Master's Thesis: Optimizing performance of a dry bulk terminal by integrating the seaside and the stockyard

MARK VAN DER GOOT 458221

ERASMUS UNIVERSITY ROTTERDAM
ERASMUS SCHOOL OF ECONOMICS
MSC OPERATIONS RESEARCH AND QUANTITATIVE LOGISTICS
DECEMBER 17, 2017

SUPERVISOR: DR. W. VAN DEN HEUVEL
SECOND ASSESSOR: DR. T.A.B. DOLLEVOET
COMPANY SUPERVISOR: A. DIJKSTRA

Abstract

In this research, different solution strategies are presented to create complete equipment schedules for dry bulk terminals. At the seaside, this contains the berth allocation problem and the quay crane assignment problem. At the stockyard, the stacker/reclaimer assignment, the stockpile allocation and the belt conveyor routing are considered. An existing heuristic to solve the seaside problem for container terminals is adjusted for dry bulk terminals. The adjusted method performs close to optimal. Next, a scheduling method for the stockyard equipment is presented. Finally, three solution strategies are developed to integrate the seaside scheduling problem with the stockyard scheduling problem to compute a complete dry bulk terminal schedule. The first method computes feasible schedules in relative short computation time by considering individual vessels sequentially, where the second method constructs multiple schedules and creates a lower and upper bound. The third method integrates the complete terminal and performs well. The results prove that a well-performing integration between the seaside and stockyard is possible. This integration can be used to obtain good equipment schedules for the complete dry bulk terminal.

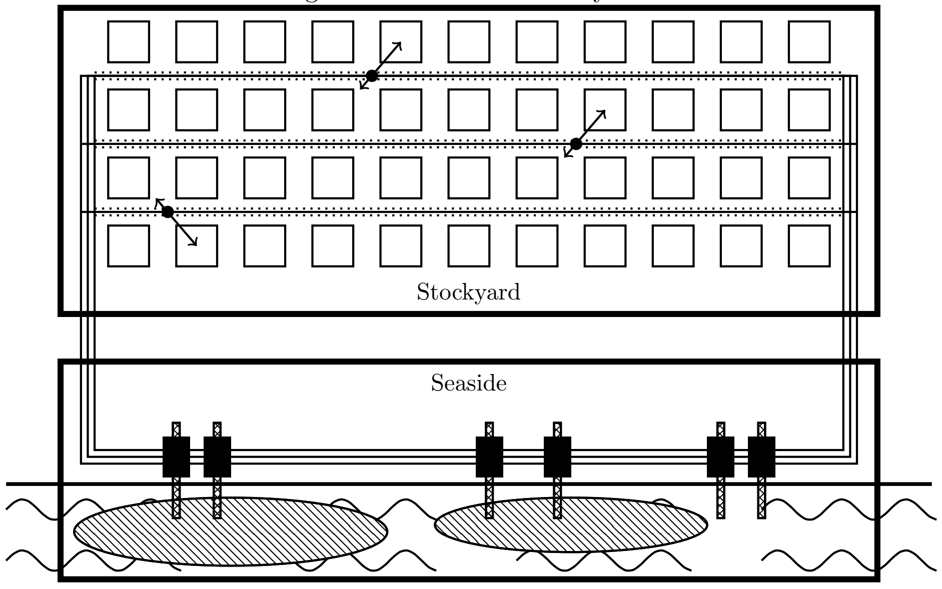
Keywords: Dry Bulk Terminal, Berth Allocation Problem, Quay Crane Scheduling, Stockyard Scheduling

Contents

1 Introduction

Although for the first time in this century the world coal production declined in 2014, there is still a significant increase in coal production since 2000. The world coal production increased from approximately 4000 megaton to 7269 megaton between 2000 and 2016 (IEA [2017]).
All produced coal needs to be transported. An efficient way of transporting coal and other bulk materials globally is through dry bulk vessels. These bulk vessels are loaded and unloaded at dry bulk terminals. Due to the growing production of coal this century, dry bulk terminals need to transfer more bulk materials. Therefore, it is essential for terminals to operate efficiently.
An important aspect of dry bulk terminal efficiency is the planning of the equipment. Usually, a dry bulk terminal is composed of three parts. The first part is the seaside, where the vessels berth and are serviced by quay cranes. The second part is the stockyard, where not directly transported materials are stored. Thirdly, we have the landside, where trucks and trains are serviced to transport materials. For simplicity, the landside is omitted in this research. This implies that vessels create the inward and outward flow for the terminal, hence either loading or unloading vessels will be considered. A vessel contains multiple holds, these holds are used to store the bulk material during transport. At the stockyard, the material is transported on belt conveyors to stacker/reclaimers in order to store the material on a stockpile. These stacker/reclaimers are able to move on rails, which are located between lanes of stockpiles. In these rails, a belt conveyor is implemented. A schematic overview of a basic terminal without a landside is given in Figure 1.

Figure 1: Basic terminal layout

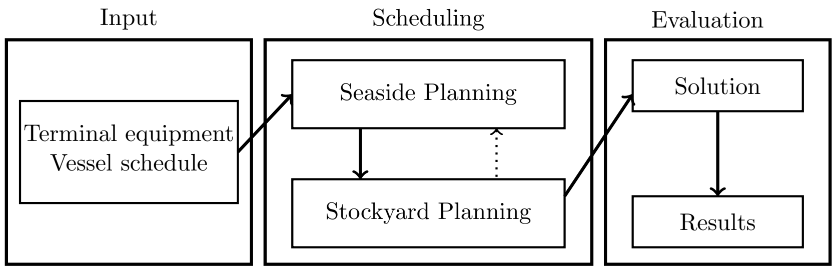


In this example, the seaside and the stockyard are illustrated. We assume that the vessels are unloading in this example. At the seaside two vessels are berthed. Both vessels are serviced by two cranes, and two other cranes are idle. When materials are unloaded from a vessel, the materials are transported on belt conveyors to the stockyard. These belt conveyors are represented by a single line. At the quay and on both sides of the stockyard, three parallel belt conveyors are pictured. Three single belt conveyors are shown between the stockpiles. When the materials arrive at the stockyard, they are stacked by a stacker/reclaimer (represented by a double-sided arrow) on a stockpile, which is represented by a small square. A stacker/reclaimer is able to move on rails, which is represented by the two dotted lines enclosing the belt conveyor. This belt conveyor is implemented in the rails to execute the final part of the route to the stacker/reclaimers.

The focus of this research is to find a good method to create an equipment schedule for the stockyard and the seaside, and a procedure to integrate these two schedules. A seaside schedule consists of the berthing location of a vessel, the assignment of the quay cranes and the berthing time. The stockyard schedule includes the assignment of the belt conveyors, stacker/reclaimers and stockpiles.
In order to increase the efficiency of a dry bulk terminal, well-performing scheduling methods are needed. Concerning the seaside, many existing methods for container terminals can be used. For the stockyard, there is a lack of existing methods. Furthermore, significant performance improvements may be gained through integrating both parts of a dry bulk terminal.
For the seaside schedule we will adjust an already existing method developed by Meisel and Bierwirth [2009]. The method considered is developed for container terminals. To adjust this method for dry bulk terminals, we assume that a hold needs to be serviced by one crane without interruption. However, this assumption has implications for the individual crane assignments. Consequently, the individual crane assignment is also included in the adjusted method. In comparison, the method by Meisel and Bierwirth [2009] only assign a number of cranes, it is not specified which crane will service which vessel.
A new method to create stockyard schedules will be presented. This method uses the schedule created at the seaside as input, and will verify whether a corresponding feasible stockyard schedule can be created. If this is possible, the corresponding schedule will be created. If it is not possible, the stockyard scheduling method will validate which vessel or equipment is causing this. This feedback information will be returned and used.
Finally, three approaches to integrate both parts are developed. The first approach schedules the vessels individually sequenced on their arrival time. The second approach creates multiple schedules at the seaside and tests each schedule on feasibility at the stockyard. The feasible schedule with the best objective value will be selected. The third approach incorporates integration between the stockyard and the seaside. In this third approach, the seaside will use the feedback information received from the stockyard. This information is

sent back to the seaside, where a new seaside schedule is created taking the stockyard infeasibilities of the first schedule into account. This continues until a feasible solution is found.
The different methods are evaluated by creating a score for each schedule. This is compared with an optimal schedule created by CPLEX.
The main contributions of this thesis are the adjustment of the method by Meisel and Bierwirth [2009] for dry bulk terminals, the development of a stockyard scheduling method for dry bulk terminals and the integration of the seaside and stockyard by three solution strategies.
The scope of this research is represented in Figure 2, where the dotted line represents the third integration approach.

Figure 2: Overview research



The remainder of this thesis is structured as follows: in Section 2 relevant literature is presented, and the contribution of this research is explained. Section 3 explains the problem, and the seaside and stockyard models are presented in Section 4. The heuristics which create the separate schedules for both the seaside and the stockyard are explained in Section 5. Then, in Section 6 the method to integrate the two parts is explained. In Section 7 the data is presented, and the experimental design is explained. The results are presented in Section 8. Finally, the research will be discussed and concluded in Section 9.2. All abbreviations used in this report are stated in Appendix 10.1.

2 Literature review

The problems at the seaside are the Berth Allocation Problem (BAP) and the Quay Crane Scheduling Problem (QCSP). The BAP allocates vessels at the berth, taking time and berth length constraints into account to prevent overlapping of ships in either time or space. The main objective of the BAP is to minimize total service time of the vessels. The QCSP assigns quay cranes to vessels, assuming the berth planning is given.

All the following papers are focused on container terminals. Because the main operations at the seaside are similar at container and bulk terminals, this

literature is relevant for the problem at hand.
Because the BAP is known to be NP-Complete (Lim [1998]) most BAPs are solved by heuristics. Lim [1998] proves the NP-completeness of the BAP by reducing the set partitioning problem to the BAP. To solve the BAP, the BAP is transformed into a two dimensional bin packing problem. One dimension is time, and the other dimension is the length. A berthing vessel occupies a space in this time-length space. This two dimensional bin-packing problem is solved by transforming the problem into a network. The approach of Cordeau et al. [2005] is to solve the dynamic BAP with discrete and semi-continuous berths by using tabu search heuristics. These heuristics perform well and have short computation times. Guan and Cheung [2004] present a tree search procedure to solve the BAP exactly up to 15 vessels. For larger problem instances a composite heuristic is presented. This combines the tree search procedure with a construction heuristic and a pair-wise improvement heuristic. In order to use the berth more efficiently, a continuous scheme is preferred. This is considered in combination with location-dependent handling time by Imai et al. [2005] and solved by a heuristic algorithm based on the discrete berth. In these BAP studies, the handling time per vessel is independent of the QCSP as given. In reality, this handling time depends on the QCSP.
The QCSP is introduced by Daganzo [1989], and he presents exact and approximate solution methods. These methods are extended by a branch-and-bound method by Peterkofsky and Daganzo [1990]. In these early papers, an important property of the QCSP is not considered: the non-crossing constraint. Because the movement of quay cranes is restricted by a rail, they are not able to cross each other. This constraint is included by Lim et al. [2007], where the QCSP is modeled as an m-parallel machine scheduling problem and solved heuristically by a simulated annealing heuristic. Other research including the non-crossing constraint is done by Zhu and Lim [2006], where the NP-completeness of the problem is proved and another simulated annealing approach is presented. Liu et al. [2006] solve an MIP (also including the non-crossing constraint) by decomposing the problem into two smaller problems on berth-level and on vessel-level. More recently, Legato et al. [2012] solved very rich and extended QCSPs by a branch-and-bound approach and a timed Petri net. In addition to this a lower bound is presented using Lagrangian relaxation.
Because the BAP and the QCSP are interrelated, Park and Kim [2003] presented a first MIP to solve the integrated BAP and QCSP problem. They also propose a two-phase approach to solve the related problems. In the first phase, the berthing position and number of cranes is determined. The second phase constructs a feasible crane schedule. The MIP of Park and Kim [2003] is improved and extended with crane productivity by Meisel and Bierwirth [2009]. Also the formulation is made more compact to improve the computation time. The problem is solved by a squeaky wheel optimization and a tabu search heuristic. In a squeaky wheel optimization, a solution is created based on a priority list. Solution components with bad performance will obtain a higher priority, and a new solution is created. Because the squeaky wheel optimization performs well, we consider a variant of this squeaky wheel optimization in this

thesis. Blazewicz et al. [2011] consider the combined BAP and QCSP as a moldable task scheduling model. This problem is solved by a heuristic. Imai et al. [2008] formulate a MIP which includes the crane and berth scheduling simultaneously. This problem is solved by a genetic algorithm because of a large number of constraints and variables. Likewise, Han et al. [2010] solve the BAP and QCSP simultaneously, and consider stochastic arrival and handling times, just as Zhou and Kang [2008], who included chance constraints. For more interesting references about the BAP and the QCSP on container terminals, readers are referred to the surveys by Bierwirth and Meisel [2010] and Bierwirth and Meisel [2015]. Furthermore, a survey on meta heuristics used to solve the BAP is presented by Kovač [2017].

Considering the BAP and QCSP on bulk terminals, Umang et al. [2011] present a MIP, and solve this with CPLEX. Because solving this is only feasible for small instances, and can already take significant computing time, Umang et al. [2013] propose a general set partitioning method and a heuristic method based on squeaky wheel optimization to solve the BAP. Barros et al. [2011] also present a MIP, which is solved by CPLEX. Next to this, a simulated annealing approach is used. Pratap et al. [2017] include cranes in a BAP and solve this using a genetic algorithm and a chemical reaction optimization technique, which is a combination of a genetic algorithm and a simulated annealing approach.

