4.20}$$
$$z_{ikj}, z_{ikjm} = 0 \; or \; 1 \qquad \forall i,k \in \phi_1 \cup \phi_2, \forall j \in \{1,2,3,4\} \tag{4.21}$$

Note that whenever $j+1$ is listed the next stage is meant, not necessarily being stage $j+1$. This is because for unloading the next stage is stage $j-1$. Furthermore, since $1 \leq j \leq 4$, $j \in \{1,2,3\}$ whenever $j+1$ is required and loading containers are considered. Whenever unloading is considered and $j-1$ is required; $j \in \{2,3,4\}$.
Since the sequence of operations on stage 4 is already known entirely, all the required z variables are known and can thus be excluded as decision variables.
Additionally, variable z_{iij} does not exist; a job can never immediately proceed or precede itself. Therefore, these variables are excluded.

Mathematical model description

In this section we give a textual description to all constraints presented in the previous section. In the objective function (Equation 4.2) the makespan is minimized adding the most weight to the completion time of operations at the QC stage. Since each operation is considered in the objective function, most weight is indirectly added to the first few operations on each QC. This is because delays in the beginning propagate to other operations. This is a desired feature of the objective function since the first few operations are most accurately predicted and are most important to be finished. Besides that, distribution of jobs to different machines is stimulated by the small negative factor β. Whenever the makespan is equal, the jobs should be evenly distributed over all machines. Although $\beta \ll 1$, the difference should not be too large. This is because solvers might disregard changes in the sums multiplied by β whenever the remaining sum is dominant to determine the objective value. This is because when the objective value changes less than the objective threshold value, solvers consider their job done.
Constraint 4.3 enforces that jobs start after they are being released. This is only required at the starting stage and the first job of the sequence since other operations and jobs are forced to start later. Practically, this constraint enforces that the YC and the QC should finish their current operation before they become available for a next planning horizon. Constraint 4.4 has a similar function as Constraint 4.3 but, focuses on the Lift-AGV stage. This is required since it is unknown which container is handled by which Lift-AGV and each Lift-AGV can become available independently.
Constraint 4.5 ensures that a job can only start at the next stage once the current stage is completed. Constraint 4.7 is required to make use of the sequence dependent processing times for twin moves. However, at the YC stage containers cannot be twin containers. Therefore, the processing time remains unchanged and thus Constraint 4.5 applies for all containers in ϕ. This is expressed by Constraint 4.6.
Constraint 4.8 is only active once the machine that should handle a job is already known. It ensures that a sequence of z_{ikj} is only selected from a subset of ϕ since not all i and k are available. Furthermore, the number of sequence variables is constraint to the number of jobs that should be handled on the machine plus one. This is because each sequence starts and ends with a dummy job. In our specific case this constraint is only active at the YC stage.
Constraint 4.9 ensures that the number of started sequences is equal to the number of machines. However, once a sequence cannot start at an arbitrary job, a machine might not be able to process every job, those jobs are excluded for that particular machine. Subsequently that machine receives its personal constraint that means that a sequence should start at one of the available jobs. Constraint 4.10 is similar to Constraint 4.9 but now to ensure that each sequence is ended at a dummy job. Similarly to Constraint 4.9, if a machine cannot process every job that machine is given its personal variant of Constraint 4.10.
Constraint 4.11 ensures that each operation is proceeded by another operation or is the end of the sequence. Similar is Constraint 4.12, which states that each operation is preceded by another operation or is the start of the sequence. Both of these constraints exclude infeasible job pairs whenever they are obliged to be performed on different machines.
Constraint 4.13 ensures that a pair of twin moves is handled by the same machine. This is not required at stage 1 since different YCs might work on the same twin pair.
Constraint 4.14 is the blocking constraint; it states that whenever job i immediately precedes job k on stage j, job k cannot start earlier than the start time of job i on the next stage plus the setup time between job i and job k. This is because the start of job i on the next stage marks the point in time where the container is released from the current stage. A timeline is added to better visualize this process on stage j (see Figure 4.3). It can be seen that a container can be processed on the next stage while the considered stage is busy with a setup. However, it could be possible that the next stage is

not directly available at $t_{ij} + p_{ij}$ that is when Constraint 4.14 makes sure that operation k has to wait. This effect is shown in Figure 4.4.
Constraint 4.15 is used to separate operations by their processing times and setup times. Practically, this constraint is only required at the first and last stage since intermediate stages automatically satisfy this constraint by a combination of Constraint 4.5 and Constraint 4.14. Furthermore, since the entire sequence of operations is known at the QC stage, z_{ik3} becomes superfluous as a decision variable and becomes an ordinary parameter. Constraint 4.17 is the same as Constraint 4.15 only for twin moves which have their processing time dependent on the sequence of operations. Practically, since pairs of twin moves do not really have a sequence at the QC, the same sequence as the sequence on the Lift-AGV is used at the QC stage. This results in lesser variables and constraints since z_{ik4} is superfluous. For the same reason as for Constraint 4.6, Constraint 4.16 is added specifically for the first stage since the twin moves are separated here and thus processing times are sequence independent.
Constraint 4.18 ensures that containers that have already passed a stage or that are currently processed on a stage start their operation at time zero. However, this constraint is not valid whenever an unloading twin container is currently assigned to a QC or to a Lift-AGV. This is because twin containers are modelled as sequential single moves with different processing times. Therefore, Constraint 4.19 is introduced which specifies that at least one of the twin containers starts at zero. Constraint 4.13 subsequently ensures that the other container of the twin pair is processed directly after its twin pair.
Constraint 4.20 ensures that operations that are assigned to equipment but, that have not currently started can only proceed: containers that are in current progress or containers that have passed that stage already or their twin pair or the dummy start container. This situation occurs frequently whenever a machine is driving empty to the grab location whenever a new schedule is desired.
Finally, Constraint 4.21 is used to enforce binary solutions to the decision variables z_{ikj} and z_{0kjm}.

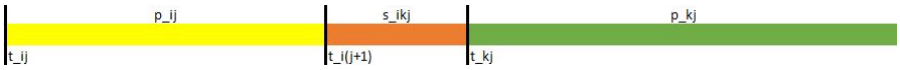


Figure 4.3: Timeline illustrating the setup process with regards to Constraint 4.14.

Figure 4.4: Timeline illustrating the blocking Constraint 4.14.

The total number of decision variables is dependent on the number of stages, the number of containers and the number of Lift-AGVs. The number of variables can be expressed with Equation 4.22.

$$\#\ of\ variables\ = N \cdot (\#\ of\ stages) + (N+1) \cdot (n_2 + n_3) + (N+1)^2 + 2(N^2 + N) \tag{4.22}$$

Equation 4.22 is based on the situation at TBA where each container can only be processed by one specific YC and one specific QC. The Lift-AGVs are allowed to handle any container. However, machines are not allowed to preempt a job. This means that only the first container is constrained to be processed on a specific Lift-AGV whenever it was already assigned to that Lift-AGV. Furthermore, the sequence of operations at the QC is known in advance and therefore requires no decision variables.

4.2. Initial solution heuristic

Almost all numerical solvers require an initial solution to start an optimization process. Generally, the closer the initial solution is to the optimal solution, the quicker the optimization algorithm converges to the optimal solution. Therefore, the construction of the initial solution is an important part of the optimization process. For HFS problems typically dispatching rules are used to find an initial solution. Dispatching rules are simple rules of thumb used to rank jobs and assign them to machines [50]. In Chen et al. [19] three different heuristics are compared with each other to determine which of them

most quickly produces an optimal solution when combined with a Tabu Search algorithm. The three compared heuristics are: Minimum Extra Time (MET), Minimum Completion Time (MCT) and Non-Delay (ND). It was concluded that the ND heuristic performed best. Therefore, the ND heuristic will be used in this research as well.

4.2.1. Minimum Extra Time heuristic

MET relies on minimization of the setup times. The idea is that processing times are unavoidable. Therefore, the quality of a solution is determined primarily by the length of the setup times. The pseudo-code of the algorithm is as follows [19]:

1. Arrange the containers in numerical order for use on the first stage;
2. Prioritize containers in numerical order. Insert container i at every possible position in the machine sequence developed so far. Subsequently, calculate the extra time c_{ij} for each insertion;
3. Insert container i to the position which leads to a minimum extra time;
4. On subsequent stages, the priority follows a First In First Out (FIFO) strategy. Afterwards, the extra time can be calculated for each insertion. Finally, the container is inserted at the position which results in the MET.

The extra time of an insertion of container i on stage j between containers a and b can be calculated with Equation 4.23 [19]:

$$c_{ij} = s_{aij} + s_{ibj} - s_{abj} \tag{4.23}$$

The assumption that the quality of a solution is primarily determined by setup times is not necessarily valid for the integrated container terminal scheduling. This is because the quality of a solution is primarily determined by the time that equipment is waiting for preceding stages to finish their jobs and waiting on proceeding stages to relieve them of their current job.

4.2.2. Minimum Completion Time heuristic

MCT is another heuristic that could be used to prioritize moves. A job which can be completed as early as possible is given the most priority [129]. Subsequently, jobs are assigned to the machines completely relying on the priority alone.

4.2.3. Non-Delay heuristic

The ND heuristic schedules jobs in numerical order such that a machine is never idle while a task is waiting to be processed [53]. This heuristic performed best of all three heuristics during the research of Chen et al. [19]. Therefore, this heuristic is selected for the problem in this research. Compared to the problem described in [19] the heuristic is expected to perform even better on the problem of this research since the entire sequence of operations is known at the QC stage. Furthermore, also the YC that should be selected is already known so the decision space tightens. The pseudocode of the used ND scheduling heuristic is as follows:

1. Sort containers in numerical order such that precedence relations are satisfied;
2. Start processing container i on the first stage (QC/YC stage) as soon as the machine becomes available plus the setup time to operation i.
3. Find the machine on the next stage that comes first available and that is allowed to handle container i. Availability is the sum of the start time of the current job, the processing time of the current job plus the setup time to job i;
4. Start processing container i on the next stage ($j+1/j-1$) at the maximum time of the availability of the machine and the completion time at the current stage (j);
5. Repeat steps three and four until all stages have been accounted for;
6. Repeat steps two to four until all containers have been planned.

This heuristic uses a FIFO strategy on all stages which seems reasonable since the containers should be handled in FIFO order on one stage (the QC stage). For the second container of a twin pair and already assigned containers, the machine which should handle the container at the Lift-AGV stage is already known. Therefore, that machine is the sole machine which is allowed to handle the respective container.
To improve the performance of the schedule, both sequences of a twin pair are evaluated and the best one is selected. This is because the sequence of a twin move is not fixed by the QC.

Another advantage of the ND heuristic strategy is that multiple solutions can be generated quickly. This is because there are multiple sorts available that satisfy the precedence relations. This seems controversial but, since precedence relations are only active for each QC individually, moves using different QCs can still be sorted in a different way. This is best explained by a simple example shown in Table 4.3. As can be seen in Table 4.3, the precedence relations for each individual QC are satisfied. A move with a higher sequence number for one QC is not performed before a move with a lower sequence number at the same QC. The number of available sorts rapidly increases with the number of containers and QCs.

Numerical order	QC	Move of QC		Numerical order	QC	Move of QC
1	1	1		1	1	1
2	1	2		4	2	1
3	1	3		2	1	2
4	2	1		5	2	2
5	2	2		3	1	3
6	2	3		6	2	3

Table 4.3: Two different sorts of containers that both satisfy precedence relations for all QCs

To find an initial solution the ND heuristic is used on multiple feasible sorts. Subsequently, the sort with the best objective value is selected as an initial solution. Whenever multiple initial solutions are required, the best number of required solutions could be selected.

Whenever a receding horizon is used during planning, there will be containers which are already in the system whenever the scheduling starts. These jobs are exceptional jobs that could have already passed certain stages or are in current progress by a certain machine. The last option is that a machine is currently underway to grab a certain container. These actions cannot be interrupted. Therefore, for these operations the machine that the container is to be processed on is fixed and the starting time is also predetermined. This is accommodated in the initial solution in the following way: for each stage all the known information is first assigned. Subsequently, the remaining containers enter the ND heuristic. This method implies that blocking and precedence relations are no longer naturally satisfied. Therefore, whenever a new container has to be assigned to a machine. Machines that will violate blocking and precedence relations should be excluded.

4.2.4. TBA's uncoordinated scheduling heuristic

Another available heuristic is the uncoordinated scheduling heuristic that is currently in use by TBA. This heuristic is based on a machine perspective instead of a container perspective. Whenever a machine completes a task and thus comes available for a new task, a scoring strategy is used to determine which container should be handled next. The detailed outline of TBA's uncoordinated scheduling heuristic can be found in Figures 2.10 and 2.11. This heuristic is effective whenever there is great uncertainty in processing times and sequence order. Therefore, the heuristic is adaptive and easily accommodates to unexpected delays. However, once all processing times are considered deterministic, and the sequence of operations at each QC is fixed, the benefits of a machine perspective heuristic disappear. This is because the scores of the heuristic are easily maximized. This is because the sequence of orders is known by each machine and can be distributed to each machine is the appropriate order. Furthermore, the urgency score is a score based on the desired starting time of each machine. The desired starting

times of containers on one machine are parallel to the optimal sequence. Therefore, whenever all containers are sorted on desired starting times, considering all QCs simultaneously, both the sequence score and the urgency scores are maximized. Subsequently, also the setup time to a container could be minimized since the setup time to a container is known in advance for each machine. Therefore, a container perspective outperforms a machine perspective. However, the idea that containers should be supplied to a machine following an urgency score is promising. Therefore, one of the sorts within the total number of sorts will be a sort where containers are sorted based on their urgency score. Subsequently, the containers are assigned to machines with the ND heuristic.

Nevertheless, the uncoordinated TBA scheduling technique was developed as well. This is for two reasons. First, it allows for an easy comparison between TBA's uncoordinated scheduling heuristic and the optimized schedule. Furthermore, the heuristic was used numerous times to find bugs in the mathematical model and the initial schedule generation. This is because it was easy to spot which container was the bottleneck in the assignment process. Moreover, since the initial solution is generated quickly it was later added as a second initial solution technique. Therefore, there are now two initial solution methods of which the best one is selected to proceed with. TBA's uncoordinated scheduling heuristic was implemented as follows:

1. Assign all containers which are passed certain stages, are in current progress or are pre-assigned to specific machines.
2. Whenever a machine comes available select the most urgent available container as the next operation.
3. Update the availability of the container on the next stage and the availability of the machine.
4. Repeat steps two and three until all containers are processed on each stage.

4.3. Solution method

In this section we evaluate the efficiency and effectiveness of different solution methods. The evaluation will be based on solving the mathematical Lift-AGV model.
It should be noted that often there are multiple optimal solutions to a HFS problem. This is because there are multiple identical vehicles. Therefore, a container could be assigned to different vehicles without changing the objective value. Furthermore, there are multiple solutions with approximately the same objective value. This means that the optimal value is in one of the multiple figurative valleys of the solution space. This complicates the solution process since finding improvements in a valley is difficult. First, the exact solution method B&B is applied to the HFS problem formulated as a MIP since it delivers the best possible solutions. Subsequently, SA will be used as a metaheuristic since metaheuristics approach the optimal solution quicker compared to exact solutions. As discussed in Section 3.5.11, SA is selected as metaheuristic since it is superior to Tabu Search and Genetic Algorithms when working with large HFS problems [95].

4.3.1. Mixed integer programming

First, MIP is used to model the HFS. A common way to solve MIPs is B&B. B&B provides optimal solutions within reasonable time. However, the computational time increases exponentially with increasing problem size. Therefore, B&B is only feasible when solving small scale problems. The computational speed can be increased significantly by passing an initial solution. However, the algorithm is still not quick enough to solve any of the benchmark problems described in Table 2.1 within reasonable time. Both MATLAB's internal B&B algorithm and the industrial level B&B algorithm of Gurobi were used for evaluation. It was found that Gurobi's B&B is much quicker compared to MATLAB's B&B algorithm. This is because Gurobi performs complex pre- and post-solving which make the algorithm no longer strictly B&B. One of differences between traditional B&B and the Gurobi's B&B algorithm is that Gurobi's B&B algorithm does not only phantom branches, it is also able to perform plane cuts in the solution space. With this action the number of attainable solutions reduces quicker. However, the computational time is reduced significantly by these actions, the computational time remains too excessive. The solution speed of the B&B algorithm is indicated in Table 4.8.

4.3.2. Simulated Annealing

SA was selected as approximative solution technique whenever the exact B&B solution technique proved insufficient considering the required computational time. SA works sufficiently quick and delivered proper results. This is in line with what was found in [95], SA is superior to Tabu Search and Genetic Algorithms when working with large HFS problems.
Several different techniques are required to enable SA to work with the HFS problem considered in this report. First, the global outline of the SA program is presented. Subsequently, the different steps will be explained in more detail.

Global outline

The first step of SA is to find an initial solution. To find an initial solution the non-delay heuristic was used on the sequence sorted based on the urgency score. The details of this heuristic can be found in Section 4.2.3. Additionally, TBA's uncoordinated scheduling heuristic was used, the details of this initial solution can be found in Section 4.2.4. Subsequently, the best one of these two objective values is saved as $f(x)$. Thereafter, an iterative process is initiated. The pseudocode of this process is presented on the next page.

WHILE # $of iterations \leq maximum$ # of $iterations$ AND # $of stalls \leq maximum$ # of $stalls$

Draw a random number $0 \leq R_1 \leq 1$.

IF $R_1 \leq YC$ $insertion$ $percentage$

Select a random container B from the YC stage which is not currently in progress or already removed from the stage.

Select another container A at random which is being processed on the same machine within $t_{limit.YC}$ seconds of container B and is currently not preceding container B.

Insert container B directly after container A (see Section 4.3.2).

IF a reparation is required, repair the sequences of other stages in the following order (see Section 4.3.2):

Repair Rack stage by YC stage.

Repair Lift-AGV stage by Rack stage.

Repair Lift-AGV stage by QC stage.

Repair Rack stage by Lift-AGV stage.

Repair YC stage by Rack stage.

ELSE

Select a container B at the Lift-AGV stage which is not currently in progress or already removed from the stage. The selection process can be found in Section 4.3.2.

Select another container A following the selection process described in Section 4.3.2 which is being processed within $t_{limit.AGV}$ seconds of container B and is currently not preceding container B.

Draw a random number $0 \leq R_2 \leq 1$.

IF container A cannot switch machines OR IF $R_2 \leq Insert$ $percentage$ AGV

Insert container B directly after container A (see Section 4.3.2).

ELSE

Swap container B with container A (see Section 4.3.2).

IF a reparation is required, repair the sequences of other stages in the following order (see Section 4.3.2):

Repair Lift-AGV stage by QC stage.

Repair Rack stage by Lift-AGV stage.

Repair YC stage by Rack stage.

Obtain the quality of the new obtained sequence and save it as $f(y)$ (see Section 4.3.2).

Draw a random number $0 \leq R_3 \leq 1$.

IF $f(y) \leq f(x)$ OR $R_3 \leq e^{\frac{f(x)-f(y)}{T}}$

$f(x) = f(y)$ and store the new obtained solution.

Update the temperature based on the temperature schedule.

As can be seen in the pseudocode, YCs do only allow inserts whereas Lift-AGVs allow both inserts and swaps. Since containers on the YC can only be processed on one machine an insert is productive. This is because the entire sequence remains intact and only one containers shifts. Generally, an optimal sequence is a sequence that ensures that loading containers are always available whenever a Lift-AGV arrives. Unloading containers can be processed at any time and only have priority whenever they prevent another machine from continuing to process its next container. Therefore, a swap could be too destructive whereas an insert is more productive on the YC stage.
Lift-AGVs benefit more of swaps instead of inserts. Since containers can be handled by any Lift-AGV an optimal sequence is generally a sequence that distributes workload evenly over all Lift-AGVs. This is more natural when using swaps instead of inserts. This is because the initial solution is likely to produce an evenly distributed workload to all Lift-AGVs. The problem with inserts alone, is that two inserts are required to restore the initial workload. This is unlikely to be productive for two reasons.

First, after the first insertion the chance that the new sequence has a better objective value compared to the current best objective is slim. Secondly, since containers are chosen at random, it is unlikely that the machine that received an extra container is chosen in the second insert iteration to discard a container. Therefore, workload becomes unbalanced when inserts alone are used at the Lift-AGV stage which is unlikely to lead to a low objective value. On the other hand, swaps alone do not guarantee to produce the optimal sequence either. This is because with swaps alone, the number of processed containers on each Lift-AGV can never change. Therefore, a balance is required between the chance to swap and the chance to insert a container. To complicate things further, whenever a more intelligent selection procedure is used, inserts on the Lift-AGV stage become more productive than swaps. Since containers are specifically selected to be removed and added to other sequences (see Section 4.3.2) Therefore, a balance between swaps and inserts is also determined by the intelligence of the selection process.
When performing swaps and inserts on the Lift-AGV stage, one has to deal with twin containers as well. Twin containers should always remain together. Therefore, they are inserted or swapped as a pair with another container. The other container does not need to be a twin pair as well. Furthermore, the sequence among the twin pair could also change by an insertion or swap. This is natural by the definition of a swap or insert, for instance if container B is directly inserted after container A. This means that when container B is the second container of a twin pair, it will become the first container after the insert. A similar logic is applied with a swap.

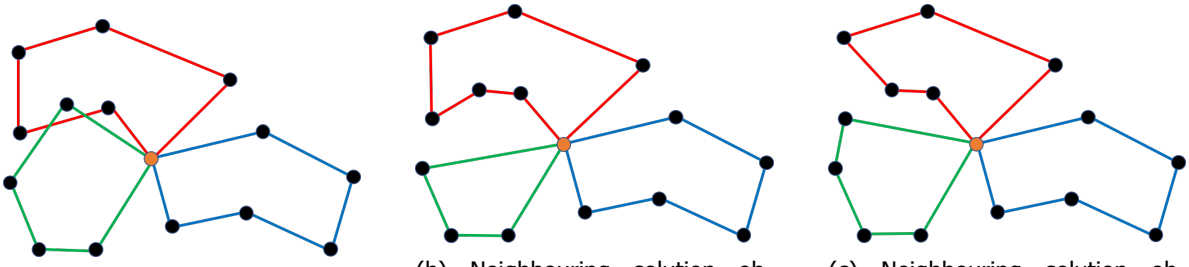
Throughout the SA process, several parameters are used. The values assigned to these parameters can be found in Table 4.4. The parameters will be explained in more detail throughout this chapter.

Parameter	**Value**
Maximum number of iterations	550
Maximum number of stalls	350
% YC inserts	0.45
% AGV inserts	0.75
Initial population	1
time_limit.YC	400
time_limit.AGV.advance	1000
time_limit.AGV.after	100
Horizon length	500

Table 4.4: Values assigned to the parameters used by SA.

Neighbour solution generation

Many heuristics are based on the fact that from a given initial solution a neighbouring solution is selected and accepted based on certain criteria. This is called local search. A solution is considered a neighbouring solution whenever the solution has exactly the same sequences of containers on each machine except for one container which is inserted after another container or one container which is swapped with another container. This is because this definition of a neighbour ensures that the neighbour is close to the original solution while maintaining feasibility. Furthermore, the same definition of neighbours is used in [19, 35, 64, 66, 78]. The insertion or swap should be performed whilst respecting precedence and machine assignment constraints. To remove a container from one sequence and insert it somewhere else requires the modification of three sequence variables. Whereas a swap requires the modification of four variables. To better visualize this, the analogy with a Vehicle Routing Problem (VRP) is used. On a single stage, the containers handled by machines could be perceived as cities. The different machines are the available pool of vehicles. The arcs represent the setup times between the different containers. Finally, the depot represents the dummy-start and dummy-end container which initializes and completes each sequence. The removal and construction process is illustrated with Figure 4.5. However, the analogy of the HFS problem with a VRP process raises the suspicion that the HFS problem is an addition of different VRPs. This suspicion is wrong: the HFS problem is much more complicated since the different VRPs on each stage are coupled with each other (see Section 4.3.2). Furthermore, the costs of a route is not equal to the sum of the arc lengths. Therefore, proven VRP solution techniques do not necessarily work on HFS problems.



(a) Initial solution

(b) Neighbouring solution obtained by an insertion.

(c) Neighbouring solution obtained by a swap.

Figure 4.5: Two possible neighbours of an initial solution for a VRP obtained by a swap of two jobs or by inserting one job in the job sequence of another machine.

Container selection strategy

To select a container at the Lift-AGV stage, either to swap or to insert, a predictive selection procedure is used. The selection procedure predicts which containers are most likely to reduce the objective value. Subsequently, a container is selected based on the likelihood to reduce the objective value. The selection procedure is explained with an illustration. In Figure 4.6 all containers that cause a delay in the QC schedule are marked yellow. Furthermore, it can be seen that the container highlighted pink causes the most delay in the QC schedule. Therefore, it is most likely that by reassigning this container an improvement in the overall schedule could be realized. However, to include randomness in the algorithm, which is often beneficial [78], the chance of selecting a container is a function of the caused delay. Therefore, each of the yellow marked containers could be selected with their probability proportional to the caused delay in the QC schedule. The cumulative distribution function used to determine this probability can be found in Figure 4.7. This function shows that the higher the caused delay a container produces, the more likely it becomes to select that container. The caused delay by a single container is always scaled with the total delay that is currently in the system. Therefore, the chance that a container is selected is always equal to one.

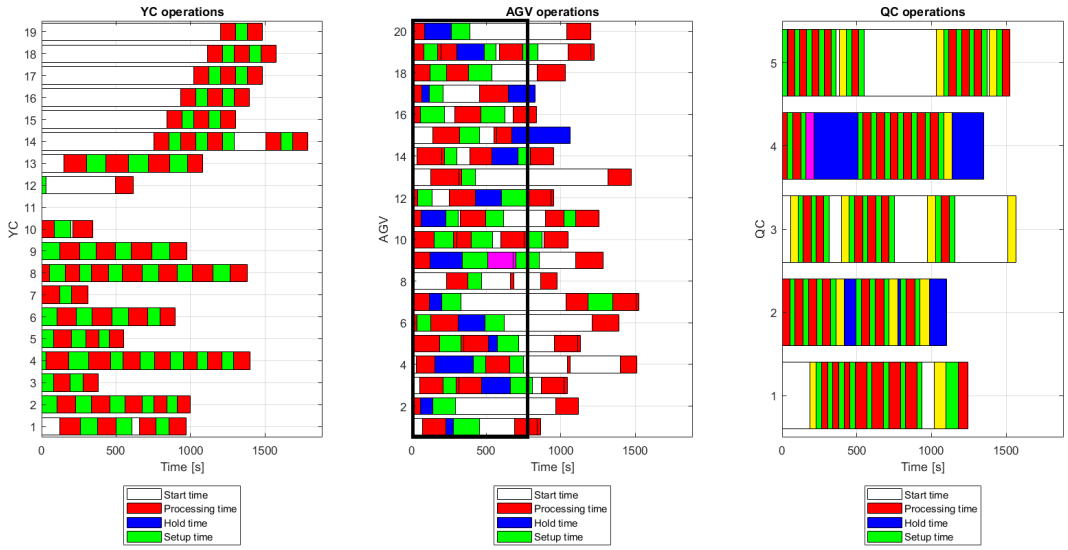


Figure 4.6: A schedule of the YC, Lift-AGV and QC, the yellow boxes represent containers that cause delay in the QC schedule. The pink box represents the container that causes the most delay in the QC schedule. Processing time is the productive working time of a machine on a container. Setup represents the driving empty time between two containers. Hold time is the time that a machine cannot proceed with its next operation since there are no discharge opportunities for the current operation. Within white areas a machine is idle.

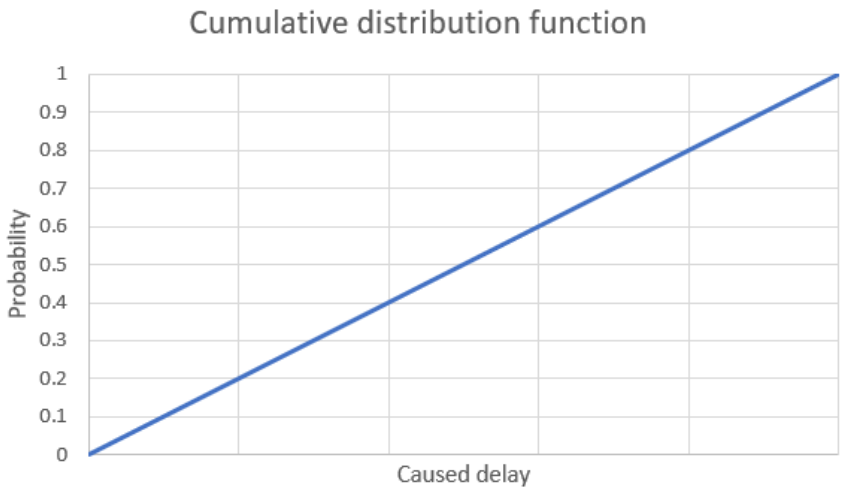


Figure 4.7: Cumulative distribution function used to select a container at the Lift-AGV stage.

Whenever container B, currently handled by Lift-AGV B, that causes delay in the QC schedule is selected to be removed, it needs to swap or be inserted with/after another container A, processed by Lift-AGV A. This selection process could also be done in a predictive way. A swap or insert becomes more likely to be productive whenever Lift-AGV A is idle during the handling time of container B. This means that containers after which an idle period is present become more likely to be selected. The likeliness can be expressed with Figure 4.7. Only now not the caused delay but the idle time is on the x-axis. Similar to the selection procedure for container B, the waiting time is expressed as a factor of the total waiting time. Therefore, the chance that a container is selected is always equal to one. In the situation of Figure 4.6, it is most likely that reassigning the pink marked container to Lift-AGV 13 is the most productive. Since that Lift-AGV has the longest idle time within the black rectangle. The black rectangle is drawn based on $time\_limit.AGV.advance$ and $time\_limit.AGV.after$ surrounding the pink marked container on the Lift-AGV stage. $time\_limit.AGV.advance$ and $time\_limit.AGV.after$ are not equal since it is more likely that bringing a container forward reduces delay instead of sending it further back.

A similar selection strategy could be applied to the YC stage. However, tests revealed that the quality of the obtained solution does not improve significantly within an equal number of iterations. Since the computational time did increase with respect to a random selection procedure, the predictive selection procedure is not used at the YC stage. The lagging performance at the YC stage could be attributed to the fact that the YC and the QC stage are not successive. This means that delay in the QC schedule is not a proper estimator for delays originating at the YC stage.

Repair algorithm

As said in Section 4.3.2, not every insertion of one container into another sequence of containers leads to a new feasible solution. This is because since there are no infinite buffers between different stages, it could be that two subsequent stages have containers sorted in different orders. Such a solution satisfies all equality constraints (constraints containing sequence variables only) but, is infeasible for the inequality constraints. Therefore, after an insertion of a container is performed on one stage, the other stages are repaired to ensure that precedence and blocking constraints are taken into account. This means that all containers which are present on two machines operating on subsequent stages should be handled in the same order. A sequence that requires reparation is shown on the left of Table 4.5. The sequence of operations on AGV1 conflicts with the sequence of operation on QC1. For this particular problem, the QC sequence is always leading when repairing another sequence. This is because the precedence relations on the QC stage completely define the sequence. On the other stages, it was chosen that the stage on which the insertion took place is leading when repairing another sequence.

AGV1	2	7	1	9	5
AGV2	6	3	4	8	10

AGV1	1	7	2	9	5
AGV2	6	3	4	8	10

QC1	1	2	3	4	5
QC2	6	7	8	9	10

QC1	1	2	3	4	5
QC2	6	7	8	9	10

Table 4.5: Left: A faulty sequence. Right: A repaired sequence. The sequence in the left is faulty since the order of containers handled on AGV1 is different from the order of handled containers on QC1

The type of required reparation indicated in Table 4.5 is most common when inserting containers to another position in the sequence. However, there is also another possible sequence conflict that could manifest when adjusting sequences. This conflict could occur whenever the sequence of handled containers is different over multiple stages. Therefore, this conflict has to do with multi-stage blocking. The conflict is best explained by an example (see Table 4.6). Each YC has access to two different racks, racks 1 and 2 for YC1 and racks 3 and 4 for YC2. Although the sequence of containers handled by both subsequent machines is valid, the complete schedule is not valid. This is because container 5 cannot be discharged to Rack4 as long as container 3 is not removed. But, the AGV is unable to remove container 3 since it is first waiting for container 1 which is handled after container 5 by the YC. This problem could be resolved similar to the previous conflict. However, it was chosen to ignore this possible conflict. This is because it is rather conservative to schedule all containers along every stage in the same order. There are many possible sequences that do not experience the described flaw whenever the sequence of containers is different along different stages. The flaw is quite rare and mainly manifests itself whenever an insertion is applied on really small problems as described in the example. Therefore, whenever a neighbour is suggested that contains this flaw it is simply ignored and a new neighbour is sought.