Considering the stockyard of a bulk terminal, three problems can be distinguished: stockyard allocation, stacker/reclaimer scheduling and belt conveyor routing. Kim et al. [2009] consider a storage yard allocation at a large-scale steelworks terminal, where the raw material is coming in by vessels. The solution is computed using CPLEX. Tang et al. [2016] integrate the stockyard allocation with a vessel scheduling problem. Concerning stacker/reclaimer scheduling Angelelli et al. [2016] presents theorems on the optimality of solutions. Hu and Yao [2012] developed a genetic algorithm to solve the stacker/reclaimer scheduling. Van Vianen et al. [2012] studied the belt conveyor routing by developing a support system for terminal operators.
In all this literature, only one part of the complete stockyard problem is considered, where the other parts are assumed to be given. In contradiction to this, Boland et al. [2012] developed a stockyard planning technology which improves the efficiency on a stockyard by using an advanced heuristic. Ago et al. [2006] combine the allocation with the routing on the belt conveyor network, but does not takes route overlapping into account. Robenek et al. [2014] and Menezes et al. [2016] consider the complete problem but assume the handling time is fixed. Consequently the handling time is not dependent on the equipment used. Klaassen [2007] tested various decision criteria for different parts of equipment on a bulk terminal. The research done by Van Vianen [2015] contains extensive information on the design of a dry bulk terminal, these designs were tested with simulation. More general information about bulk terminals can be found in Lodewijks [2002], Lodewijks and Ottjes [2003], Lodewijks et al. [2007], Lodewijks et al. [2009] and Willekes [1999].

In contrast to all aforementioned studies, this research combines the operational difficulties on the stockyard and the seaside. The algorithms developed in this research will create a complete dry bulk terminal schedule. This includes the BAP, the QCSP, the stockyard allocation, the stacker/reclaimer scheduling and the belt conveyor routing. Also, methods to integrate the problems are presented.

3 Problem formulation

3.1 Problem

The main focus of this research is on the optimization of dry bulk terminal performance. The increase in performance of the dry bulk terminal is achieved by creating a well-performing integrated schedule for the seaside and the stockyard.
The main objective is to minimize the turnaround time of vessels. Since the turnaround time of vessels is mainly affected by the scheduling at the seaside, the goal at the seaside is to have a heuristic which creates schedules with low turnaround times. For the stockyard, this converts to a scheduling method that is able to construct feasible stockyard schedules corresponding to well-performing seaside schedules.
A dry bulk terminal executes given tasks, by assigning equipment to a task. These tasks for the dry bulk terminal are created by the following 5 questions:

1. Where and when will the vessel berth?
   The time a vessel berths is important for the time windows of the tasks. The location is important for the cranes assigned.

2. Which cranes will service the vessel?
   The assignment of cranes is important for the tasks, especially the number of cranes. If more cranes are assigned to a vessel, the number of tons transported per time unit will be larger and the vessel will be finished earlier. Next to this, the quay crane will be the end or beginning of a route on the belt conveyor network.

3. Which stacker/reclaimer will execute this?
   The stacker/reclaimer chosen will be the end or beginning of a route on the belt conveyor network.

4. To which stockpile will we (un)load the bulk material?
   When the stacker/reclaimer is known, a corresponding stockpile can be assigned.

5. Which route will be used on the belt conveyor network?
   When the stacker/reclaimer is known, a route to the stacker/reclaimer can be established.

Questions 1 and 2 correspond to the seaside planning problem. Questions 3, 4 and 5 correspond to the stockyard planning. In summary, a method to develop seaside schedules is needed, as well as a method to create a stockyard schedule. The aim of this study is to create a scheduling method for the complete terminal by distinguishing two parts of the terminal: the seaside and the stockyard.

3.2 Seaside

The seaside part of a terminal contains the following important parts: the vessels, the quay, the quay cranes and the belt conveyors.

3.2.1 Vessels

Different types of vessels arrive at a dry bulk terminal. These differences are mainly in the following characteristics: deadweight, length, beam, draft and number of holds. The deadweight is the amount of bulk material a vessel can carry. The length, beam and the draft correspond to the size of a vessel. A vessel can carry different types of material, but in one hold the type of material needs to be homogeneous.

3.2.2 Quay & Quay Cranes

Bulk vessels moor at a quay, the quay-layout can be of three different types: continuous, discrete or hybrid. A dry bulk terminal can contain multiple quays. In the continuous case, vessels can moor everywhere at the quay. In the discrete case, the quay is divided in parts, and at every part only one vessel can be moored. In the hybrid case, the quay is divided in parts, but a vessel can occupy different parts.
The bulk material is (un)loaded from a ship using quay cranes. Different types of quay cranes are suitable for bulk materials. Continuous cranes create a continuous flow of material, whereas discrete cranes pick up an amount of material every cycle. Important information about the quay cranes is the (un)loading capacity. Quay cranes are able to either load or unload. Quay cranes belonging to the same quay move on the same rails, hence these quay cranes are not able to cross each other. In the remainder of this report, the notation QCs is used to refer to quay cranes.

3.3 Stockyard

The stockyard consists of the following equipment: the belt conveyor network, the stacker/reclaimers and the stockpiles. In the remainder of this report, the notation SRs is used to refer to stacker/reclaimers.

3.3.1 Belt conveyor network

In this report, the belt conveyor network belongs completely to the stockyard, including the belt conveyors under the quay cranes. The belt conveyor network contains several belt conveyors. These belt conveyors connect the QCs with the SRs. Different routes are possible to transport bulk materials from a QC to a SR.

3.3.2 Stacker/reclaimers

When a bulk material arrives at a SR, it is stacked at a stockpile. If the bulk material is requested from a stockpile, it is reclaimed by a SR. The stacking capacity and the reclaim capacity can be different. SRs move on a rail between the stockpiles. A SR can reach the stockpiles adjacent to the rail. In this rail, a belt conveyor is implemented to transport the material to the SR(as seen in Figure 1).

3.3.3 Stockpiles

Stockpiles are located on a lane. After bulk material is unloaded and stacked, it remains on a stockpile until it is reclaimed. A stockpile contains one type of bulk material. It is possible that a stockpile can be reached by SR from two different SR rails.

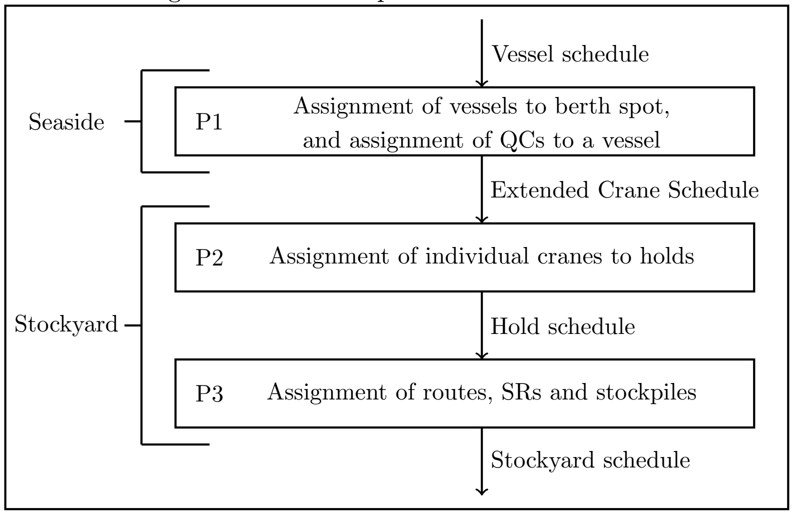
4 Model

In this section, we first present 3 types of schedules used in this report. Next, both models will be presented. First the model for the seaside, and secondly the model for the stockyard.

4.1 Schedule types

For clarification, the different schedules and planning methods in this report are explained in Figure 3. Note that the distinction between the planning methods (seaside and stockyard) is important is this research. This distinction is made because the problem is considered too large for one planning method. In Figure 3, the planning problems are stated in blocks. The arrows between these problem blocks are the input and/or output.
The first problem (P1) needs a vessel schedule as input, and creates an extended crane schedule (ECS). In this ECS the berthing position of the vessel is determined and the assignment of the individual QCs at each time period is stated. The ECS is used as an input for the stockyard, where in the second planning problem (P2) the hold schedule is created. In P2, it is determined in which order the holds of a vessel are serviced. In this research, P2 is part of the stockyard planning.
Finally, in P3 the SRs, stockpiles and routes are assigned to each hold in the hold schedule. The output of P3 is the stockyard schedule.

Figure 3: Overview problems and schedules


Vessel schedule
Seaside
P1
Assignment of vessels to berth spot, and assignment of QCs to a vessel
Extended Crane Schedule
P2
Assignment of individual cranes to holds
Stockyard
Hold schedule
P3
Assignment of routes, SRs and stockpiles
Stockyard schedule

In this research, and hence in all schedules, time and length is discretized. Note that all schedule examples in this section are aligned. The problem classification (P1, P2, P3) is important in the remainder of this report.

4.1.1 Vessel Schedule

The vessel schedule is the main input of the model. This schedule contains all important information about the arriving vessels, such as the arrival time, the type, whether the vessel is loading or unloading, the amount and type of materials and other information. An example of a vessel schedule is given in Table 1. In this table, the minimum finishing time (MinFT) is included. This is the finishing time in the optimal case. Next to this, the maximum finishing time (MaxFT) is included. The MinFT and MaxFT are used in the objective value. The HHT is the handling time for one hold in the used time units. For the arrival time we use AT. In this table, and in the remainder of this report we use numbers to represent different materials.

Table 1: Example of a vessel schedule

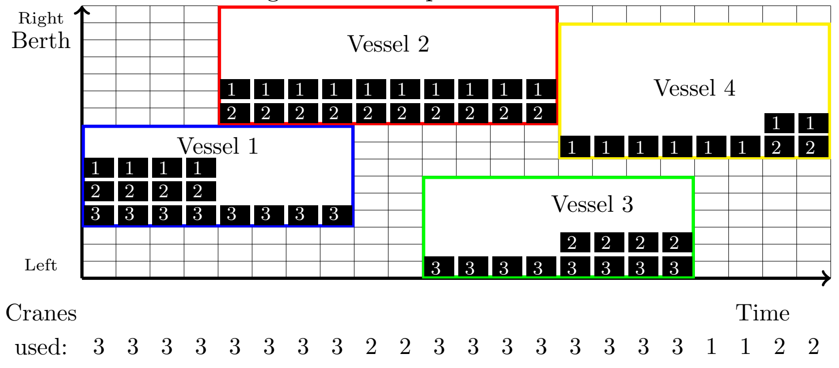
Vessel	AT	Length	HHT	# of holds	MinFT	MaxFT	Materials	Loading/ Unloading	Max # of Cranes
1	1	7	5	4	14	20	4,4,2,2	Unloading	3
2	4	6	4	4	8	12	1,1,1,1	Loading	3
3	10	6	4	3	18	24	2,3,3	Unloading	2
4	14	8	2	5	20	24	4,4,4,4,4	Unloading	2

In this example 4 vessels arrive with different properties. Note that, in this section, the examples of the schedules are aligned with this vessel schedule.

4.1.2 Extended Crane Schedule

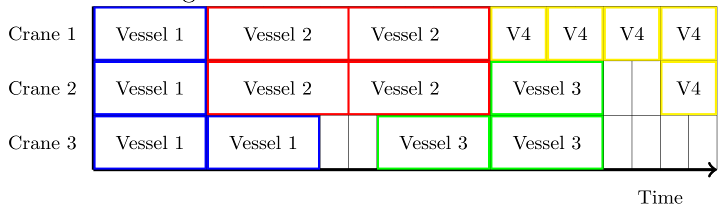
At the seaside an ECS is created. In Figure 4, an ECS is shown.

Figure 4: Example of an ECS



In this figure, 4 vessels are scheduled on a terminal with 3 cranes. The big rectangles represents vessels, and the small black rectangles represent a time period of a crane working. For example, vessel 1 berths from the first until the 8th time period, and has crane 1, 2 and 3 equipped until the 4th time period. From the 5th until the 8th period, only crane 1 is equipped at vessel 1.
Another way to visualize an ECS can be found in Figure 5.

Figure 5: Extensive crane schedule

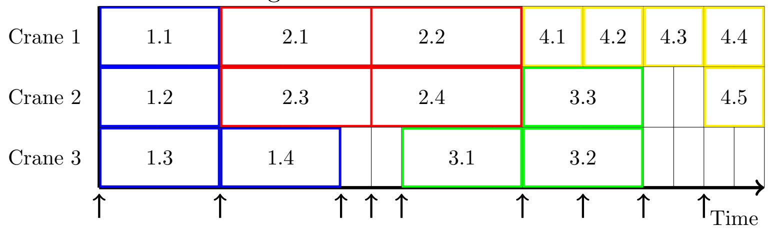


In this visualization of the ECS the different holds are distinguishable. Every rectangle represents a hold of a vessel. Note that Figure 4 and Figure 5 contain the same information.

4.1.3 Hold Schedule

When a QC works on a vessel, it is assigned to a given hold on the vessel. This is scheduled in the hold schedule, an example is shown in Figure 6.

Figure 6: Hold Schedule

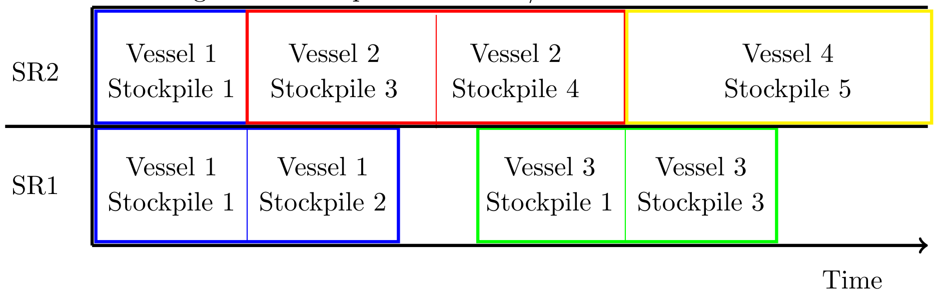


In this picture, a number in rectangle represents the vessel and the hold. For example, 1.2 represents hold 2 of vessel 1. The arrows will be explained in Section 5. This hold schedule is created from the ECS.