YC1	2				
YC2	3	5	1	4	

Rack1	2				
Rack2					
Rack3	1	4			
Rack4	3	5			

AGV1	1	2	3	4	5

QC1	1	2	3	4	5

Table 4.6: A faulty sequence of operations. The yard crane cannot start processing container 1 since there is no ability to discharge container 5.

Quality of an obtained sequence

As was indicated in Section 4.3.2, the sum of the arc lengths in the 'VRP' is not equal to the objective value. This is because a vehicle has to remain in a given city for a while (the processing time). Furthermore, the time of arrival in a city influences the time spend by another vehicle from a different stage in the same city. Either it has to wait because it is too early, or the vehicle cannot leave because the next vehicle is not ready. In other words, the objective value is strongly dependent on the starting time of each machine on each container since it influences the starting times on other stages. This has been tried to visualize with Figure 4.8. Therefore, to calculate the objective value; the complete problem, with exclusion of the sequence variables, should be solved once the quality of a new sequence is desired. Fortunately, this can be done rather efficiently. This is because when the sequence of operations is known, by insertion and reparation, the only unknown variables left are the continuous variables representing the starting times of different operations. Furthermore, each equality constraint is naturally satisfied whenever a feasible sequence was found. Therefore, to find optimal starting times corresponding to a given sequence, an optimization problem consisting solely of continuous variables and inequality constraints should be solved. An optimization problem that consists solely of continuous variables can be solved much quicker compared to problems consisting of both discrete and continuous

variables.
The remaining linear program was solved with both MATLAB's internal Linear Programming (LP) solver and with Gurobi's solver. It was found that Gurobi's solver was approximately five times as fast as MATLAB's solver. Therefore, Gurobi's LP solver was selected to solve the continuous LP problem. Gurobi's algorithm is very advanced and allows users to control many different settings. Whereas MATLAB's solver is much more closed off. Gurobi's default settings are generally very good and do not need to be changed unless there is a problem. Therefore, all settings remained at their default values. During future research; it is advised to better investigate the effect of tuning Gurobi's solver parameters. One of the features that speeds up Gurobi's solver is that it employs a presolver. Presolving means that the constraints and variables are examined and removed whenever possible [130]. This could be because they are redundant. Furthermore, the problem is properly scaled by presolving algorithms which improves the computational efficiency. However, the benefits of presolving scale with the quality of the initial model.

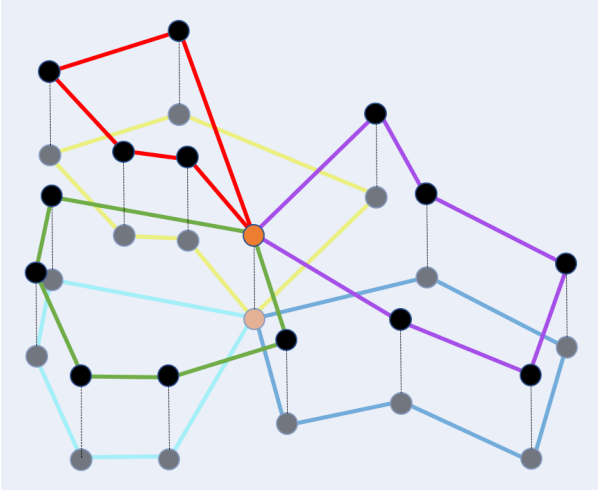


Figure 4.8: A visualization of the dependencies of different VRPs on two different stages. The pink, green and red sequences are on a different stage than the yellow, cyan and blue sequences.

Temperature schedule

Important when using SA is the temperature schedule. In essence, the temperature schedule defines the probability of accepting a worsening move. The presence of a probability to accept a worsening move is important since it offers the possibility of escaping a local minimum. The temperature schedule that was used during this research is shown in Figure 4.9. The initial temperature is set to 100. Each 7^{th} iteration the temperature is multiplied by 0.5 which means an exponential decrease in temperature. Each 50^{th} iteration a reheating with 75 degrees takes places. This means that the temperature attains a stable envelope whenever the temperature reaches 75 after reheating. This temperature schedule was chosen for two reasons. First, it is assumed that the initial solution is quite good so an investigation to that solution should start immediately. Whenever that in depth search is complete, a relatively new solution could be explored which is subsequently investigated. Besides that, the objective value could change very rapidly even after a single swap or insertion. Therefore, differences in objective value above 1000 are not uncommon. These solutions are nearly always discarded but, this illustrates the need for relatively high temperatures somewhere along the process. All constants that are used to specify the temperature schedule are based on the performance on a limited number of samples. Therefore, it is not excluded that there exist better temperature schedules. The constants that are used are displayed in Table 4.7.

Start Temperature	100
Multiplication	0.5
Reheat	75
Iterations till multiplication	7
Iterations till reheat	50

Table 4.7: Parameters used in the temperature schedule by the SA algorithm.

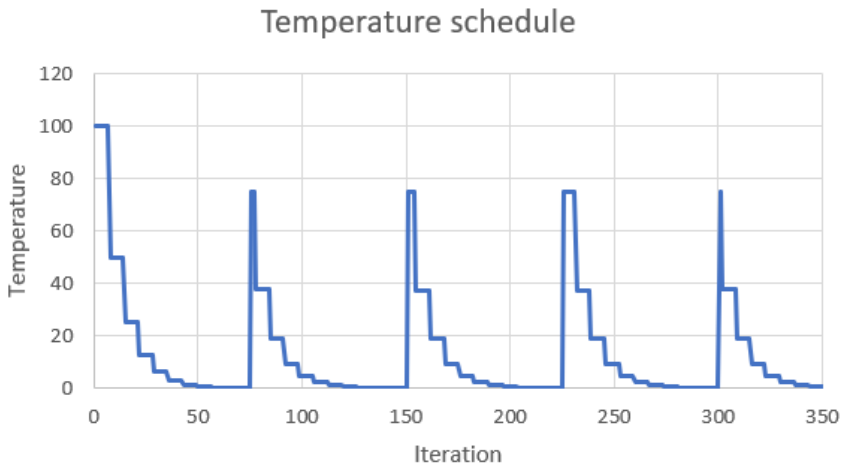


Figure 4.9: The temperature schedule used by the SA algorithm.

Stopping criteria

The SA process is stopped once a stopping criterion is encountered. Three different stopping criteria are present in the SA algorithm of this thesis. The first, most dominant, stopping criterion stops the optimization process as soon as a certain number of iterations are performed. To find a proper value for the maximum number of iterations; a study into the development of the objective value is performed. The study is based on the development in objective value for 40 different samples. The progress in objective value with respect to the number of iterations is shown in Figure 4.10. To compare performance of different samples, the objective values are scaled between 0 and 1. The highest attained objective value is set to 1 and the lowest attained objective value within 800 iterations is set to 0. Subsequently, intermediate objective values are scaled proportionally.

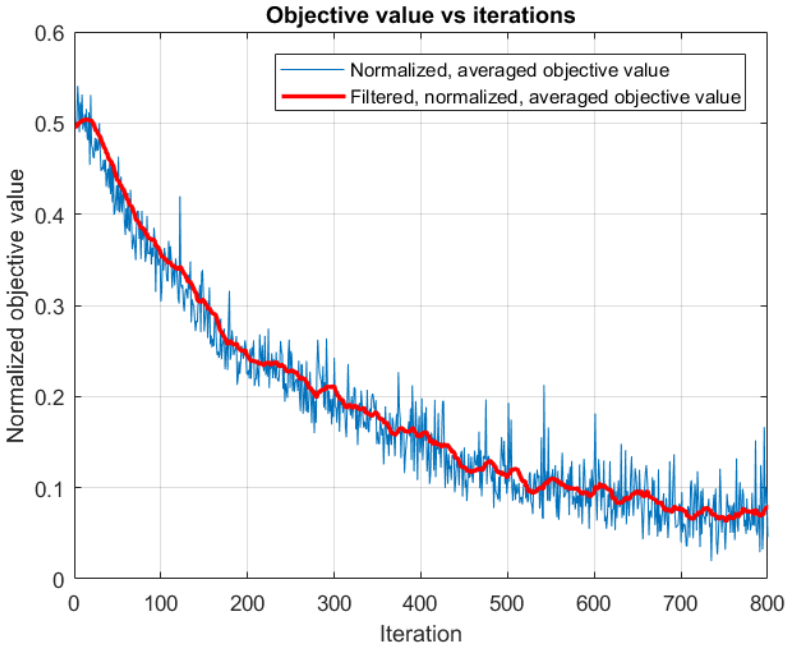


Figure 4.10: Averaged, normalized progress of the objective value of 40 different instances with 5 QCs.

As expected, the normalized averaged objective value shows high frequency behaviour. This is expected since the objective value could change rapidly even by only one swap/insert. Furthermore, also as expected, the low frequency behaviour approximates a negative exponential function. This is logical since gradually delays are removed focusing on the larger delays first.
To determine the maximum number of iterations, one should balance between the potential gain and the extra effort in terms of computational time. As can be seen in Figure 4.10, the objective value stabilizes between 700 and 800 iterations. This is at an objective value that is 6.5 times as low as the initial objective value. At approximately 550 iterations the objective value has decreased by a factor 5. The extra 150 iterations are considered too expensive in terms of computational time with regards to the potential gain. Therefore, 550 iterations is selected as the maximum number of iterations. However, it should be stressed that constraining the number of iterations is just a trade-off between performance and computational time.

Additionally, the SA iterative cycle is ceased once the objective value does not improve for 350 successive iterations (see Table 4.4). This happens rarely but, prevents the algorithm from spending too much time on cycling iterations.

Lastly, the SA algorithm is terminated when the QC schedule contains no delays anymore. Once the QC schedule can no longer be improved; the potential gain is considered insufficient. Therefore, the optimization can be stopped to save time.

4.3.3. Summary of the simulated annealing algorithm

Summarizing, as a solution technique a tailored variant of SA is selected. This is because B&B did not find solutions sufficiently quick. Therefore, an approximative solution technique is selected. This solution technique is SA since it balances well between performance and computational time. The different blocks of the SA algorithm are visualized with Figure 4.11. This figure highlights the separation of the different blocks of the algorithm. Furthermore, it indicates the relative position of each block with respect to the others. First, find and select an initial solution. Subsequently, select two containers on the same stage that are used to generate a neighbouring solution. If required, repair this neighbouring solution to ensure feasibility. As soon as the neighbouring solution is feasible, evaluate the quality with LP. Depending on the quality of the newly obtained solution, accept or reject the neighbouring solution.

Finally, reset and update the parameters to repeat the cycle. This process is repeated until a stopping criterion is encountered.

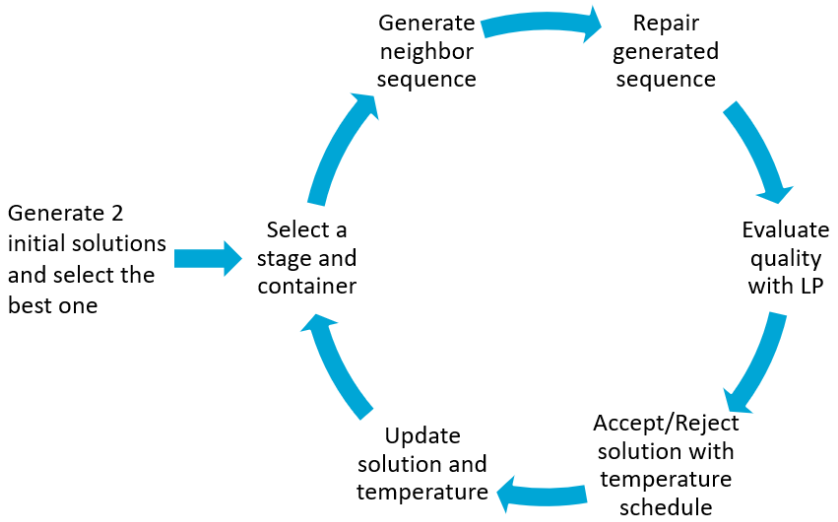


Figure 4.11: The SA algorithm visualized by breaking it up in successive blocks.

4.3.4. Algorithm performance

The performance of B&B and SA is compared in this section. The comparison is made for increasing problem sizes. For each problem size, only one sample is evaluated. The SA algorithm was repeated for each different sample with five different seeds to mitigate the effects of randomness. Besides that, the SA algorithm is also evaluated when more time is available. This obtained solution is called the better solution. To obtain the better solution the maximum number of iterations is increased to 1000 and the SA is repeated 10 times with different initial solutions. Subsequently, the best found solution is selected. The results of the comparison between B&B and the two SA variants are shown in Table 4.8. All of these experiments are performed with an Intel i7-7700HQ CPU @ 2.8 GHz processor. It can be seen that the optimality gap grows with increasing problem sizes. However, the terminal with two and seven QCs do not follow this pattern. This is because for certain samples the SA algorithm performs better than on others. This is not related to the terminal size but, to the characteristics of the container orders. Overall, the performance of the SA algorithm is good. Especially compared to performance of algorithms developed by other researchers (see Table 3.3). On top of that, the SA algorithm is quick compared to other research [19, 59–62, 64] (see Table 3.3). Therefore, this research delivered a new algorithm to solve HFS problems very efficiently (balance between performance and computational time). To achieve this performance, the number of iterations was set to 550 for each experiment. More iterations will lead to a lower optimality gap but also, a longer computational time.
The computational time of the SA algorithm is more stable compared to the B&B algorithm. It should be stressed that there is no polynomial formula to describe the computational time since the HFS problem is NP-hard [79].
It is possible to qualitatively say something about the expected computational time. Whenever B&B solution technique is selected, the computational time is dependent on the number of variables and the number of constraints. Where the number of variables has a bigger effect than the number of constraints [78]. After several experimental runs, it was found that the computational time is also correlated strongly with the average number of jobs per machine. Whenever a machine is very busy, the computational time increases significantly compared to an instance where machines are predominantly idle. This effect could be attributed to the fact that when there is enough idle time available, the consequences of swapping two jobs in the sequence are marginal or absent. This means that the solution converges quicker to the global optimum since there are lesser local minimums. Therefore, the global optimum is in a larger valley. Instead of a single gap. To arrive in a large valley is much easier compared to finding a single deepest gap within a rough landscape.

To considerably improve the computational speed of both the SA and the B&B algorithm. The buffer between the YC and the Lift-AGV could be assumed to have infinite capacity. This is a major simplification to the mathematical model since an entire stage, associated with the most machines, is removed. The computational time reduces since two complete sets of blocking constraints disappear. However, since buffers are not really infinite the model separates itself from reality. The simplification could be justified because benchmark runs revealed that machines did never have to wait for buffers to free up (see Section 2.3.4). Therefore, buffers seemed infinite since capacity was always available. For the SA algorithm specifically, the removal of the rack stage removes the requirement to repair YC and Lift-AGV sequences by each other. This reduces the computational time significantly. All results shown do consider infinite buffer capacity between the YC and the Lift-AGV.
To indicate the scaleability of the algorithm, results obtained on larger sized terminals are presented in Table 4.8. However, the comparison with B&B is not made since computational times of B&B solutions are too excessive.

Problem size				Branch and Bound			Simulated annealing quick solution					Simulated annealing better solution				
QC	AGV	YC	CNT	Objective value	Makespan (MS) [s]	Time [s]	Objective value	Makespan (MS) [s]	Time [s]	Opt gap	MS gap	Objective value	Makespan (MS) [s]	Time [s]	Opt gap	MS gap
1	4	3	10	7299.8	1568	0.1	7313.4	1568	1.3	0%	0%	7313.4	1568	21.7	0%	0%
2	8	5	20	19316.8	1784	0.9	20619.7	1971	1.5	7%	11%	19429.4	1816	23.9	1%	2%
3	12	8	30	40856.5	2857	16.3	42015.3	2932	1.8	3%	3%	41006.8	2857	30.5	0%	0%
4	16	10	40	41764.1	1960	71.1	42332.2	2008	2.1	1%	2%	42310.9	2008	29.7	1%	2%
5	20	13	50	50791.1	1910	2928.4	52672.1	1969	3.1	4%	3%	51517.2	1917	48.7	1%	0%
6	24	15	60	63987.5	2093	3762.8	66139.6	2169	2.5	3%	4%	65900.6	2169	43.1	3%	4%
7	28	18	70	69608.4	2373	7206.1	77214.0	2600	2.8	11%	10%	73937.1	2373	52.2	6%	0%
8	32	20	80	90604.23	2340	7206.1	95052.6	2392	3.3	5%	2%	93617.7	2340	57.6	3%	0%
9	36	23	90	-	-	-	102894.6	3330	3.8	-	-	101007	3330	64.3	-	-
10	40	25	100	-	-	-	104969.8	2173	4.4	-	-	103365.4	2141	69.6	-	-
11	44	28	110	-	-	-	110268.7	2369	4.6	-	-	109714.6	2369	78.0	-	-
12	48	30	120	-	-	-	131096.5	2819	4.9	-	-	128267.2	2634	84.3	-	-
13	52	33	130	-	-	-	142717.3	2687	5.4	-	-	140837.6	2468	100.5	-	-
14	56	35	140	-	-	-	164543.3	3012	6.0	-	-	163711.6	2787	108.7	-	-
15	60	38	150	-	-	-	159654.5	2423	6.7	-	-	157237.3	2397	117.0	-	-
16	64	40	160	-	-	-	169312.9	2589	7.5	-	-	164655.5	2550	135.8	-	-
17	68	43	170	-	-	-	192067.5	2599	8.9	-	-	190296.5	2599	148.6	-	-
18	72	45	180	-	-	-	212399.7	2566	9.5	-	-	206932.7	2420	169.3	-	-
19	76	48	190	-	-	-	218903.8	2854	10.0	-	-	213542.3	2816	178.4	-	-
20	80	50	200	-	-	-	214730.0	2434	11.3	-	-	209483.9	2404	194.2	-	-
21	84	53	210	-	-	-	245257.8	3011	12.5	-	-	239076	2995	216.6	-	-
22	88	55	220	-	-	-	244952.5	2299	14.4	-	-	237181.5	2163	254.0	-	-
23	92	58	230	-	-	-	288390.8	2966	14.8	-	-	278780.2	2930	282.4	-	-
24	96	60	240	-	-	-	302116.9	2777	21.0	-	-	297615	2609	398.9	-	-

Table 4.8: Optimization results for 8 different sized instances. The SA results have a quick solution (1 replication 550 iterations) and a better solution (10 replications 1000 iterations). The number of AGVs, YCs and Containers (CNT) is a function of the number of QCs. A - indicates that no results are available. This is because B&B did not find solutions within reasonable time.

In many cases the SA annealing algorithm is able to find a solution with the shortest makespan. However, this solution is not always considered as the best solution. This is because the objective function does not consist of makespan alone. Furthermore, there is a considerable improvement for the SA variants. However, also the time to arrive at the solution increased significantly. Therefore, for operational terminals it is advised to use the quick solution.

4.4. Concluding remarks

Concluding, a model is developed that integrates the operations of three different types of equipment operating on the waterside of a container terminal. These equipment types are YCs, Lift-AGVs and QCs. The aim was to construct a coordinated schedule that approximates optimal handling of containers. The model used to construct the coordinated schedule is an extended version of the HFS model. The extension lies in the two added features: bidirectional and job pairs. The former enables jobs to move through the HFS in both directions, the latter constrains certain jobs to be performed simultaneously by a single machine. Since the mathematical model is too complex to be solved with exact methods within a realistic time, an approximative solution technique is developed. This solution technique is based on SA. The exact solution is approximated by splitting the problem in a discrete and a continuous part. The discrete part uses a local search method to find better alternatives to the current solution by predicative reassignment of jobs to machines. Where predictive means that, with hindsight of the problem, it is

predicted which job reassignments are most likely to improve the current solution. The continuous part can subsequently be solved quickly with LP. Computational results prove that the developed solution technique is capable of finding solutions with relatively small optimality gaps within a fraction of the time that B&B requires even for very large terminals.

To proceed with the developed model and solution technique several future research directions are indicated:

1. The scheduling algorithm could be coupled to a real terminal to validate the schedule and to enable comparison with the current scheduling techniques applied by container terminals.
2. A library of HFS test problems is desired that allow for comparison of different solution techniques developed by multiple researchers.
3. The mathematical model could be extended with more features such as battery changes.
4. The mathematical model could be extended by integrating the landside of the terminal as well.
5. The solution technique could be updated by using dynamic settings for different parameters based on the problem at hand.
6. A new objective function that truly represents the container terminal could be formulated. When integrating this new objective function to the mathematical model, the solution technique should be updated as well.

For the remaining part of this research the focus will be at future work number one: integrating the scheduling algorithm with a real terminal. This is revolutionary with regards to container terminal scheduling since most other published scheduling algorithms [2, 19, 35, 56, 59–66] remain purely theoretical.

5

Simulation results

In this chapter we first describe how the mathematical model is integrated with the simulation model of TBA. Subsequently, we will present the results obtained with the simulation program when the coordinated schedule is integrated with TBA's simulation software. Three types of experiments are conducted. First, some small scale experiments are performed to analyze the performance of the coordinated schedule. However, more importantly: expose the differences between the mathematical model and the simulation model. Besides that, these small scale experiments helped to determine proper settings for the mathematical model in combination with the simulation model. These small scale experiments consist of small controllable instances that allow for exact logging. Second, controllable simulation experiments of 8 hours are performed to obtain statistical proof for claims based on the small scale experiments. Finally, large scale experiments are performed to compare performance of the uncoordinated scheduling heuristic and the coordinated schedule when applied to a random realistic terminal. Once all these results are obtained, comments on the research are presented and a preliminary conclusion is drawn.

5.1. Coupling between eM-Plant and MATLAB

The simulation model of TBA is built in a simulation program called eM-Plant. This program is unable to perform linear programming or to perform quick calculations on large matrices. Therefore, MATLAB is used to construct and solve the mathematical model. Besides that, there is personal preference for using MATLAB since I was more familiar with that program. The disadvantage of using two different programs is that communication needs to be established between the different programs. MATLAB and eM-Plant do no support a direct communication between each other. Therefore, a third program should be used to store data which both eM-Plant and MATLAB can access; to read and to write. A SQL database is selected to do this since: it supports all required datatypes, is accessible by both eM-Plant and MATLAB and allows quick access.

5.1.1. Communication between eM-Plant and MATLAB

The communication protocol is designed as follows: eM-Plant is ordered to generate a table that contains all the required input data. Subsequently, eM-Plant writes this data to the SQL database. In the mean time, MATLAB is waiting for updates to the SQL database. Whenever an update is registered, MATLAB constructs a schedule based on the imported data. Once the construction of a schedule is complete, a table is generated and stored in the SQL database with output data. This data is accessible by eM-Plant whenever required. A summarized example of all input and output tables is presented in Tables 5.1, 5.2 and 5.3. To keep track of the updates, a counter of the update is implemented. Each time something is stored in the database, the counter is increased by one. To ensure that no data is lost, eM-Plant is only allowed to access the database whenever the counter is even. MATLAB is granted access to the database whenever the counter is odd. Because of this extra check, it is guaranteed that the output of each program arrives at the other program.

order	container_id	YC	AGV	QC	order_type	sequence_no	sequence_key	processing_YC	processing_AGV	processing_QC	twin_pair	current_position	current_progress	time_remaining
QC1_1_37	CID_15271	Module11	AGVL05	QC1	loading	37	QC1_1	0.0	207.9	41.7		AGVL05	TRUE	186.51
QC1_1_38	CID_12933	Module13	AGVL20	QC1	loading	38	QC1_1	0.0	185.5	41.0		AGVL20	TRUE	87.53
QC1_1_39	CID_17120	Module07	AGVL09	QC1	loading	39	QC1_1	0.0	114.1	40.3		AGVL09	TRUE	124.13
QC1_1_44	CID_12928	Module13		QC1	loading	44	QC1_1	0.0	201.7	76.5		Module13	FALSE	
QC1_1_45	CID_5703	Module06		QC1	loading	45	QC1_1	80.4	118.4	76.1		Module06	FALSE	
QC1_2_1	CID_7909	Module07		QC1	loading	46	QC1_2	91.6	130.3	60.0		Module07	FALSE	
QC2_1_37	QC2_1_D_15	Module13	AGVL10	QC2	unloading	37	QC2_1	131.5	168.8	0.0		AGVL10	TRUE	149.09
QC2_1_38	QC2_1_D_36	Module01	AGVL18	QC2	unloading	38	QC2_1	116.3	116.0	0.0		AGVL18	TRUE	123.75
QC2_1_39	QC2_1_D_31	Module10	AGVL03	QC2	unloading	39	QC2_1	113.1	215.0	54.3		QC2	TRUE	51.80
QC3_2_1	CID_6722	Module04	AGVL16	QC3	loading	26	QC3_2	0.0	30.0	50.9	QC3_2_2	AGVL16	TRUE	55.91
QC3_2_2	CID_10565	Module04	AGVL16	QC3	loading	26	QC3_2	0.0	30.0	50.9	QC3_2_1	AGVL16	TRUE	55.91
QC3_2_3	CID_14450	Module12	AGVL02	QC3	loading	27	QC3_2	0.0	30.0	51.5	QC3_2_4	AGVL02	TRUE	58.25
QC3_2_4	CID_11604	Module12	AGVL02	QC3	loading	27	QC3_2	0.0	30.0	51.5	QC3_2_3	AGVL02	TRUE	58.25

Table 5.1: A short example of the data exported by eM-Plant to the SQL database.

In Table 5.1 all the known order information is combined. The first two columns show the respective order and the container-id. The next three columns state the planned vehicles. Subsequently, the order type, the sequence number and the sequence key specify the order of operations along the different machines and while processing the ship. The processing times define the times that a container should spend at a given stage. In the column twin-pair, the twin pair of the respective container is mentioned. Finally, the last three columns provide information concerning the current status of the container.

AGV	order	location	time2grab	idle
.Models.RMG_AGV_Lift.AGVL01	QC2_1_4	WS03		FALSE
.Models.RMG_AGV_Lift.AGVL02	QC2_1_2	QC2		FALSE
.Models.RMG_AGV_Lift.AGVL03	QC3_1_3	WS09		FALSE
.Models.RMG_AGV_Lift.AGVL04	QC5_1_9	QC5	30	FALSE
.Models.RMG_AGV_Lift.AGVL05	QC2_1_3	QC2		FALSE
.Models.RMG_AGV_Lift.AGVL06	QC1_1_5	QC1		FALSE
.Models.RMG_AGV_Lift.AGVL07	batterychange	BatCharge1	103.63	TRUE
.Models.RMG_AGV_Lift.AGVL08	QC1_1_7	QC3	42.97	FALSE
.Models.RMG_AGV_Lift.AGVL09	QC5_1_7	QC5		FALSE
.Models.RMG_AGV_Lift.AGVL10	QC4_1_7	WS08		FALSE
.Models.RMG_AGV_Lift.AGVL11	QC4_1_8	WS04		FALSE

Table 5.2: The information of the different AGV statuses generated by eM-Plant.

In Table 5.2 the current status of the AGVs is presented. This information is required to determine the time required to reach the pickup position of the first order. In the first column, the respective AGV is listed. In the second column, the order that is currently claimed by that respective AGV is stated. This could be while the container is loaded on the AGV or while the AGV is driving to the pickup location of the container. The location signifies the location on the terminal of the current destination of the AGV. In the column 'time2grab' the time required for the AGV to finish grabbing a container is presented. Therefore, this time is only communicated when the AGV carries no container at the time of constructing the table. When the AGV is in the battery station, the time required to exit the battery station is presented here. The last column indicates whether an AGV is assignable to any order or whether it is currently reserved to complete another order first.

AGV	Order 1	Order 2	Order 3	Order 4	Order 5	Order 6	Order 7
AGV1	QC1_2_11	QC1_2_12	QC1_2_13	QC1_2_14	QC1_2_23		
AGV2	QC1_2_24	QC1_2_27	QC1_2_28				
AGV3	QC2_1_36	QC2_1_37	QC2_1_38	QC2_1_39			
AGV4	QC1_2_29	QC1_2_30	QC2_1_35				
AGV5	QC2_1_40	QC2_2_1	QC2_2_2	QC2_2_3	QC2_2_4	QC3_1_34	
AGV6	QC3_1_35	QC3_1_36	QC3_1_37	QC3_1_38	QC3_1_39		
AGV7	QC3_1_40	QC3_2_1	QC3_2_2	QC3_2_3			
AGV8	QC4_1_25	QC4_1_26	QC4_1_27	QC4_1_28	QC4_1_29		
AGV9	QC4_1_30	QC4_1_31	QC4_1_32				
AGV10	QC4_1_33	QC4_1_34	QC5_1_15	QC5_1_16	QC5_2_17	QC5_2_18	
AGV11	QC5_2_19	QC5_2_20	QC5_2_21	QC5_2_22			

Table 5.3: The produced output generated by MATLAB and exported to eM-Plant for the AGV stage.

In Table 5.3 a possible output of the MATLAB program is presented. In the first column, a specific vehicle is mentioned. In the columns to the right, the sequence of orders for that vehicle is presented. The orders correspond to the orders imported from eM-Plant as in Table 5.1. A similar table is generated for the other stages.

5.1.2. Order assignment

To ensure that eM-Plant obeys the schedule generated with MATLAB; a job assignment process was implemented. This process consists partly of TBA's job assignment described in Figures 2.11 and 2.10. The exclusion procedure remains intact, only the scoring mechanism is changed. An order, remaining after the exclusion process, receives a huge bonus if it is the first order in the newly generated sequence of operations received from the MATLAB program. This ensures that the first order scheduled for a vehicle receives the highest bonus and is therefore selected. The exclusion process is maintained because this offers robustness and a validated order selection. Therefore, some of the intelligence currently available in the eM-Plant model is used.

5.1.3. Parameters introduced by coupling eM-Plant to MATLAB

By coupling eM-Plant and MATLAB, several parameters need to be specified. These parameters are: the horizon length and the updating frequency. The horizon length determines the period for which the schedule is generated. The longer this period, the more computational effort. However, further horizons potentially offer a more optimal planning. The horizon length in itself is a bit fictitious since at the start of the scheduling process; it is unknown how long an order takes. Therefore, the cumulative sum of the processing times at the QC is used to determine which containers are contained within the horizon. Regarding the updating frequency, the total computational time required to generate schedules increases with increasing frequency. However, there will be a more accurate representation of the current situation at the terminal once a higher updating frequency is used. The used horizon length and updating frequency are indicated in Table 5.4.

	Value
Horizon length [s]	800
Updating frequency [Hz]	0.1

Table 5.4: Values assigned to parameters required to couple eM-Plant with MATLAB.

5.1.4. AGV interference

As was introduced in Section 1.1, the coordinated planning of this thesis considers scheduling and dispatching but, it ignores routing. It is expected that this will introduce a big difference between the planned schedule and the realised schedule. This is because without considering routing, the terminal is prone to congestions which drastically increase processing times of Lift-AGV jobs. Therefore, the estimated drivetimes based on the distance between origin and destination are compared with the realised drivetimes during simulation. The result of this experiment is presented in Figure 5.1. As can be seen, for the majority of the samples, the estimated drivetime was too optimistic. Especially on

shorter paths the estimated drivetime is too short, whereas on longer paths the estimated drivetime becomes pessimistic. The realised drivetime is obtained by using the average of over 100 samples for each origin-destination pair. Unfortunately, due to memory issues and SQL limitations, only the average could be obtained. Ideally, individual logs are desired such that standard deviations and distributions could be estimated. However, since this data is not available; averages are the best estimates. After obtaining Figure 5.1, the realised drivetime is used to estimate the processing time of the AGVs instead of the estimate based on the distance. The disadvantage of this approach is that data needs to be acquired of the specific terminal before the algorithm could be used. Therefore, to make the algorithm more robust the algorithm would benefit from a more accurate prediction of the driving times without using terminal specific information. Another option would be to dynamically update the estimated drivetime along with the simulation.