4.1.4 Stockyard Schedule

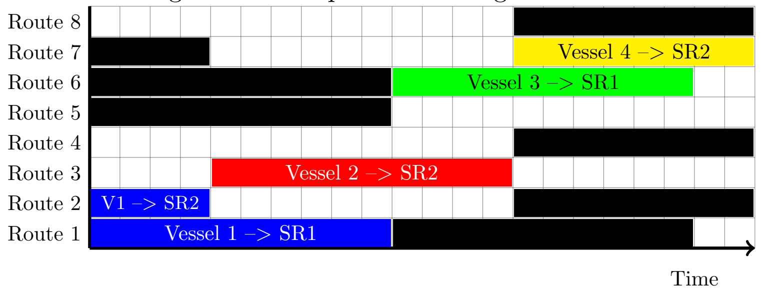
When the hold schedule is created, every hold can be seen as a job. Every job needs the right equipment at the stockyard. The stockyard schedule represents a time planning for the routes on the belt conveyor network, the stacker/reclaimers and the assignment of the stockpiles. An example of a time schedule of SRs is shown in Figure 7.

Figure 7: Example of a stacker/reclaimer schedule



In this example, SR1 services vessels 1 and 3, whereas S2 first services vessel 1, 2 and 4.
A similar schedule can also be used for the routes, where each route has a schedule, and where routes are made unavailable if they overlap with a used route. Then it is easy to see which routes are accessible to use. An example of such a route schedule is shown in Figure 8.

Figure 8: Example of a routing schedule



In this example, the material of vessel 1 is transported on route 1 from the quay to SR1. Because route 5 and 6 overlap with route 1, these are made unavailable. Vessel 1 also uses route 2 to SR2.
In this research we do not consider individual belt conveyors, only complete routes on the network are taken into account. We argue this is a reasonable assumption, as the amount of possible routes on a dry bulk terminal is usually limited.

4.2 Seaside

4.2.1 General formulation

In the seaside problem, the main objective is to minimize the turnaround time of vessels and to minimize the use of different QCs. To achieve this, all the equipment restrictions need to be satisfied. We can view this problem as follows:

Seaside Problem

Input:	Vessel schedule
Objective:	Minimize turnaround time & changes in equipment
Subject to :	Feasible berth planning
	Feasible crane planning
Output:	Extended crane schedule

In the model, the following assumptions are made:

- The terminal has one quay
- All holds on a vessels are loading, or all holds on a vessel are unloading

- Setup times are included in the service times.
- Beam and draft of a vessel is not restricting
- Cranes (un)load continuously
- The berth is continuous
- All quay cranes have the same capacity
- All belt conveyors have the same capacity
- Time is discrete
- When a vessel is berthing, it has to be serviced.
- Loading vessels desire a berthing position at the left side of the berth
- Unloading vessels desire a berthing position at the right side of the berth
- Quay cranes cannot cross each other
- Holds have to be (un)loaded without interruption
- Starting on a new hold is only possible at berthing time of the vessel, or at the finishing time of another hold (of the same vessel)

The last assumption is made to simplify the problem for the stockyard.
This implies that starting to work on a hold is only possible when another hold is finished. Hence, starting to work on a hold of a vessel is not allowed when a QC is working on another hold of the same vessel. Consequently, when multiple QCs are equipped on a vessel, they start simultaneously on a new hold. When all assumptions and constraints are satisfied, the output will be an ECS.

4.2.2 Mathematical formulation

To define the problem precisely, a mathematical formulation is presented. The presented MIP is based on Meisel and Bierwirth [2009]. This MIP solves P1 and creates an ECS.
Note that in this report, a variable superscript belongs to the variable name due to the number of variables. The subscript denotes the index. Hence $x^{*}_{1,1}$ and $x^{**}_{1,1}$ are different variables. An important adjustment is the addition of the individual crane assignment. In some constraints, a number of cranes is assigned, this is represented by q. In other constraints, an specific crane is used, this is represented by c. Hence if $x_{v,t,q} = 3$, this means that 3 cranes are assigned to vessel v at time t. When $x^{c}_{c,v,t} = 3$, crane 3 is assigned to vessel v at time t. In Table 2, first all sets and parameters are stated, then the variables are presented.

Table 2: Symbols (Seaside)

Symbol	Description	type
V	set of vessels v $(1, 2, 3, ..., N)$	set
V^u	set of unloading vessels	subset
V^l	set of loading vessels	subset
C	set of QCs c $(1, 2, 3, ..., C)$	set
C_v	QCs which are allowed to handle vessel v	subset
T	set of time periods t $(1, 2, 3, ..., H)$	set
L	length of berth (in berth segments)	parameter
l_v	length of vessel in berth segments v	parameter
m_v	workload per hold of vessel v in QC time periods	parameter
h_v	number of holds on vessel v	parameter
t_v^a	expected arrival time of vessel v	parameter
t_v^m	minimum finishing time for vessel v	parameter
t_v^r	maximum finishing time for vessel v	parameter
R_v	set of possible number of assigned cranes for vessel v	set
λ_v^1	penalty cost if the vessel finishes after the desired finishing time	parameter
λ_v^2	penalty cost if the handling time is longer than the minimum handling time	parameter
λ_v^3	penalty cost for not berthing on the desired position	parameter
λ_v^4	penalty cost for changing number of cranes	parameter
b_v	berthing position of vessel v	integer variable
t_v^s	starting time of handling vessel v	integer variable
t_v^f	finishing time of handling vessel v	integer variable
x_v^L	penalty variable, if the vessel finishing time is later than the maximum finishing time	integer variable
$x_{v,t}$	1 if at least one QC is assigned to vessel v at time t	binary variable
$x_{v,t}^*$	artificial variable which measures a change in crane assignment on vessel v at time t	binary variable
$x_{v,t}^n$	artificial variable to force a hold of vessel v to start at $t_v^s + n \cdot m_v$	integer variable
$x_{v,t}^{**}$	artificial variable to ensure constraint 21 holds for vessel v at time t, if no crane change occurs	integer variable
$x_{c,v,t}^c$	1 if QC c is assigned to vessel v in time period t	binary variable
$p_{c,t}^c$	position of QC c at time t	integer variable
$x_{v,t,q}$	1 if q QCs are assigned to vessel v at time t	binary variable
$y_{v,v'}$	1 if vessel v is berthed left of vessel v'	binary variable
$z_{v,v'}$	1 if handling of vessel v ends before the handling of vessel v' starts	binary variable

The MIP is formulated as follows:

$$\textbf{Obj.:} \quad \min \sum_{v\in V}(\lambda_v^1 x_v^L + \lambda_v^2(t_v^f - t_v^m)) + \sum_{v\in V^u}\lambda_v^3(b_v - 1) + \sum_{v\in V^l}\lambda_v^3(L - b_v - l_v) + \lambda^4 \sum_{v\in V}\sum_{t\in T} x_{v,t}^* \tag{1}$$

S. t.:

$$t_v^s \geq t_v^a \qquad \forall v \in V \tag{2}$$
$$t_v^f \leq H \qquad \forall v \in V \tag{3}$$
$$b_v \leq L - l_v \qquad \forall v \in V \tag{4}$$
$$t_v^f - t_v^r \leq x_v^L \qquad \forall v \in V \tag{5}$$
$$\sum_{t\in T}\sum_{q\in R_v} q x_{v,t,q} = h_v m_v \qquad \forall v \in V \tag{6}$$
$$\sum_{q\in R_v} x_{v,t,q} = x_{v,t} \qquad \forall v \in V, t \in T \tag{7}$$
$$\sum_{t\in T} x_{v,t} = t_v^f - t_v^s \qquad \forall v \in V \tag{8}$$
$$(t+1)x_{v,t} \leq t_v^f \qquad \forall v \in V, t \in T \tag{9}$$
$$t x_{v,t} + H(1 - x_{v,t}) \geq t_v^s \qquad \forall v \in V, t \in T \tag{10}$$
$$y_{v,v'} + z_{v,v'} + z_{v',v} \geq 1 \qquad \forall v\in V^l, v'\in V^u \tag{11}$$
$$b_v + L(1 - y_{v,v'}) \geq b_{v'} + l_{v'} \qquad \forall v,v'(v\neq v')\in V \tag{12}$$
$$t_v^s + H(1 - z_{v,v'}) \geq t_{v'}^f \qquad \forall v,v'(v\neq v')\in V \tag{13}$$
$$y_{v,v'} + y_{v',v} + z_{v,v'} + z_{v',v} \geq 1 \qquad \forall v,v'(v\neq v')\in V \tag{14}$$
$$\sum_{c\in C_v} x_{c,v,t}^c = \sum_{q\in Q} q x_{v,t,q} \qquad v \in V, t \in T \tag{15}$$
$$p_{c,t}^c \leq b_v + (1 - x_{c,v,t}^c)H \qquad c \in C, v \in V, t \in T \tag{16}$$
$$p_{c,t}^c \geq b_v - (1 - x_{c,v,t}^c)H \qquad c \in C, v \in V, t \in T \tag{17}$$
$$p_{c,t}^c \geq p_{c-1,t}^c \qquad c \in C, t \in T \tag{18}$$
$$x_{c,v,t}^c - x_{c,v,t-1}^c \geq -x_{v,t}^* \qquad v \in V, t \in T, c \in C \tag{19}$$
$$x_{c,v,t}^c - x_{c,v,t-1}^c \leq x_{v,t}^* \qquad v \in V, t \in T, c \in C \tag{20}$$
$$t x_{v,t}^* = t_v^s + x_{v,t}^n m_v - x_{v,t}^{**} \qquad v \in V, t \in T \tag{21}$$
$$H(1 - x_{v,t}^*) \geq x_{v,t}^{**} \qquad v \in V, t \in T \tag{22}$$
$$b_v, t_v^s, t_v^f, x_v^L, x_{v,t}^n, x_{v,t}^{**}, p_{c,t}^c \in \mathbb{N} \qquad \forall v \in V, t \in T, c \in C \tag{23}$$
$$x_{v,t}, x_{v,t}^*, x_{c,v,t}^C, x_{v,t,q}, y_{v,v'}, z_{v,v'} \in \mathbb{B} \qquad \forall v,v'\in V, t\in T. q\in R \tag{24}$$
$$\tag{25}$$

The objective (1) is a sum of different variables. The first objective variable creates a penalty when a vessel is finished after the maximum finishing time. The second objective variable gives a penalty for finishing after the minimum finishing time. Hence, a vessel is penalized double if it is still serviced after the desired finishing time. The third objective variable measures the distance between the vessel and the end of the berths.
We assume the desired berthing position of loading vessels is close to the left end of the berth, whereas unloading vessels desire to berth closer to the right end of the berth. The fourth objective variable measures changes in crane equipment. Constraints (2) ensure that a vessel is not handled before arrival time, where constraints (3) ensure all vessels are handled before the end of the planning horizon. Constraints (4) limit the berth positioning. Constraints (5) define the penalty variable when a vessel is finishing too late. Constraints (6) ensure that enough cranes are assigned to handle the complete vessel. Note that the set R_v includes information about the maximum number of cranes allowed to be assigned to a vessel. Constraints (7) enforces $x_{v,t}$ to be 1, if any number of cranes is assigned. Constraints (8) - (10) set the starting and ending time of a vessel, and enforce a crane assignment at every time unit a vessel is being serviced. Constraints (11) enforce unloading ships to berth right from loading ships. Constraints (12) - (14) prevent vessel overlapping in time and space. To create the crane schedule, constraints (15) assign cranes to vessels. Constraints (16) - (18) ensure that cranes with a higher number always have a position left of the cranes with a lower number. To ensure that working on a hold only starts at the starting time or at the finishing time of another hold constraints (19) - (22) are included. Constraints (19) - (20) enforce the variable $x^*_{v,t}$ to be 1 if a crane change occurs. Constraints (21) ensure that $x^*_{v,t}$ can only be 1 at $t^s_v + x^n_{v,t} m_v$, where $x^n_{v,t}$ is an integer variable. If $x^*_{v,t} = 0$, constraints (22) will allow x^{**} to get the value needed to satisfy constraint (21). Note that constraints (16) - (22) are implemented to create the individual crane assignment. Finally, (23) - (24) define the domains of all variables.

4.3 Stockyard

4.3.1 General formulation

The output created by the seaside problem, the ECS, is used as input for the stockyard problem. At the stockyard, 2 problems occur, P2 and P3, as explained in Figure 3. In P2 the hold schedule is created. The formulation of P2 can be found in Appendix 10.2. In this section, we only consider P3, where we assume that P2 is already solved. Hence, the hold schedule is the input of the following model. Working on a hold is called a job in this section.

	Hold Schedule
Input:	Extensive crane schedule
Objective:	Minimize changes in equipment
Subject to :	Feasible conveyor belt planning
	Feasible stacker/reclaimer planning
	Feasible stockpile planning
Output:	Stockyard schedule

The following assumptions are made:

- a job is serviced by 1 route and 1 SR
- routes sharing the same belt are overlapping, and can not be used simultaneously
- crossing routes are non-overlapping routes
- a stockpile is fixed in size, capacity and material
- the setup time of a route/SR is ignored

When a solution is found, the output can be described in two figures. The time schedule of SRs, and the time schedule of the routes. If needed, the time schedule of routes can be decomposed in a schedule of belt conveyor parts.
If multiple routes use the same belt conveyor in the same order, they are considered as one unique route. For example, if two routes are equal except for the starting position at the quay conveyor, they are considered as the same route.

4.3.2 Mathematical formulation

To define the model precisely, a mathematical formulation for P3 is created. The notation of this mathematical formulation is explained in Table 3. A job is defined as the uninterrupted working of one crane on a hold.

Table 3: Symbols (Stockyard)

Symbol	Meaning	type
J	set of jobs (indexed by j)	set
R	set of routes (indexed by r)	set
S	set of stockpiles (indexed by s)	set
S^j	subset of stockpiles suitable for job j	set
A	set of stacker/reclaimers (indexed by a)	set
λ	penalty for changing equipment	parameter
z_{ij}	1 if job i and j belong to same vessel	binary parameter
m_j	amount of material for job j	parameter
L_j	1 if job j is loading, -1 if job is unloading	{-1,1} parameter
$L_{j,r}$	1 if job j and route r have same function (both loading, or both unloading)	binary parameter
$\alpha^a_{s,a}$	1 if stockpile s can be serviced by SR a	binary parameter
$\alpha^r_{r,a}$	1 if route r contains SR a	binary parameter
γ_j	flow per time period for job j	parameter
c^s_s	maximum capacity of stockpile s	parameter
$c^s_{s,0}$	Inventory level of stockpile s at time 0	parameter
c^r	capacity of route r per time period	parameter
c^a	capacity of SR a per time period	parameter
q_{ij}	if job i is finished before job j	binary parameter
p_{ij}	1 if jobs i and j are at the same time (1 if $i = j$)	binary parameter
$w_{r,r'}$	1 if route r and r' have overlap (1 if $r = r'$)	binary parameter
M	Large positive number	parameter
$x^s_{j,s}$	1 if stockpile s is used for job j	binary variable
$x^r_{j,r}$	1 if route r is used for job j	binary variable
$x^a_{j,a}$	1 if SR a is used for job j	binary variable
$c^s_{j,s}$	remaining capacity of stockpile s before job j starts	variable
$m^s_{j,s}$	amount of material stored on stockpile s by job j	variable

$$\textbf{Obj.:} \quad \min \sum_{a \in A} \sum_{v \in V} \lambda x^a_{j,a} \tag{26}$$

$$\text{S.t.:} \sum_{s \in S^j} m^s_{j,s} \geq m_j \qquad \forall j \in J \tag{27}$$

$$m^s_{j,s} \leq c^s_{j,s} \qquad \forall j \in J, s \in S^j \tag{28}$$

$$c^s_{j,s} = c^s_{s,0} - \sum_{i \in J} L_i q_{ij} m^s_{s,i} \qquad \forall j \in J, s \in S^j \tag{29}$$

$$c^s_{j,s} \leq c^s_s \qquad \forall j \in J, s \in S^j \tag{30}$$

$$M x^s_{j,s} \geq m^s_{j,s} \qquad \forall j \in J, s \in S^j \tag{31}$$

$$\sum_{a \in A} \alpha^a_{s,a} x^a_{j,a} = x^s_{j,s} \qquad \forall j \in J, s \in S^j \tag{32}$$

$$\sum_{a \in A} x^a_{j,a} = 1 \qquad \forall j \in J \tag{33}$$

$$(1 - z_{ij}) p_{ij} (x^a_{j,a} + x^a_{i,a}) \leq 1 \qquad \forall i, j \in J, a \in A \tag{34}$$

$$\sum_{r \in R} L_{j,r} \alpha^r_{r,a} x^r_{j,r} = x^a_{j,a} \qquad \forall j \in J, a \in A \tag{35}$$

$$\sum_{i \in J_c} \gamma_i x^a_{i,a} \leq c^a \qquad \forall c \in C, J_c, a \in A \tag{36}$$

$$\sum_{i \in J_c} \gamma_i x^r_{i,r} \leq c^r \qquad \forall c \in C, J_c, r \in R \tag{37}$$

$$\sum_{i \in J/\{j\}} \sum_{r' \in R/\{r\}} w_{r,r'} p_{ij} x^r_{r',i} \leq M(1 - x^r_{j,r}) \qquad \forall j \in J, r \in R \tag{38}$$

$$\sum_{j \in J_v} x^a_{a,j} \leq M x^a_{v,a} \qquad \forall v \in V, a \in A \tag{39}$$

$$m^s_{j,s}, c^s_{j,s} \in \mathbb{R}, \quad x^s_{j,s}, x^r_{j,r}, x^a_{j,a} \in \mathbb{B} \qquad \forall j \in J, s \in S^j, a \in A, r \in R, \tag{40}$$

The objective (26) minimizes distance and changes in equipment. Constraints (27) ensure that enough storage space is allocated, where constraints (28) validates if the chosen stockpiles have enough remaining storage capacity. These remaining storage capacities are calculated by constraints (29). Constraints (30) restrict the remaining storage capacity to the maximum capacity of the stockpile.
Constraints (31) enforce variable $x^s_{j,s}$ to be 1, if $m^s_{j,s}$ is positive.. Constraints (32) assign a SR to a stockpile. Constraints (33) - (35) assure that only one SR is assigned to a job, and assign one route to the SR. Constraints (36) and (37) ensure that the capacity of the SR and route is respected. Constraints (38) prevent the usage of overlapping routes at the same time. Constraints (39) define the SRs used per vessel. The last constraints (40) define the domain of the variables.

4.4 Extensions

4.4.1 Integration between seaside and stockyard

In the basic model, there is no integration between the seaside and stockyard. In order to incorporate this, a few additional constraints can be added to the seaside problem. The parameters of these additional constraints can be updated if the stockyard problem is infeasible, in order to make the stockyard problem feasible.
For example, constraints (41) enforce a minimum handling time of t_v^h for a vessel, and constraints (42) prevent a vessel to be serviced before β. Constraints (43-44) prevent given combinations of vessels to be planned at the same time, where $\phi_{v,v'}$ can be set to 1 if a certain combination is not allowed. The same holds for $\phi_{v,v',v''}$.

$$t_v^f - t_v^s \geq t_v^h \qquad \forall v \in V \tag{41}$$

$$t_v^s \geq \beta \qquad \forall v \in V \tag{42}$$

$$z_{v,v'} + z_{v',v} \geq \phi_{v,v'} \qquad \forall v, v'(v \neq v') \in V \tag{43}$$

$$\sum_{i \in v,v',v''} \sum_{j \in \{v,v',v'']\}/i} z_{i,j} \geq \phi_{v,v',v''} \qquad \forall v,v',v''(v \neq v', v \neq v'', v'' \neq v') \in V \tag{44}$$

How these additional constraints are constructed, is explained in Section 6. Next to this, the set R_v can be adjusted. Hence, for some vessels it is not allowed to have their maximum number of cranes.
The summation in constraint 44 is equal to:

$$\sum_{i \in v,v',v''} \sum_{j \in \{v,v',v'']\}/i} z_{i,j} = z_{v,v'} + z_{v,v''} + z_{v',v} + z_{v',v''} + z_{v'',v} + z_{v'',v}$$

5 Scheduling heuristics

In this section, the methods to solve the scheduling problems are presented. For the seaside, an existing heuristic is adjusted and used. For the stockyard, a new approach is applied.

5.1 Seaside

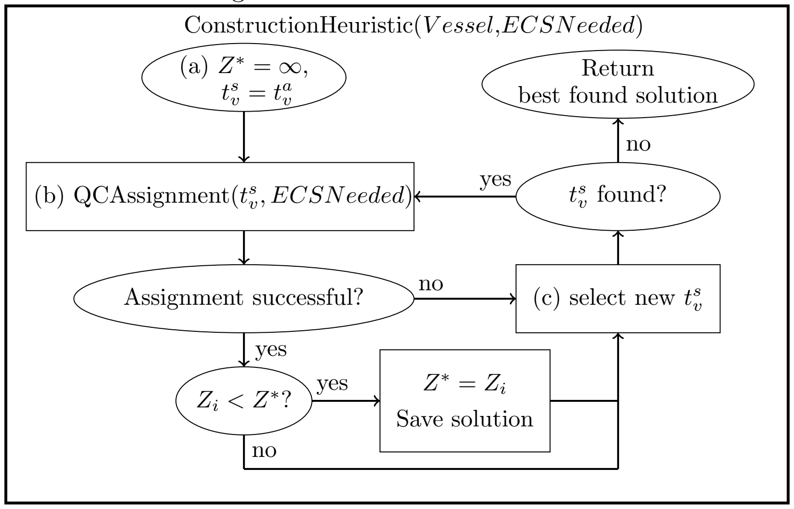
In this subsection, we use the terminology from Meisel and Bierwirth [2009]. Three methods will be explained. In addition to Meisel and Bierwirth [2009], the individual crane assignment is considered. However, because this increases the computation time, this is not always computed in the first two methods. When this is not solved, P1 is not completely finished. This schedule is mentioned as

the reduced crane schedule (RCS) in the remainder of this report. Whether the RCS or the ECS is calculated is stated with the Boolean variable $ECSNeeded$. When this Boolean variable is false, the RCS will be calculated and only the number of cranes will be assigned. The ECS can be obtained from the RCS in most cases. For a few cases, this is not feasible, then the ECS need to be calculated in every iteration of the heuristic.
The first method is a simple construction heuristic, which plans the vessel sequentially without taking future vessels into account. The second method takes all future vessels into account when assigning cranes to a vessel. The third method tries the second method multiple times with another priority list of vessels. The following abbreviations are used: Construction heuristic (CH), Local Refinement (LR), Squeaky Wheel Optimization (SWO).

5.1.1 Construction heuristic

The CH is an adjusted version of the construction heuristic by Meisel and Bierwirth [2009]. In this thesis, the distance to the favorite position does not influence the assignment of the quay cranes or the handling time. In the remainder of this report, all heuristic are explained as functions. It is assumed that the vessel schedule is accessible for all functions, so it is not shown as input. The CH validates different starting times iteratively, by checking if a feasible assignment of QCs exist. If this succeeds, the solution is saved. At the end, the best solution is chosen. The heuristic is explained in Figure 9.

Figure 9: Construction Heuristic


ConstructionHeuristic($Vessel$,$ECSNeeded$)
(a) $Z^* = \infty$, $t_v^s = t_v^a$
Return best found solution
no
yes
(b) QCAssignment(t_v^s, $ECSNeeded$)
t_v^s found?
Assignment successful?
no
(c) select new t_v^s
yes
$Z_i < Z^*$?
yes
$Z^* = Z_i$
Save solution
no

In the CH, all vessels are planned individually. The input of the function is the vessel to be planned. The second input is a Boolean variable whether the complete ECS is needed as output. First, in step (a), the variables are initialized.

Secondly, at step (b), QCs are assigned to the vessel. If this succeeds and the new solution improves on other solutions, the solution is saved. At step (c) a possible starting time is selected. If a new starting time is found, the procedure starts again at (b). If no new starting time is found, the best solution found
The most important step of this heuristic is the assignment of the QCs in step (b). In this step, the position of the vessel at the berth is also determined.
Another important step is the selection of the berthing time. Since time is discretized, all possible berthing times can be considered (because a finite time horizon is used). At step (c), t_v^s is increased by one, and a lower bound of the objective value for starting at t_v^s is calculated.
This lower bound is based on the minimum handling time of a vessel. To speed up the heuristic, the lower bound on the objective value while starting at t_v^s will be compared to the best found solution. If the lower bound exceeds the best found solution, t_v^s is again increased by one and the lower bound is calculated again. Because increasing the starting time after the finishing time of the final vessel will not improve the solution, this is used as stopping criteria.
The pseudo-code for the function QCAssignment is presented in Algorithm 1.

Algorithm 1: QCAssignment

```
1  Function QCAssignment($t_v^s$, ECSNeeded)
2      HoldToPlan = 1;
3      while Not all Holds planned do
4          HoldStartTime = GetHoldStartTime($t_v^s$,CranesAvailable,
             CranesOnVessel, HoldToPlan);
5          if No Holdstart is found then
6              return No QC Assignment Found;
7          end
8          if ECSNeeded then
9              ECS = CreateECS(ECS,CranesOnVessel, CranesAvailable,
                 HoldStartTime);
10         end
11         if no ECSNeeded OR ECS is Feasible then
12             Update CranesOnVessel and CranesAvailable;
13             HoldToPlan = HoldToPlan + 1;
14         end
15     end
16     $b_v$ = CalculateBerthingPosition($t_v^s$,CranesOnVessel);
17     if no $b_v$ is found then
18         return No QC Assignment Found;
19     end
20     $Z$ = GetObjectiveValue($t_v^s$,CranesOnVessel,$b_v$);
21     return QC Assignment and $Z$;
22 end
```

At the start of the function, the variable *HoldToPlan* is initialized (Line 2).

Then, the actual planning of the holds start (Line 3). The function $GetHoldStart$ (Line 4) calculates the starting time for a hold, and validates if enough cranes are available to finish the hold. If no starting time can be calculated, the algorithm terminates (Line 6), and no QC Assignment is found. If a starting time of a hold is found, the algorithm validates whether an ECS is needed (Line 8). If this is needed, an ECS will be created by the function $CreateECS$. The function also determines a feasible berthing position of the vessel. The function $CreateECS$ is explained in the appendix. If a $CraneSchedule$ is found, the hold planning is finished for the current hold and saved in the variables $CranesOnVessel$ and $CranesAvailable$ (Line 12). Both variables are vectors, in vector $CranesOnVessel$ the values for the $HoldStartTime$ until the time the hold is finished are increased by one. For the vector $CranesAvailable$ the same occurs, but the values are decreased by one. Next, the following hold will be planned. In the case no crane schedule can be created, the function $GetHoldStart$ will be called again for the same hold ($HoldToPlan$). If another different possible starting time exist, this will be returned by the function $GetHoldStart$. When this is not possible, the function ends (Line 6). When all holds are planned, the berthing position is calculated (Line 16), if this is not possible the algorithm terminates. Otherwise, the objective value is calculated and returned with the QC Assignment (Line 21).
When the CH is used and an ECS is needed, it is first carried out without calculating the ECS. Only the RCS is calculated, and the ECS will be created using the RCS at the end of the procedure. When this is not possible, the heuristic will be carried out again with calculating the ECS during the procedure. This increases the computation time. This approach to use the CH is referred to as the function $applyConstructionHeuristic$ in the remainder of this report.

5.1.2 Local refinement procedure

In order to improve the results of the construction heuristic, Meisel and Bierwirth [2009] created a local refinement procedure. This procedure is explained in Algorithm 2. The LR plans each vessel a few times, and each planning is completed with all other vessels. The planning with the best total objective value determines the planning for the current vessel.