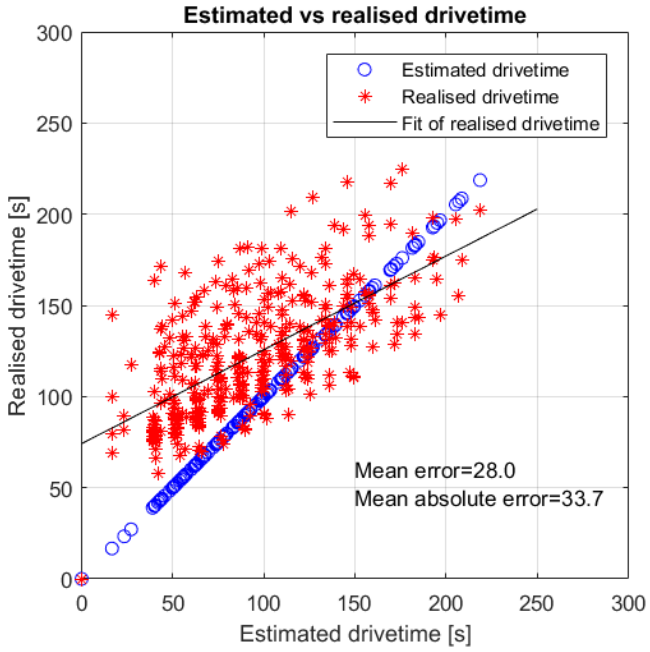


Figure 5.1: The difference between estimated and realised drivetime.

5.2. Small scale experiments

In this section we clarify what is meant by a small scale experiment and why these are performed and with what settings. Subsequently, results of these small scale experiments are presented. Finally, an extensive analysis into these results is performed.

Small scale experiments are experiments that span a specific set of containers. This set of containers spans all containers that are available within one horizon length. Small scale experiments allow users to compare performance of different settings or strategies on the same controlled environment. This is because, during these small scale experiments, data could be gathered and logged on a detailed level. The set of containers within one horizon length is called an instance. Therefore, small scale experiments are used to asses the performance during these instances. The process of generating an instance is described with Figure 5.2.
First, the simulation is started and uses TBA's uncoordinated scheduling heuristic. At a random point in time (P_0); the simulation is halted. Subsequently, data is exported to MATLAB and a schedule is constructed. Thereafter, the simulation is performed twice for all containers in the current plan. The first simulation applies TBA's uncoordinated scheduling heuristic and the second simulation applies the coordinated schedule. When desired, more duplicates of the same instance could be evaluated with different settings/strategies. By the time that all containers are processed, the realised execution plans can be compared to the theoretical plan made at P_0 and to each other. Figure 5.2 shows that when

applying the coordinated schedule, the schedule is generated repetitively. This is because input data may change during simulation. With a control layer the schedule is able to anticipate to these input changes by adjusting the schedule designed at P_0.

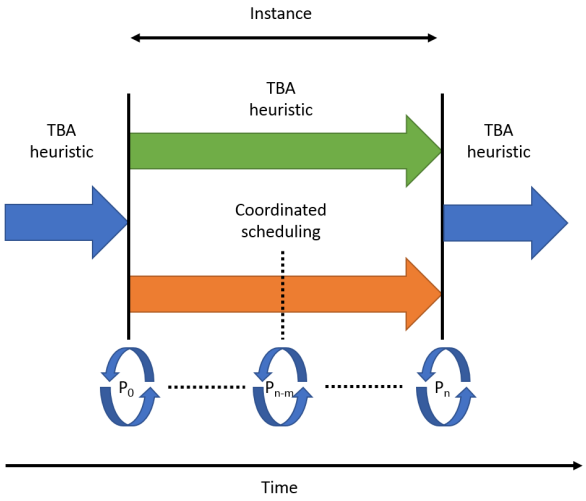


Figure 5.2: Description of an instance with respect to time.

It is stressed that the purpose of small scale experiments is: to see the effect of different settings and strategies, to analyze the similarities and differences between the mathematical model and the simulation model and to indicate which scenarios are likely to produce positive/negative results when used in a larger experiment. The purpose of small scale experiments is not to accept/reject the hypothesis that a coordinated schedule is better than TBA's current uncoordinated scheduling heuristic.

Prior to generating an instance, a scenario should be selected. Since analysis is one of the purposes of small scale experiments, it was decided to simplify the terminal such that the QCs all have the same function (loading, unloading, twinlift loading and twinlift unloading). This ensures that during each instance there is sufficient data regarding a QC function to analyze. Moreover, this enables users to predict which of the QC functions is likely to benefit from a coordinated schedule. Besides that, the size of the terminal is maintained constant and the terminal employs Lift-AGVs.

5.2.1. Results of small scale experiments

Results of the small scale tests are presented in this section. In the rightmost column of Tables 5.5 to 5.8 the balance in performance between the TBA realisation with the uncoordinated scheduling heuristic and the realisation with the coordinated schedule is indicated. Whenever the balance is positive, the coordinated schedule outperformed TBA's uncoordinated scheduling heuristic and vice versa. Besides that, all instances presented in Tables 5.5 to 5.8 are completely independent of each other.

Loading	Nr. Lift-AGVs	TBA realisation [box/hr]	Planned productivity [box/hr]	Coordinated realisation [box/hr]	Balance [box/hr]
Instance 1	3	23.2	21.9	24.1	0.9
Instance 2	3	32.4	32.6	31.5	-0.9
Instance 3	3	30.6	30.6	30.7	0.1
Instance 1	4	28.3	23.9	28.7	0.4
Instance 2	4	30.5	32.3	31.9	1.4
Instance 3	4	34.2	35.5	34.1	-0.1
Instance 1	5	25.0	24.6	25.4	0.4
Instance 2	5	27.5	21.0	27.3	-0.2
Instance 3	5	35.5	36.6	35.2	-0.4
Average	3,4,5				**0.2**
Average	4				**0.6**

Table 5.5: Results of small scale experiments on a terminal with 5 QCs and a variable number of Lift-AGVs. All instances consist solely of loading operations.

Unloading	Nr. Lift-AGVs	TBA realisation [box/hr]	Planned productivity [box/hr]	Coordinated realisation [box/hr]	Balance [box/hr]
Instance 1	4	32.8	31.9	33.7	0.9
Instance 2	4	32.0	32.1	32.1	0.1
Instance 3	4	34.3	31.7	33.9	-0.3
Average					**0.2**

Table 5.6: Results of small scale experiments on a terminal with 5 QCs and a fixed number of Lift-AGVs. All instances consist solely of unloading operations.

Twinlift loading	Nr. Lift-AGVs	TBA realisation [box/hr]	Planned productivity [box/hr]	Coordinated realisation [box/hr]	Balance [box/hr]
Instance 1	4	35.7	34.4	37.9	2.2
Instance 2	4	26.0	28.6	27.1	1.1
Instance 3	4	42.1	39.5	45.9	3.8
Average					**2.4**

Table 5.7: Results of small scale experiments on a terminal with 5 QCs and a fixed number of Lift-AGVs. All instances consist solely of twinlift loading operations.

Twinlift unloading	Nr. Lift-AGVs	TBA realisation [box/hr]	Planned productivity [box/hr]	Coordinated realisation [box/hr]	Balance [box/hr]
Instance 1	4	35.3	56.5	36.1	0.8
Instance 2	4	41.2	56.7	40.6	-0.6
Instance 3	4	41.9	55.5	38.6	-3.3
Average					**-1.1**

Table 5.8: Results of small scale experiments on a terminal with 5 QCs and a fixed number of Lift-AGVs. All instances consist solely of twinlift unloading operations.

All of the results presented above in Tables 5.5 to 5.8, are also summarized and visualized with a graph presented in Figure 5.3.

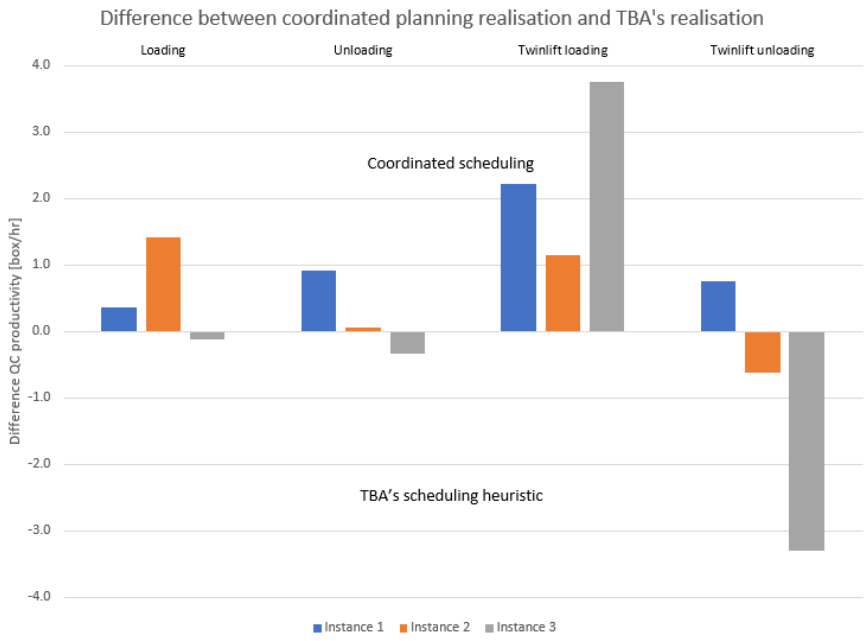


Figure 5.3: Difference in performance between the realisation using TBA's uncoordinated scheduling heuristic and the realisation using the coordinated schedule.

5.2.2. Analysis of results

Based on the results presented in Tables 5.5 to 5.8, it can be concluded that the coordinated scheduling algorithm does not lead to the expected increase in performance for each QC function during every instance. Nor does the realised productivity match the planned productivity. This is attributed to the discrepancy between the mathematical model and the simulation model. This discrepancy manifests itself in a mismatch between assumed and realised time for different actions and some simplifications made within the mathematical model. The mismatches can be divided into two different categories. The first category is related to the pure stochastic nature of certain operations. The second category is the simplification of the mathematical model with respect to the simulation model. Since the mathematical model is deterministic, both category problems are tried to be resolved by assuming a fixed time or ignoring certain operations entirely. The differences between the two categories is that the effects of the mismatches of the first category can only be reduced by using a stochastic model whereas mismatches of the second category could be resolved by using a more complex deterministic model. In Table 5.9 several mismatches between the mathematical and the simulation model are categorized. These mismatches are sorted from largest to smallest effect on the performance of the coordinated schedule.

Pure stochastic mismatches	**Mathematical simplification mismatches**
Interchange time of containers	AGV interference
Twistlock handling	Availability of buffer positions
Arrival pattern of trucks	Container weight
	Dynamics of the equipment
	Position of containers in the ship
	Setup times of the YC
	Battery changes
	Interference with landside

Table 5.9: Differences between realised and estimated times subdivided into two categories and sorted in order of largest to smallest effect on the difference in performance between the theoretical plan and the realised plan.

Within the simulation model, all pure stochastic mismatches occur because a random number is

drawn from a distribution. This is desired in the simulation model since in the real world there is some variation to these values as well. When sticking to a deterministic model, there is no better estimate to these values than assuming the average of the probability distribution. Therefore, mismatches related to the simplification of the mathematical model with respect to the simulation model are far more interesting to investigate.

Interference of AGVs

The most dominant of all mismatches between simulation model and mathematical model is the AGV interference. Since multiple Lift-AGVs share the road between the stacking yard and the QCs, it is unavoidable that these Lift-AGVs interfere with each other. This means that sometimes a Lift-AGV has to wait for another Lift-AGV to avoid collisions. This waiting time introduces the mismatch between the mathematical model and the simulation model. The severity of this mismatch is identified with Figure 5.4. Figure 5.4 shows the probability distribution of the difference between the estimated and the realised processing times at the Lift-AGV stage for different numbers of Lift-AGVs per QC. Figure 5.4 shows that even when the processing times obtained by the method described in Section 5.1.4 are used, the realised processing time is very variable. Furthermore, it shows that the more Lift-AGVs operate on the terminal, the more interaction there is, the harder it becomes to predict the processing times. Although this effect is small. This is because when there are more Lift-AGVs on the terminal, there are not necessarily more Lift-AGVs on the road. This seems controversial but, with more Lift-AGVs on the terminal, more Lift-AGVs are stationary in the buffer not causing interference. Moreover, the probability distribution is a little skewed to the right since it is more likely that a Lift-AGV has to wait very long instead of travelling much faster than expected.
A similar effect can be seen in Figure 5.5. This figure shows the probability distribution of the difference between the estimated and the realised processing time for each QC function. Figure 5.5 shows that loading operations can be predicted with the highest accuracy. This is because during loading there is minimal activity on the Lift-AGV road. Furthermore, unloading operations can be predicted quite accurately as well. This is because during the stack assignment, the crowdedness at a particular module is considered as well. This results in an even distribution of workload over the different stack modules which reduces the likeliness of interference. On the other hand, some unloading operations have an unknown destination at the moment of obtaining the instance. These containers are scheduled to the center of all modules until a destination is known. Therefore, some unloading operations cannot be predicted with the same accuracy. On top of that, each unloading operation requires a doorturn. This introduces more activity on the Lift-AGV highways increasing the chances of interference. Therefore, some unloading operations show long delays. Twinlift operations are also harder to estimate compared to their single variants since there is more traffic on the terminal. This seems controversial but, this is due the requirement of doorturns whenever the two containers of the twin pair need to be picked up at different locations. And because heavier loaded vehicles claim a larger space on the terminal while driving slower, increasing the chance of interference with other vehicles.

To find out whether it is better to overestimate or underestimate the Lift-AGV drive time, small scale experiments for loading containers were performed. The results of these experiments are shown in Table 5.10. These experiments indicated that the used instances were not very sensitive to changes to the Lift-AGV drivetime. However, the highest productivity was achieved when the drivetime is estimated as accurately as possible. Therefore, all other experiments were performed without deliberately under or overestimating the Lift-AGV drivetime.

	Realised QC productivity [box/hr]		
Factor drivetime	**0.85**	**1.00**	**1.15**
Instance 1	33.3	33.0	32.9
Instance 2	32.9	33.5	32.9
Instance 3	34.9	35.0	34.5
Average	**33.7**	**33.8**	**33.4**

Table 5.10: The effect of under/overestimating the Lift-AGV drivetime for a terminal with 5 QCs and 20 Lift-AGVs in case of pure loading.

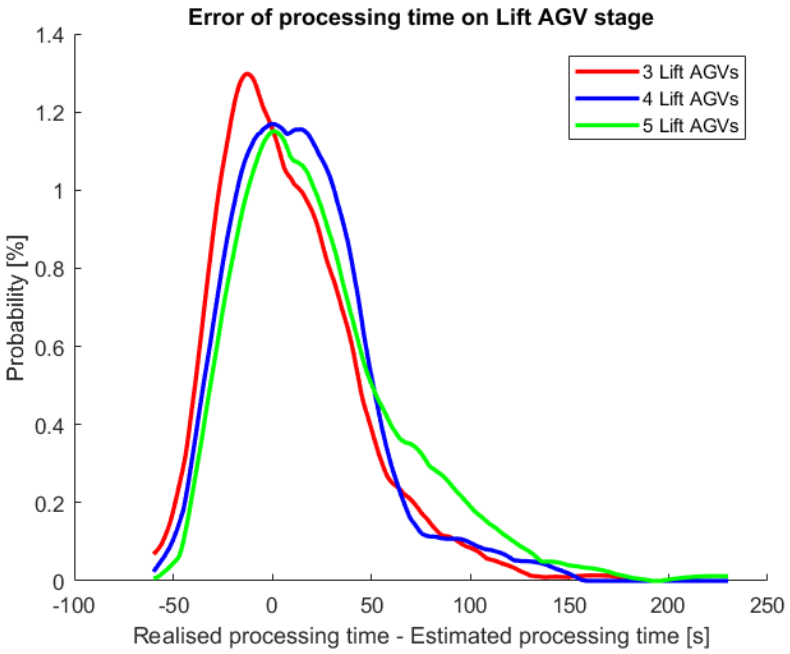


Figure 5.4: The probability distribution of the mismatches between realised and estimated processing times of Lift-AGVs in case of loading.

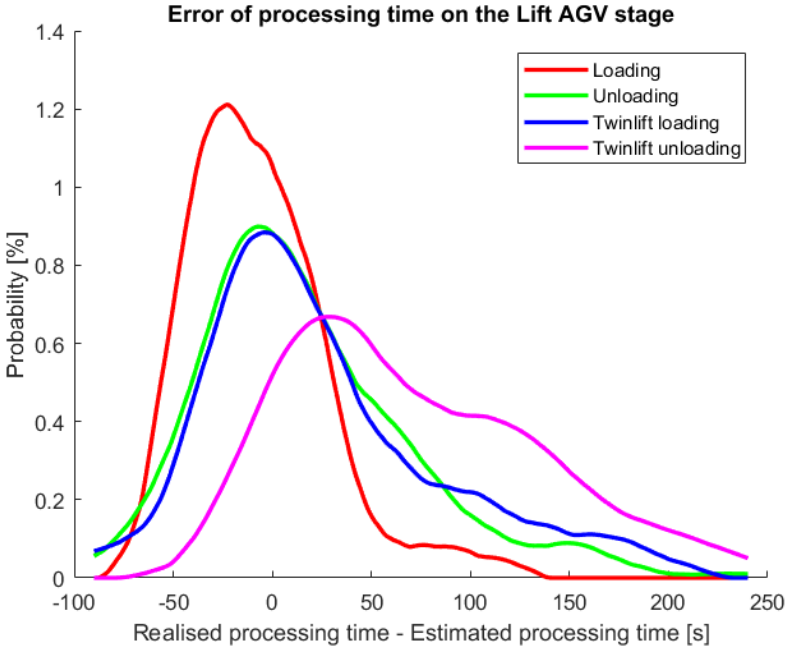


Figure 5.5: The probability distribution of the mismatches between realised and estimated processing times of Lift-AGVs on a terminal with 5 QCs and 20 Lift-AGVs.

When a more complex model would be used, the routing of different Lift-AGVs could be considered during the scheduling phase. This would eliminate the mismatch between the mathematical model and the simulation model and therefore lead to a more accurate planning. Since this is such a common scheduling problem, the routing problem is studied often [18, 45, 46]. These researches aim to resolve the routing problem separate from the scheduling problem. This could be advantageous since it is far

less complex compared to a fully integrated routing module to the scheduling and dispatching model. Nevertheless this has also been studied in for example [47, 48].

Availability of buffer positions

Another mismatch related to the simplification of the mathematical model with respect to the simulation model is the availability of buffer positions. Currently, the mathematical model does not consider the buffer positions. What is meant by buffer positions are areas where Lift-AGVs are allowed to stand still. Either carrying a container or being idle. This means that within the mathematical model, Lift-AGVs are assumed to be at the position of their last completed task while they are actually in the buffer. Lift-AGVs should move to the buffer since otherwise other Lift-AGVs could not reach the location they are currently at. Therefore, Lift-AGVs are directed to a buffer awaiting a travel order. Since there are multiple buffer positions, the time to reach the buffer and to reach the next destination is no longer accurate. Le et al. [131] recognize four different policies to handle this buffer problem. Furthermore, references to previous developed models are given. These models could be integrated with the mathematical model developed in this research.
Besides that, in the mathematical model the rack stage between the YCs and the Lift-AGVs was removed to increase the computational speed significantly. However, this is also a major simplification to the model. This explains the large difference in performance between the current TBA realisation, the coordinated scheduling plan and the realised performance with the coordinated schedule for twinlift unloading. When racks are fully occupied, the current scheduling algorithm incorporates this by withholding the order from being assigned. Since the mathematical model does not incorporate this, Lift-AGVs spend more time waiting for racks to free up. Moreover, the time of assigning an order is important, the later, the better. This is because this leads to a more accurate representation of the crowdedness and the occupancy of the rack. Unfortunately, the mathematical model experiences difficulties with unknown grounding locations. Therefore, grounding locations are determined further in advance. This also explains the difference between twinlift unloading and single unloading, with regards to the Lift-AGV processing times (see Figure 5.5), the number of Lift-AGV moves that already possess a destination is larger with single unloading. This is because the container location of 5 containers is planned in advance. This results in 5 single moves but only 2.5 twin moves. Additionally, since YCs empty the racks one by one and the rack is filled with two containers at a time, the racks get full more frequently with twinlift unloading compared to single unloading.
However, on a real terminal it is unlikely that all QCs are twinlift unloading. Therefore, there will be less pressure on the YCs. Furthermore, Lift-AGVs will help with emptying racks if there are some loading operations as well.

Dynamics of equipment

The third predictable mismatch, between simulation model and mathematical model, are the dynamics of the equipment. Currently, the average speed of the equipment is used to calculate the time required to reach another location. However, equipment does not work with average speed, there is constant acceleration and deceleration. Furthermore, especially for Lift-AGVs, interference with other equipment and extra curves lead to disturbances to the average speed. On top of that, sometimes Lift-AGVs perform crab movement. This means that the Lift-AGV travels sideways, leading to totally different dynamics.

Container weight

In order to incorporate dynamics of equipment, container weight should be considered as well. This is because the weight of a container determines the acceleration of equipment. Furthermore, the speed of equipment could be dependent on the weight. For instance, the spreader hoist speed of a QC is reduced for heavy containers. With a more accurate prediction model, container weight could be incorporated to determine processing speeds of equipment more accurately.

Position in the ship

Currently, the mathematical model does not consider the position on the ship of a container. This means that the QC assumes an equal processing time for each container within a bay. However, when the container position on the ship is further from the quay or at a lower tier in the hold, the trolley and spreader need more time to reach this position.

In Figure 5.6 the probability distribution of the difference between estimated and realised processing times at the QC stage is shown. When containers are placed on the deck of a ship, twistlocks need to be attached to prevent the containers from falling off the ship. During discharge, these twistlocks need to be removed as well. Therefore, containers above deck require more processing time at the QC stage compared to containers in the hold. This possible extra processing time is disregarded within the mathematical model. Therefore, a gap between the simulation and the mathematical model is formed. The effect of the different processing times for containers in the hold and on the deck explains the bumps to the right of the main peak in Figure 5.6. With more detailed input, with respect to container position on the ship, the processing times at the QC stage can be estimated more accurately.

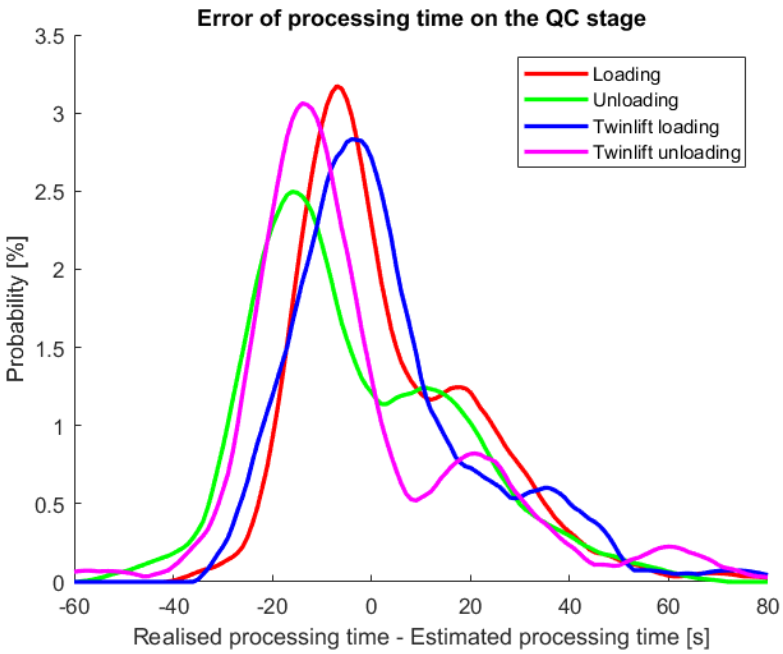


Figure 5.6: The probability distribution of the mismatches between realised and estimated processing times of QCs on a terminal with 5 QCs and 20 Lift-AGVs.

Setup times of the yard crane

Within the mathematical model, the setup time of the YC is estimated based on the following rules. If two moves of the same type succeed each other, the setup time is equal to the processing time of the first move. This prediction is based on the the fact that succeeding containers are stored near each other within the stack. Whenever a (twin)loading container succeeds an (twin)unloading container, a fixed intermediate setup time is assumed since the YC requires to move within the stack. Whenever an (twin)unloading container succeeds a (twin)loading container, a fixed short setup time is assumed since the YC does not need to move because both containers are located on the rack. Within the simulation model; these assumptions might not be valid. However, it would require too much communication to communicate a complete setup table for every possible container combination. Therefore, the mathematical model cannot access the more accurate predictions. Besides that, not every setup time could be predicted in advance since the location of each container within the yard and the rack is not known in advance. This mismatch explains why the planned productivity does not match the realised productivity when the coordinated schedule is applied. Accompanying the inability to accurately predict the setup time, it is also unknown to the mathematical model how long a YC will be busy with the setup when the schedule is made somewhere halfway the setup time.

Battery changes

Within the simulation model, Lift-AGVs do not have infinite battery capacity which means that they require to swap batteries from time to time. The mathematical model does not possess this information and is only notified that a Lift-AGV is in the battery station whenever it arrives there. From the moment

that a Lift-AGV has a status related to a battery swap the time until the Lift-AGV becomes available again is predicted and incorporated in the scheduling algorithm. When the battery status is transferred to the mathematical model as well, a more accurate prediction of the Lift-AGV availability can be made.

Interference with the landside

Within the simulation model; both land- and waterside operations exist. Whereas within the mathematical model only the waterside of the container terminal is considered. The landside of the container terminal is served from the same stack as the waterside but, with another YC. This leads to disturbances to the waterside YC since the landside YC could be occupying a certain area of the yard. Furthermore, sometimes a container arriving by truck is required at the waterside. This requires arrival of this truck before the container could be transshipped. The landside operations could be integrated with the current mathematical model. Jung et al. [70] proposed a model where multiple cranes are working within the same yard block. In Figure 5.7 the probability distribution of the difference between estimated and realised processing times at the YC stage is shown. The things that introduce mismatches are landside operations, availability of buffer positions, attaching the spreader to the container and the dynamics of the crane. The unload operations are more variable compared to loading operations. This is because when importing the instance not all positions of unloading containers in the stack are known. Therefore, a rougher estimate is made for these containers. When time progresses, the location of these containers becomes known exactly and by then the processing time can be predicted more accurately.

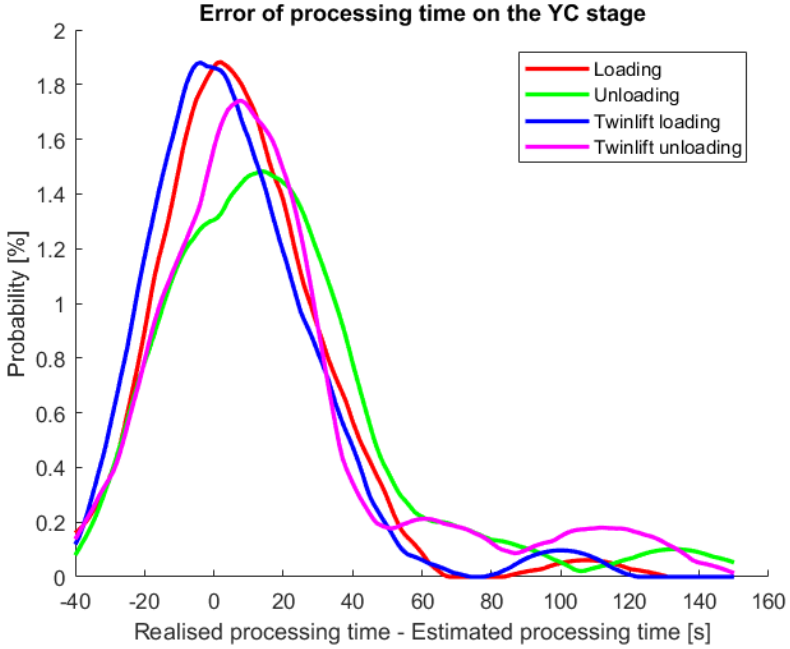


Figure 5.7: The probability distribution of the mismatches between realised and estimated processing times of YCs on a terminal with 5 QCs and 20 Lift-AGVs.

Other reasons for lack of performance

What strikes first when observing the results presented in Tables 5.5 to 5.8 is that especially twinlift unloading performs very badly. On top of the mismatches highlighted in Table 5.9 and in particular the availability of buffer positions discussed in Section 5.2.2, there are two more related things that impair the performance of the coordinated schedule with respect to twinlift unloading. These are the length of the horizon with relation to the number of containers on the terminal and the used number of iterations in the SA algorithm. During twinlift unloading, the most containers are present in the planning. This is because the horizon length is based on the cumulative processing times of operations at the QC. However, during unloading, containers are not removed from the planning whenever processed by the QC. Instead they have a processing time of zero seconds. The containers are only removed when they arrive in the stack. Therefore, the number of containers to be scheduled easily reaches four times

as many as for single loading. With more containers present, more iterations of the SA algorithm are required as well. However, it was chosen not to increase the number of iterations for the small scale experiments since they are only used to predict performance on the larger experiments, and to analyze the performance. Ideally, the number of iterations in the SA algorithm is a function of the number of containers.

Differences in performance per QC function

In the previous sections the differences between the mathematical and the simulation model were indicated, categorized and explained. In this section we will clarify why certain QC functions perform better compared to others. This is done based on Table 5.11. In this table several aspects that influence the performance of the coordinated schedule are listed in the first column. In the second column the severity of these aspects is reflected with a weight factor. In the subsequent columns a score is given for each QC function. A high score means that the effect is the least negative/most positive for that particular QC function. A low score means that the effect is the worst/least positive for that particular QC function.

	Weight factor	Loading	Unloading	Twinlift loading	Twinlift unloading
Chance of full racks between Lift-AGVs and YCs	0.20	3	2	4	1
Criticality of the YC stage vs criticality of the Lift-AGV stage	0.18	1	2	4	3
Complexity of the function with TBA's uncoordinated scheduling heuristic	0.15	3	2	4	1
Vehicle weight	0.12	4	4	1	1
Number of required doorturns	0.10	4	2	3	1
Preference of the QC function in the objective function	0.10	2	1	4	3
Number of containers that require scheduling	0.10	4	2	3	1
Number of unknown grounding locations	0.05	4	2	4	1
Weighted average		**2.9**	**2.1**	**3.4**	**1.6**

Table 5.11: Influence of different aspects on a specific QC function. A high score is positive.

As was described in Section 5.2.2 the chance of full racks has a negative effect on the performance. The chance of racks getting full is related to the filling rate with respect to the emptying rate (explains the difference between twinlift vs single lift). Furthermore, the type of equipment that is kept waiting is important (explains the difference between loading vs unloading).
The balance between the criticality between the YC stage and the Lift-AGV stage is important since YC moves are easier to estimate and are less prone to disruptions compared to Lift-AGV moves. Therefore, functions where the YC is the most critical and the Lift-AGV is the least critical perform best.
The complexity of a QC function is important to indicate the potential gain in performance compared to TBA's uncoordinated scheduling heuristic. Unloading and single moves are easier to predict compared to loading and twin moves. Therefore, the harder the operation is to schedule with TBA's uncoordinated scheduling heuristic the more gain the coordinated schedule offers.
Vehicle weight influences the maximum speed and the claimed distance by vehicles. This in turn influences the likelihood of AGV interference. Therefore, lighter single moves are better handled with the coordinated schedule.
The more necessary doorturns, the more activity on the AGV highways, the more congestion occurs. These congestions are not considered within the mathematical model. This means that the theoretical plan does not match reality. Therefore, moves that require the minimal amount of doorturns perform best.
The objective function slightly favors loading moves and twinlift moves over unloading and single moves. This is because the objective function accumulates delays. Therefore, the penalty for delaying

one container increases with increasing number of containers per QC.
As was indicated in Section 5.2.2 during unloading there are more containers on the terminal that require scheduling. This influences the ratio between iterations and number of containers. The higher this ratio, the better the coordinated schedule performs.
Finally, the number of unknown grounding locations has an effect on the accuracy of the planning. Unloading containers do not have a definitive stack module and location within the stack directly from the start. Therefore, the processing time estimates are inaccurate for these moves. This problem is partly solved with the control layer that updates all known information with a specified frequency.