Algorithm 2: Local refinement procedure

```
1  Function LocalRefinement(PriorityList,ECSNeeded)
2      for CurrentVessel in PriorityList do
3          for TempMaxCranes in 1:MaxCranes[Vessel] do
4              MaxCranes[CurrentVessel] = TempMaxCranes;
5              for Vessel in Prioritylist [ $\geq$CurrentVessel ] do
6                  ConstructionHeuristic(Vessel,ECSNeeded);
7              end
8              ReassignCranes(CurrentVessel);
9              SaveCurrentSolution(TempMaxCranes);
10         end
11         PlanBestSolution(CurrentVessel);
12     end
13 end
```

The input of this algorithm is a priority list of vessels. When the LR is used individually this is the sequence 1,2,3,...,N. The third method uses other priority lists as input for the LR. The vessels are planned in the order of the priority list. For each vessel, different numbers of the maximum cranes allowed are considered (Line 3). The maximum number of cranes is adjusted, and the construction heuristic is carried out for all the vessels which are later on the priority list (Line 5). When all vessels are planned, the algorithm validates if it is possible to assign additional cranes to the vessel considered (called $CurrentVessel$ (Line 8)). If this is possible, additional cranes are assigned. At Line 9, the solution value is calculated. When this is done for all possible values for the maximum number of cranes allowed, the best solution is obtained and the $CurrentVessel$ is planned (Line 11). Then the next vessel in the priority list is considered.
When the LR is used and an ECS is needed, the LR is used without calculating the ECS during the procedure. The ECS is created using the RCS at the end. If this is not possible, the same approach used in the CH is used. The function $ApplyLocalRefinement$ is used in the remainder of this report to refer to this.

5.1.3 Squeaky Wheel Optimization

The squeaky wheel optimization (SWO) is similar to the approach of Meisel and Bierwirth [2009]. Using the SWO, the vessels are planned using the LR procedure based on a priority list. Next, the contribution of the individual vessels to the overall objective is calculated. If a vessel has a higher contribution than the vessel planned before, these two vessels are swapped in the planning order, and the LR procedure is performed again. If a given order occurs for the second time, the SWO heuristic is trapped in a cycle. If the heuristic remains stuck in a cycle, it is stopped. To speed up the procedure, the RCS is calculated during the procedure to determine the priority order. If the procedure stops, the final schedule with ECS is calculated based on the best priority order. The pseudo-code for the SWO can be found in Algorithm 3

Algorithm 3: SWO

```
1  Function SWO
2      Z = $\infty$; PriorityOrder = 1,2,3,.., N; Continue = TRUE;
3      while Continue do
4          RCS = LocalRefinement(PriorityOrder, FALSE);
5          ObjectiveValue = getObjectiveValue(RCS);
6          PriorityOrder = getPriorityOrder(ObjectiveValue);
7          SavePriorityOrder(ObjectiveValue,PriorityOrder);
8          Cycle = CheckNew(PriorityOrder);
9          while Cycle and Continue do
10             RCS = UseConstructionHeuristic(FALSE);
11             ObjectiveValue = getObjectiveValue(RCS);
12             PriorityOrder = getPriorityOrder(ObjectiveValue);
13             Cycle = CheckNew(PriorityOrder);
14             Continue = ContinueAlgorithm(PriorityOrder);
15         end
16     end
17     PriorityOrder = GetBestPriorityOrder();
18     FinalSolution = LocalRefinement(PriorityOrder, TRUE);
19 end
```

The SWO starts with creating a RCS with the initial priority order (1,2,3...N) (Line 2). At Line 4, the RCS is created. Note that the RCS is the ECS without the individual crane assignments, hence only a number of cranes is assigned. Using this RCS, the function *getObjectiveValue* calculates the objective value per vessel. Next, the function *getPriorityOrder* determines the new priority order. The function *SavePriorityOrder* in Line 6 saves the priority order together with the objective value of the RCS. The *CheckNew* function validates if a priority order is already used (Line 8). If an order is already used, the algorithm is trapped in a cycle. To solve this cycle, the construction heuristic (Line 10) is used to create another priority order in the same way (Line 11-12). The function *UseConstructionHeuristic* applies the CH for all vessels with the argument ECSNeeded set to False. In this cycle, every iteration a new priority order is created which is validated in *CheckNew* whether these priority order are new. The function *ContinueAlgorithm* (Line 14) checks if the construction heuristic produces a priority order which was produced earlier in the same cycle. If this occurs, the loop ends.
If the loop is finished, the best priority order is obtained and an ECS is created at Line 18.

5.1.4 CPLEX & SWO + CPLEX

To compare the solutions of the SWO, the MIP (in Section 4.2.2) for the seaside is implemented in CPLEX to obtain optimal solutions. CPLEX refers to the solver with only the MIP as input without a base solution. SWO + CPLEX refers to the CPLEX solver, where the solution of the SWO is used as a base solution.

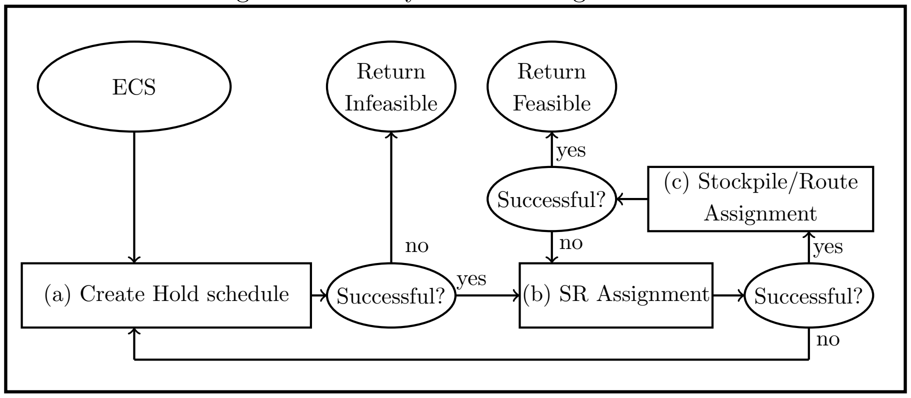
5.2 Stockyard schedule

If an ECS is created, the stockyard will validate if a feasible corresponding stockyard schedule can be created.

5.2.1 Scheduling heuristic

The scheduling heuristic consists of two parts, the first part assigns holds to different slots in the extensive crane schedule (this solves problem P2). The second part assigns the equipment (P3). The scheduling heuristic is explained in Figure 10.

Figure 10: Stockyard scheduling Heuristic

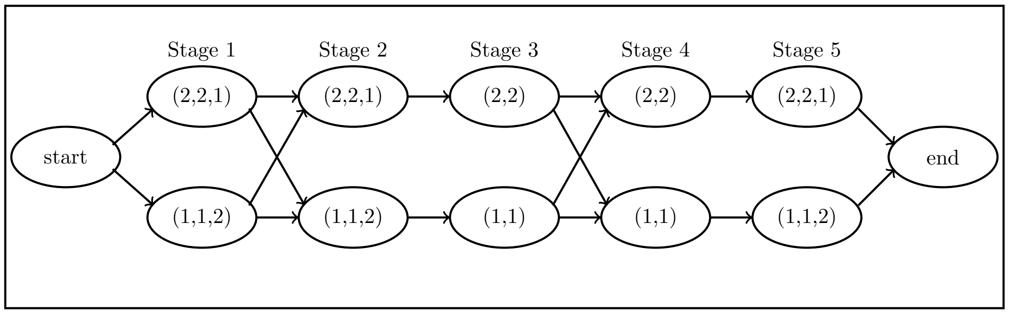


The input in this heuristic is the ECS. An example of this input is given in Figure 4. This needs to be converted into a schedule as pictured in Figure 6. In step (a), a hold schedule will be created and hence P2 will be solved. In the different solution strategies, different methods to solve step (a) will be used.
When the ECS contains one vessel, all possible hold schedules will be created iteratively. These hold schedules are ordered on a score based on the types of a material being serviced at the same time. This score is calculated by calculating the sum of unique materials which are served at each time unit. If at time t, 2 unique materials are handled, the score for time t for the corresponding hold schedule is 2. This is done for all possible t The hold schedule with the lowest score is used first.
When an ECS is created for a complete vessel schedule, two approaches are used. In the first approach, just one hold schedule is made and if this one is not feasible, the heuristic terminates. In the second approach, a hold schedule is created for one vessel, and every step one vessel is added to the hold schedule.

When the algorithm is not able to find a feasible solution, the hold schedules of the last two vessels considered are changed based on the same scoring method as explained before.

If the hold schedule is obtained, we will solve P3. In the hold schedule, each hold will be seen as a job for the stockyard. Important information about a job is the starting and ending time, the type of material and the flow per time unit. If this schedule is created, the SR assignment can be done in step (b). The arrows in Figure 6 are the times at which a equipment changes take place. A hold starts, or a hold is finished. At these moments, a decision needs to made. These moments are called stages in the remainder of this report. Based on these stages, a multipartite network is constructed. Every stage contains nodes which represent an SR-assignment for the different jobs. All nodes are connected with the feasible SR-assignments of the next stage, given the current SR-assignment. Each stage represents the start of a new job (the arrows in Figure 6 refer to the different stages). At each stage, SRs needs to be assigned. This assignment remains the same until the next stage. In the network, a node represents the SR-assignment at a stage. Hence the node (1,1,3) at stage 1 implies that the jobs on crane 1 and 2 are serviced by SR 1 and the job on crane 3 is serviced by SR 3. All nodes in a stage are connected with nodes in the subsequent stage. The distance on these edges is the cost of the assignment in the next stage (with an additional cost if equipment is changed for a hold belonging to the same vessel). A route from the source to the sink in this network corresponds to a feasible schedule. These routes are constructed by Dijkstra's algorithm.
Assume for the example in Figure 6, that hold 1.1, 1.2, 2.1, 2.3, 2.2 and 2.4 contain the same material. Similarly, hold 1.3, 1.4, 1.5, 3.1 contain the same material. Assume that 2 SRs are available, and both materials can be stored in the range of both SRs. Then a corresponding example network (for the first part of the schedule in Figure 6) can be constructed as follows:

Figure 11: Example SR assignment network



In this example, Stage 1 corresponds to working on hold 1.1, 1.2 and 1.3. Stage 2 represents working on hold 2.1, 2.3 and 1.4. The node (2,2,1) at stage 1 means

that SR 2 serves hold 1.1 and 1.2, and that SR 1 serves hold 1.3. Note that the distance of the crossing arrows will be higher, because of the equipment change for crane 3, while the crane is working on the same ship. Also observe that from stage 2 to 3 there is no possibility to change, this is because in both stage 2 and 3 holds 2.1 and 2.3 are included and it is not allowed to change equipment during a job. A solution is found by finding the shortest path from the start to end.
When an SR assignment is done, corresponding stockpiles and routes are assigned to the jobs in step (c). This is done sequentially per stage by first assigning a route, by looking which routes are available per job. Then, all feasible combinations are calculated by checking if routes have overlap. If no feasible combinations exist, the route assignment fails. If at least one feasible combination exist, the routes with the minimal total distance are picked.
When the routes are assigned, the stockpiles available for each job are calculated by checking which stockpiles are accessible by the used SR. The stockpiles with the highest remaining capacity are picked in the case of unloading. When an vessel is loading, the stockpiles with the largest amount of material are chosen. If not enough material(or remaining capacity) is available, the stockpile assignment fails.
Because the route and stockpile assignment happens sequentially per stage, it is straightforward to identify the stage which causes the infeasibility. Hence, the edge used in the SR assignment network to the infeasible stage will be removed, and the shortest path algorithm will be executed again. If a successful assignment of all equipment is done, the schedule is feasible.

6 Solution strategies

The aim of the different solution strategies is to create a complete schedule for a dry bulk terminal, incorporating all the equipment restrictions. The schedule is divided in two parts as explained before, the seaside and the stockyard. For the seaside, the SWO method will be used to create the ECS. This solution will be feasible for the seaside, but it may not be feasible for the stockyard. Hence, the value of this solution will serve as a lower bound. Note that this is not a lower bound on the optimal objective value. Because the second and third method use the SWO as best solution, they will never improve on the objective value of the SWO. The first method uses the CH for each vessel individually. Hence, with these solution strategies, the SWO is a lower bound on the performance.
In this section, different methods to obtain solutions are presented. In the different methods, various scheduling methods are used. The scheduling methods are presented in Section 5. Here we explain how these scheduling methods are used, and how the stockyard and seaside communicate.
Different solution methods will be carried out in order to find the best solution method. A simple benchmark solution method will be used to compare with. The benchmark solution method plans the vessels at arrival, in a simple way.

Next to the benchmark solution a basic solution method is presented. The basic solution method is a two-phase method. In the first phase multiple ECS are created, and the stockyard tries to find a feasible stockyard schedule for the ECS. To improve upon the basic solution method, an extended solution method will be applied. In the extended solution method, a few constraints will be added to P1, and will be updated if necessary. An overview of the methods is shown in Table 1.

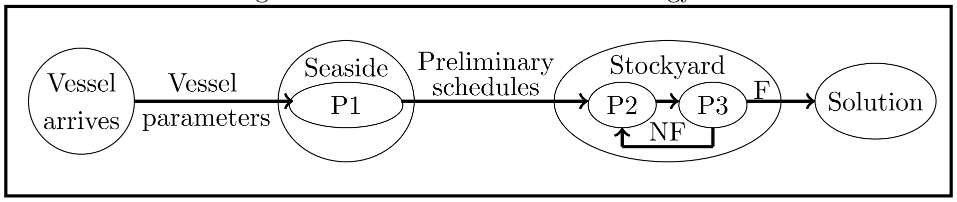
Table 4: Solution Strategies

Method	Remarks
Benchmark	Simple method
Basic	Basic method, creates upper and lower bound
Extended	Incorporates communication between the seaside and stockyard problem

6.1 Benchmark solution strategy

In this method a planning is made per vessel at the moment a vessel arrives. All the previously planned vessels are taken as fixed. Multiple solutions for P1 are calculated, all these solutions are send to the stockyard. At the stockyard, the best feasible solution is picked.
When the vessel arrives, the vessel parameters are passed to the seaside. The seaside calculates several possible ECS, by considering different starting times and locations. Because vessels are planned in arrival order, the number of good possibilities is limited. These ECSs are validated by the stockyard in order of increasing objective value. The stockyard scheduling heuristic iteratively solves P2 and P3 to find a solution per vessel (as explained in section 5.2). If there is no solution for P2 for which P3 is feasible, the next ECS is considered. This method is pictured in Figure 12. Note that F represents feasible, and NF represents not feasible.