5.2.3. Summary

Based on the limited number of experiments on different single function instances it can be concluded that the coordinated planning does lead to a slightly better performance compared to TBA's uncoordinated scheduling heuristic.
Together with the results obtained with the small scale experiments the weighted average scores in Table 5.11 indicate which QC function is most likely to perform better compared to TBA's uncoordinated scheduling heuristic. In particular twinlift loading moves are better handled with the coordinated schedule. The rest of Table 5.11 provides information that lead to the weighted average score. An explanation of how these scores were obtained is provided in Section 5.2.2.

5.3. Controllable simulation experiments

Now that the small scale experiments are completed, the experimental phase is moved to controllable simulation experiments. These experiments are used to obtain statistical proof for claims based on the small scale experiments. This is why these experiments are called controllable, all QCs have the same function contrary to the more generic reality where each QC could have a different function.
The small scale experiments indicate that twinlift loading is most likely to produce positive results in favour of the coordinated schedule. To confirm this, peak simulations of 8 hours are performed repetitively. The repetition is required to mitigate effects of randomness. Totally, 56 simulation hours were obtained where the simulation used the coordinated schedule. The results of these experiments are subjected to statistical tests to support the conclusions statistically. A brief description of the statistical tests that will be used are introduced in Section 5.3.1.

5.3.1. Statistical methods

Statistics is an important part of research since it allows researchers to support or reject theories on a quantitative basis. To enable distinction, with sufficient statistical back-up, between QC performance when using an uncoordinated and a coordinated schedule, the Welch's t-test will be used [132]. Welch's t-test is a modification to the Student t-test. The modification enables users to compare datasets with unequal standard deviations and an unequal number of samples. Therefore, the Welch's t-test is more generic compared to the ordinary Student's t-test. The Welch's t-test is valid when the following three assumptions are satisfied [133, 134]:

1. Both populations follow a normal distribution. This is generally true once multiple independent samples are used based on the central limit theorem [135]. If desired, this could be verified with the Chi-Square Goodness-of-Fit Test [133];
2. The standard deviation should follow a Chi-Squared distribution;
3. Measurements should be independent of each other.

Once all the assumptions are valid; the statistic t-value is determined with Equation 5.1.

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}}} \tag{5.1}$$

By the time that the statistic t-value is obtained, the Student t-distribution table could be used to find the associated p-value. The p-value expresses the probability of observing the test statistic as extreme or more extreme than the observed value assuming the null hypothesis is true [133, 134]. Besides

the statistic t-value, the p-value is dependent on the degrees of freedom of the test. The degrees of freedom for Welch's t-test can be calculated with Equation 5.2 [132].

$$\nu = \frac{\left(\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}\right)^2}{\frac{s_1^4}{N_1^2(N_1-1)} + \frac{s_2^4}{N_2^2(N_2-1)}} \tag{5.2}$$

The p-value offers the ability to accept or reject a hypothesis on a quantitative basis. The null hypothesis will be rejected whenever the p-value is smaller than the prespecified threshold variable α. A common accepted value for α is 0.05 which is also used in this research.

5.3.2. Hypotheses

The null hypothesis used in this experiment is that the average realised performance of the coordinated schedule and the average realised performance when TBA's uncoordinated scheduling heuristic is applied are equal. The alternative hypothesis is that the average realised performance of the coordinated schedule is higher than the average realised performance when TBA's uncoordinated scheduling heuristic is applied.

5.3.3. Assumptions

The average crane productivity per hour as a sample is independent of other samples since the average is used over multiple cranes. The other two preconditions that should be satisfied in order to lawfully use Welch's t-test are valid based on the chi-square goodness-of-fit test. Therefore, the Welch t-test can be used lawfully.

5.3.4. Results

The results of the experiments are digitally available to save paper. However, the summarized results are presented in Table 5.12 and Figure 5.8. In Figure 5.8 the probability distribution of the results is presented. In Table 5.12 it can be seen that there are more simulation hours with TBA's uncoordinated scheduling heuristic (94) compared to the number of simulation hours with the coordinated schedule (56). Since more samples make it easier to reject the null hypothesis; many samples are desired. However, obtaining more samples is a slow process especially with the coordinated schedule. Therefore, after a while the simulation experiments with the coordinated schedule were ceased while the simulation experiments with TBA's uncoordinated scheduling heuristic continued. On top of that, the standard deviation while applying TBA's uncoordinated scheduling heuristic is larger so more samples are required to reduce this.

	Realisation with the coordinated schedule	Realisation with TBA's uncoordinated scheduling heuristic
Average productivity [box/hr]	37.55	35.81
Standard deviation [box/hr]	4.86	5.44
Samples	56	94
Confidence interval [box/hr]	1.30	1.11
p-value	**0.0225**	

Table 5.12: Results of the realisation with the coordinated schedule and TBA's uncoordinated schedule on a terminal with 5 QCs, 20 Lift-AGVs and pure twinloading.

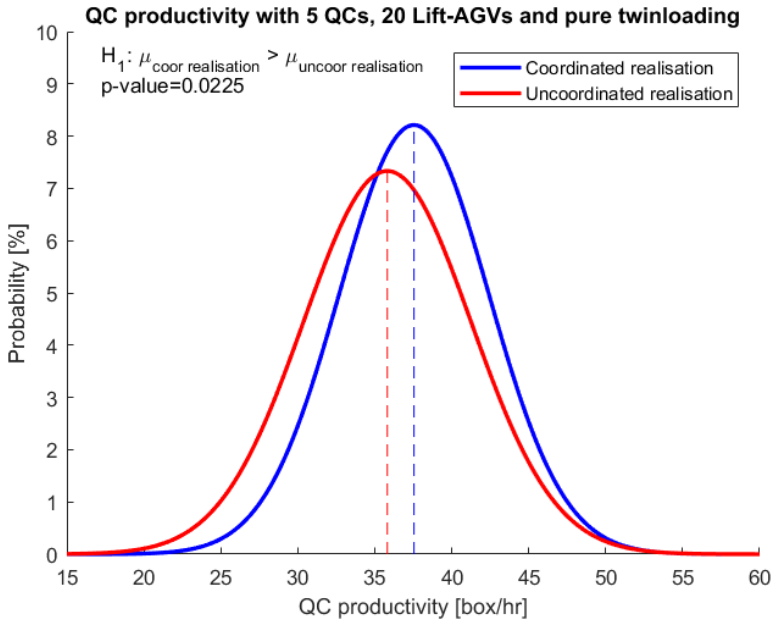


Figure 5.8: Results of the realisation with the coordinated schedule and TBA's uncoordinated schedule on a terminal with 5 QCs, 20 Lift-AGVs and pure twinloading.

5.3.5. Conclusion

Since the p-value (see Table 5.12) is below the threshold value of 0.05 the null hypothesis is rejected and the alternative hypothesis is accepted. Therefore, it can be concluded, with sufficient statistical backup, that the coordinated schedule outperforms TBA's uncoordinated scheduling heuristic on terminals with 5 QCs, 20 Lift-AGVs and pure twinloading. With respect to Figure 5.8 it can be concluded that indeed two different normal distributions are present where the blue one is located right of the red one.

5.4. Large scale experiments

Large scale experiments are experiments where all QCs can perform each function independent of each other. These experiments are generally used by TBA and mimic operations on a real container terminal. The same statistical test as was used during the analysis of the controllable simulation experiments could be used for the analysis of the large scale experiments. On top that, the null hypothesis and the alternative hypothesis remain unchanged as well.

Since the small scale experiments indicated that there is a possible gain in performance for three of the four QC functions and since controllable simulation experiments verified this indication for one of the three functions, a large scale experiment is started. In total this produced 70 usable hours with the coordinated schedule and 63 with TBA's uncoordinated scheduling heuristic. Based on the obtained results all requirements to justify the use of Welch's t-test are satisfied. Therefore, a valid p-value can be calculated which is shown in Table 5.13. Unfortunately, positive results could not be maintained during the large scale experiments as can be seen in Table 5.13 and Figure 5.9. The average QC productivity did not increase by using the coordinated schedule with respect to TBA's uncoordinated scheduling heuristic. Figure 5.9 shows that the coordinated schedule performed worse for every QC function. Therefore, the average QC productivity is lower as well, which explains the large p-value. Therefore, the null hypothesis is accepted.

	Realisation with the coordinated schedule	Realisation with TBA's uncoordinated scheduling heuristic
Average productivity [box/hr]	36.48	37.46
Standard deviation [box/hr]	3.63	3.14
Samples	70	63
Confidence interval [box/hr]	0.87	0.79
p-value	**0.9511**	

Table 5.13: Results of the realisation with the coordinated schedule and TBA's uncoordinated schedule on a terminal with 5 QCs, 20 Lift-AGVs and all QC functions simultaneously present.

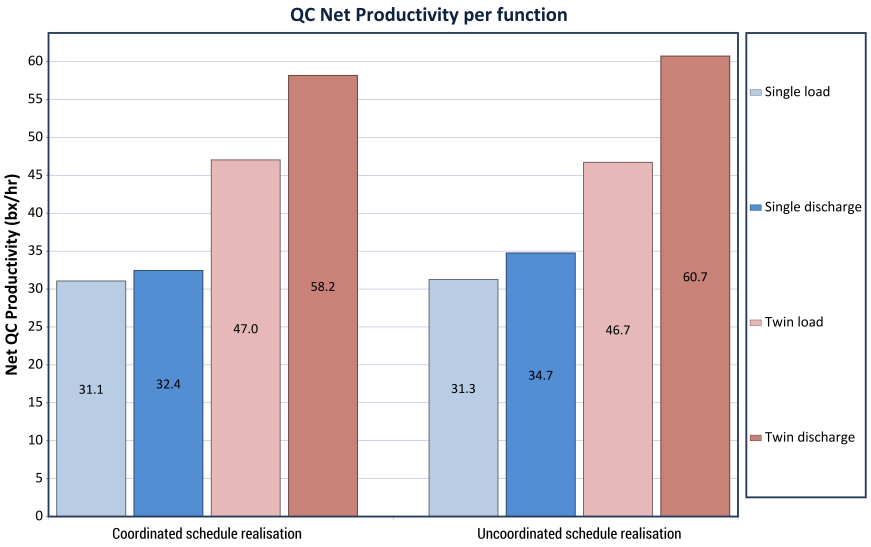


Figure 5.9: The QC productivity on a terminal with 5 QCs and 20 Lift-AGVs while using the coordinated schedule and TBA's uncoordinated schedule.

5.4.1. Analysis of results

After examining the results of the large scale experiments it was found that the coordinated algorithm especially performs worse on unloading moves. It could be that this has affected the performance on loading moves as well. This is because Lift-AGVs spend much more time inside the buffer. This means that the Lift-AGV is not available for other operations and productivity of all QC functions decreases. However, it could also be that the coordinated schedule is too greedy with loading moves sending more Lift-AGVs to the buffers with containers loaded. Both options result in long waiting times and unavailability of Lift-AGVs. This claim is supported by Figure 5.10. In Figure 5.10 it can be seen that the Lift-AGVs spend much more time waiting for a QC approach with the coordinated schedule compared to TBA's uncoordinated scheduling heuristic. On the other hand, waiting and interchanging at the QC is slightly smaller with the coordinated schedule. However, summed up, the Lift-AGVs wait longer while using the coordinated schedule.

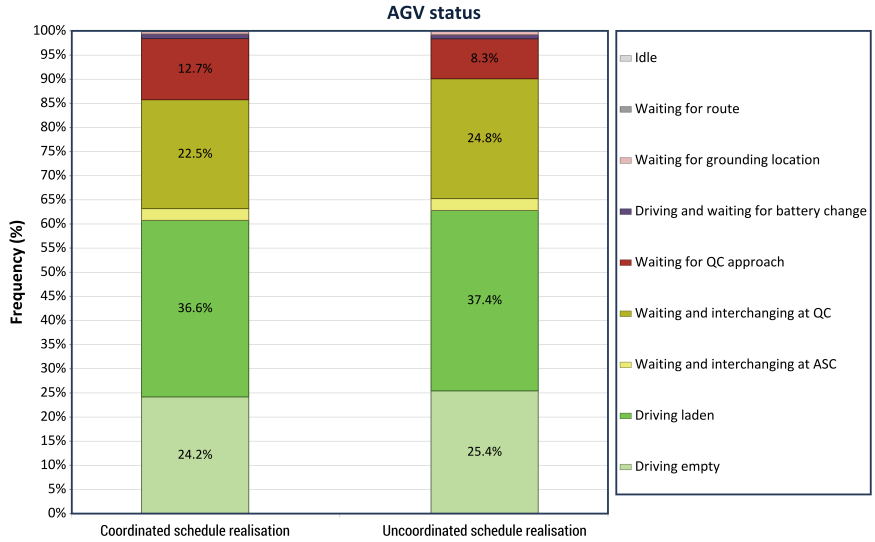


Figure 5.10: The Lift-AGV status on a terminal with 5 QCs and 20 Lift-AGVs while using the coordinated schedule and TBA's uncoordinated schedule.

Moreover, Figure 5.10 confirms that Lift-AGVs carrying unloading containers do not need to wait in the buffer awaiting a grounding location whenever a mixed situation is used instead of purely (twin)unloading. Other mismatches that particularly affected (twin)unloading operations were that there are more containers in the planning and that there is more AGV interference because of door-turns. Both of these mismatches scale with the number of (twin)unloading containers so, with a mixed situation these effects decreased.

5.5. Iteration two of the large scale experiment

Since large scale experiments showed that the QC performance could not be improved by using the coordinated schedule as it is. The coordinated schedule is modified in order to investigate whether the coordinated schedule, in the modified version, could lead to improvement of the QC performance.

5.5.1. Model improvement

Two things are changed before new results are acquired: a new constraint is added to decrease the time that Lift-AGVs wait in the buffer and, the QC processing times are more accurately estimated incorporating the position of containers on the ship.

Just in time constraint

The new constraint that is added is a Just in time (JIT) constraint. This constraint is used to ensure that the Lift-AGVs only arrive at the QCs once they can be immediately served. This is called JIT delivery and is one of the pillars of lean manufacturing [136, 137]. The idea is that by delivering products just in time the waiting times decrease and the required buffer capacity reduces. These two things make this concept attractive for the coordinated schedule. JIT is enforced by Constraint 5.3.

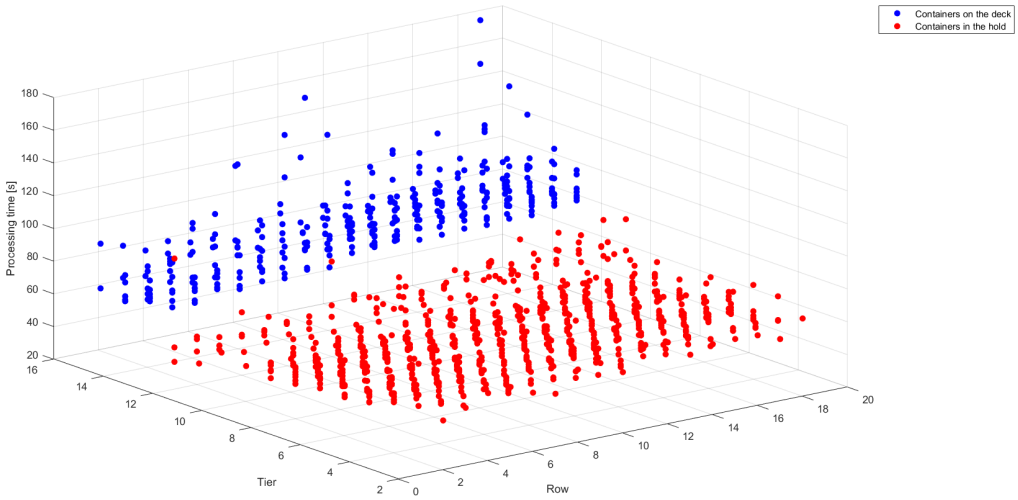
$$t_{k2} + p_{k2} \geq ti2 + p_{i2} + pi3 + s_{ik3} \quad \forall i, k \in \phi \cap A^C \cap L \tag{5.3}$$

Basically, this constraint specifies that operation k can only finish once operation i is finished on both the Lift-AGV and the QC stage. This holds for all containers handled by the same QC, not assigned to any other machine so far, are of the loading type and are not two containers of the same twin pair.

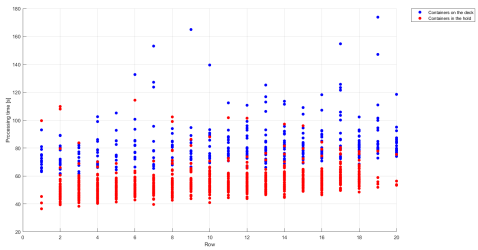
Quay crane processing time

Once the large scale experiments using the coordinated schedule did not produce better results compared to TBA's uncoordinated scheduling heuristic, one of the mathematical simplification mismatches is selected from Table 5.9 to investigate whether this could be reduced. The position of containers on

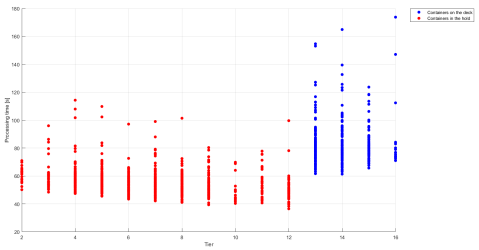
the ship mismatch is selected to be reduced since this is easily changed without huge modifications to the model. The position of containers on the ship has an effect on the estimated processing time of the QC to handle a container. The mismatch manifests itself because of a difference in processing times for containers in the hold and on the deck and the travel distance of the QC's spreader. There is a difference in processing time for containers on the deck and in the hold because twistlocks must be attached/detached to containers on deck. This is the cause of the bumps right to the biggest peak for every QC function in Figure 5.6. Furthermore, the row and tier position of a container on the ship also effect the processing time since it is directly related to the travelled distance of the spreader. Therefore, a study into the processing time with respect to the container position is performed for every QC function. The results of this study for loading operations is shown in Figure 5.11.



(a) 3D viewpoint



(b) Processing time vs row position viewpoint



(c) Processing time vs tier position viewpoint

Figure 5.11: The processing time of the QC for loading containers with respect to the position on the ship separated for containers in the hold and on the deck.

From Figure 5.11 it can be concluded that moves on deck clearly take longer compared to moves in the hold. Furthermore, the higher the row, the further the spreader should move, the longer the processing time. Finally, for containers in the hold, the processing time increases with decreasing tier. For containers on deck this relation is reversed. Both are logical since it increases the distance that the spreader should travel to reach the desired tier. Two linear surfaces are fitted to the datapoints shown in Figure 5.11, for hold and deck containers separately, to estimate the QC processing time more accurately. Besides the predictable effects of the container position on the QC processing time, Figure 5.11 also indicates the likeliness of containers requiring processing times much longer than expected. Similar scatterplots could be obtained for the other QC functions. Confirming expectations, the time required to pass rows is independent of the QC function whereas the time to move tiers is. This is because the time required to move along different tiers is weight dependent.

To prove that the new prediction of the QC processing time is indeed more accurate than the rough estimate, the small scale experiments were repeated. The new distribution of the QC processing time is shown in Figure 5.12. The distribution shown in Figure 5.12 became more narrow compared to Figure 5.6. However, the distribution could not be reduced to a single spike because of other factors disturbing the QC processing time remained present. These are: the attachment of twistlocks, the container weight and the interchange time. Besides that, the more accurate estimate could not always be used since the location of containers on the ship of another bay than the current bay that a QC is handling is unknown. Therefore, for these containers the more general assumption is used.
With the aid of a more accurate prediction of the QC processing time, it was found that underestimating the QC processing time slightly results in better productivity. This is because it is better to have an Lift-AGV idling compared to a QC. By underestimating QC processing times, Lift-AGVs are stimulated to arrive too early at the QC making sure that the QC never idles waiting for a Lift-AGV.

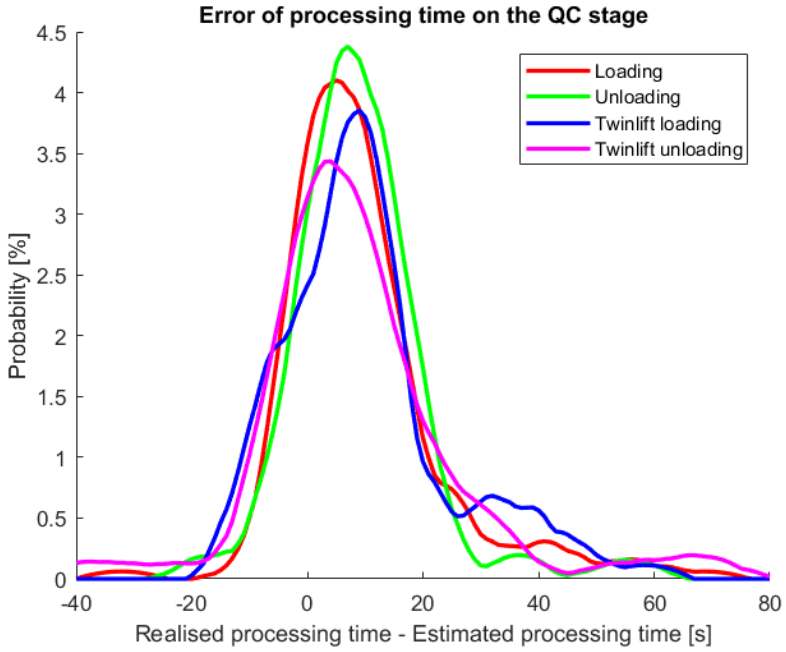


Figure 5.12: The probability distribution of the mismatches between realised and estimated processing times of QCs on a terminal with 5 QCs and 20 Lift-AGVs once incorporating the position of containers on the ship within the prediction.

5.5.2. Results of iteration two

The large scale experiment performed in Section 5.4 is repeated. The null hypotheses is that the average realised performance of the coordinated schedule and the average realised performance when TBA's uncoordinated scheduling heuristic is applied are equal. The alternative hypothesis is that the average realised performance of the coordinated schedule is higher than the average realised performance when TBA's uncoordinated scheduling heuristic is applied. This new large scale experiment produced new results that are shown in Table 5.14. The number of samples for both experiments is not equal for the same reasons as for the previous experiments. Based on the obtained results all requirements to justify the use of Welch's t-test are satisfied. Therefore, a valid p-value can be calculated which is also shown in Table 5.14.

	Realisation with the coordinated schedule	Realisation with TBA's uncoordinated scheduling heuristic
Average productivity [box/hr]	38.02	37.76
Standard deviation [box/hr]	2.80	3.41
Samples	105	98
Confidence interval [box/hr]	0.54	0.68
p-value	**0.2731**	

Table 5.14: Results of the realisation with the coordinated schedule and TBA's uncoordinated schedule on a terminal with 5 QCs, 20 Lift-AGVs and all QC functions simultaneously present.

5.5.3. Analysis of results

Based on Table 5.14 we conclude that results have improved with respect to the first iteration. However, we are unable to reject the null hypothesis based on the obtained number of samples and achieved performance. Therefore, we conclude that the coordinated schedule and TBA's uncoordinated scheduling heuristic have an equal performance.

The results presented in Table 5.14 are examined in more detail with the aid of Figures 5.13 and 5.14.

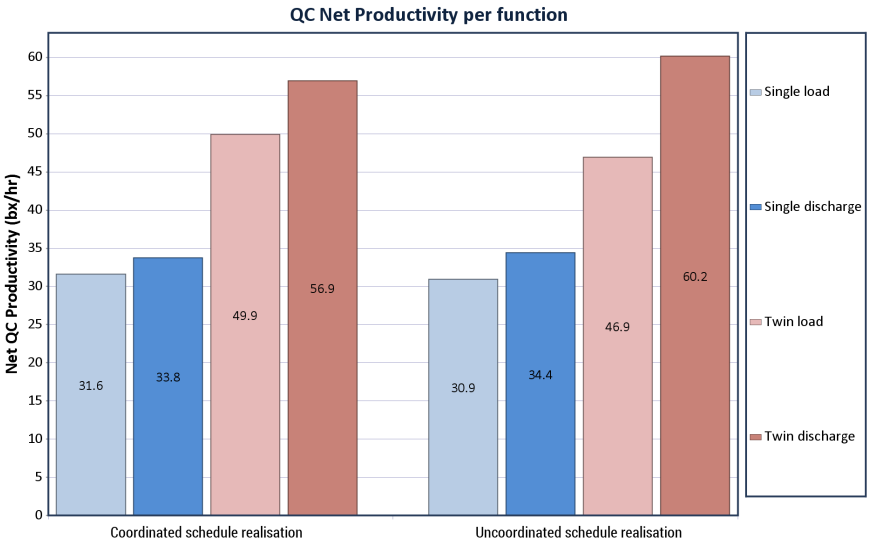


Figure 5.13: The QC productivity on a terminal with 5 QCs and 20 Lift-AGVs while using the coordinated schedule and TBA's uncoordinated schedule.

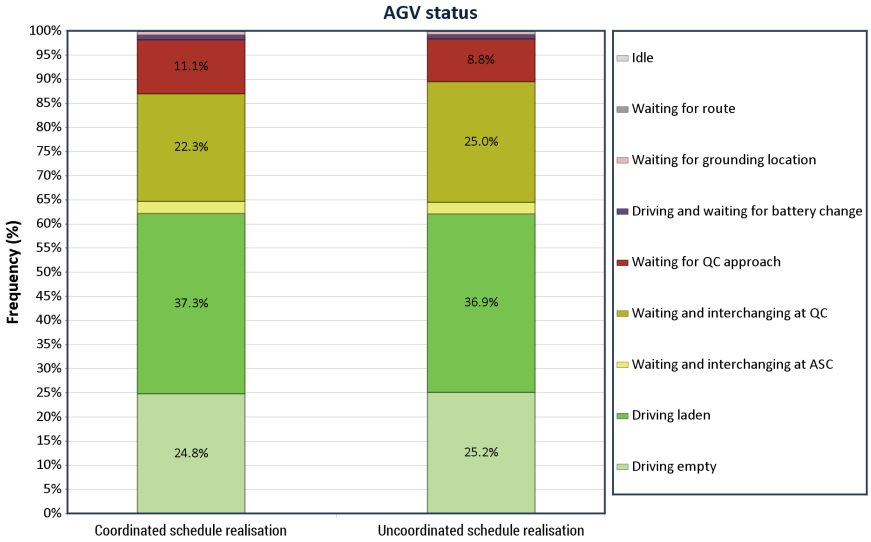


Figure 5.14: The Lift-AGV status on a terminal with 5 QCs and 20 Lift-AGVs while using the coordinated schedule and TBA's uncoordinated schedule.

From Figure 5.13 it becomes clear that the coordinated schedule performs well on loading moves but, unloading moves are better performed with TBA's uncoordinated scheduling heuristic. This is in line with the observed performance during the small scale experiments. Furthermore, from Figure 5.14 it can be seen that with respect to Figure 5.10 the AGVs spend less time waiting for a QC. Compared to TBA's uncoordinated scheduling heuristic the AGVs spend more time in the buffer but, less time at the QC. Summed up, the waiting time while using the coordinated schedule (33.4%) is not much different from the waiting time while using TBA's uncoordinated scheduling heuristic (33.8%). More charts that could be used to analyze the performed experiments are presented in Appendix B.

5.5.4. Discussion of the results

It is expected that by reducing the mismatches indicated in Table 5.9 the performance of the coordinated schedule could surpass the performance of TBA's uncoordinated scheduling heuristic. Possibly, acquiring more samples could lead to the same conclusion. However, the simulation time of the coordinated schedule was much longer compared to the simulation time with the uncoordinated scheduling heuristic. On top of that, the coordinated schedule requires resets of the scheduling computer to prevent the memory from getting full. The result of the memory issue is that gradually the computational time per optimization cycle increases. Therefore, the interval between the resets determines the total required computational time of the coordinated schedule. Iteration 2 required 7 hours with the uncoordinated scheduling heuristic and approximately 72 hours with the coordinated schedule. Possibly an experienced programmer could prevent the memory from getting full. However, the difference in time will remain large.

5.6. Model improvement directions

In this section we present different options to improve the performance of the coordinated schedule. These options are indicated in Figure 5.15. The first option is to disregard the current deterministic model and to develop a new stochastic model. This is because stochastic models are able to cope with uncertainties. Therefore, they deliver a better representation of reality. One possible outcome of a stochastic model is that operations associated with possible long delays are not scheduled on the same equipment to reduce the risk of disruptions at other machines. Developing an entire new stochastic model would require a lot of effort and is therefore not desired. Another option to increase the performance of the coordinated schedule, is to improve the current deterministic model. Three different directions to do this have been identified.

The first direction is to add a control layer. By adding a control layer the schedule is constantly

updated based on current information. This could be done in a receding horizon approach. The length of the horizon and the frequency of updating the schedule will influence the performance [20]. Longer horizons lead to a more optimal schedule but, also require longer computational times. A higher updating frequency leads to a more accurate representation of the current situation but, also increases the total time required for scheduling. Moreover, not only the containers currently in progress should be updated, also the estimated end times of the current operations should be recalculated. This also requires more computational power. Another disadvantage is that control only offers improvement once the model is stable enough. Once the model is inaccurate, it can still produce meaningless output. What greatly reduces the benefits of control is that Lift-AGVs are assigned to a move rather early and are not allowed to preempt their operation. Therefore, once it turns out that it was a bad choice to assign a particular Lift-AGV to a particular move; control cannot stop this. The basic control strategy of rescheduling containers within a receding horizon is already performed within the current coordinated schedule. However, the control layer could be extended with dynamically updating parameters based on the current stage of the terminal. Furthermore, a study into the balance between horizon length, re-planning frequency and the number of iterations per optimization cycle could be performed to better set these parameters.

The second direction is to extend the current mathematical model by removing the different mismatches introduced in Table 5.9 as mismatches related to the simplification of the mathematical model with respect to the simulation model. The major advantage is that this will lead to a far more accurate representation of reality. This enables the mathematical model to create an achievable schedule. However, by integrating more functions, the model also becomes more complex. This leads to longer computational times which is undesired. This could be mitigated by introducing multiple layers to the scheduling algorithm. First, on a higher layer a schedule is made. Second, given the created schedule optimal choices should be selected to execute this schedule. For instance, with regards to routing. By separating these decisions, the model becomes far less complex since there is only a one-way dependency. However, during scheduling; it is still inconclusive whether the created schedule is executable as scheduled. This means that an optimal schedule cannot be guaranteed. Therefore, research is required to determine which modules should be placed at what levels to ensure a good balance between performance and computational time.

The final identified future research direction to improve the deterministic model is to better predict the times required for certain operations. A suggested technique is machine learning. The advantage of machine learning is that patterns could be recognized which are invisible to the human eye. This can only be done by supplying lots of data to a machine to establish patterns. This directly introduces the disadvantage of machine learning; lots of data is required and it is unknown which data specifically is required. This means that a research is required aimed at studying patterns and relations between times required to perform certain actions and the nature of these actions. Another complicating factor is that more insight in the times required to perform certain actions could also lead to longer computational times. This is because first a prediction should be performed before the scheduling may start. Furthermore, better predictions will most likely also reduce the robustness of the coordinated scheduling algorithm. This is because each terminal has its own characteristics that influence the processing times. This claim is supported by Figure 5.4. Each different number of Lift-AGVs has different processing times. Second, it could be that some times are dependent on the sequence of operations. Since this sequence is unknown before scheduling, the prediction should be integrated within the scheduling algorithm. This complicates the mathematical model, increasing the computational time.

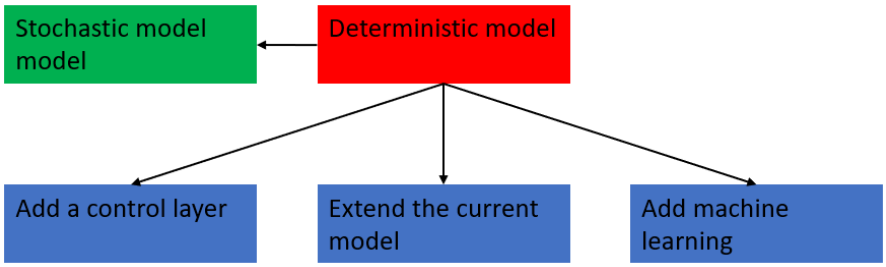


Figure 5.15: Suggested model improvements to the current deterministic model.