Figure 12: Benchmark solution strategy

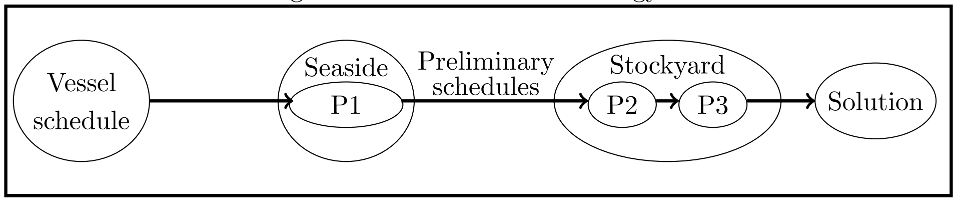


In summary: the seaside problem sends multiple schedules to the stockyard planning, where the stockyard picks the best feasible schedule, and creates a corresponding stockyard planning for the considered vessel.

6.2 Basic solution strategy

The basic solution strategy is similar to the benchmark solution strategy, but it incorporates the complete schedule at the beginning of a planning horizon. Multiple ECS are created at the seaside. The stockyard tests each ECS for feasibility by creating a hold schedule and a stockyard schedule. The feasible schedule with the lowest objective values will be used. Note that for every ECS, one hold schedule is created and tested at the stockyard problem.

Figure 13: Basic solution strategy



In Figure 13, the basic solution strategy is explained. For this solution strategy, the creation of the ECSs is an important step. This is done by creating different arrival times for vessels. For all these schedules, the CH is used to create ECSs.

Algorithm 4: Basic Solution Strategy

```
Function BasicSolutionStrategy
    AllECS := CreateSchedules;
    for ECS in AllECS do
        HoldSchedule = CreateHoldSchedule(ECS);
        Solution = StockyardConstructionHeuristic(HoldSchedule);
        if Feasible(Solution) then
            return Solution ;
        end
    end
end
```

At Line 2 in Algorithm 4 of the basic solution strategy, many ECSs are created. All these ECSs are tested in order of objective value. When, a feasible solution is found, the algorithm terminates. The most important step of this algorithm is the creation of the ECS, this is done in the function *CreateSchedules*, which is explained in Algorithm 5.

Algorithm 5: Create Schedules

```
1  Function CreateSchedules
2      for Vessel in 1,2,3,...,N do
3          for Schedules in PreliminarySchedules do
4              AddVessel(PreliminarySchedules,Vessel);
5              if length(PreliminarySchedules) >
                 MaximumNumberOfSchedules then
6                  PreliminarySchedules = Sample(PreliminarySchedules,
                     MaximumNumberOfSchedules);
7              end
8          end
9      end
10     for Schedules in PreliminarySchedules do
11         ECS = applyConstructionHeuristic();
12         FinalSchedules = FinalSchedules + ECS;
13     end
14     FinalSchedules = FinalSchedules + CreateLastSchedule();
15     FinalSchedules = FinalSchedules + CreateFeasibleSchedule();
16     FinalSchedules = FinalSchedules + SWO();
17     Order(FinalSchedules);
18     return FinalSchedules;
19 end
```

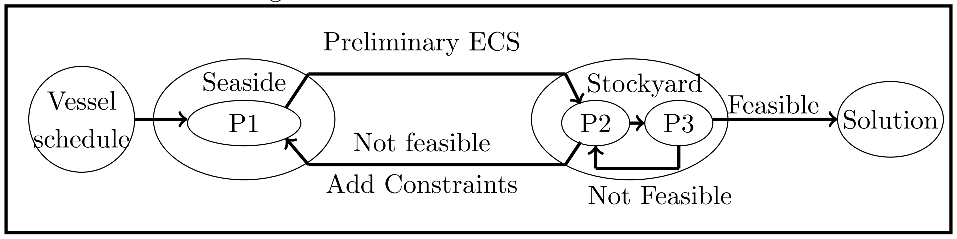
The function *CreateSchedules* starts with a loop of all vessels (Line 2). In the function $AddVessel$ (Line 4), two berthing times of a vessel are added to every schedule of the collection of preliminary schedules (in the first step, a starting preliminary schedule is created by just initialize a schedule by adding the first arrival time). The first berthing time added is the original arrival time, the second berthing time added is the arrival time of the previous vessel with the minimum handling time added (if this is later than the original arrival time). Hence, for each existing schedule, 2 schedules are added to the set $PreliminarySchedules$. Because the number of schedules increases significantly (at maximum with rate 2^n), a given number of schedules are sampled if the number of created schedules is larger (Line 6).
If these vessel schedules are created, an ECS for each vessel schedule is created (Line 11). Hence, P1 is solved multiple times for different input vessel schedules. The construction heuristic is used to save time. After this, three additional schedules are created. Firstly, *CreateLastSchedule* creates a schedule where every vessel starts after the previous vessel is handled completely(Line 14). Because this will not always have a feasible solution for the stockyard, another ECS is produced. This schedule will have all vessels being handled sequentially with a maximum of one crane (Line 15). This ECS is always feasible. Hence this will serve as an upper bound. Next to this, the SWO solution will be added to serve as a lower bound (Line 16).
The ECS created will be ordered on objective value, starting from the schedule

with the best objective value (Line 17). In this order, the ECS are sent to the stockyard. At the stockyard, one hold schedule will be created for each ECS. This hold schedule will be sent to P3, to create a stockyard schedule. If this is feasible, the algorithm terminates. If this is infeasible, the next ECS will be considered.

6.3 Extended solution method

Because the solution in the basic solution method can be far from optimal because of the limited numbers of schedules which can be created, feedback from the stockyard is expected to improve the solution. In this method, one ECS is created at the seaside by using SWO and if it is infeasible at the stockyard the additional constraints as explained in Section 4.4.1 are added. Then a new schedule is made and sent to the stockyard, if this is again infeasible, more constraints will be added.
This continues in an iterative way until the stockyard schedule is feasible.

Figure 14: Extended solution method



In Figure 14, we can see that P1 is solved at the seaside for the vessel schedule. This is done using the SWO. An ECS is sent to P2, where a hold schedule is created. This is processed at P3, and if a feasible stockyard schedule is created the algorithm terminates. If this is not the case, another hold schedule from P2 will be validated. This continues until there are no possible hold schedules anymore for the given ECS. Then, feedback is sent to the Seaside, and a new ECS is created.
The algorithm is explained in detail in algorithm 6.

Algorithm 6: Extended solution strategy

```
1  Function Extended solution strategy
2      CurrentECS = applySWO();
3      HoldSchedule = CreateHoldSchedule(CurrentECS);
4      StockyardSchedule = CreateStockYardSchedule(HoldSchedule);
5      while Stockyard schedule not feasible do
6          ProblemVessels = ObtainProblemVessels();
7          DifferentHoldSchedules = CreateHoldSchedules(ProblemVessels);
8          HoldSchedule = FirstHoldSchedule();
9          while PartlyStockyard schedule not feasible OR all Holdschedules
             tested do
10             PartlyStockyardSchedule =
                 CreateStockYardSchedule(HoldSchedule);
11             HoldSchedule = NextHoldSchedule();
12         end
13         if partly Stockyard schedule not feasible then
14             ProblemVessels = ObtainProblemVessels();
15             if length(ProblemVessels) == 1 then
16                 VesselSchedule = DecreaseCranes(ProblemVessels);
17             else
18                 VesselSchedule = DelayVessel(ProblemVessels);
19             end
20             CurrentECS = applyLocalRefinement();
21         else
22             HoldSchedule = CreateHoldSchedule(CurrentECS);
23             StockyardSchedule =
                 CreateStockYardSchedule(HoldSchedule);
24         end
25     end
26 end
```

In this solution strategy, we start with the ECS obtained from the SWO (Line 2). With this ECS, a hold schedule is made (Line 3), and the stockyard schedule is created (Line 4). If this stockyard schedule is feasible, the algorithm terminates. If the stockyard schedule is not feasible, the vessels causing the infeasibility are calculated at Line 6. The function *ObtainProblemVessels* tries to create a stockyard schedule for every number of vessels. Hence, first a stockyard schedule is created for the first vessel, then for the first two vessels, and this continues in this way. At the number of vessels where it is not possible to create a stockyard schedule, the last vessel is considered as the limiting vessel. If this vessel is serviced at the same time as previous vessels, all previous vessels being serviced at the same time are considered in the set $ProblemVessels$. At Line 7, different hold schedules are created. For all vessels before the first vessel in the set $ProblemVessels$, nothing is changed. For all the vessels in the set $ProblemVessels$, all possible hold schedules are created. All vessels after the last vessel in the set $ProblemVessels$ are not included in the hold schedule. At Line 8, the first hold schedule is selected. In the loop starting at Line 9, all hold schedules are tested on feasibility. The variable $PartlyStockyardSchedule$ is a stockyard schedule for all vessels until the last vessel in the set $ProblemVessels$. If a feasible partly stockyard schedule is found, a holdschedule for all vessels is created (using the hold schedules for the $ProblemVessels$ from Line 11). At Line 23, a stockyard schedule is created. If this is feasible, the algorithm terminates. If this is not feasible, the algorithm starts again at Line 6. If a feasible partly stockyard schedule is not found, the last vessel in the set $ProblemVessels$ will be delayed, so that no time overlap exist with previous vessels. If there is just one vessel in the set $ProblemVessels$, the number of cranes is reduced. All vessels after the first vessel in the set $ProblemVessels$ are planned again using the LR heuristic at Line 20.

7 Data & Experiments

7.1 Data

7.1.1 Terminal layout

In this research, one terminal layout is considered. This terminal layout consist of the following information:

Table 5: Scenario information

Part	Number	Remarks
Berth	1	Length of 750 meter
Cranes	3	Cranes are able to load and unload vessels with a capacity of 500 tons an hour
SR	4	Reclaiming capacity of 500 tons an hour, stacking capacity is 500 tons an hour
Lanes	5	Each lane is filled with 18 stockpiles
Conveyors	13	The capacity of the conveyors is is 500 tons an hour
Vessels		4 different vessel types are used: the Handysize, Handymax, Panamax and Capesize

The layout is similar to the terminal in Figure 1, where one additional lane and an SR is added. In this case, 27 routes are possible on the belt conveyor network. Because processing a vessel can take multiple days, time units of 4 hours are used. We assume the capacity of every stockpile is 10,000 tons. The material on the stockpiles is divided as in Table 6

Table 6: Stacklanes, Material distribution

Lane	Materials
1	9 piles with Material 1, 9 piles with Material 2
2	9 piles with Material 1, 9 piles with Material 2
3	6 piles with Material 3, 6 piles with Material 4 6 piles with Material 5
4	4 piles with Material 6, 4 piles with Material 7 5 piles with Material 8, 5 piles with Material 9
5	5 piles with Material 6, 5 piles with Material 7 4 piles with Material 8, 4 piles with Material 9

7.1.2 Vessel input data

In the remainder of this thesis we use the Vessel classification in Table 7, this classification is based on Van Vianen [2015].

Table 7: Classification Bulk vessels

Name	Length	# Holds	Weight	Max Cranes	Occurrence
Handysize	190	4	20,000	2	2/6
Handymax	210	5	50,000	2	1/6
Panamax	250	7	60,000	2	2/6
Capesize	270	8	120,000	3	1/6

The maximum finishing time is calculated by dividing the total amount of material by the crane capacity per time unit.

7.1.3 Input Generation

Because there is no landside in this research, the input data of the vessels needs to be really balanced to be feasible. The material loaded need to be in the stockyard, and the material unloaded need to be less than the remaining stockyard capacity for the given material. Hence, the vessel schedule needs to be aligned with the initial stock. To create initial stock 1/3 of all stockpiles will be chosen randomly and filled up to maximum capacity. The creation of vessel schedules is explained in Algorithm 7.

Algorithm 7: Vessel schedule generation

```
1  Function Create Vessel Schedules(NoOfVessels)
2      for 1 to NoOfVessels do
3          VesselType = SampleType(VesselTypeDistr);
4          ArrivalTime = SampleArrivalTime();
5          FunctionDistr = CreateFunctionDistr();
6          VesselFunction = SampleFunction(FunctionDistr);
7          for All Holds do
8              if VesselFunction = Unloading then
9                  RemainingCapacities = CalculateRemainingCapacities();
10                 MaterialDistr =
                     RemainingCapacities/sum(RemainingCapacities);
11             else
12                 StockLevels = CalculateStockLevels();
13                 MaterialDistr = StockLevels/sum(StockLevels);
14             end
15             PreviousMaterial = Materials[Hold-1];
16             MaterialDistr[PreviousMaterial] = 3 ·
                 MaterialDistr[PreviousMaterial] ;
17             MaterialDistr = MaterialDistr/sum(MaterialDistr)
                 Materials[Hold] = sample(MaterialDistr);
18         end
19     end
20 end
```

The vessel schedule is created sequentially. The type of a vessel is chosen randomly at Line 3, using the distribution from Table 7. The first vessel will arrive at time 1, and the probability of loading or unloading is 0.5. If the function of the vessel is unloading, the inventory levels of all materials will be counted, and a probability distribution will be based on the remaining capacity to store materials (Line 9). This probability distribution is proportional to the remaining capacity per material. For loading vessels, a similar method is used based on the available amount of materials (here the probability distribution is proportional to the amounts of material in stock).
When the remaining capacity is calculated at Line 10, all loading vessels in the 3 previous vessels are not taken into account, to make sure enough remaining capacity is available. When the stock level is calculated at Line 13 the same approach is used, but now unloading vessels in the 3 previous vessels are not taken into account. This is done, because the service order of the vessels can be changed in the SWO method. For all materials except the first, the probability of having the same material as in the previous hold is multiplied by 3 at Line 17 to obtain a more homogeneous material distribution on a vessel.
The arrival times of the second vessel is calculated in the function $SampleArrivalTime$ (Line 4) by taking a random number between 0.1 and 0.5, and multiply this with the minimum handling time of the first vessel. Hence, the second vessel always

arrives while the first vessel is being handled. For the other vessels, a random number between -5 and 5 is taken. This number is added to the minimum finishing time of the vessel before.

7.2 Experiments

To test the heuristic, scenarios with different numbers of vessels will be tested. This ranges from 3 until 25 vessels. For every number of vessels, 10 scenarios are tested. The penalty costs are given in Table 8 and are (except for λ_4) proportional to the size of a vessel.

Table 8: Penalty values Bulk vessels

	λ_1	λ_2	λ_3	λ_4
Handysize	1	1	0.1	0.01
Handymax	2.5	2.5	0.25	0.01
Panamax	3.5	3.5	0.35	0.01
Capesize	6	6	0.6	0.01

8 Results

The results shown in this section are averages of 10 instances for every number of vessels. All results can be found in Appendix 10.4.

8.1 Seaside planning method

The methods are all limited to 5 minutes (300 seconds), and performed on the same input. For every number of vessels, 10 scenarios are created, and the averages are shown in Table 9. The time is presented in seconds. All results can be found in the Appendix.

Table 9: Seaside Results

# of	SWO + CPLEX		CPLEX		CH		LR		SWO	
vessels	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time
3	11.001	3.939		169.597	11.282	0.004	11.282	0.044	11.282	0.350
4	15.253	16.600		279.470	19.477	0.014	15.962	0.070	15.962	0.820
5	31.363^{2}	102.156		300.228	34.198	0.024	33.344	1.113	33.344	3.934
6	31.259^{2}	99.167			39.122	0.028	32.929	1.286	31.565	11.163
7	37.978^{2}	113.414			42.952	0.100	40.614	0.983	39.735	24.428
8	45.600^{4}	178.341			50.556	0.040	49.954	0.293	48.764	29.353
9	33.335^{4}	174.923			36.250	0.040	35.134	0.311	34.609	42.057
10	30.429^{3}	230.165			37.825	0.052	36.883	2.624	33.302	33.601
15	30.911^{10}	300.631			36.682	0.065	32.013	11.781	31.014	163.789
20					54.294	0.087	53.569	1.201	53.569	277.722
25					63.666	0.107	57.881	1.861	57.069	303.092

The number at the objective value of the SWO + CPLEX method denotes the number of solutions which are not proven to be optimal. No objective values exist for the CPLEX method, because for every number of vessels, at least one scenario existed where CPLEX was not able to find a solution. The computation time of SWO + CPLEX is without the computation time for the SWO solution. We can infer from the results that the LR procedure improves significantly on the construction heuristic. SWO shows small improvements on the LR for some cases while the computation time increases. SWO performs close the solution by SWO + CPLEX.
The computation time for the LR with 15 vessels is remarkably high. This is due to one problem instance (problem instance 8), where the RCS was not convertible to an feasible ECS. Hence the LR needed to be carried out with calculating the ECS every iteration. This causes a large increase in the computation time.
From the results we can conclude that the LR procedure is a good method when a short computation time is needed. When it is important to get the best solution possible, the SWO works better but also has a large increase in computation time compared to the LR. When computation time is not important, the SWO + CPLEX method works best for smaller problem instances. It is expected that an LR + CPLEX method will perform well, because the total computation time will be less than the SWO + CPLEX and the final objective value is expected to be similar.
In Table 10 the gap to optimality is shown, here the results are compared with the optimal solutions from the SWO + CPLEX method. All problem instances without an optimal solution are omitted in calculating these percentages.

Table 10: Average gap to optimality

	CH	LR	SWO
3^{10}	2.55%a	2.55%	2.55%
4^{10}	27.68%	4.64%	4.64%
5^{8}	7.65%	7.65%	7.65%
6^{8}	38.66%	5.07%	0.46%
7^{8}	19.44%	18.49%	12.32%
8^{6}	32.99%	32.99%	32.99%
9^{6}	22.26%	11.01%	5.71%
10^{7}	27.89%	27.89%	23.56%
Average	21.51%	12.39%	10.04%

From this table, we can infer that for some cases SWO performs close to optimal. In some scenarios the SWO performs bad, as we can see at the scenarios with 8 vessels. In this case this occurs because in one scenario the objective of the SWO was double the optimal objective value. The superscript at the number of vessels is the number of scenarios which are solved to optimality. The average is a weighted average, taking the number of scenarios into account

8.2 Complete methods

The results of the complete methods can be found in Table 11.

Table 11: Complete Results

	Lowerbound	Upperbound	Benchmark		Basic		Extended	
	Obj.	Obj.	Obj.	Time	Obj.	Time	Obj.	Time
3	11.282	395.660	79.311	1.270	42.791	0.142	11.649	0.897
4	15.960	848.447	172.460	2.779	149.444	0.280	42.659	1.768
5	33.135	1,276.967	235.200	2.350	589.021	0.491	68.111	7.034
6	31.564	1,388.420	262.763	4.682	881.950	0.939	69.050	5.561
7	39.735	2,494.607	367.723	4.294	471.315	1.796	84.415	11.508
8	48.764	3,562.227	597.950	4.437	2,496.675	3.449	123.243	58.140
9	34.609	3,944.780	526.146	5.908	1,032.307	2.362	53.316	7.465
10	33.302	4,604.167	463.777	7.358	283.323	3.250	47.807	7.871
15	31.014	9,976.100	419.406	12.852	2,483.180	9.686		
20	53.569	14,334.170	1,069.788	12.483	9,781.699	33.731	128.302	157.981
25	56.585	22,606.700	1,418.314	16.258	16,663.290	23.892	92.347	95.775

For one scenario with 15 vessels, the extended solution was not able to compute a solution, hence the average is not shown. The lower bound in this table is the value of the SWO. Because all three solution strategies are based on the CH or the SWO, an objective value lower than the SWO is not possible. Note that this is not a lower bound on the optimal value for the problem. The upperbound is calculated by creating a schedule with all vessels serviced sequentially with one crane equipped. The objective value is explained in Section 4, it is a sum of penalty values based on the handling time, berthing position and the number of changes in cranes assigned.
We can see that the basic method performs significantly worse compared to the benchmark model, especially for larger problem instances. We can infer that in the way the basic solution strategy is constructed in this research, that it shows a poor performance. The benchmark solution strategy performs well compared to the basic solution strategy, but is still not close to the lower bound. The extended solution strategy outperforms the benchmark and basic solution strategies. The objective values are relatively close to the lower bound.
in Table 12 we can see the gap to the lower bound of the methods. Here we can conclude again that the extended solution strategy performs better than the other two methods.

Table 12: Gap to lower bound

Vessels	Benchmark	Basic	Extended
3	584.00%	270.49%	3.15%
4	366.86%	312.91%	62.58%
5	296.67%	816.15%	51.35%
6	334.83%	1,231.54%	54.28%
7	388.54%	511.26%	52.93%
8	445.61%	1,986.25%	60.43%
9	921.94%	1,871.30%	35.08%
10	900.44%	522.97%	30.34%
15	879.34%	5,551.79%	29.03%
20	792.05%	7,582.21%	58.24%
25	1,474.58%	17,983.00%	38.72%
Mean	671.35%	3,512.71%	43.28%

For 15 vessels, one value is missing for the extended solution strategy, this scenario is also removed from the other two solution strategy to create this table.
In this table, we can see that basic and benchmark solution strategies are not close to the lower bound. The gap to the lower bound increases when the number of vessels increases for these two methods. For the extended solution strategy, the gap to the lower bound is relatively small. Furthermore, this gap does not increase when the number of vessels increases. We can conclude that the extended solution strategy performs well compared to the other two strategies.

9 Conclusion and future research

9.1 Conclusion

First, three methods to create seaside schedules were considered and adjusted. The first method was a simple construction heuristic. In order to improve on this simple construction heuristic, a local refinement procedure was presented. This local refinement procedure was used by a squeaky wheel optimization(SWO) to obtain objective values close to optimal. The local refinement procedure performed close to the objective values of the SWO, with a lower computation time. Without a base solution, CPLEX was not able to come up with a solution for the most scenarios. When CPLEX had access to the solution computed by the SWO, optimal solutions were found for most scenarios.
Second, a method to create stockyard schedules was presented. This method created stockyard schedules using the output of the seaside schedule method as input.
Third, three solution strategies were proposed to integrate the stockyard and the

seaside methods. The first solution strategy planned all vessels sequentially. The second method created multiple seaside schedules, and validated these schedules at the stockyard. The third method created a seaside schedule by using the SWO method, and this seaside schedule was validated at the stockyard. If it was not possible to create a feasible stockyard schedule, some feedback about the limiting vessels was returned. A new seaside schedule was created, and the procedure was repeated until the a feasible stockyard schedule was created.
The first solution strategy performed better than the second solution strategy. The second solution strategy had a large distance to the lower bound. The third solution strategy outperformed the first two solution strategies.

9.2 Future research

Because the used SWO optimization around 10 % from the optimal solution, a better way to create seaside schedules can improve the performance of the solution strategies. The solution strategies presented in this research are limited by the performance of the SWO.
To validate the presented methods and solution strategies better it is advised to test multiple terminal layouts in future research. Another suggestion to validate the presented solution strategies better is to create a tighter lower bound for the solution strategies. The current lower bound only uses information about the seaside. Using information about the stockyard can improve the lower bound. A way to improve the performance of the basic solution strategy is by developing a different way to create schedules. In addition, incorporating information from the stockyard while creating the seaside schedules can improve the basic method. The extended solution strategy can be improved by creating different ways of processing the feedback instead of only delaying vessels or decreasing the number of cranes.

References

M. Ago, T Nishi, and M Konishi. Simultaneous optimization of storage allocation and routing problems for belt-conveyor transportation. In *Journal of Advanced Mechanical Design, Systems, and Manufacturing*, volume 1, pages 250–261, 2006.

E. Angelelli, T. Kalinowski, R. Kapoor, and M.W.P. Savelsbergh. A reclaimer scheduling problem arising in coal stockyard management. *Journal of Scheduling*, 19(5):563–582, 2016.

V.H. Barros, T. S. Costa, A.C.M. Oliveira, and L.A.N. Lorena. Model and heuristic for berth allocation in tidal bulk ports with stock level constraints. *Computers & Industrial Engineering*, 60(4):606–613, 2011.

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

C. Bierwirth and F. Meisel. A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research*, 244(3):675–689, 2015.

J. Blazewicz, T.C.E. Cheng, M Machowiak, and C. Oguz. Berth and quay crane allocation: a moldable task scheduling model. *Journal of the Operational Research Society*, 62(7):1189–1197, 2011.

N. Boland, D. Gulczynski, and M. Savelsbergh. A stockyard planning problem. *EURO Journal on Transportation and Logistics*, 1(3):197–236, 2012.

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

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

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

X. Han, Z. Lu, and L. Xi. A proactive approach for simultaneous berth and quay crane scheduling problem with stochastic arrival and handling time. *European Journal of Operational Research*, 207(3):1327–1340, 2010.

D. Hu and Z. Yao. Stacker-reclaimer scheduling in a dry bulk terminal. *International Journal of Computer Integrated Manufacturing*, 25(11):1047–1058, 2012.

A. Imai, X. Sun, E. Nishimura, and S. Papadimitriou. Berth allocation in a container port: using a continuous location space approach. *Transportation Research Part B: Methodological*, 39(3):199–221, 2005.

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

B. Kim, J. Koo, and B. S. Park. A raw material storage yard allocation problem for a large-scale steelworks. *The International Journal of Advanced Manufacturing Technology*, 41(9):880–884, 2009.

M.J.A. Klaassen. *Operationele besturing en inrichting van een ijzererts exportterminal,*. PhD thesis, Report Number 2007.TL.7119., 2007.

N. Kovač. Metaheuristic approaches for the berth allocation problem. *Yugoslav Journal of Operations Research*, 27(2), 2017.

P. Legato, R. Trunfio, and F. Meisel. Modeling and solving rich quay crane scheduling problems. *Computers & Operations Research*, 39(9):2063–2078, 2012.

A. Lim. The berth planning problem. *Operations research letters*, 22(2):105–110, 1998.

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

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

G. Lodewijks. Two decades dynamics of belt conveyor systems. *Bulk Solids Handling*, 22(2):124–132, 2002.

G. Lodewijks and J.A. Ottjes. Reliability of large scale bulk material handling systems. *Bulk Solids & Powder Science & Technology*, 1:9–17, 2003.

G. Lodewijks, D.L. Schott, and J.A. Ottjes. Modern dry bulk terminal design. *Bulk Solids Handling*, 27(6):364, 2007.

G. Lodewijks, D.L. Schott, and J.A. Ottjes. Logistic control of modern dry bulk terminals. In *Beltcon conference*, volume 15, 2009.

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

G.C. Menezes, G.R. Mateus, and M.G. Ravetti. A hierarchical approach to solve a production planning and scheduling problem in bulk cargo terminal. *Computers & Industrial Engineering*, 97:1–14, 2016.

Y.M. Park and K.H. Kim. A scheduling method for berth and quay cranes. In *Container Terminals and Automated Transport Systems*, pages 159–181. Springer, 2003.

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

S. Pratap, A. Nayak, A. Kumar, N. Cheikhrouhou, and M.K. Tiwari. An integrated decision support system for berth and ship unloader allocation in bulk material handling port. *Computers & Industrial Engineering*, 106: 386–399, 2017.

T. Robenek, N. Umang, M. Bierlaire, and R. Stefan. A branch-and-price algorithm to solve the integrated berth allocation and yard assignment problem in bulk ports. *European Journal of Operational Research*, 235(2): 399–411, 2014.

L Tang, D Sun, and J. Liu. Integrated storage space allocation and ship scheduling problem in bulk cargo terminals. *IIE Transactions*, 48(5):428–439, 2016.

N. Umang, M. Bierlaire, and I. Vacca. The berth allocation problem in bulk ports. In *Swiss Transport Research Conference, STRC, Monte Verita/Ascona, Switzerland*. Citeseer, 2011.

N. Umang, M. Bierlaire, and I. Vacca. Exact and heuristic methods to solve the berth allocation problem in bulk ports. *Transportation Research Part E: Logistics and Transportation Review*, 54:14–31, 2013.

T.A. Van Vianen. *Simulation-integrated Design of Dry Bulk Terminals*. PhD thesis, TRAIL Research School, 2015.