All of the four options to improve the current deterministic model do not exclude another. Therefore, it is also possible to improve the model by combining different options. Furthermore, all options can be performed to a variable extend. This means that any possible mix of these four options could be selected to improve the current deterministic model. It is expected that the model will improve asymptotically by adding any of these four options to various extend. Therefore, it is advised to first investigate which improvement is likely to produce a maximum gain with minimum effort. By repeating this procedure it is expected that the model will improve most quickly without making it far too complex to solve. Probably, this would produce a mixture of all four options introduced in Figure 5.15. On the other hand each option increases the complexity of the problem. Therefore, it is advised to question whether the current solution technique remains sufficient. In [59, 67] constraint programming is suggested as an alternative to MIP.

5.7. Discussion on the results

During the gathering of results it was found that the coordinated schedule is very sensitive to small changes. Therefore, it is hard to determine, based on a small number of samples, which settings result in the optimal realisation. For each instance a set of settings resulting in a positive and a negative outcome could be found. However, it remains inconclusive which set of parameters results in the best overall performance. Nevertheless, this is a valuable conclusion; the scheduling algorithm used by TBA is not perfect but, it is a very good alternative since it is robust and delivers proper results in most cases. On top of that, it is very quick compared to the coordinated schedule used in this research.
Furthermore, as was already indicated in Section 5.5.4, the simulation time of the coordinated schedule is much longer compared to the simulation time of the uncoordinated schedule. This is because of the long computational time associated with the creation of coordinated schedules. Further efforts to improve the quality of the coordinated schedule are likely to increase the computational time even further.

6

Conclusion and future research

In this chapter we answer the research questions presented in section 1.3. The main research question of this report is:

Can Quay crane productivity be increased compared to an uncoordinated schedule by developing a coordinated schedule considering Quay cranes, Yard Cranes and Horizontal Transport Vehicles?

Based on the results obtained during this research, the QC productivity whilst using TBA's uncoordinated scheduling heuristic cannot be increased with a coordinated schedule that uses a HFS to model the container terminal and a SA algorithm to solve that model (p-value 0.2731). On the other hand, it can neither be concluded that TBA's uncoordinated scheduling heuristic results in higher productivity compared to the coordinated schedule. Therefore, the coordinated schedule is a solid alternative to TBA's uncoordinated scheduling heuristic. However, it is expected that with certain changes to the coordinated schedule the QC productivity could be increased.

The main research question could not have been answered without answering the underlying sub questions:

1. The current uncoordinated scheduling algorithm that is used to schedule operations considers all equipment types operating on the terminal separately. The scheduling and dispatching strategy applied at the YC is as follows: all orders known to the central planning for a specific YC are gathered. Subsequently, orders outside a time window of 30 minutes are excluded. Thereafter, orders that could lead to sequence conflicts at the QC are excluded. Subsequently, a scoring algorithm is initiated that consists of an urgency score, a sequence score and a transferpoint penalty. Where the urgency score and the sequence score are influenced by QC operations. Finally, the order with the best score is selected.
   At the HTV stage orders are selected following a similar technique. Orders outside a time window of 30 minutes are excluded. Similar to orders that have an origin or destination that is already the origin/destination of N other vehicles. Finally, orders that could lead to sequence conflicts are excluded. Now, the scoring mechanism is initiated again; consisting of penalties for containers that have not yet started or are underway. Furthermore, a sequence and urgency score is granted similar to the one for YCs. Additionally, a traffic and a distance score is used to score orders. The traffic score promotes orders in quiet places of the terminal and the distance score promotes orders that are close to the current location of the HTV. This procedure is repeated for every HTV and container combination. Subsequently, the combination of HTV and container with the best score is selected.
   There is no scheduling algorithm at the QC stage since orders should be processed in a pre-specified sequence. This is because QCs should obey a loadplan.

2. The current productivity of the QCs with the uncoordinated schedule cannot be defined with a single number. This is because there is great variation in productivity with different scenarios. Throughout the report different scenarios are used and each time the current productivity for that

specific scenario is compared with the achieved productivity of the realisation with the coordinated schedule.

3. In literature, three promising coordinated scheduling approaches are available for operations involving multiple stages. These are HFS models, MPS and Markov chains. Of these three approaches, HFS models are selected as most applicable to container terminals using a coordinated schedule. This is because the relation with other research is strong. Therefore, the likelihood that this research will be used by other researchers and industry increases. Furthermore, HFS problems can be solved optimally given sufficient computational time. When computational time is limited, many alternative heuristic solutions exist that speed-up the calculation process. Moreover, the HFS problem allows many additional features which can easily be turned on and off dependent on the user's wishes. Besides that, HFS problems appear better structured and organized compared to MPS and Markov chains. Finally, HFS problems can easily be scaled to the requirements of the user.

4. To apply coordinated scheduling to real container terminals the computational time should be short. This is because the planning has to be adjusted frequently since operations vary quickly and predictions with regards to processing times are unreliable. Therefore, exact solution techniques are not feasible and heuristic solutions should be used. A tailored SA algorithm proved to work very efficiently (balance between performance and computational time) to solve the HFS model formulated in this research. The exact solution is approximated with the tailored SA algorithm by splitting the problem in a discrete and a continuous part. The discrete part uses a local search method to find better alternatives to the current solution by predictive reassignment of jobs to machines. The continuous part can subsequently be solved quickly with LP.

5. The theoretical performance of the coordinated schedule consisting of a HFS model and a tailored SA algorithm to solve it is shown in Table 4.8. The optimality gap between the exact solution and the approximated solution never exceeds 11% for terminals with sizes of up to 8 QCs, 32 Lift-AGVs, 20 YCs and 80 containers when using a quick solution (maximum computational time 3.3 seconds). When the SA is granted more time (57.6 seconds) the optimality gap never exceeds 6% for terminals with sizes of up to 8 QCs, 32 Lift-AGVs, 20 YCs and 80 containers. The balance between quality and computational time can be controlled by adjusting the number of iterations used to solve the HFS problem.

6. The realised performance of the coordinated schedule is 38.02 boxes/hour on a terminal with 5 QCs, 20 Lift-AGVs, 13 YCs. The realised performance of TBA's uncoordinated scheduling heuristic on the same terminal is 37.76 boxes/hour. The difference between these two is too small with 105 and 98 samples respectively to conclude that the coordinated schedule has a higher productivity compared to TBA's uncoordinated scheduling heuristic.

6.1. Recommendations to TBA

In this section we present our advise to TBA. Based on the results obtained during this research, it cannot be concluded that a coordinated schedule performs better compared to their current uncoordinated scheduling technique. However, with certain changes it is expected that a coordinated schedule could achieve higher QC productivity. But, TBA should determine if this is worth the investment. The disadvantages of the coordinated schedule are that it requires long computational times. This makes a coordinated schedule not feasible for use in the simulation department where results are required rapidly. Furthermore, the coordinated schedule requires detailed data of the terminal where it is applied. This costs extra time and is especially expensive once multiple layouts of the same terminal should be investigated. Moreover, in its current form, a lot of input data should be supplied manually to the coordinated schedule making it expensive in terms of labor costs. Additionally, manual resets are required to prevent the memory from getting full. Finally, the coordinated schedule, in its current form, is programmed in MATLAB and Gurobi, both are software programs that are not owned by TBA. This means that either licenses should be bought or the code should be transferred to software programs owned by TBA.

As far as I could see the potential gain and costs associated with a coordinated schedule, I discourage TBA to continue development of the coordinated schedule aiming for optimal productivity. However,

this research exposed that the current scheduling heuristic is not flawless. Therefore, I suggest that TBA tries to improve their current scheduling heuristic. Maybe new scoring attributes could be added to their current dispatching strategy that promote coordination.

6.2. Future research

In this section several suggested further research directions are indicated to improve the results that were obtained during this research.

- It would be interesting to see the effects of using a stochastic mathematical model instead of a deterministic one. Currently, all setup and processing times are assumed to be deterministic. However, the reality is that everything is stochastic as disruptions occur. It could be that a stochastic model is better at predicting the time required to perform certain operations. But, more importantly a stochastic model is expected to better distribute operations with uncertain processing times over multiple machines.

- Besides that, the mathematical model could also be improved by adding more functions. Currently, certain operations and actions on the terminal are ignored within the coordinated schedule. A more extensive but, at the same time more complex model could be used that integrates more functions on the container terminal. Operations and actions that could be added to the mathematical model are: routing of equipment, slot assignment, equipment dynamics, landside operations, buffer positions and battery changes.

- Another approach to improve the quality of the model is to get a more accurate insight in the deterministic time values. This could be done by a better prediction of these values by considering multiple factors. A promising technique would be to use machine learning. With more accurate predictions the coordinated schedule approaches reality more closely. This results in equipment that is more likely to be able to follow the coordinated schedule as planned.

- Moreover, the performance of the coordinated schedule could be increased by adding a more sophisticated control layer. Currently, the only applied control strategy is that after a fixed time the entire schedule is updated to ensure that data remains accurate. However, with a more sophisticated control layer errors in the coordinated schedule could be resolved dynamically. Furthermore, an investigation into the balance between computational time and performance could be performed. By lengthening the planning horizon and increasing the update frequency the coordinated schedule is expected to perform better at the cost of a longer computational time.

- To increase the effect of the control layer it is advised that equipment is allowed to switch orders until the moment that a container is grabbed. The advantage is that operations can be rescheduled for a longer period. This decreases the likeliness that jobs are wrongfully assigned to specific machines. Currently the model offers no possibility to recover from bad choices. It is expected that both the coordinated schedule and TBA's scheduling heuristic will benefit from this added freedom.

- Other scheduling techniques could be applied to investigate whether they are able to outperform the current scheduling heuristic of TBA. It is suggested to start looking at scoring attributes that could be added to TBA's scheduling heuristic that promote coordination of the different equipment types.

- Furthermore, it is recommend to set parameters used during optimization to their optimal values. This was not performed during this research since this would have required much more time and computational power which was not available. It is expected that results will improve by a few percentages when parameters are set optimally.

- More advanced would be not to pre-specify parameters but, to let them update dynamically. This to ensure that throughout every instance the optimal set of parameters is used instead of the optimal average set of parameters.

- Since many of the above future research directions increase the complexity of the problem it might be wise to review the current solution technique. Constraint programming might be a suitable alternative to MIP combined with SA.
- Additionally, it could be considered to update the objective function for instance to balance performance between the four different QC functions. Currently, it was observed that the coordinated schedule showed a different increase in performance with regards to all four QC functions. With another objective function the performance could possibly be balanced more evenly. Furthermore, another objective function could be used to emphasize either QC productivity or the total makespan of operations. Currently, TBA is inconclusive which of the two is their objective.
- From an academic perspective it is desired that there are better options that isolate situations in the simulation model and allow for exact logging of operations. This is because currently the simulation model is very centered around doing all sorts of operations simultaneously. To gain more insight into the effects of certain changes to the mathematical model it is desired to replicate scenarios exactly or to isolate a certain part of the terminal or to turn off stochastic effects. Currently, this is not always possible or requires a lot of programming work.
- Research could be extended towards different terminals with another layout. Furthermore, it would be interesting whether the coordinated schedule works for other equipment types. In case the coordinated schedule is not directly applicable, it could be investigated what needs to change and what is shared between the different terminals. Besides that, research performed on different terminals could be used to investigate the scaleability of the algorithm.
- Research would benefit from a library with HFS problems that enables fair comparison between different algorithms used by different researchers. Currently, new algorithms can only be compared with itself since there is no general test scenario defined.

Many of the future research directions indicate some form of desired additional robustness. This is because with the current coordinated schedule quite some time is required to make the model ready for tests with a different terminal.

Bibliography

[1] R. Stahlbock and S. Voß, *Operations research at container terminals: a literature update,* OR spectrum **30**, 1 (2008).

[2] E. K. Bish, *A multiple-crane-constrained scheduling problem in a container terminal,* European Journal of Operational Research **144**, 83 (2003).

[3] H. Schim van der Loeff, *Quay crane productivity at the ECT Delta Terminal*, Master's thesis, TU Delft (2008).

[4] B. K. Lee and K. H. Kim, *Optimizing the block size in container yards,* Transportation Research Part E: Logistics and Transportation Review **46**, 120 (2010).

[5] Liebherr, *Container gantry crane,* http://www.nauticexpo.com/prod/liebherr-international-deutschland/product-30468-189567.html (2017), [Online; accessed 2018-1-11].

[6] T. online, *Automatische smeersystemen en oilmaster populair in container handling,* https://www.transport-online.nl/site/15145/automatische-smeersystemen-en-oilmaster-populair-in-containerhandling/ (2013), [Online; accessed 2018-1-11].

[7] Paceco, *Quay cranes,* http://www.paceco.es/en/quay-cranes (2017), [Online; accessed 2018-1-11].

[8] Y. Saanen, *Agv versus lift agv versus alv,* Port Technology International , 63 (2016).

[9] A. Verbraeck, M. Seck, and M. Fumarola, *Validation Simulation Models Maasvlakte 2*, Tech. Rep. 1.0 (TU Delft, 2009).

[10] K. Lewandowski, *Growth in the size of unit loads and shipping containers from antique to wwi,* Packaging Technology and Science **29**, 451 (2016).

[11] J.-P. Rodrigue, C. Comtois, and B. Slack, *The geography of transport systems* (Routledge, 2009).

[12] J.-P. Rodrigue, *Composition of the global fleet of containers, 2008,* https://web.archive.org/web/20141121102626/http://people.hofstra.edu/geotrans/eng/ch3en/conc3en/containerfleet.html (2008), [Online; accessed 2018-1-12].

[13] A. De Waal, *Container terminal & performance,* TBA internal documents (2012), [Online; accessed 2018-1-11].

[14] A. De Waal and P. Bierhuizen, *Timesquare functional description,* TBA internal documents (2017), [Online; accessed 2018-1-11].

[15] D. Steenken, S. Voß, and R. Stahlbock, *Container terminal operation and operations research-a classification and literature review,* OR spectrum **26**, 3 (2004).

[16] L. Zhen, X. Jiang, L. H. Lee, and E. P. Chew, *A review on yard management in container terminals,* Industrial Engineering and Management Systems **12**, 289 (2013).

[17] M. E. Petering, *Effect of block width and storage yard layout on marine container terminal performance,* Transportation Research Part E: Logistics and Transportation Review **45**, 591 (2009).

[18] B. Vuuren, *Route plan scheduling for automated guided vehicles at container terminals*, Master's thesis, TU Delft (2017).

[19] L. Chen, N. Bostel, P. Dejax, J. Cai, and L. Xi, *A tabu search algorithm for the integrated scheduling problem of container handling systems in a maritime terminal,* European Journal of Operational Research **181**, 40 (2007).

[20] P. J. Meersmans and A. P. Wagelmans, *Effective algorithms for integrated scheduling of handling equipment at automated container terminals,* ERIM Report series (2001).

[21] J. Rijsenbrij, *Container handling in mainports: a dilemma about future scales,* The future of intermodal freight transport , 109 (2008).

[22] A. Staff, *Top 5: Ways container ships have evolved in size,* www.arabiansupplychain.com/article-11234-top-5-ways-container-ships-have-evolved-in-size/ (2015), [Online; accessed 2018-1-11].

[23] N. Al-Dhaheri and A. Diabat, *The quay crane scheduling problem,* Journal of Manufacturing Systems **36**, 87 (2015).

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

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

[26] K. H. Kim and Y.-M. Park, *A crane scheduling method for port container terminals,* European Journal of operational research **156**, 752 (2004).

[27] J. Wiese, N. Kliewer, and L. Suhl, *A survey of container terminal characteristics and equipment types,* Decision Support & Operations Research Lab. University of Paderborn (2009).

[28] J. Wiese, L. Suhl, and N. Kliewer, *Planning container terminal layouts considering equipment types and storage block design,* Handbook of terminal planning , 219 (2011).

[29] T. Boontjes, *The vertical container terminal: stacking strategies and job dispatching*, Master's thesis, TU Delft (2014).

[30] R. E. Shannon, *Introduction to the art and science of simulation,* in *Proceedings of the 30th conference on Winter simulation* (IEEE Computer Society Press, 1998) pp. 7–14.

[31] O. Balci, *Verification, validation, and testing,* Handbook of simulation **10**, 335 (1998).

[32] M. J. Sharma and S. J. Yu, *Benchmark optimization and attribute identification for improvement of container terminals,* European Journal of Operational Research **201**, 568 (2010).

[33] R. J. van Zijverden and R. R. Negenborn, *Survey of approaches for integrated control of intermodal container terminals,* in *Networking, Sensing and Control (ICNSC), 2012 9th IEEE International Conference on* (IEEE, 2012) pp. 67–72.

[34] M. De Koster, B. Balk, and W. Van Nus, *On using dea for benchmarking container terminals,* International Journal of Operations & Production Management **29**, 1140 (2009).

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

[36] J. C. Rijsenbrij and A. Wieschemann, *Sustainable container terminals: a design approach,* in *Handbook of terminal planning* (Springer, 2011) pp. 61–82.

[37] J. Van Duin and H. Geerlings, *Estimating co 2 footprints of container terminal port-operations,* International Journal of Sustainable Development and Planning **6**, 459 (2011).

[38] M. B. Duinkerken, J. A. Ottjes, and G. Lodewijks, *Comparison of routing strategies for agv systems using simulation,* in *Proceedings of the 38th conference on Winter simulation* (Winter Simulation Conference, 2006) pp. 1523–1530.

[39] B. Cao and G. Uebe, *Solving transportation problems with nonlinear side constraints with tabu search,* Computers & Operations Research **22**, 593 (1995).

[40] K. H. Kim, Y. M. Park, and K.-R. Ryu, *Deriving decision rules to locate export containers in container yards,* European Journal of Operational Research **124**, 89 (2000).

[41] C. Zhang, J. Liu, Y.-w. Wan, K. G. Murty, and R. J. Linn, *Storage space allocation in container terminals,* Transportation Research Part B: Methodological **37**, 883 (2003).

[42] K. H. Kim and J. W. Bae, *Re-marshaling export containers in port container terminals,* Computers & Industrial Engineering **35**, 655 (1998).

[43] M. Bielli, A. Boulmakoul, and M. Rida, *Object oriented model for container terminal distributed simulation,* European Journal of Operational Research **175**, 1731 (2006).

[44] M. B. Duinkerken and J. A. Ottjes, *A simulation model for automated container terminals,* in *Proceedings of the Business and Industry Simulation Symposium*, Vol. 10 (2000) pp. 134–139.

[45] S. A. Reveliotis, *Conflict resolution in agv systems,* Iie Transactions **32**, 647 (2000).

[46] L. Qiu and W.-J. Hsu, *Conflict-free agv routing in a bi-directional path layout,* in *Proceedings of the 5th International Conference on Computer Integrated Manufacturing*, Vol. 1 (2000) pp. 392–403.

[47] M. Grunow, H.-O. Günther, and M. Lehmann, *Dispatching multi-load agvs in highly automated seaport container terminals,* Container Terminals and Automated Transport Systems Part I , 231 (2005).

[48] G. Ulusoy, F. Sivrikaya-Şerifoğlu, and Ü. Bilge, *A genetic algorithm approach to the simultaneous scheduling of machines and automated guided vehicles,* Computers & Operations Research **24**, 335 (1997).

[49] I. Ribas, R. Leisten, and J. M. Framiñan, *Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective,* Computers & Operations Research **37**, 1439 (2010).

[50] R. Ruiz and J. A. Vázquez-Rodríguez, *The hybrid flow shop scheduling problem,* European Journal of Operational Research **205**, 1 (2010).

[51] P. Fattahi, S. M. H. Hosseini, F. Jolai, and R. Tavakkoli-Moghaddam, *A branch and bound algorithm for hybrid flow shop scheduling problem with setup time and assembly operations,* Applied Mathematical Modelling **38**, 119 (2014).

[52] A. Vignier, J.-C. Billaut, and C. Proust, *Les problèmes d'ordonnancement de type flow-shop hybride: état de l'art,* RAIRO-Operations Research **33**, 117 (1999).

[53] M. Pinedo, *Scheduling* (Springer, 2012).

[54] E. González-Neira, J. Montoya-Torres, and D. Barrera, *Flow-shop scheduling problem under uncertainties: Review and trends,* International Journal of Industrial Engineering Computations **8**, 399 (2017).

[55] E. M. González-Neira, R. G. García-Cáceres, J. P. Caballero-Villalobos, L. P. Molina-Sánchez, and J. R. Montoya-Torres, *Stochastic flexible flow shop scheduling problem under quantitative and qualitative decision criteria,* Computers & Industrial Engineering **101**, 128 (2016).

[56] Y. Lu and M. Le, *The integrated optimization of container terminal scheduling with uncertain factors,* Computers & Industrial Engineering **75**, 209 (2014).

[57] A. Elyasi and N. Salmasi, *Stochastic flow-shop scheduling with minimizing the expected number of tardy jobs,* The International Journal of Advanced Manufacturing Technology , 1 (2013).

[58] A. A. Juan, B. B. Barrios, E. Vallada, D. Riera, and J. Jorba, *A simheuristic algorithm for solving the permutation flow shop problem with stochastic processing times,* Simulation Modelling Practice and Theory **46**, 101 (2014).

[59] L. Chen, A. Langevin, and Z. Lu, *Integrated scheduling of crane handling and truck transportation in a maritime container terminal,* European Journal of Operational Research **225**, 142 (2013).

[60] H. Y. Lau and Y. Zhao, *Integrated scheduling of handling equipment at automated container terminals,* International journal of production economics **112**, 665 (2008).

[61] J. Xin, *Control and Coordination for Automated Container Terminals* (Citeseer, 2015).

[62] J. Xin, R. R. Negenborn, F. Corman, and G. Lodewijks, *Control of interacting machines in automated container terminals using a sequential planning approach for collision avoidance,* Transportation Research Part C: Emerging Technologies **60**, 377 (2015).

[63] K. H. Kim and J. W. Bae, *A look-ahead dispatching method for automated guided vehicles in automated port container terminals,* Transportation science **38**, 224 (2004).

[64] J. He, Y. Huang, W. Yan, and S. Wang, *Integrated internal truck, yard crane and quay crane scheduling in a container terminal considering energy consumption,* Expert Systems with Applications **42**, 2464 (2015).

[65] N. Kaveshgar and N. Huynh, *Integrated quay crane and yard truck scheduling for unloading inbound containers,* International Journal of Production Economics **159**, 168 (2015).

[66] J. Cao, Q. Shi, and D.-H. Lee, *Integrated quay crane and yard truck schedule problem in container terminals,* Tsinghua Science & Technology **15**, 467 (2010).

[67] D. Kizilay, D. T. Eliiyi, and P. Van Hentenryck, *Constraint and mathematical programming models for integrated port container terminal operations,* in *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research* (Springer, 2018) pp. 344–360.

[68] J. Xin, R. R. Negenborn, and G. Lodewijks, *Energy-aware control for automated container terminals using integrated flow shop scheduling and optimal control,* Transportation Research Part C: Emerging Technologies **44**, 214 (2014).

[69] J. Xin, R. R. Negenborn, and G. Lodewijks, *Event-driven receding horizon control for energy-efficient container handling,* Control Engineering Practice **39**, 45 (2015).

[70] S. H. Jung and K. H. Kim, *Load scheduling for multiple quay cranes in port container terminals,* Journal of Intelligent manufacturing **17**, 479 (2006).

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

[72] F. Van Boetzelaer, T. J. van den Boom, and R. Negenborn, *Model predictive scheduling for container terminals,* IFAC Proceedings Volumes **47**, 5091 (2014).

[73] T. Van Den Boom and B. De Schutter, *Modelling and control of discrete event systems using switching max-plus-linear systems,* Control engineering practice **14**, 1199 (2006).

[74] Y.-L. Cheng, H.-C. Sen, K. Natarajan, C.-P. Teo, and K.-C. Tan, *Dispatching automated guided vehicles in a container terminal,* Supply chain optimization , 355 (2005).

[75] A. Alessandri, S. Sacone, and S. Siri, *Modelling and optimal receding-horizon control of maritime container terminals,* Journal of Mathematical Modelling and Algorithms **6**, 109 (2007).

[76] A. Alessandri, C. Cervellera, M. Cuneo, M. Gaggero, and G. Soncin, *Modeling and feedback control for resource allocation and performance analysis in container terminals,* IEEE Transactions on Intelligent Transportation Systems **9**, 601 (2008).

[77] J. Pearl, *Heuristics: intelligent search strategies for computer problem solving,* (1984).

[78] F. S. Hillier and G. J. Lieberman, *Introduction to Operations Research, 10th Ed.* (McGraw-Hill, New York USA, 2015).

[79] J. N. Gupta, *Two-stage, hybrid flowshop scheduling problem,* Journal of the Operational Research Society , 359 (1988).

[80] Anonymous, *P versus np,* https://en.wikipedia.org/wiki/P_versus_NP_problem (2018), [Online; accessed 2018-1-26].

[81] B. Esfahbod, *Np- hardness,* https://commons.wikimedia.org/w/index.php?curid=3532181 (2017), [Online; accessed 2018-1-26].

[82] J. Leeuwen, *Handbook of theoretical computer science*, Vol. 1 (Elsevier, 1990).

[83] A. Cobham, *The intrinsic computational difficulty of functions,* (1965).

[84] A. Guinet and M. Solomon, *Scheduling hybrid flowshops to minimize maximum tardiness or maximum completion time,* International Journal of Production Research **34**, 1643 (1996).

[85] J. Cheng, Y. Karuno, and H. Kise, *A shifting bottleneck approach for a parallel-machine flowshop scheduling problem,* Journal of the operations research society of Japan **44**, 140 (2001).

[86] X. Wang and L. Tang, *A tabu search heuristic for the hybrid flowshop scheduling with finite intermediate buffers,* Computers & Operations Research **36**, 907 (2009).

[87] B. Wardono and Y. Fathi, *A tabu search algorithm for the multi-stage parallel machine problem with limited buffer capacities,* European Journal of Operational Research **155**, 380 (2004).

[88] C. Low, *Simulated annealing heuristic for flow shop scheduling problems with unrelated parallel machines,* Computers & Operations Research **32**, 2013 (2005).

[89] B. Naderi, M. Zandieh, A. K. G. Balagh, and V. Roshanaei, *An improved simulated annealing for hybrid flowshops with sequence-dependent setup and transportation times to minimize total completion time and total tardiness,* Expert systems with Applications **36**, 9625 (2009).

[90] B. Naderi, M. Zandieh, and V. Roshanaei, *Scheduling hybrid flowshops with sequence dependent setup times to minimize makespan and maximum tardiness,* The International Journal of Advanced Manufacturing Technology **41**, 1186 (2009).

[91] M. E. Kurz and R. G. Askin, *Scheduling flexible flow lines with sequence-dependent setup times,* European Journal of Operational Research **159**, 66 (2004).

[92] R. Ruiz and C. Maroto, *A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility,* European Journal of Operational Research **169**, 781 (2006).

[93] V. Yaurima, L. Burtseva, and A. Tchernykh, *Hybrid flowshop with unrelated machines, sequence-dependent setup time, availability constraints and limited buffers,* Computers & Industrial Engineering **56**, 1452 (2009).

[94] J. Jungwattanakit, M. Reodecha, P. Chaovalitwongse, and F. Werner, *Algorithms for flexible flow shop problems with unrelated parallel machines, setup times, and dual criteria,* The International Journal of Advanced Manufacturing Technology **37**, 354 (2008).

[95] J. Jungwattanakit, M. Reodecha, P. Chaovalitwongse, and F. Werner, *A comparison of scheduling algorithms for flexible flow shop problems with unrelated parallel machines, setup times, and dual criteria,* Computers & Operations Research **36**, 358 (2009).

[96] D. Abramson and M. Randall, *A simulated annealing code for general integer linear programs,* Annals of Operations Research **86**, 3 (1999).

[97] P. Baptiste, C. Le Pape, and W. Nuijten, *Constraint-based scheduling: applying constraint programming to scheduling problems*, Vol. 39 (Springer Science & Business Media, 2012).

[98] F. Rossi, P. Van Beek, and T. Walsh, *Constraint programming,* Foundations of Artificial Intelligence **3**, 181 (2008).

[99] D. Riera and N. Yorke-Smith, *An improved hybrid model for the generic hoist scheduling problem,* Annals of Operations Research **115**, 173 (2002).

[100] A. H. Land and A. G. Doig, *An automatic method of solving discrete programming problems,* Econometrica: Journal of the Econometric Society , 497 (1960).

[101] J. Clausen, *Branch and bound algorithms-principles and examples,* Department of Computer Science, University of Copenhagen , 1 (1999).

[102] Gurobi, *Mixed integer programming basics,* MixedIntegerProgrammingBasics (2017), [Online; accessed 2018-1-27].

[103] M. Pinedo, *Planning and scheduling in manufacturing and services* (Springer, 2005).

[104] Y. Mati, N. Rezg, and X. Xie, *A taboo search approach for deadlock-free scheduling of automated manufacturing systems,* Journal of Intelligent Manufacturing **12**, 535 (2001).

[105] F. Glover, *Future paths for integer programming and links to artificial intelligence,* Computers & operations research **13**, 533 (1986).

[106] F. Glover, *Tabu search—part i,* ORSA Journal on computing **1**, 190 (1989).

[107] F. Glover, *Tabu search—part ii,* ORSA Journal on computing **2**, 4 (1990).

[108] F. Glover, *Tabu search: A tutorial,* Interfaces **20**, 74 (1990).

[109] M. Malek, M. Guruswamy, M. Pandya, and H. Owens, *Serial and parallel simulated annealing and tabu search algorithms for the traveling salesman problem,* Annals of Operations Research **21**, 59 (1989).

[110] M. Langelaar, *Me46060(2016-2017) blackboard slides,* (2017), [Online; accessed 2018-1-29].

[111] V. Granville, M. Krivánek, and J.-P. Rasson, *Simulated annealing: A proof of convergence,* IEEE transactions on pattern analysis and machine intelligence **16**, 652 (1994).

[112] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, *Optimization by simulated annealing,* science **220**, 671 (1983).

[113] D. Connolly, *General purpose simulated annealing,* Journal of the Operational Research Society **43**, 495 (1992).

[114] J. Teghem, M. Pirlot, and C. Antoniadis, *Embedding of linear programming in a simulated annealing algorithm for solving a mixed integer production planning problem,* Journal of Computational and Applied Mathematics **64**, 91 (1995).

[115] C. Darwin and W. F. Bynum, *The origin of species by means of natural selection: or, the preservation of favored races in the struggle for life* (Penguin, 2009).

[116] M. Mitchell, *An introduction to genetic algorithms* (MIT press, 1998).

[117] Anonymous, *Selection (genetic algorithm),* https://en.wikipedia.org/wiki/Selection_(genetic_algorithm) (2018), [Online; accessed 2018-1-29].

[118] Anonymous, *Crossover (genetic algorithm),* https://en.wikipedia.org/wiki/Crossover_(genetic_algorithm) (2018), [Online; accessed 2018-1-29].

[119] P. Larranaga, C. M. Kuijpers, R. H. Murga, and Y. Yurramendi, *Learning bayesian network structures by searching for the best ordering with genetic algorithms,* IEEE transactions on systems, man, and cybernetics-part A: systems and humans **26**, 487 (1996).

[120] Anonymous, *Mutation (genetic algorithm),* https://en.wikipedia.org/wiki/Mutation_(genetic_algorithm) (2018), [Online; accessed 2018-1-29].

[121] Y. Zhang, S. Wang, and G. Ji, *A comprehensive survey on particle swarm optimization algorithm and its applications,* Mathematical Problems in Engineering **2015** (2015).

[122] J. Kennedy and R. Mendes, *Population structure and particle swarm performance,* in *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, Vol. 2 (IEEE, 2002) pp. 1671–1676.

[123] D. Bratton and J. Kennedy, *Defining a standard for particle swarm optimization,* in *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE* (IEEE, 2007) pp. 120–127.

[124] Y. Shi and R. C. Eberhart, *Parameter selection in particle swarm optimization,* in *International conference on evolutionary programming* (Springer, 1998) pp. 591–600.

[125] R. Roy, S. Dehuri, and S. B. Cho, *A novel particle swarm optimization algorithm for multi-objective combinatorial optimization problem,* in *Trends in Developing Metaheuristics, Algorithms, and Optimization Approaches* (IGI Global, 2013) pp. 1–16.

[126] W.-N. Chen, J. Zhang, H. S. Chung, W.-L. Zhong, W.-G. Wu, and Y.-H. Shi, *A novel set-based particle swarm optimization method for discrete optimization problems,* IEEE Transactions on evolutionary computation **14**, 278 (2010).

[127] M. Dorigo, V. Maniezzo, and A. Colorni, *Ant system: optimization by a colony of cooperating agents,* IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **26**, 29 (1996).

[128] K. Alaykỳran, O. Engin, and A. Döyen, *Using ant colony optimization to solve hybrid flow shop scheduling problems,* The international journal of advanced manufacturing technology **35**, 541 (2007).

[129] H. Izakian, A. Abraham, and V. Snasel, *Comparison of heuristics for scheduling independent tasks on heterogeneous distributed environments,* in *Computational Sciences and Optimization, 2009. CSO 2009. International Joint Conference on*, Vol. 1 (IEEE, 2009) pp. 8–12.

[130] E. D. Andersen and K. D. Andersen, *Presolving in linear programming,* Mathematical Programming **71**, 221 (1995).

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

[132] B. L. Welch, *The generalization of student's' problem when several different population variances are involved,* Biometrika **34**, 28 (1947).

[133] F. M. Dekking, *A Modern Introduction to Probability and Statistics: Understanding why and how* (Springer Science & Business Media, 2005).

[134] T. Jonker, *Predictive control for model AGVs on trajectories with unreliable measurements and communication*, Research assignment, TU Delft (2017).

[135] G. E. Box, W. G. Hunter, and J. S. Hunter, *Statistics for experimenters: an introduction to design, data analysis, and model building*, Vol. 1 (JSTOR, 1978).

[136] Y. Monden, *Toyota production system: an integrated approach to just-in-time* (CRC Press, 2011).

[137] J. P. Womack, J. P. Womack, D. T. Jones, and D. Roos, *Machine that changed the world* (Simon and Schuster, 1990).

Coordinated optimization of equipment operations on a container terminal

Tim Jonker, *4281004*
Supervisors: M.B. Duinkerken, R.R. Negenborn, N. Yorke-Smith, A. de Waal

***Abstract*—Increased international marine transport has increased the use of containers. This goes hand in hand with the increased demand for well-functioning container terminals. The speed at which containers can be handled on the container terminal is an important performance indicator. In particular, the productivity of the Quay crane (QC)s is used to determine the performance of a container terminal. To achieve a high performance, a well designed schedule of operations is required. This research investigates whether an all-encompassing model can be constructed that represents the waterside of a container terminal with the aim to promote coordination of multiple machines. Currently, an uncoordinated scheduling heuristic is used to dispatch the equipment operating on a container terminal. With a coordinated schedule, operations of all equipment working on the terminal can be considered at once to achieve optimal QC productivity. A Hybrid Flow Shop (HFS) model with revolutionary features is used to model the waterside of a container terminal. The revolutionary features are called bi-directional and job-pairs. The former enables jobs to move through the HFS in both directions, the latter constrains certain jobs to be performed simultaneously by a single machine. Subsequently, a tailored Simulated Annealing (SA) algorithm is used to obtain a heuristic solution to this mathematical model. To enable the coordinated schedule to be applicable to operational container terminals the SA algorithm should balance between both quality and computational time. Computational time should remain short since operations on a container terminal vary quickly and predictions with regards to processing times are unreliable.**

***Keywords*—Scheduling, Hybrid flowshop, coordination, Simulated Annealing**

I. Introduction

Today, 60% of the total transport volume is transported in standardized boxes called containers [1]. Therefore, the demand for well-functioning container terminals to process these containers increases. The function of a container terminal is to transship containers. Deep-sea vessels, barges, trains and trucks all arrive to deliver and pick up containers. The container terminal ensures that the correct containers are loaded and unloaded to and from their respective carriers. The speed at which this is achieved is an important benchmark for container terminals. In particular, the turnaround time of deep-sea vessels is used to determine the performance of a container terminal [2], [3]. This is because the berthtime of deep-sea vessels is expensive. Furthermore, container terminals wish to decrease the turnaround time of deep-sea vessels since the container terminal becomes more attractive for shipping lines to move their wares through. Moreover, the shorter the turnaround time of deep-sea vessels, the higher the container terminal capacity.

To achieve low turnaround times of deep-sea vessels, containers should be handled as fast as possible on the container terminal. An existing terminal can do three things to increase the handling speed of containers: 1) using state-of-the-art equipment or 2) buying more equipment or 3) using the current equipment more efficiently. The latter option is investigated within this research. By coordinating the operations of the equipment operating on the terminal, the equipment could be used more efficiently. Currently, the available scheduling models do not match with the operations on a container terminal. Furthermore, the available models cannot be used in real-time since the time to solve them is too long. Computational times should remain low since re-planning is often required at a container terminal because the current situation can change rapidly.

Our main contribution is the development of (i) a HFS model that contains two revolutionary features called, bi-directional and job-pairs to represent the container terminal mathematically. (ii) a new solution technique that efficiently approximates exact solutions to the developed mathematical model which allows for real time usage.

II. Container terminal layout and problem formulation

In this section we briefly introduce the layout of a container terminal and their known scheduling problems. A container terminal is primarily a material handling system; it is a link in the intercontinental transportation chain. Within this chain, the container terminal fulfills a buffer function between sea and land transportation. Furthermore, the container terminal provides secondary services like inspection, washing, repairs and cargo consolidation [4]. This research focuses on the waterside operations on a container terminal: transporting containers between the yard and the deep sea vessels. Three different types of equipment are used to perform waterside operations. These are Yard Crane (YC)s, Automated Guided Vehicle (AGV)s and Quay crane (QC)s. A schematic drawing of a typical container terminal is given in Figure 1. From this figure the tasks of the different equipment types becomes clear; QCs move containers between the ship and the quay, the AGVs move containers between the quay and the transfer point and YCs move containers within the stack.

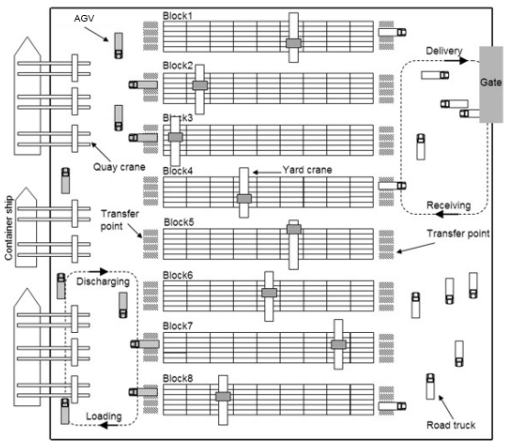


Fig. 1. Schematic drawing of a container terminal [5].

Master thesis, Transport Engineering and Logistics

Our main research topic is to investigate how a coordinated schedule could lead to increased productivity of Quay cranes on a container terminal.

A coordinated schedule applied to container terminals is a schedule that considers the operations of all equipment operating on the container terminal simultaneously to achieve overall optimal performance. Contrary to an uncoordinated schedule that considers operations of all equipment separately and tries to find optimal solutions for all equipment individually. Since different equipment operating on a container terminal could have conflicting objectives an uncoordinated planning does not lead to the optimal holistic solution. Therefore, a richer coordinated scheduling model than currently exists is desired to find the optimal holistic solution.

In general, schedules could be subdivided into three different categories which are strongly connected. These categories are: scheduling, dispatching and routing. Scheduling signifies the process of assigning operating times to the individual containers. Subsequently, dispatching is a process during which equipment is assigned to handle individual containers. Thereafter, routing describes the process of determining the routes of equipment required to handle the containers. Coordinated schedules are schedules that recognize the connection between the different categories and adapt their schedule accordingly.

Different scheduling problems on a container terminal are identified in Figure 2. The first two scheduling problems encountered on a container terminal upon arrival of a deep-sea vessel are berth allocation and bay scheduling [6]–[9]. Another problem which should be solved prior to the individual container scheduling problem is the problem of determining the yard location of containers [10]. This problem is addressed in [11]–[13]. The solutions to these scheduling problems define the starting point of the scheduling problem considered within this research. Individual container schedules and yard allocation are closely related. Therefore, the quality of the yard allocation solution partially determines the performance of the individual container handling schedule. This is why certain researchers integrated yard allocation with individual container scheduling [14].

This research focuses on the scheduling and dispatching problem on a container terminal. This problem consists of assigning equipment to handle operations and to determine the starting times of these operations. Scheduling and dispatching are strongly intertwined; operations cannot be performed simultaneously on the same machine and a machine requires a sequence dependent setup time to start a new operation. Good scheduling and dispatching increases the productivity of a container terminal [15]. Therefore, the investigation of optimal scheduling and dispatching receives great interest. Another problem present on container terminals is routing [10]. Routing considers the problem of determining the route of Lift Automated Guided Vehicle (Lift-AGV)s on the container terminal. Routing is often used to avoid congestions and collisions between different automated Lift-AGVs while guaranteeing short driving times. Routing involves path planning, area claiming, energy consumption and prioritization. Typically, routing problems are solved after scheduling and dispatching problems are solved. Since this research considers both scheduling and dispatching, routing is the first encountered problem after the individual container schedule is constructed. Routing problems are frequently studied and interested readers are referred to [16]–[18].

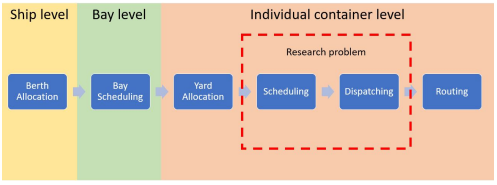


Fig. 2. Relation of the research problem with other logistic problems encountered on container terminals.

III. Literature overview

In this section we introduce the modelling strategy that is used to mathematically model the container terminal which is used to find optimal solutions. Furthermore, we place the research into perspective by comparing previous developed mathematical models.

A. Hybrid Flowshop Problem

The container terminal is formulated mathematically as a Hybrid Flow Shop (HFS) problem. HFS problems are problems where n different jobs should be processed in a series of m stages while optimizing an objective function [30], [31]. A job consists of multiple operations which should be performed on the different stages. Problems classify as HFS problems once the following characteristics apply [19], [30]:

1) The number of processing stages m is at least 2.
2) Each stage k has $M^k \geq 1$ machines in parallel and at least one stage has $M^k > 1$.
3) All jobs follow the same production flow through the stages. However, a job is allowed to skip one or more stages.
4) Each job j requires a processing time p_{jk} on stage k.

Within the standard form of the HFS problem, each job is available from the start and can be processed on any of the different parallel machines within a stage. Furthermore, each machine is equal and has the same deterministic processing time which is known in advance. Moreover, machines can only serve one job simultaneously. Additionally, sequence dependent setup times are negligible, buffers between stages are unlimited and preemption of operations is not allowed. However, the standard form of the HFS problem is not directly applicable to the container terminal at hand. Only a few aspects need to be modified.

At a container terminal, a job signifies an order since it requires operations in a specified order by different types of machines. All machines of equal type compose a stage. Therefore, the waterside of a container terminal can be represented by a three-stage HFS problem that operates in both directions.

Modelling a container terminal as a HFS problem is not new. Therefore, most research is dedicated either to additional features [20]–[22], [25], [32], [33] or to the solution technique [2], [15], [19], [20], [23]–[28]. The aim of our research is to reduce the computational time required to solve the HFS problem. Reduction of the computational time is a major hurdle before a scheduling algorithm can be used on an operational terminal. This is because schedules should be created and adapted in real time. HFS problems are complex NP-hard problems [34]. NP-hard problems are problems which cannot be solved in polynomial time as a function of the problem size [35]. Furthermore, NP-hard problems do not necessarily need to be verifiable in polynomial time either [36]. However, NP or NP-hard problems are not necessarily "hard problems" [37]. This is also the case for the HFS problem [35]. Therefore, many larger HFS problems

TABLE I. CLASSIFICATION SCHEME OF OTHER PAPERS CONSIDERING INTEGRATED CONTAINER TERMINAL SCHEDULING. EXTENDED WITH THE MAXIMUM MANAGED PROBLEM SIZE (# OF CONTAINERS, QCS, AGVS AND YCS) AND THE APPLIED SOLUTION TECHNIQUE.

Article		α	β	γ	Cont	QC	HTV	YC	Solution technique
Chen	[19], [20]	$FH3,((RM)^k)_{k=1}^3$	$M_j,S_{sd},block$	C_{max}	500	5	30	20	Tabu Search, Constraint programming
Lau	[15]	$FH3,((RM)^k)_{k=1}^3$	$prec,S_{sd},M_j$	$\bar{C}^w$	16	2	4	2	Multi Layer Genetic Algorithm
Bish	[2]	$FH2,((PM)^k)_{k=1}^2$	$M_j,S_{sd},block$	$\bar{C}^w$	2500	4	40	-	Transshipment Based Heuristic
Xin	[21], [22]	$FH3,((RM)^k)_{k=1}^3$	$block$	C_{max}	40	5	10	8	Exact, Variable Neighbourhood Search
Lu	[23]	$FH2,((PM)^k)_{k=1}^2$	$p_{ij}\sim N(\mu,\sigma),S_{sd},block$	T_{max}	30	-	4	1	Particle Swarm Optimization
Kim	[24]	$FH2,((PM)^1,(RM)^k)_{k=2}^3$	S_{sd}	$\bar{L}+\bar{C}$	100	2	7	5	Dispatching method
He	[25]	$FH3,((RM)^k)_{k=1,3},(PM)^2$	$prec,S_{sd}$	$\bar{L}^w+energy$	500	10	50	20	Genetic Algorithm
Kaveshgar	[26]	$FH2,((RM)^k)_{k=1}^2$	$block,prec$	C_{max}	1050	2	15	-	Genetic Algorithm
Cao	[27]	$FH2,((PM)^k)_{k=2}^3$	$S_{sd},M_j,block$	C_{max}	500	-	60	40	MIP+Benders Cut
Cao	[28]	$FH2,((PM)^k)_{k=2}^3$	$S_{sd},M_j,block$	C_{max}	15	1	4	-	Genetic Algorithm + Heuristic
Kizilay	[29]	$FH2,((PM)^k)_{k=1,3}$	$prec,S_{sd},M_j$	C_{max}	550	4	∞	6	MIP, Constraint programming

use (meta)heuristics to approximate the exact solution within reasonable time. However, if the problem is small enough, a branch and bound technique is typically used to arrive at an exact solution. Popular metaheuristics are Genetic Algorithm (GA), tabu search and Simulated Annealing (SA). Jungwattanakit et al. [38] found that SA is superior to GA and Tabu Search for large scale HFS problems. Therefore, we will use SA in this research.

B. Simulated annealing

Simulated Annealing (SA) is a stochastic metaheuristic involving randomness. The metaheuristic is based on the annealing process of metals. It is particularly effective to locate the global minimum of large problems. The core thought is that the solution iteratively progresses towards the global optimum. Initially, the algorithm quickly explores the search space by accepting a large portion of moves, both improving and worsening. This allows the algorithm to escape local minimums. As time progresses, the algorithm becomes more selective and starts favouring improving moves only [39], [40].

The basic working principle of SA is as follows [39]:

1) Select an initial temperature T and solution $\mathbf{x}$;
2) Randomly generate a new solution $\mathbf{y}$ in the neighbourhood of $\mathbf{x}$ and subsequently evaluate $f(\mathbf{y})$;
3) If $f(\mathbf{y}) < f(\mathbf{x})$, accept $f(\mathbf{y})$. Otherwise, compute the probability $P_{accept}(T)$ (Equation 1) of accepting the new solution $f(\mathbf{y})$. Use a random number generator to decide whether or not to accept $f(\mathbf{y})$;
4) Reduce the temperature and return to step 2.

The probability to accept a worsening move is expressed with Equation 1. This chance gradually decreases towards zero with decreasing temperature, meaning that only improving moves are selected [41]. In other words, in the beginning SA is exploratory, whereas later SA becomes more investigative.

$$P_{accept}(T) = e^{\frac{f(\mathbf{x})-f(\mathbf{y})}{T}} \tag{1}$$

C. Hybrid flow shops applied to integrated container terminal scheduling

To compare different HFS problems with each other a classification scheme is introduced in [30], [31], [42]. The classification scheme consists of three categories: α,β,γ. These categories respectively identify the structure of the shop, the additional features and the objective function. We use this classification to classify previous efforts to construct integrated container terminal schedules applying HFS models. The collection of previous efforts is shown in Table I. This table also indicates the problem size and the applied solution technique.

IV. MATHEMATICAL MODEL FORMULATION

In this section we present the mathematical model and the applied solution technique to solve this mathematical model.

A. Model description

Since a HFS is used to describe the coordinated container scheduling problem, the problem can be classified with the $\alpha|\beta|\gamma$ classification scheme introduced in [30], [31], [42]. First, the problem will be described in detail to highlight all facets of the HFS. Subsequently, the $\alpha|\beta|\gamma$ classification will be provided.

- Within the HFS model that we used there are three stages with unrelated parallel machines.
- Precedence relations exist at the QC stage to ensure that containers can only be (un)loaded in a specific sequence.
- Sequence dependent setup times are used to describe the back route of equipment. A back route is a route that equipment should travel to pickup their next container after delivering a container.
- Machine eligibility is applied at the YC and QC stage to ensure that certain containers can only be handled by specific machines.
- Blocking is used to constrain the available buffer space between different stages. Blocking signifies that equipment can only proceed with its next operation once the next equipment type becomes available. This means that certain containers block a machine for as long as the next machine is not available. In our case blocking applies on all stages.
- Bi-directional flow shop is a new feature of the HFS model developed during this research. It enables containers to move in both directions through the shop. This is required since both loading and unloading containers are simultaneously present on the container terminal.
- Release dates signify that machines and containers become available not directly from the start. This enables the mathematical model to be constructed at an arbitrary time during operation of the container terminal.
- Job pairs is a revolutionary feature that is used to enable machines to simultaneously handle two jobs. More specifically, job pairs should finish at the same time but, could be started at different times. Job pairs are used to enable the HFS model to handle twin lift moves. Twin lift moves are moves consisting of two containers that require simultaneous handling at the QC stage and the Lift-AGV stage. Job pairs could be formed differently on different stages and could have a different processing time dependent on their pick-up sequence.
- The objective of the HFS is to minimize the makespan of the entire container plan. Primarily, the QC should be finished as early as possible.

All together the HFS problem can be classified with the $\alpha|\beta|\gamma$ classification as:

$$FH4,((RM)^k)_{k=1}^4|r_j,M_j,S_{sd},block,prec,Bidirec,Jobpairs|\bar{C}$$

The features Bi-direc and Job-pairs are newly introduced in this research. Bi-direc is used to describe the relaxation

TABLE II. A TABLE TO COMPARE THE MODEL DEVELOPED IN THIS RESEARCH WITH PREVIOUS EFFORTS, INDICATING THE STRUCTURE, FEATURES AND OBJECTIVE VALUE ACCORDING TO THE CLASSIFICATION SCHEME OF [30], [31], [42]

Article		Stages	Machine type	M_j	S_{sd}	block	prec	$p_{ij} \sim N(\mu,\sigma)$	Bidirec	Jobpairs	Objective
Chen	[19], [20]	3	R	x	x	x					C_{max}
Lau	[15]	3	R	x	x		x				$\bar{C}^w$
Bish	[2]	3	P	x	x	x					$\bar{C}^w$
Xin	[21], [22]	3	R			x					C_{max}
Lu	[23]	2	P		x	x		x			T_{max}
Kim	[24]	2	P		x						$\bar{L}+\bar{C}$
He	[25]	3	P,R		x		x				$\bar{L}^w + energy$
Kaveshgar	[26]	2	R			x	x				C_{max}
Cao	[27]	2	P	x	x	x					C_{max}
Cao	[28]	2	P	x	x	x					C_{max}
Kizilay	[29]	2	R	x	x		x				C_{max}
This research		**3**	**R**	**x**	**x**	**x**	**x**		**x**	**x**	$\bar{\mathbf{C}}^{\mathbf{w}}$

of unidirectional flow in the flow shop. Job-pairs is a feature introduced to specify that some containers require simultaneous handling on a stage by one machine. All together the HFS model developed within this research is the most extensive HFS model that was ever used to model a container terminal. Table II can be used to compare different efforts to model container terminals. Figure 3 shows a schematic overview of the container terminal considered in this research. Each container can only be handled by one specific Yard Crane (YC) but the sequence of operations is variable. Each YC has access to its own transfer point modelled as a buffer. This means that the buffer stage contains multiple spots to place containers. The buffer is not shared between different YCs. At the QC stage, not only the specific machine but, also the sequence of operations is known in advance.

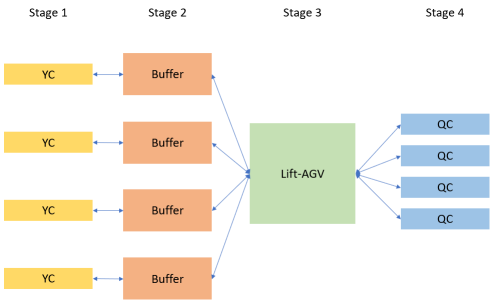


Fig. 3. Illustration of the HFS applied to a container terminal. The YC and QC stages are represented by multiple small blocks since the machine assignment is fixed. The Lift-AGV stage is represented by a large block since the machine selection is an element of the to be developed schedule. Arrows are used to indicate how the different blocks are connected. Containers can only move along the arrows.

All together the HFS model used in this research is the most extensive HFS that has ever been used to model a container terminal. In Table II the HFS model used within this research is compared with the HFS models used by other researchers.

B. Mathematical model

In this section our HFS model is translated into a mathematical formulation. Since the problem size is often a bottleneck for HFS problems it is of paramount importance to keep the number of variables to a minimum. Nevertheless, two artificial jobs are created which define the first and the last job of a sequence [27].

The first step towards creating a model is to declare certain parameters.

i,k job index $i,k \in \phi_1 \cap \phi_2$
j stage index $j \in 1,2,3,4$
m machine index $m \in 1,2,...n_j$
n_j number of machines on stage j
N total number of containers
ϕ set of all jobs excluding $i=0, i=N+1$
ϕ_1 set of all jobs including the first dummy job $i=0$
ϕ_2 set of all jobs including the last dummy job $i=N+1$
P set of job pairs with precedence relations
O_{ij} operation i at stage j
M_{ij} set of machines which can process O_{ij}
E_m set of jobs that can be performed on machine m, includes jobs $i=0$ and $i=N+1$.
K_{ij} set of jobs that can form a pair with job i on stage j, includes jobs $k=0$ and $k=N+1$.
T set of twinlift operations.
T_i the twin pair of container i.
A set of containers that is already assigned to equipment.
Pr set of containers that is already assigned to equipment but not in current progress.
J_i set of stages that a container i already past or is being currently processes job i.
Q_{jm} number of jobs that must be performed on machine m
r_{ij} release date of job i on stage j.
p_{ij} processing time of operation O_{ij}
p_{ij}^{ik} processing time of operation O_{ij} whenever operation i directly precedes operation k.
p_{ij}^{ki} processing time of operation O_{ij} whenever operation k directly precedes operation i.
s_{ikj} setup time between job i and job k at stage j
s_{0kjm} setup time between the dummy job and job k on stage j for machine m.
H a sufficiently large constant.
β a constant indicating the weight of completion time of operations on the YC stage, $\beta \ll 1$.

The stage indices 1 to 4 respectively represent: the YC stage, the rack stage, the Lift-AGV stage and the QC stage.

Next, the decision variables are declared:

- t_{ij} The start time of job i on stage j
- z_{ikj} 1, if O_{ij} immediately precedes O_{kj}; 0, otherwise $\forall i \in \phi, \forall k \in \phi_2, \forall j \in \{1,2,3,4\}$.
- z_{0kjm} 1, if O_{0jm} immediately precedes O_{kj}; 0, otherwise $\forall k \in \phi_2, \forall j \in \{1,2,3,4\}, \forall m \in M_{ij}$.

The parameters p_{ij}^{ik} and p_{ij}^{ki} are only required for twin moves ($\forall i,k \in T$) since twin moves are represented by two sequential single moves with different processing times. Whenever i,k are pairs of a twin move p_{ij} is replaced by $p_{ij}^{ik} z_{ikj} + p_{ij}^{ki} z_{kij}$.

The integrated scheduling problem can be formulated in a HFS style.

$$\text{Minimize} \quad Z = \beta \sum_{i=1}^{N} t_{i1} + \sum_{i=1}^{N} t_{i4} - \beta \sum_{i=1}^{m} \sum_{k=1}^{N+1} z_{0kjm} \tag{2}$$

$$\text{Subject to} \quad t_{ij} \geq r_{ij} \qquad \forall i \in \phi, \forall j \in \{1,4\} \tag{3}$$

$$H(1 - z_0kjm) + t_kj \geq s_0kjm \qquad \forall k \in \phi \cap A^C, j \in \{3\}, \forall m \in n_j \tag{4}$$

$$t_{ij} + p_{ij} \leq t_{i(j+1)} \qquad \forall i \in \phi \cap T^C, \forall j \in \{3,4\} \tag{5}$$

$$t_{ij} + p_{ij} \leq t_{i(j+1)} \qquad \forall i \in \phi, \forall j \in \{1,2\} \tag{6}$$

$$t_{ij} + p_{ij}^{ik} z_{ikj} + p_{ij}^{ki} z_{kij} \leq t_{i(j+1)} \qquad \forall i \in T, \forall j \in \{3,4\} \tag{7}$$

$$\sum_{i=1}^{N} \sum_{k=1}^{N} z_{ikj} = Q_{jm} + 1 \qquad \forall i,k \in E_m, \forall j \in \{1,2,3,4\}, \forall m \in M_{ij} \tag{8}$$

$$\sum_{k=1}^{N} z_{0kj} = n_j \qquad \forall k \in E_m, \forall j \in \{1,2,3,4\} \tag{9}$$

$$\sum_{i=1}^{N} z_{i(N+1)j} = n_j \qquad \forall i \in E_m, \forall j \in \{1,2,3,4\} \tag{10}$$

$$\sum_{k=1}^{N+1} z_{ikj} = 1 \qquad \forall i \in \phi, \forall k \in \phi_2 \cap K_{ij}, \forall j \in \{1,2,3,4\} \tag{11}$$

$$\sum_{i=0}^{N} z_{ikj} = 1 \qquad \forall i \in \phi_1 \cap K_{kj}, \forall k \in \phi, \forall j \in \{1,2,3,4\} \tag{12}$$

$$z_{ikj} + z_{kij} = 1 \qquad \forall i,k \in T, j \in \{3,4\} \tag{13}$$

$$t_{i(j+1)} + s_{ikj} \leq t_{kj} + H(1 - z_{ikj}) \qquad \forall i,k \in \phi \cap T^C, \forall j \in \{1,2,3,4\} \tag{14}$$

$$t_{ij} + p_{ij} + s_{ikj} \leq t_{kj} + H(1 - z_{ikj}) \qquad \forall i,k \in \phi \cap T^C, \forall j \in \{4\} \tag{15}$$

$$t_{ij} + p_{ij} + s_{ikj} \leq t_{kj} + H(1 - z_{ikj}) \qquad \forall i,k \in \phi, \forall j \in \{1,2\} \tag{16}$$

$$t_{ij} + p_{ij}^{ik} z_{ikj} + p_{ij}^{ki} z_{kij} + s_{ikj} \leq t_{kj} + H(1 - z_{ikj}) \qquad \forall i,k \in T, \forall j \in \{3,4\} \tag{17}$$

$$\sum_{i=1}^{N} t_{ij} \leq 0 \qquad \forall i \in A, \forall j \in J_i \tag{18}$$

$$\sum_{i=1}^{N} t_{ij} \leq H(1 - z_{ik3}) \qquad \forall i,k \in A \cap T, \forall j \in J_i \tag{19}$$

$$\sum_{i=1}^{N} z_{0kjm} + z_{ikj} = 1 \qquad \forall i \in A \cup T_k, \forall k \in Pr, \forall j \in \{1,2,3\} \tag{20}$$

$$z_{ikj}, z_{ikjm} = 0 \ or \ 1 \qquad \forall i,k \in \phi_1 \cup \phi_2, \forall j \in \{1,2,3,4\} \tag{21}$$

C. Mathematical model description

This section will be used to give a textual description to all constraints presented in the previous section. In the objective function (Equation 2) the makespan is minimized adding the most weight to the completion time of operations at the QC stage. Since each operation is considered in the objective function, most weight is indirectly added to the first few operations on each QC. This is because delays in the beginning propagate to other operations. This is a desired feature of the objective function since the first few operations are most accurately predicted and are most important to be finished on time. Besides that, distribution of jobs to different machines is stimulated by the small negative factor β. Whenever the makespan is equal, the jobs should be evenly distributed over all machines.

Constraint 3 enforces that jobs can only start after they are being released. This is only required at the starting stage and the first job of the sequence since other operations and jobs are forced to start later. Constraint 4 has a similar function as Constraint 3 but, focuses on the Lift-AGV stage. This is required since it is unknown which container is handled by which Lift-AGV and each Lift-AGV can become available independently.

Constraint 5 ensures that a job can only start at the next stage once the current stage is completed. Constraint 7 is required to make use of the sequence dependent processing times of twin moves. However, at the YC stage containers cannot be twin containers. Therefore, the processing time remains unchanged and thus Constraint 5 applies for all containers in ϕ. This is expressed by Constraint 6.

Constraint 8 describes machine eligibility at the YC stage. It ensures that a sequence of z_{ikj} is only selected from a subset of ϕ. Furthermore, the number of sequence variables is constraint to the number of jobs that should be handled on the machine plus one. This is because each sequence starts and ends with a dummy job.

Constraint 9 ensures that the number of started sequences is equal to the number of machines. However, once a sequence cannot start at an arbitrary job, that machine receives its personal constraint. The personal constraint enforces that a sequence should start at one of the available jobs. Constraint 10 is similar to Constraint 9 but, now to ensure that each sequence is ended. Similarly to Constraint 9, if a machine cannot process every job; that machine is given its personal variant of Constraint 10.

Constraint 11 ensures that each operation is proceeded by another operation or is the end of the sequence. Similar is Constraint 12, which ensures that each operation is preceded by another operation or is the start of the sequence. Both of these constraints exclude infeasible job pairs whenever they are obliged to be

performed on different machines.
Constraint 13 ensures that a pair of twin moves is handled by the same machine. This is not required at stage 1 or 2 since different YCs might work on the same twin pair.
Constraint 14 is the blocking constraint; it states that whenever job i immediately precedes job k on stage j, job k cannot start earlier than the start time of job i on the next stage plus the setup time between job i and job k. This is because the start of job i on the next stage marks the point where the container is released from the current stage.
Constraint 15 is used to separate operations by their processing times and setup times. Practically, this constraint is only required at the first and last stage since intermediate stages automatically satisfy this constraint by a combination of Constraint 5 and Constraint 14. Constraint 17 is the same as Constraint 15 only for twin moves which have their processing time dependent on the sequence of operations. For the same reason as for Constraint 6, Constraint 16 is added specifically for the first stage since the twin moves are separated here and thus processing times are sequence independent.
Constraint 18 ensures that containers that have already passed a stage or that are currently processed on a stage start their operation at time zero. However, this constraint is not valid whenever an unloading twin container is currently assigned to a QC or to a Lift-AGV. This is because twin containers are modelled as sequential single moves with different processing times. Therefore, Constraint 19 is introduced which specifies that at least one of the twin containers starts at zero. Constraint 13 subsequently ensures that the other container of the twin pair is processed directly after its twin pair.
Constraint 20 ensures that operations that are assigned to equipment but, that have not currently started can only proceed: containers that are in current progress or containers that have passed that stage already or their twin pair or the dummy start container.
Finally, Constraint 21 is used to enforce binary solutions to the decision variables z_{ikj} and z_{0kjm}.

Note that whenever $j+1$ is listed the next stage is meant, not necessarily being stage $j+1$. This is because for unloading the next stage is stage $j-1$. Furthermore, since $1 \leq j \leq 4$, $j \in \{1,2,3\}$ whenever $j+1$ is required and loading containers are considered. Whenever unloading is considered and $j-1$ is required; $j \in \{2,3,4\}$.
Since the sequence of operations on stage 4 is already known entirely, all the associated z variables become superfluous as decision variables.
Additionally, variable z_{iij} does not exist; a job can never immediately proceed or precede itself. Therefore, these variables are excluded.

The total number of decision variables is dependent on the number of stages, the number of containers and the number of Lift-AGVs. The number of variables can be expressed with Equation 3.

$$\begin{aligned} \#\ of\ variables\ &= N \cdot (\#\ of\ stages) + (N+1) \\ &\cdot (n_2 + n_3) + (N+1)^2 + 2(N^2+N) \end{aligned} \tag{3}$$

Equation 3 is based on the situation at TBA where the containers can only be planned on one YC and one QC. The Lift-AGVs are allowed to handle any container except that they cannot preempt a container handling. Furthermore, the sequence of operations at the QC is known in advance and thus requires no decision variables.

V. Solution technique

In this section we describe how the mathematical model can be solved efficiently. The efficiency is defined as the balance between the performance and the computational time of the algorithm. First, a general overview of the solution technique is presented in Figure 4 followed by a brief description of each block. Subsequently, the different blocks will be discussed in more detail.
The algorithm used to solve the HFS model is a tailored SA algorithm. The algorithm uses local search and a temperature based accepting/rejection criterion to arrive at new solutions. The first step is to generate an initial solution. Two different approaches are used for this and the best solution is selected as initial solution. Subsequently, the initial solution is modified locally to investigate better alternatives. The local modification is performed by altering the sequence of operations on the different machines. To do this, 2 containers are selected to be either inserted or swapped. This creates a neighbouring sequence. The neighbouring sequence might require a repair to remain feasible. Thereafter, the quality of the newly obtained neighbouring sequence can be evaluated with LP. The quality of the neighbouring sequence and the temperature schedule are used to decide whether the new solution is accepted or rejected. Whenever all these steps are completed everything will be updated to be ready for the next cycle. Once a stopping criterion is satisfied, the incumbent solution is returned as the final solution.

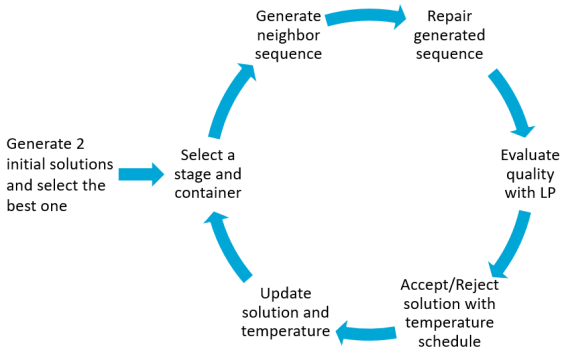


Fig. 4. The SA algorithm visualized by breaking it up in successive blocks.

A. The initial solution

To find an initial solution, two different solutions are generated from two different perspectives. The first solution is created from a container perspective and the other from a machine perspective.

For the container perspective, all containers are sorted based on their desired starting times at the first stage. Subsequently, the containers are scheduled in that order on all stages. This initial solution exploits the fact that the sequence of operations is known at the QC stage. Another name for this initial schedule is a non-delay schedule since containers are never deliberately delayed.

When scheduling from a machine perspective, all machines and all containers are sorted based on their availability. Machine availability does also consider the setup time to a particular container. Subsequently, the machine which comes first available is paired with the container that becomes first available. This is different from the container perspective since containers are not scheduled through the entire shop immediately.

B. The selection algorithm

After obtaining an initial solution the solution could be modified locally to investigate whether there is a better adjacent solution. The modification consists of a swap or insert of two containers to update the sequence of operations. A prerequisite is to select two containers that will be used to modify the original sequence. Since the loadplan is fixed, the sequence of operations at the QC cannot be changed. Therefore, containers are selected from the YC stage and from the Lift-AGV stage. At the YC stage; containers are selected randomly. This is because the possible variation of the schedule is small since containers cannot switch machines.

At the Lift-AGV stage; containers are selected in a predictive way. The aim is to predict which combination of two containers is most likely to improve the schedule the most. This means one container that is associated with delays and one container that precedes an idle period of the Lift-AGV. This selection process involves two steps. First, containers are sorted based on the delay that they cause in the QC schedule. These containers are marked yellow in Figure 5. The longer the delay, the more likely it becomes to select that container. Randomness is important to promote diversity and to prevent the algorithm from getting stuck in a local minimum. Let us assume that the container marked pink in Figure 5 is selected. This becomes the first of two containers. The second step is to select another container with some slack. This is done by selecting the containers with slack within the black box in Figure 5. The black box is drawn surrounding the first selected container. Thereafter, the more slack a container has, within the black box, the more likely it is to be selected. This selection ensures that two containers are selected which are likely to reduce the objective value the most. During the selection process, machine eligibility is taken into account. Furthermore, containers that are already in progress cannot be reassigned to another machine since preemption is not allowed.

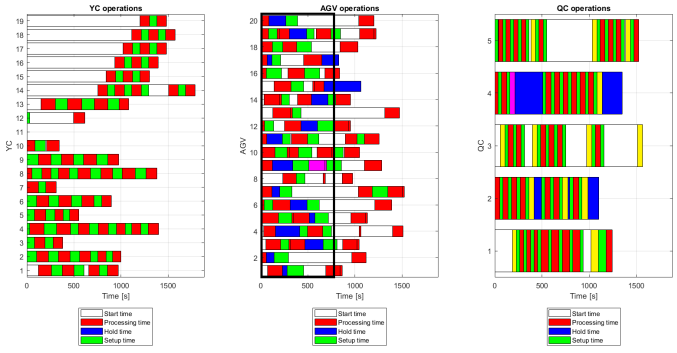


Fig. 5. A schedule of the YC, Lift-AGV and QC, the yellow boxes represent containers that cause delay in the QC schedule. The pink box represents the container that causes the most delay in the QC schedule. Processing time is the productive working time of a machine on a container. Setup represents the driving empty time between two containers. Hold time is the time that a machine cannot proceed with its next operation since there are no discharge opportunities for the current operation. Within white areas a machine is idle.

C. Generate the neighbour sequence

To obtain a new solution that reassigns the previously selected containers. The initial solution is first split into two parts. Namely, the sequence of operations (the discrete variables) and the associated starting times (the continuous variables). Subsequently, the sequence of operations is altered by swapping or inserting one container after another. The containers used for this are obtained with the selection algorithm. Generally, inserts are more productive than swaps. However, it was found that allowing a small portion of swaps does lead to a better performance compared to inserts alone. Presumably this is because diversity is maintained. Important is that upon each iteration, only one swap/insert can be performed on either the YC or Lift-AGV stage. Randomness determines which of the four options is performed.

D. Repair algorithm

The downside of performing a swap or insert in the sequence of operations is that it could produce an infeasible sequence of operations. This is because precedence or blocking constraints could be violated. Whenever this happens, the sequence of operations should be repaired. An example of this phenomenon is presented in Table III. These types of errors are easily repaired by sorting the containers on the machine to be repaired in the same order as they are on the sequence that is used to fix the faulty sequence. This type of fix is also indicated in Table III.

TABLE III. LEFT: A FAULTY SEQUENCE. RIGHT: A REPAIRED SEQUENCE. THE SEQUENCE IN THE LEFT IS FAULTY SINCE THE ORDER OF CONTAINERS HANDLED ON AGV1 IS DIFFERENT FROM THE ORDER OF HANDLED CONTAINERS ON QC1

AGV1	2	7	1	9	5
AGV2	6	3	4	8	10

QC1	1	2	3	4	5
QC2	6	7	8	9	10

AGV1	1	7	2	9	5
AGV2	6	3	4	8	10

QC1	1	2	3	4	5
QC2	6	7	8	9	10

E. Quality of the solution

With all sequence variables feasibly assigned, the starting time of each operation should be determined. These starting times also determine the quality of the new solution. The starting times can be determined optimally with Linear Programming (LP). LP is fast since there are only continuous variables left. Gurobi's LP solver is used to obtain optimal solutions quickly.

F. Accepting criterion

Once the neighbour solution is an improvement with respect to the original solution the neighbour solution is automatically accepted. Otherwise, the probability of accepting the new solution is dependent on its quality and the temperature (see Equation 1). The temperature is dependent on a temperature schedule that progresses along with the iterations. The temperature schedule allows users of the algorithm to balance between exploratory and investigative behaviour. The temperature schedule that was used is presented in Figure 6. All constants that are used to specify the temperature schedule are based on the performance on a limited number of samples. Therefore, it is not excluded that there are better temperature schedules.

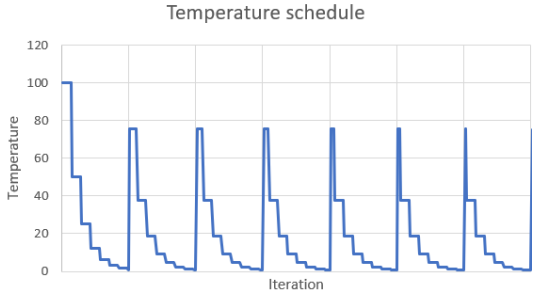


Fig. 6. The temperature schedule used by the SA algorithm.

G. Stopping criterion

The optimization cycle is terminated whenever one of the three stopping criteria is encountered. The first stopping criterion is a maximum number of iterations. This maximum number of iterations is determined based on experiments indicating the performance of the algorithm with respect to the number of iterations. The results of these experiments are shown in Figure 7. It is chosen to select 550 iterations as the maximum since the possible further gain in performance is low with respect to the required number of iterations.
The second stopping criterion specifies that whenever the incumbent solution does not update for 350 consecutive iterations, the optimization is terminated.
Finally, whenever there are no delays in the QC schedule anymore, the optimization cycle is terminated.

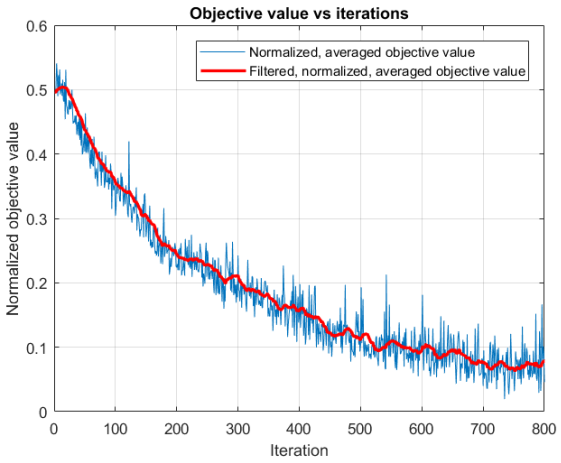


Fig. 7. Averaged, normalized progress of the objective value of 40 different instances with 5 QCs obtained from eM-Plant.

VI. Computational results

The performance of the algorithm is assessed theoretically by comparing the obtained objective values with those obtained with the exact solution technique Branch and Bound (B&B). Furthermore, the time required to arrive at these solutions is recorded as well. These experiments are repeated for eight different sized terminals. In case of SA, the average result over 5 different attempts is published to mitigate the effects of randomness. Besides that, the SA algorithm is also evaluated when more time is available. This obtained solution is called the better solution. To obtain the better solution the maximum number of iterations is increased to 1000 and the SA is repeated 10 times with different initial solutions. Subsequently, the best found solution is selected. The results of these theoretical experiments are published in Table IV. The computational times shown are obtained with an Intel i7-7700HQ CPU @ 2.8 GHz processor. To indicate the scaleability of the algorithm, results obtained on larger sized terminals are also shown in Table IV. However, the comparison with B&B is not made since computational times of B&B solutions are too excessive.

Based on the results shown in Table IV it can be concluded that the computational times realised by B&B are too long to be feasible for use on an operating terminal. The computational times obtained with the quick SA algorithm are much shorter and do allow for repetitive continuous re-planning on an operating container terminal. Furthermore, the optimality gap is relatively low. There is a considerable improvement between the different SA variants in terms of quality. However, also the time to arrive at the solution increased significantly. Therefore, for operational terminals it is advised to use the quick solution. On the other hand, the better solution verifies that the SA algorithm is able to find a solution with the shortest makespan in most cases.

When comparing the obtained results with results published in other papers [15], [19]–[22], [25] the new developed SA algorithm proves to be well balanced between performance and computational time.

TABLE IV. Optimization results for 8 different sized instances. The SA results have a quick solution (1 replication 550 iterations) and a better solution (10 replications 1000 iterations). The number of AGVs, YCs and Containers (CNT) is a function of the number of QCs. A - indicates that no results are available. This is because B&B did not find solutions within reasonable time.

Problem size				Branch and Bound			Simulated annealing quick solution					Simulated annealing better solution				
QC	AGV	YC	CNT	Objective value	Makespan (MS) [s]	Time [s]	Objective value	Makespan (MS) [s]	Time [s]	Opt gap	MS gap	Objective value	Makespan (MS) [s]	Time [s]	Opt gap	MS gap
1	4	3	10	7299.8	1568	0.1	7313.4	1568	1.3	0%	0%	7313.4	1568	21.7	0%	0%
2	8	5	20	19316.8	1784	0.9	20619.7	1971	1.5	7%	11%	19429.4	1816	23.9	1%	2%
3	12	8	30	40856.5	2857	16.3	42015.3	2932	1.8	3%	3%	41006.8	2857	30.5	0%	0%
4	16	10	40	41764.1	1960	71.1	42332.2	2008	2.1	1%	2%	42310.9	2008	29.7	1%	2%
5	20	13	50	50791.1	1910	2928.4	52672.1	1969	3.1	4%	3%	51517.2	1917	48.7	1%	0%
6	24	15	60	63987.5	2093	3762.8	66139.6	2169	2.5	3%	4%	65900.6	2169	43.1	3%	4%
7	28	18	70	69608.4	2373	7206.1	77214.0	2600	2.8	11%	10%	73937.1	2373	52.2	6%	0%
8	32	20	80	90604.23	2340	7206.1	95052.6	2392	3.3	5%	2%	93617.7	2340	57.6	3%	0%
9	36	23	90	-	-	-	102894.6	3330	3.8	-	-	101007	3330	64.3	-	-
10	40	25	100	-	-	-	104969.8	2173	4.4	-	-	103365.4	2141	69.6	-	-
11	44	28	110	-	-	-	110268.7	2369	4.6	-	-	109714.6	2369	78.0	-	-
12	48	30	120	-	-	-	131096.5	2819	4.9	-	-	128267.2	2634	84.3	-	-
13	52	33	130	-	-	-	142717.3	2687	5.4	-	-	140837.6	2468	100.5	-	-
14	56	35	140	-	-	-	164543.3	3012	6.0	-	-	163711.6	2787	108.7	-	-
15	60	38	150	-	-	-	159654.5	2423	6.7	-	-	157237.3	2397	117.0	-	-
16	64	40	160	-	-	-	169312.9	2589	7.5	-	-	164655.5	2550	135.8	-	-
17	68	43	170	-	-	-	192067.5	2599	8.9	-	-	190296.5	2599	148.6	-	-
18	72	45	180	-	-	-	212399.7	2566	9.5	-	-	206932.7	2420	169.3	-	-
19	76	48	190	-	-	-	218903.8	2854	10.0	-	-	213542.3	2816	178.4	-	-
20	80	50	200	-	-	-	214730.0	2434	11.3	-	-	209483.9	2404	194.2	-	-
21	84	53	210	-	-	-	245257.8	3011	12.5	-	-	239076	2995	216.6	-	-
22	88	55	220	-	-	-	244952.5	2299	14.4	-	-	237181.5	2163	254.0	-	-
23	92	58	230	-	-	-	288390.8	2966	14.8	-	-	278780.2	2930	282.4	-	-
24	96	60	240	-	-	-	302116.9	2777	21.0	-	-	297615	2609	398.9	-	-

VII. Conclusion and further research

In this paper, a model is developed that integrates the operations of three different types of equipment operating on the waterside of a container terminal. These equipment types are YCs, Lift-AGVs and QCs. We constructed a coordinated schedule that approximates optimal handling of containers. The model used to construct the coordinated schedule is an extended version of the HFS model. We added two new features called: bi-directional and job pairs. The former enables jobs to move through the HFS in both directions, the latter constrains certain jobs to be performed simultaneously by a single machine. All together the HFS model used within this research is the most extensive HFS that was ever used to model a container terminal (see Table II). Since the mathematical model is too complex to be solved with exact methods an approximative solution technique is developed. This solution technique is based on SA. The exact solution is approximated by splitting the problem in a discrete and a continuous part. The discrete part uses a local search method to find better alternatives to the current solution by predictive reassignment of jobs to machines. The continuous part can subsequently be solved quickly with LP. Computational results prove that the developed solution technique is capable of finding solutions with relatively small optimality gaps within a fraction of the time that B&B requires even for very large terminals.

Future work contains several parts. 1) The scheduling algorithm should be coupled to a real terminal to validate the schedule and to enable comparison with the current scheduling techniques applied on container terminals. 2) A library of HFS test problems is desired that allow for comparison of different solution techniques developed by multiple researchers. 3) The mathematical model could be extended with more features such as battery changes. This is because AGVs are battery powered. These batteries need to be changed whenever they are nearly empty. 4) The mathematical model could be extended by integrating the landside of the terminal as well. 5) The solution technique could be updated by using dynamic settings for different parameters based on the problem at hand. Parameters that are susceptible to different problems are for instance: the parameters used to construct the temperature schedule.

Acknowledgment

The author gratefully acknowledges and thanks the Delft University of Technology and TBA for providing support for this research. Special thanks goes to my supervisors, Arjen de Waal, Rudy Negenborn, Mark Duinkerken and Neil Yorke-Smith for their constructive support.

References

[1] R. Stahlbock and S. Voß, "Operations research at container terminals: a literature update," *OR spectrum*, vol. 30, no. 1, pp. 1–52, 2008.

[2] E. K. Bish, "A multiple-crane-constrained scheduling problem in a container terminal," *European Journal of Operational Research*, vol. 144, no. 1, pp. 83–107, 2003.

[3] M. De Koster, B. Balk, and W. Van Nus, "On using dea for benchmarking container terminals," *International Journal of Operations & Production Management*, vol. 29, no. 11, pp. 1140–1155, 2009.

[4] A. De Waal, "Container terminal & performance." TBA internal documents, 2012. [Online; accessed 2018-1-11].

[5] B. K. Lee and K. H. Kim, "Optimizing the block size in container yards," *Transportation Research Part E: Logistics and Transportation Review*, vol. 46, no. 1, pp. 120–135, 2010.

[6] N. Al-Dhaheri and A. Diabat, "The quay crane scheduling problem," *Journal of Manufacturing Systems*, vol. 36, pp. 87–94, 2015.

[7] C. Bierwirth and F. Meisel, "A survey of berth allocation and quay crane scheduling problems in container terminals," *European Journal of Operational Research*, vol. 202, no. 3, pp. 615–627, 2010.

[8] R. Tavakkoli-Moghaddam, A. Makui, S. Salahi, M. Bazzazi, and F. Taheri, "An efficient algorithm for solving a new mathematical model for a quay crane scheduling problem in container ports," *Computers & Industrial Engineering*, vol. 56, no. 1, pp. 241–248, 2009.

[9] K. H. Kim and Y.-M. Park, "A crane scheduling method for port container terminals," *European Journal of operational research*, vol. 156, no. 3, pp. 752–768, 2004.

[10] D. Steenken, S. Voß, and R. Stahlbock, "Container terminal operation and operations research-a classification and literature review," *OR spectrum*, vol. 26, no. 1, pp. 3–49, 2004.

[11] B. Cao and G. Uebe, "Solving transportation problems with nonlinear side constraints with tabu search," *Computers & Operations Research*, vol. 22, no. 6, pp. 593–603, 1995.

[12] K. H. Kim, Y. M. Park, and K.-R. Ryu, "Deriving decision rules to locate export containers in container yards," *European Journal of Operational Research*, vol. 124, no. 1, pp. 89–101, 2000.

[13] C. Zhang, J. Liu, Y.-w. Wan, K. G. Murty, and R. J. Linn, "Storage space allocation in container terminals," *Transportation Research Part B: Methodological*, vol. 37, no. 10, pp. 883–903, 2003.

[14] K. H. Kim and J. W. Bae, "Re-marshaling export containers in port container terminals," *Computers & Industrial Engineering*, vol. 35, no. 3-4, pp. 655–658, 1998.

[15] H. Y. Lau and Y. Zhao, "Integrated scheduling of handling equipment at automated container terminals," *International journal of production economics*, vol. 112, no. 2, pp. 665–682, 2008.

[16] S. A. Reveliotis, "Conflict resolution in agv systems," *Iie Transactions*, vol. 32, no. 7, pp. 647–659, 2000.

[17] B. Vuuren, "Route plan scheduling for automated guided vehicles at container terminals," Master's thesis, TU Delft, September 2017.

[18] L. Qiu and W.-J. Hsu, "Conflict-free agv routing in a bi-directional path layout," in *Proceedings of the 5th International Conference on Computer Integrated Manufacturing*, vol. 1, pp. 392–403, 2000.

[19] L. Chen, N. Bostel, P. Dejax, J. Cai, and L. Xi, "A tabu search algorithm for the integrated scheduling problem of container handling systems in a maritime terminal," *European Journal of Operational Research*, vol. 181, no. 1, pp. 40–58, 2007.

[20] L. Chen, A. Langevin, and Z. Lu, "Integrated scheduling of crane handling and truck transportation in a maritime container terminal," *European Journal of Operational Research*, vol. 225, no. 1, pp. 142–152, 2013.

[21] J. Xin, *Control and Coordination for Automated Container Terminals*. Citeseer, 2015.

[22] J. Xin, R. R. Negenborn, F. Corman, and G. Lodewijks, "Control of interacting machines in automated container terminals using a sequential planning approach for collision avoidance," *Transportation Research Part C: Emerging Technologies*, vol. 60, pp. 377–396, 2015.

[23] Y. Lu and M. Le, "The integrated optimization of container terminal scheduling with uncertain factors," *Computers & Industrial Engineering*, vol. 75, pp. 209–216, 2014.

[24] K. H. Kim and J. W. Bae, "A look-ahead dispatching method for automated guided vehicles in automated port container terminals," *Transportation science*, vol. 38, no. 2, pp. 224–234, 2004.

[25] J. He, Y. Huang, W. Yan, and S. Wang, "Integrated internal truck, yard crane and quay crane scheduling in a container terminal considering energy consumption," *Expert Systems with Applications*, vol. 42, no. 5, pp. 2464–2487, 2015.

[26] N. Kaveshgar and N. Huynh, "Integrated quay crane and yard truck scheduling for unloading inbound containers," *International Journal of Production Economics*, vol. 159, pp. 168–177, 2015.

[27] J. X. Cao, D.-H. Lee, J. H. Chen, and Q. Shi, "The integrated yard truck and yard crane scheduling problem: Benders' decomposition-based methods," *Transportation Research Part E: Logistics and Transportation Review*, vol. 46, no. 3, pp. 344–353, 2010.

[28] J. Cao, Q. Shi, and D.-H. Lee, "Integrated quay crane and yard truck schedule problem in container terminals," *Tsinghua Science & Technology*, vol. 15, no. 4, pp. 467–474, 2010.

[29] D. Kizilay, D. T. Eliiyi, and P. Van Hentenryck, "Constraint and mathematical programming models for integrated port container terminal operations," in *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pp. 344–360, Springer, 2018.

[30] R. Ruiz and J. A. Vázquez-Rodríguez, "The hybrid flow shop scheduling problem," *European Journal of Operational Research*, vol. 205, no. 1, pp. 1–18, 2010.

[31] I. Ribas, R. Leisten, and J. M. Framiñan, "Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective," *Computers & Operations Research*, vol. 37, no. 8, pp. 1439–1454, 2010.

[32] J. Xin, R. R. Negenborn, and G. Lodewijks, "Energy-aware control for automated container terminals using integrated flow shop scheduling and optimal control," *Transportation Research Part C: Emerging Technologies*, vol. 44, pp. 214–230, 2014.

[33] J. Xin, R. R. Negenborn, and G. Lodewijks, "Event-driven receding horizon control for energy-efficient container handling," *Control Engineering Practice*, vol. 39, pp. 45–55, 2015.

[34] J. N. Gupta, "Two-stage, hybrid flowshop scheduling problem," *Journal of the Operational Research Society*, pp. 359–364, 1988.

[35] M. Pinedo, *Scheduling*. Springer, 2012.

[36] Anonymous, "P versus np." https://en.wikipedia.org/wiki/P_versus_NP_problem, 2018. [Online; accessed 2018-1-26].

[37] A. Cobham, "The intrinsic computational difficulty of functions," 1965.

[38] J. Jungwattanakit, M. Reodecha, P. Chaovalitwongse, and F. Werner, "A comparison of scheduling algorithms for flexible flow shop problems with unrelated parallel machines, setup times, and dual criteria," *Computers & Operations Research*, vol. 36, no. 2, pp. 358–378, 2009.

[39] M. Langelaar, "Me46060(2016-2017) blackboard slides," 2017. [Online; accessed 2018-1-29].

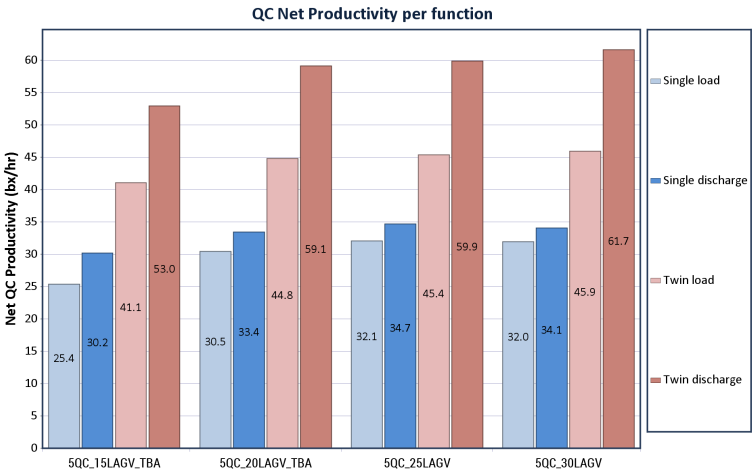
[40] F. S. Hillier and G. J. Lieberman, *Introduction to Operations Research, 10th Ed.* New York USA: McGraw-Hill, 2015.

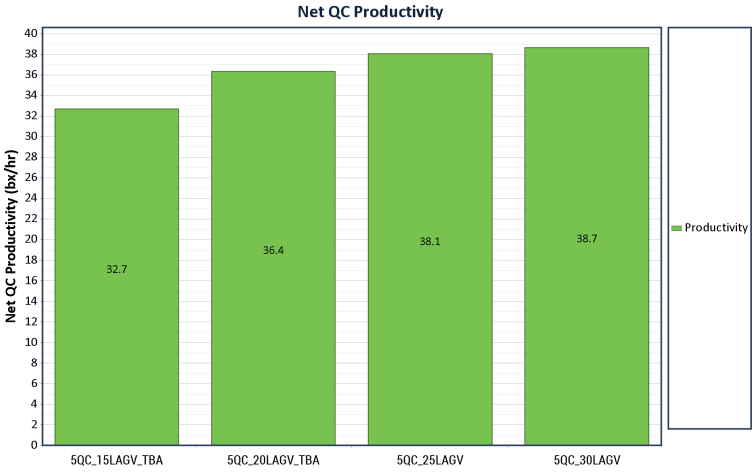
[41] V. Granville, M. Krivánek, and J.-P. Rasson, "Simulated annealing: A proof of convergence," *IEEE transactions on pattern analysis and machine intelligence*, vol. 16, no. 6, pp. 652–656, 1994.

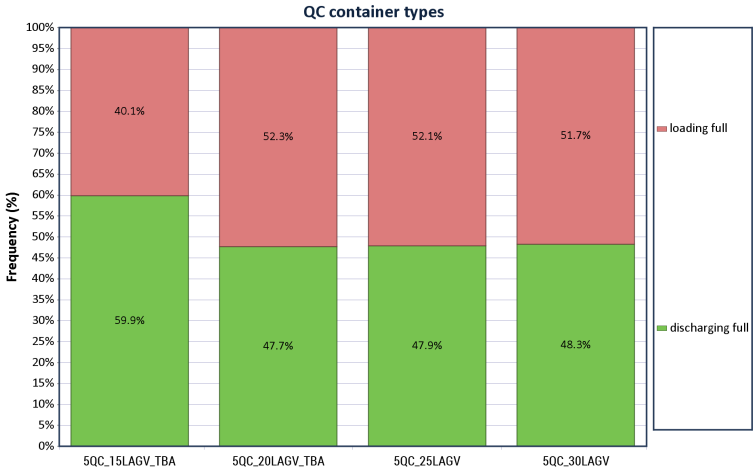
[42] A. Vignier, J.-C. Billaut, and C. Proust, "Les problèmes d'ordonnancement de type flow-shop hybride: état de l'art," *RAIRO-Operations Research*, vol. 33, no. 2, pp. 117–183, 1999.

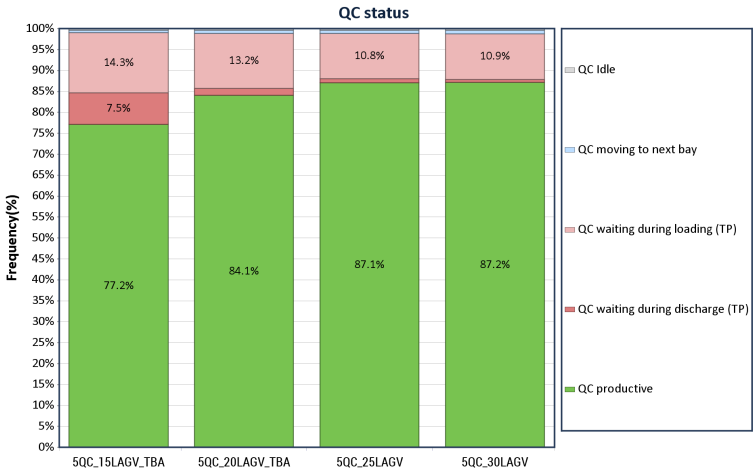
A

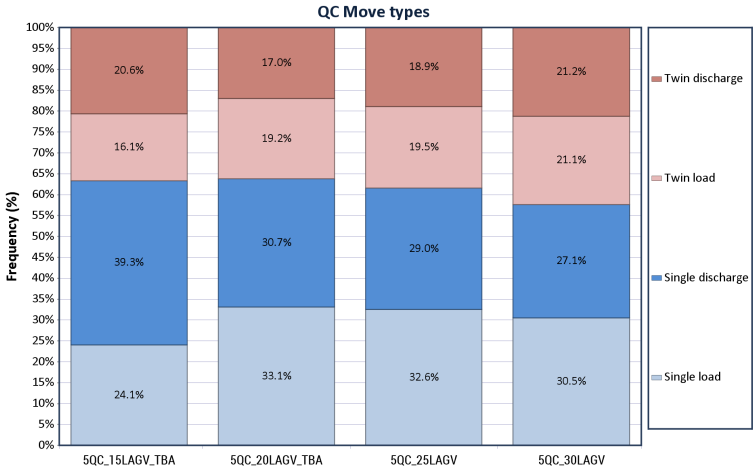
Simulation results

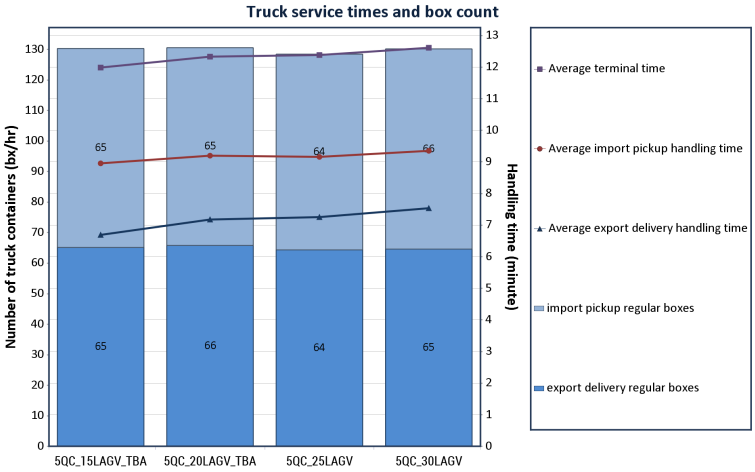


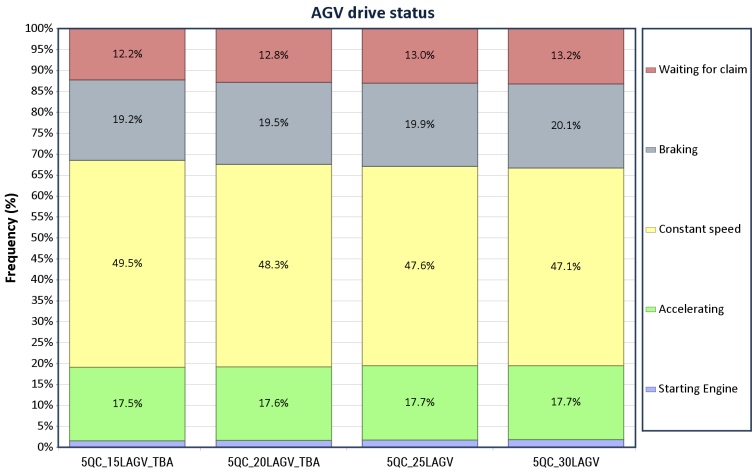


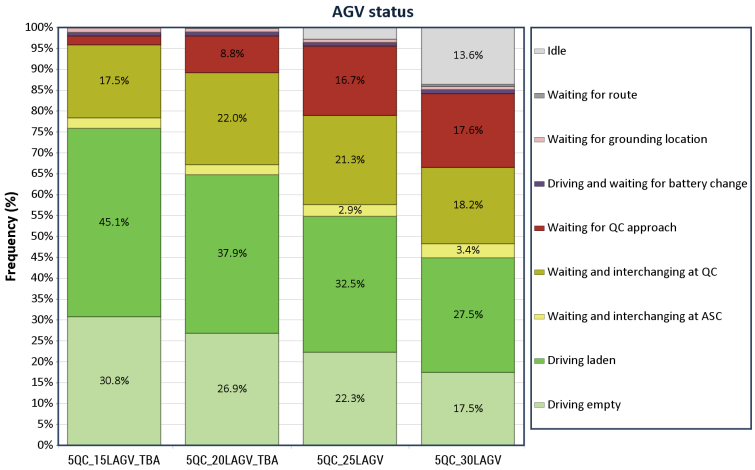

QC container types
Frequency (%)
100%
95%
90%
85%
80%
75%
70%
65%
60%
55%
50%
45%
40%
35%
30%
25%
20%
15%
10%
5%
0%
40.1%
52.3%
52.1%
51.7%
59.9%
47.7%
47.9%
48.3%
5QC_15LAGV_TBA
5QC_20LAGV_TBA
5QC_25LAGV
5QC_30LAGV
loading full
discharging full

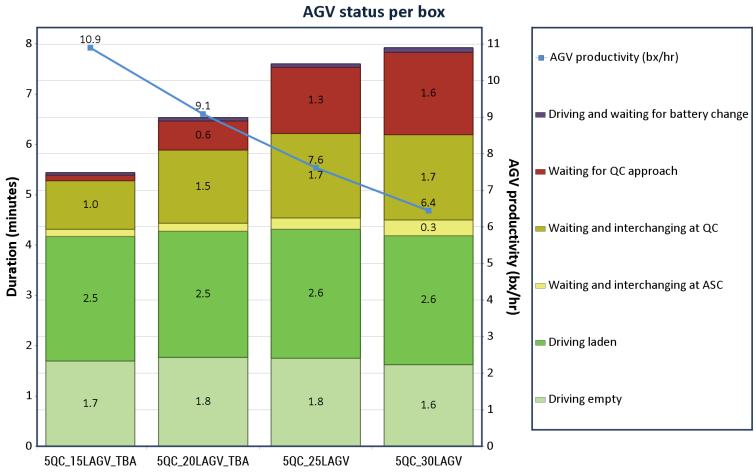

QC status
Frequency(%)
100%
95%
90%
85%
80%
75%
70%
65%
60%
55%
50%
45%
40%
35%
30%
25%
20%
15%
10%
5%
0%
14.3%
13.2%
10.8%
10.9%
7.5%
77.2%
84.1%
87.1%
87.2%
5QC_15LAGV_TBA
5QC_20LAGV_TBA
5QC_25LAGV
5QC_30LAGV
QC Idle
QC moving to next bay
QC waiting during loading (TP)
QC waiting during discharge (TP)
QC productive

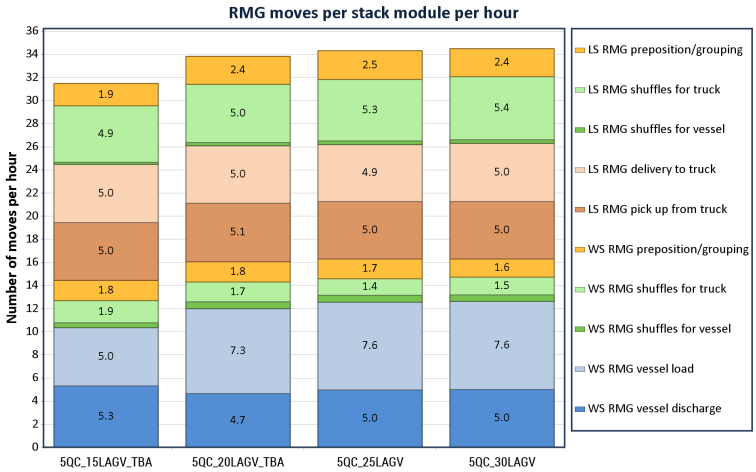

QC Move types
Frequency (%)
100%
95%
90%
85%
80%
75%
70%
65%
60%
55%
50%
45%
40%
35%
30%
25%
20%
15%
10%
5%
0%
20.6%
17.0%
18.9%
21.2%
16.1%
19.2%
19.5%
21.1%
39.3%
30.7%
29.0%
27.1%
24.1%
33.1%
32.6%
30.5%
5QC_15LAGV_TBA
5QC_20LAGV_TBA
5QC_25LAGV
5QC_30LAGV
Twin discharge
Twin load
Single discharge
Single load

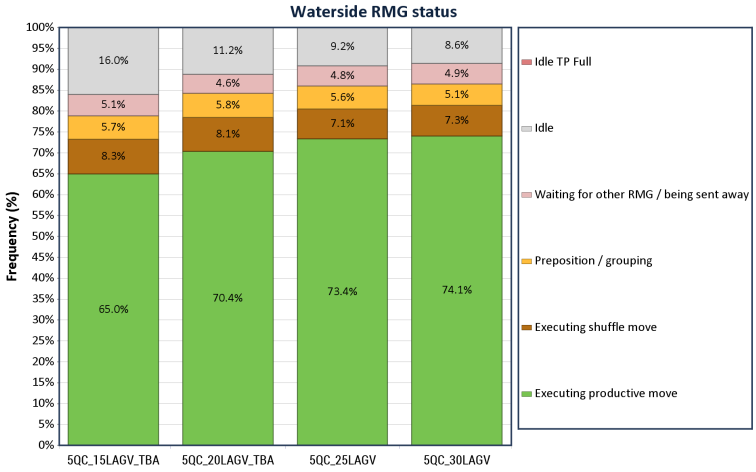

Truck service times and box count
Number of truck containers (bx/hr)
Handling time (minute)
65
65
64
66
65
66
64
65
5QC_15LAGV_TBA
5QC_20LAGV_TBA
5QC_25LAGV
5QC_30LAGV
Average terminal time
Average import pickup handling time
Average export delivery handling time
import pickup regular boxes
export delivery regular boxes

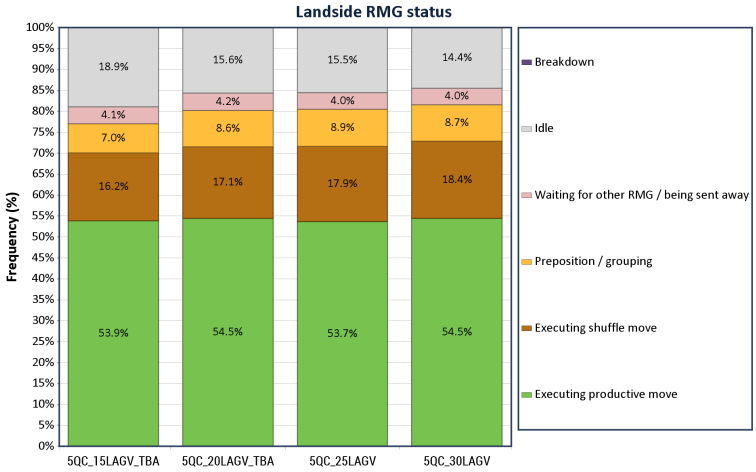

AGV drive status
Frequency (%)
12.2%
12.8%
13.0%
13.2%
19.2%
19.5%
19.9%
20.1%
49.5%
48.3%
47.6%
47.1%
17.5%
17.6%
17.7%
17.7%
5QC_15LAGV_TBA
5QC_20LAGV_TBA
5QC_25LAGV
5QC_30LAGV
Waiting for claim
Braking
Constant speed
Accelerating
Starting Engine

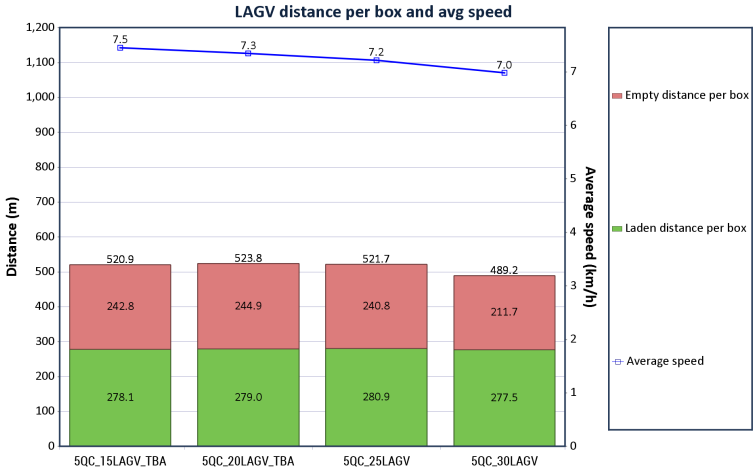

AGV status
Frequency (%)
17.5%
45.1%
30.8%
8.8%
22.0%
37.9%
26.9%
16.7%
21.3%
2.9%
32.5%
22.3%
13.6%
17.6%
18.2%
3.4%
27.5%
17.5%
5QC_15LAGV_TBA
5QC_20LAGV_TBA
5QC_25LAGV
5QC_30LAGV
Idle
Waiting for route
Waiting for grounding location
Driving and waiting for battery change
Waiting for QC approach
Waiting and interchanging at QC
Waiting and interchanging at ASC
Driving laden
Driving empty


AGV status per box
Duration (minutes)
AGV productivity (bx/hr)
10.9
9.1
6.4
1.0
1.5
1.7
0.6
1.3
1.6
0.3
2.5
2.5
2.6
2.6
1.7
1.8
1.8
1.6
5QC_15LAGV_TBA
5QC_20LAGV_TBA
5QC_25LAGV
5QC_30LAGV
AGV productivity (bx/hr)
Driving and waiting for battery change
Waiting for QC approach
Waiting and interchanging at QC
Waiting and interchanging at ASC
Driving laden
Driving empty


RMG moves per stack module per hour
Number of moves per hour
1.9
4.9
5.0
5.0
1.8
1.9
5.0
5.3
2.4
5.0
5.0
5.1
1.8
1.7
7.3
4.7
2.5
5.3
4.9
5.0
1.7
1.4
7.6
5.0
2.4
5.4
5.0
5.0
1.6
1.5
7.6
5.0
5QC_15LAGV_TBA
5QC_20LAGV_TBA
5QC_25LAGV
5QC_30LAGV
LS RMG preposition/grouping
LS RMG shuffles for truck
LS RMG shuffles for vessel
LS RMG delivery to truck
LS RMG pick up from truck
WS RMG preposition/grouping
WS RMG shuffles for truck
WS RMG shuffles for vessel
WS RMG vessel load
WS RMG vessel discharge

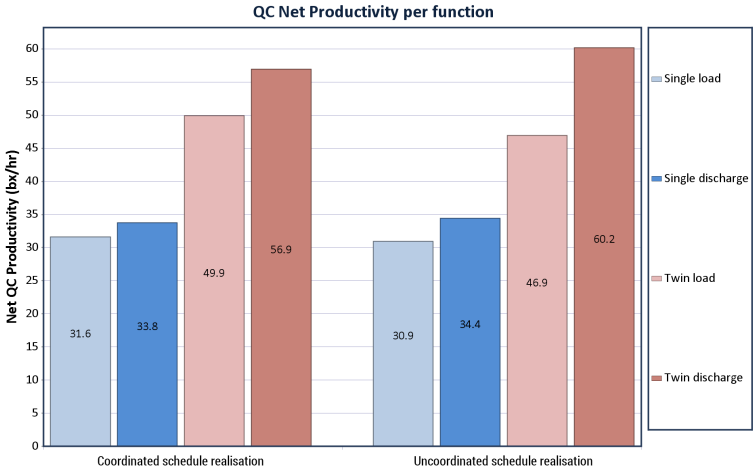

Waterside RMG status
Frequency (%)
16.0%
5.1%
5.7%
8.3%
65.0%
11.2%
4.6%
5.8%
8.1%
70.4%
9.2%
4.8%
5.6%
7.1%
73.4%
8.6%
4.9%
5.1%
7.3%
74.1%
5QC_15LAGV_TBA
5QC_20LAGV_TBA
5QC_25LAGV
5QC_30LAGV
Idle TP Full
Idle
Waiting for other RMG / being sent away
Preposition / grouping
Executing shuffle move
Executing productive move

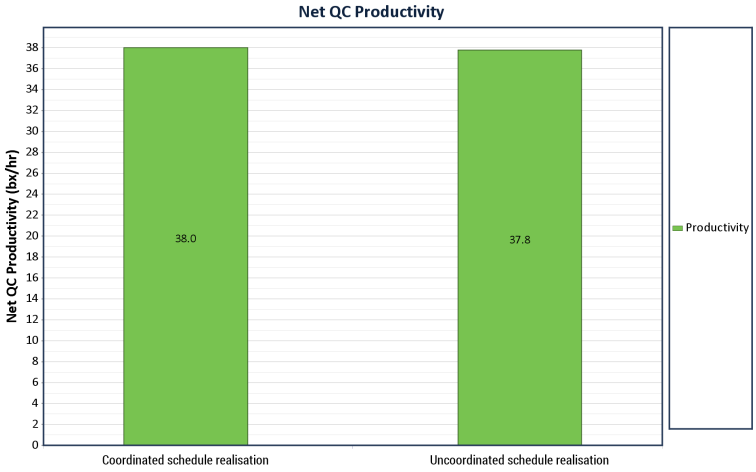

Landside RMG status
Frequency (%)
100%
95%
90%
85%
80%
75%
70%
65%
60%
55%
50%
45%
40%
35%
30%
25%
20%
15%
10%
5%
0%
18.9%
15.6%
15.5%
14.4%
4.1%
4.2%
4.0%
4.0%
7.0%
8.6%
8.9%
8.7%
16.2%
17.1%
17.9%
18.4%
53.9%
54.5%
53.7%
54.5%
5QC_15LAGV_TBA
5QC_20LAGV_TBA
5QC_25LAGV
5QC_30LAGV
Breakdown
Idle
Waiting for other RMG / being sent away
Preposition / grouping
Executing shuffle move
Executing productive move

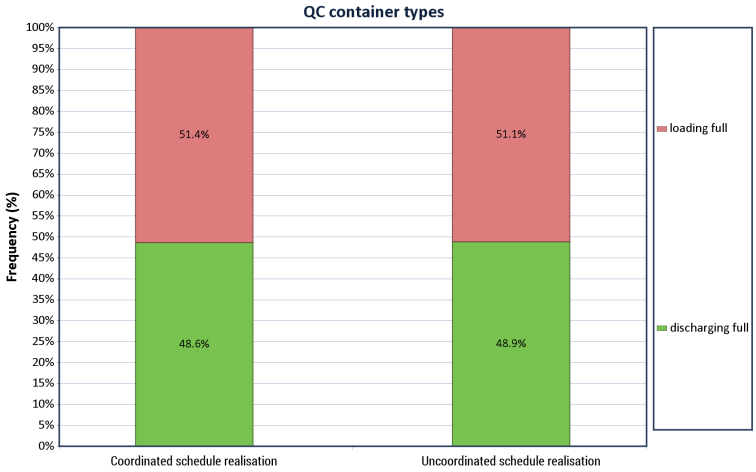

LAGV distance per box and avg speed
Distance (m)
Average speed (km/h)
1,200
1,100
1,000
900
800
700
600
500
400
300
200
100
0
7
6
5
4
3
2
1
0
7.5
7.3
7.2
7.0
520.9
523.8
521.7
489.2
242.8
244.9
240.8
211.7
278.1
279.0
280.9
277.5
5QC_15LAGV_TBA
5QC_20LAGV_TBA
5QC_25LAGV
5QC_30LAGV
Empty distance per box
Laden distance per box
Average speed

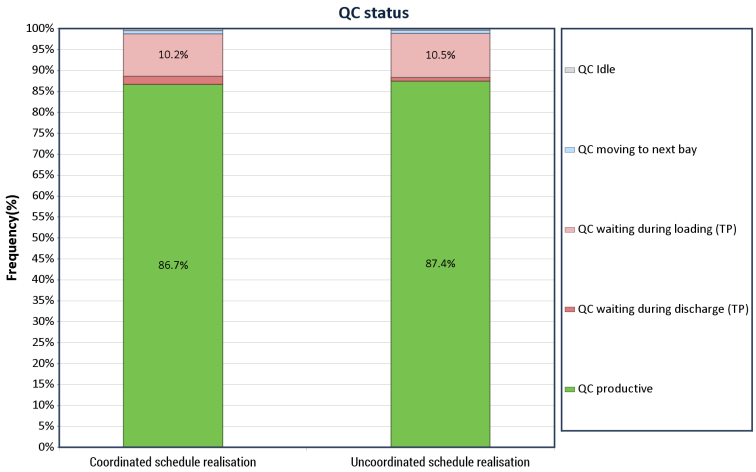
B

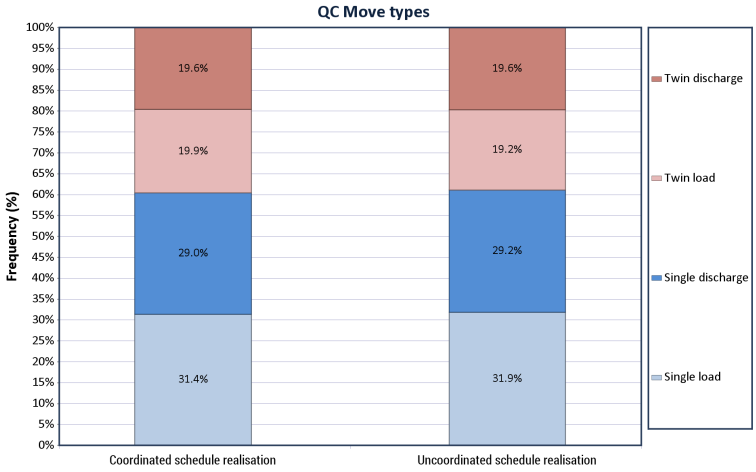
Simulation results coordinated schedule vs uncoordinated heuristic

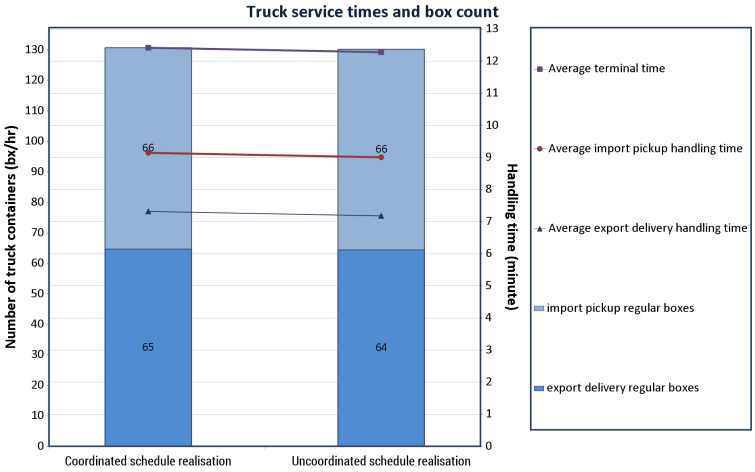


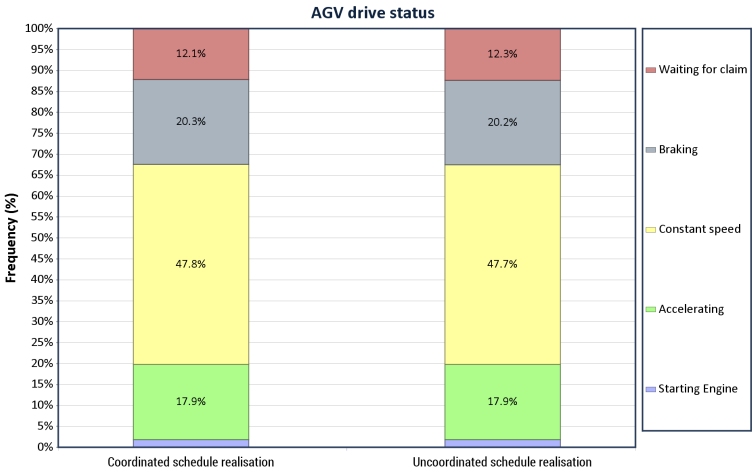


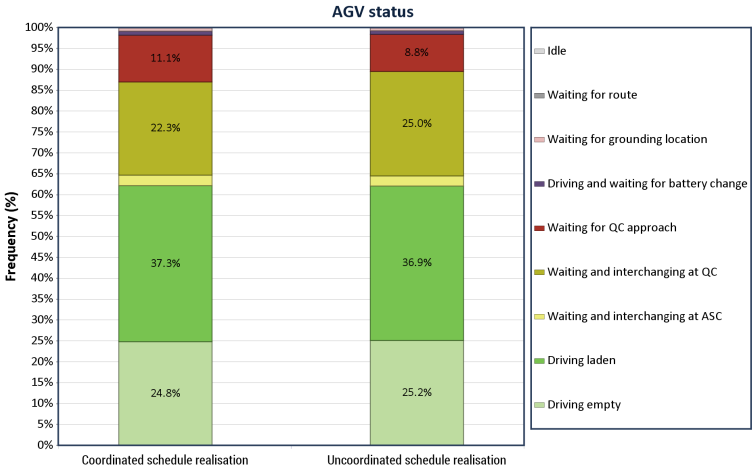

QC container types
Frequency (%)
100%
95%
90%
85%
80%
75%
70%
65%
60%
55%
50%
45%
40%
35%
30%
25%
20%
15%
10%
5%
0%
51.4%
48.6%
51.1%
48.9%
loading full
discharging full
Coordinated schedule realisation
Uncoordinated schedule realisation

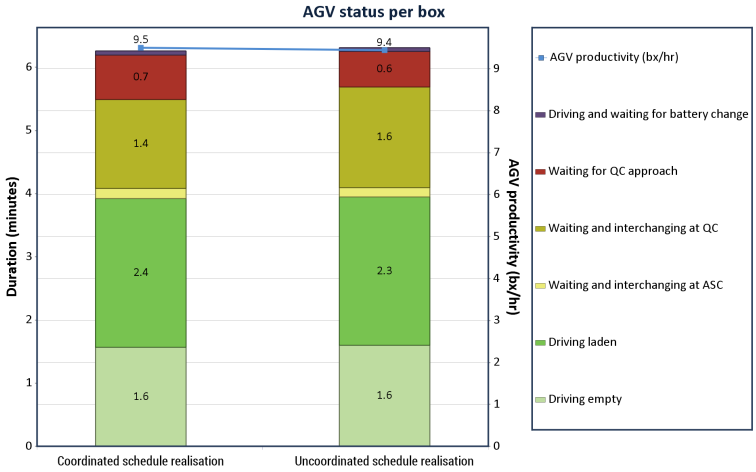

QC status
Frequency(%)
100%
95%
90%
85%
80%
75%
70%
65%
60%
55%
50%
45%
40%
35%
30%
25%
20%
15%
10%
5%
0%
10.2%
86.7%
10.5%
87.4%
QC Idle
QC moving to next bay
QC waiting during loading (TP)
QC waiting during discharge (TP)
QC productive
Coordinated schedule realisation
Uncoordinated schedule realisation

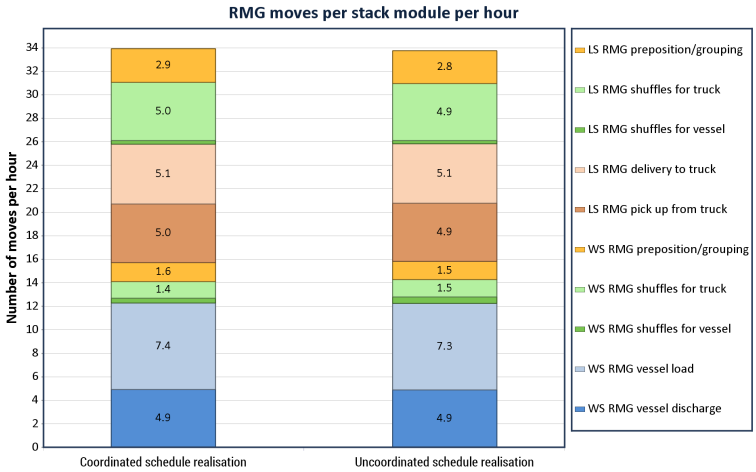

QC Move types
Frequency (%)
100%
95%
90%
85%
80%
75%
70%
65%
60%
55%
50%
45%
40%
35%
30%
25%
20%
15%
10%
5%
0%
19.6%
19.9%
29.0%
31.4%
19.6%
19.2%
29.2%
31.9%
Twin discharge
Twin load
Single discharge
Single load
Coordinated schedule realisation
Uncoordinated schedule realisation

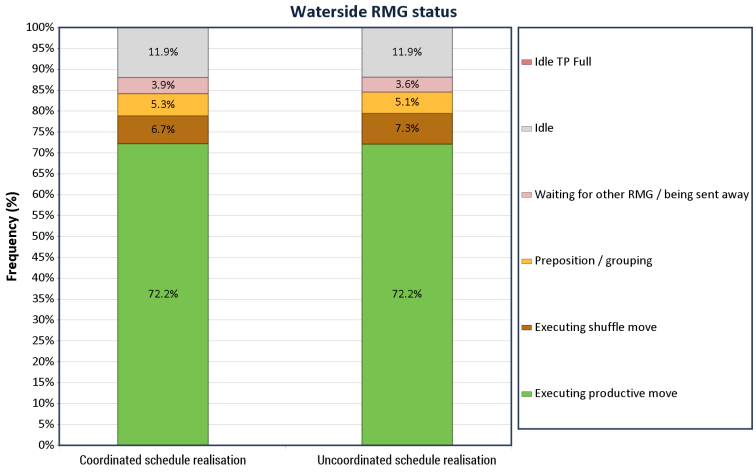

Truck service times and box count
Number of truck containers (bx/hr)
Handling time (minute)
66
66
65
64
Coordinated schedule realisation
Uncoordinated schedule realisation
Average terminal time
Average import pickup handling time
Average export delivery handling time
import pickup regular boxes
export delivery regular boxes

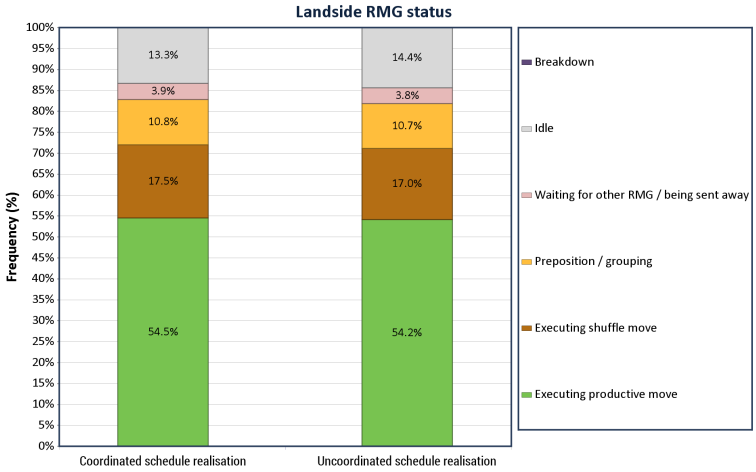

AGV drive status
Frequency (%)
12.1%
20.3%
47.8%
17.9%
12.3%
20.2%
47.7%
17.9%
Coordinated schedule realisation
Uncoordinated schedule realisation
Waiting for claim
Braking
Constant speed
Accelerating
Starting Engine

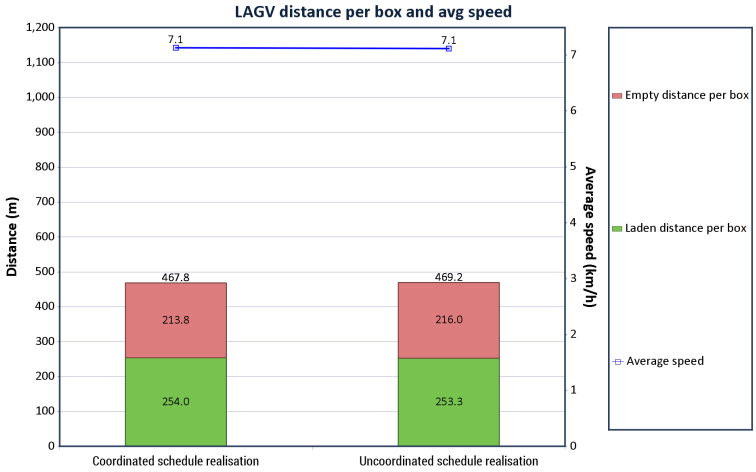

AGV status
Frequency (%)
11.1%
22.3%
37.3%
24.8%
8.8%
25.0%
36.9%
25.2%
Coordinated schedule realisation
Uncoordinated schedule realisation
Idle
Waiting for route
Waiting for grounding location
Driving and waiting for battery change
Waiting for QC approach
Waiting and interchanging at QC
Waiting and interchanging at ASC
Driving laden
Driving empty


AGV status per box
Duration (minutes)
AGV productivity (bx/hr)
9.5
9.4
0.7
1.4
2.4
1.6
0.6
1.6
2.3
1.6
Coordinated schedule realisation
Uncoordinated schedule realisation
AGV productivity (bx/hr)
Driving and waiting for battery change
Waiting for QC approach
Waiting and interchanging at QC
Waiting and interchanging at ASC
Driving laden
Driving empty


RMG moves per stack module per hour
Number of moves per hour
2.9
5.0
5.1
5.0
1.6
1.4
7.4
4.9
2.8
4.9
5.1
4.9
1.5
1.5
7.3
4.9
Coordinated schedule realisation
Uncoordinated schedule realisation
LS RMG preposition/grouping
LS RMG shuffles for truck
LS RMG shuffles for vessel
LS RMG delivery to truck
LS RMG pick up from truck
WS RMG preposition/grouping
WS RMG shuffles for truck
WS RMG shuffles for vessel
WS RMG vessel load
WS RMG vessel discharge


Waterside RMG status
Frequency (%)
11.9%
3.9%
5.3%
6.7%
72.2%
11.9%
3.6%
5.1%
7.3%
72.2%
Coordinated schedule realisation
Uncoordinated schedule realisation
Idle TP Full
Idle
Waiting for other RMG / being sent away
Preposition / grouping
Executing shuffle move
Executing productive move


Landside RMG status
Frequency (%)
100%
95%
90%
85%
80%
75%
70%
65%
60%
55%
50%
45%
40%
35%
30%
25%
20%
15%
10%
5%
0%
13.3%
3.9%
10.8%
17.5%
54.5%
14.4%
3.8%
10.7%
17.0%
54.2%
Coordinated schedule realisation
Uncoordinated schedule realisation
Breakdown
Idle
Waiting for other RMG / being sent away
Preposition / grouping
Executing shuffle move
Executing productive move


LAGV distance per box and avg speed
Distance (m)
1,200
1,100
1,000
900
800
700
600
500
400
300
200
100
0
Average speed (km/h)
7
6
5
4
3
2
1
0
7.1
7.1
467.8
213.8
254.0
469.2
216.0
253.3
Coordinated schedule realisation
Uncoordinated schedule realisation
Empty distance per box
Laden distance per box
Average speed