T.A. van Vianen, J.A. Ottjes, R.R. Negenborn, G. Lodewijks, and D.L. Mooijman. Simulation-based operational control of a dry bulk terminal. In *Networking, Sensing and Control (ICNSC), 2012 9th IEEE International Conference on*, pages 73–78. Ieee, 2012.

M.J. Willekes. Dry bulk terminals in seaports. 1999.

P. Zhou and H. Kang. Study on berth and quay-crane allocation under stochastic environments in container terminal. *Systems Engineering-Theory & Practice*, 28(1):161–169, 2008.

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

10 Appendix

10.1 Abbreviations

In Table 13, all abbreviations used are shown.

Table 13: Abbreviations

Abbreviation	Meaning
BAP	Berth Allocation Problem
QSCP	Quay Crane Scheduling Problem
QC	Quay Crane
SR	Stacker/Reclaimer
ECS	Extended Crane Schedule
RCS	Reduced Crane Schedule
MinFT	Minimum Finishing Time
MaxFT	Maximum Finishing Time
AT	Arrival Time
CH	Construction Heuristic
LR	Local Refinement
SWO	Squeaky Wheel Optimization

10.2 Mathematical formulation for Hold schedule creation

The goal of this mathematical formulation is to find a feasible formulation for a hold schedule. Because the only goal is to find a feasible formulation, there is no objective formulated.

Symbol	Meaning	type
V	set of Vessels v	set
H	set of holds h	set
S	set of slots s	set
H_v	set of holds h belonging to vessel v	subset
H_v	set of slots s belonging to vessel v	subset
$p_{s,s'}$	1 if slot s is on the same vessel as s', and s is serviced by a crane with a higher number	parameter
$x_{h,s}$	1 if hold h is assigned to slot s	binary variable

$$\sum_{h \in H_v} x_{h,s} = 1 \qquad s \in S_v, v \in V \tag{45}$$

$$\sum_{s \in S_v} x_{h,s} = 1 \qquad h \in H_v, v \in V \tag{46}$$

$$p_{s,s'} hx_{h,s} \leq hx_{h,s'} \qquad h \in H_v, s, s' \in S_s, v \in V \tag{47}$$

$$x_{h,s} \in \mathbb{B} \tag{48}$$

$$\tag{49}$$

The constraints (45) and (46), assure every hold is assigned to a slot, and every slot has a hold. The third constraints (47) ensure that a crane with a higher number always serves a hold with a higher number.

10.3 Extended crane schedule creation

The function *CreateECS* creates a crane schedule for a vessel schedule. This function plans the vessels iteratively. For every vessel, each hold is planned sequentially. When a hold is planned, the function checks the location of the vessel, and validate which cranes can be assigned to the vessel. Vessels which have a number of cranes assigned, but no specific cranes are taken into account. If enough cranes are available, these cranes are assigned and the next vessel is planned.

10.4 Results

10.4.1 Seaside Results

In this section, all seaside results are stated. For some numbers of vessels, a CPLEX method is not carried out. This is omitted because CPLEX was not expected to find solutions anymore.

Table 15: Seaside results for 3 vessels

Instance	LPBound	SWO +CPLEX		CPLEX		CH		LR		SWO	
	Obj.	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time
1	0.090	0.090	3.031		305.109	0.090	0	0.090	0.030	0.090	0.110
2	7.080	7.080	2.906	7.080	51.688	7.080	0	7.080	0.050	7.080	0.140
3	0.070	0.070	1.469	0.070	67.891	0.070	0	0.070	0.030	0.070	0.110
4	2.580	2.580	2.359	2.580	54.031	2.580	0	2.580	0.030	2.580	0.120
5	13.423	13.423	3.859	73.767	300.484	13.423	0.010	13.423	0.040	13.423	0.380
6	13.423	13.423	4.891	1,325.403	300.204	13.423	0.010	13.423	0.050	13.423	0.500
7	26.413	26.413	6.500	26.413	161.032	26.413	0.010	26.413	0.050	26.413	0.920
8	7.423	7.423	3.250	7.423	43.485	9.413	0	9.413	0.030	9.413	0.220
9	8.090	8.090	3.672	8.090	111.922	8.913	0.010	8.913	0.050	8.913	0.340
10	31.413	31.413	7.454	79.510	300.125	31.413	0	31.413	0.080	31.413	0.660

Table 16: Seaside results for 4 vessels

Instance	LPBound	SWO +CPLEX		CPLEX		CH		LR		SWO	
	Obj.	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time
1	0.110	0.110	5.610		300.172	0.110	0.010	0.110	0.070	0.110	0.450
2	0.110	0.110	5	0.110	290	0.110	0	0.110	0.040	0.110	0.300
3	8.120	8.120	7.125	8.120	290.844	20.610	0.020	8.120	0.080	8.120	0.480
4	14.453	14.453	8.719		300.375	14.453	0.010	14.453	0.060	14.453	1.220
5	5.433	5.433	4.656	5.433	111.375	15.423	0.020	5.433	0.060	5.433	0.520
6	16.370	16.370	14.110		301.671	19.620	0.020	19.620	0.110	19.620	0.860
7	60.950	60.950	77.093		300.203	64.767	0.010	64.767	0.080	64.767	0.640
8	2.100	2.100	5.860	2.100	299.329	2.120	0.020	2.120	0.040	2.120	0.510
9	39.4421	39.443	27.031		300.250	39.443	0.020	39.443	0.080	39.443	2.640
10	5.443	5.443	10.797		300.485	18.110	0.010	5.443	0.080	5.443	0.580

Table 17: Seaside results for 5 vessels

Instance	LPBound	SWO +CPLEX		CPLEX		CH		LR		SWO	
	Obj.	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time
1	2.140	2.140	8.781		300.172	2.160	0.020	2.160	0.090	2.160	0.450
2	11.463	11.463	8.891		300.250	11.463	0.020	11.463	0.070	11.463	1.830
3	38.150	38.150	48.781		300.485	38.150	0.040	38.150	0.120	38.150	1.010
4	52.150	52.150	58.859		300.375	52.150	0.030	52.150	0.140	52.150	3.800
5	48.567	48.567	127.172		300.250	56.963	0.020	56.963	3.170	56.963	10.200
6	36.323	45.503	300.156		300.485	56.197	0.010	47.663	4.300	47.663	10.410
7	14.900	14.900	50.266		300.203	23.317	0.030	23.317	2.940	23.317	4.500
8	63.160	63.160	99.297		299.329	63.983	0.030	63.983	0.170	63.983	4.340
9	30.97651563	37.453	300.297		300.250	37.453	0.020	37.453	0.090	37.453	2.580
10	0.140	0.140	19.062		300.485	0.140	0.020	0.140	0.040	0.140	0.220

Table 18: Seaside results for 6 vessels

Instance	LPBound	SWO +CPLEX		CPLEX		CH		LR		SWO	
	Obj.	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time
1	2.150	2.150	13.782			2.170	0.020	2.160	0.090	2.160	3.110
2	27.231	33.763	300.125			42.613	0.020	42.613	0.150	36.103	37.840
3	19.189	124.170	300.203			134.160	0.030	124.170	11.540	124.170	34.230
4	25.493	25.493	60.125			25.493	0.030	25.493	0.120	25.493	1.570
5	5.160	5.160	26.687			24.593	0.030	5.160	0.160	5.160	5.030
6	8.973	8.973	33.812			13.650	0.020	13.640	0.100	9.073	0.820
7	11.960	11.960	33.328			11.960	0.020	11.960	0.140	11.960	1.540
8	9.503	9.503	57.797			12.680	0.050	12.680	0.140	10.113	1
9	55.483	55.483	126.640			55.483	0.030	55.483	0.170	55.483	1.990
10	35.930	35.930	39.172			68.420	0.030	35.930	0.250	35.930	24.500

Table 19: Seaside results for 7 vessels

Instance	LPBound	SWO +CPLEX		CPLEX		CH		LR		SWO	
	Obj.	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time
1	23.850	23.850	55.265			38.200	0.030	38.200	0.240	29.410	89.490
2	46.627	46.627	164.187			49.970	0.050	48.607	0.260	48.607	23.110
3	10.190	10.190	32.344			10.200	0.050	10.200	0.150	10.200	0.790
4	50.732	134.710	300.203			134.710	0.050	134.710	0.360	134.710	7.420
5	75.983	102.523	300.234			124.543	0.050	102.523	0.310	102.523	67.680
6	12.263	12.263	87.204			16.190	0.030	16.190	0.190	16.190	10.440
7	16.700	16.700	87.843			16.710	0.030	16.710	0.190	16.710	0.750
8	9.200	9.200	32.344			9.220	0.030	9.220	0.220	9.220	2.230
9	9.450	9.450	30.203			15.513	0.650	15.513	7.710	15.513	23.060
10	14.263	14.263	44.313			14.263	0.030	14.263	0.200	14.263	19.310

Table 20: Seaside results for 8 vessels

Instance	LPBound	SWO +CPLEX		CPLEX		CH		LR		SWO	
	Obj.	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time
1	5.400	138.647	300.312			140.347	0.050	140.347	0.330	139.247	129.530
2	9.970	9.970	38.625			10.320	0.030	10.320	0.230	10.320	8.920
3	24.720	24.720	108.062			53.020	0.050	53.020	0.340	53.020	42.280
4	28.095	51.587	300.266			51.687	0.050	51.687	0.310	51.687	49.720
5	35.744	77.247	300.313			91.407	0.030	88.397	0.330	77.597	27.550
6	41.681	66.137	300.250			66.587	0.050	66.587	0.390	66.587	3.610
7	3.710	3.710	19.735			3.710	0.030	3.710	0.190	3.710	0.860
8	23.720	23.720	65.219			23.720	0.030	23.720	0.280	23.720	5.450
9	24.720	24.720	50.359			24.720	0.030	24.720	0.280	24.720	10.750
10	17.120	35.543	300.266			40.043	0.050	37.033	0.250	37.033	14.860

Table 21: Seaside results for 9 vessels

Instance	LPBound	SWO +CPLEX		CPLEX		CH		LR		SWO	
	Obj.	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time
1	14.653	14.653	42.531			15.897	0.030	14.753	0.200	14.753	9.270
2	21.553	21.553	77.531			24.183	0.030	24.173	0.250	21.573	12.360
3	27.366	90.610	300.234			90.610	0.060	90.610	0.530	90.610	300.850
4	24.663	24.663	92.828			27.333	0.040	27.313	0.260	24.663	6.720
5	19.900	28.530	300.485			30.260	0.050	30.260	0.370	30.260	4.240
6	25.800	25.800	154.328			31.350	0.050	31.350	0.330	31.350	15.760
7	56.198	65.423	300.375			65.773	0.040	65.773	0.290	65.773	9.710
8	1.280	1.250	41.625			1.250	0.030	1.250	0.250	1.250	17.960
9	20.010	49.603	300.328			54.593	0.030	54.593	0.330	54.593	30.680
10	11.260	11.260	138.968			21.250	0.040	11.260	0.300	11.260	13.020

Table 22: Seaside results for 10 vessels

Instance	LPBound	SWO +CPLEX		CPLEX		CH		LR		SWO	
	Obj.	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time
1	20.150	20.150	81.593			22.760	0.030	22.760	0.250	20.180	6.620
2	13.600	13.600	172.250			18.050	0.050	18.050	0.390	16.050	50.270
3	17.09	17.090	119.687			25.540	0.050	25.540	0.430	25.540	34.640
4	12.675	20.613	300.343			20.613	0.050	20.613	0.310	20.613	12.060
5	21.353	21.353	227.218			24.853	0.050	24.853	0.430	24.853	42.990
6	4.940	78.613	300.468			78.613	0.070	78.613	0.350	78.613	10.430
7	26.195	42.953	300.500			72.297	0.070	72.297	0.430	43.053	83.140
8	12.918	28.290	294.985			38.780	0.050	38.780	0.340	38.780	14.050
9	11.780	56.347	300.344			71.467	0.050	62.047	23.090	60.057	80.800
10	5.280	5.280	204.266			5.280	0.050	5.280	0.220	5.280	1.010

Table 23: Seaside results for 15 vessels

Instance	LPBound	SWO +CPLEX		CPLEX		CH		LR		SWO	
	Obj.	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time
1	4.869	4.920	300.343			4.920	0.050	4.920	0.550	4.920	2.980
2	8.641	8.670	300.375			18.660	0.060	18.660	0.640	8.670	301.230
3	-9.397	27.680	300.531			28.030	0.080	28.030	0.890	28.030	300.990
4	-28.084	29.180	300.766			30.700	0.080	29.280	0.570	29.280	122.200
5	10.610	10.610	300.829			11.193	0.080	11.193	0.550	11.193	31.830
6	1.813	44.150	300.766			44.150	0.060	44.150	0.720	44.150	107.920
7	-63.436	20.600	300.687			41.780	0.060	20.600	0.880	20.600	302.580
8	-30.278	133.130	300.500			151.180	0.080	133.110	111.830	133.110	412.840
9	4.322	4.410	300.625			10.440	0.060	4.420	0.600	4.420	33.520
10	-28.876	25.763	300.891			25.763	0.040	25.763	0.580	25.763	21.800

Table 24: Seaside results for 20 vessels

Instance	LPBound	SWO +CPLEX		CPLEX		CH		LR		SWO	
	Obj.	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time
1					27.370	0.030	27.370	0.660	27.370	301.310	
2					22.623	0.030	22.623	0.580	22.623	301.610	
3					113.783	0.040	111.450	0.690	109.560	301.750	
4					40.177	0.050	40.177	0.540	40.177	301.140	
5					157.910	0.050	157.910	0.890	157.910	302.280	
6					13.500	0.030	13.500	0.440	13.500	24.820	
7					38.270	0.030	37.113	0.670	37.113	301.810	
8					63.700	0.030	62.083	0.460	62.083	300.840	
9					66.570	0.040	66.560	0.690	66.560	301.860	
10					123.283	0.060	121.150	0.930	121.150	302.070	

Table 25: Seaside results for 25 vessels

Instance	LPBound	SWO +CPLEX		CPLEX		CH		LR		SWO	
	Obj.	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time
1					55.883	0.040	55.883	0.860	55.883	301.380	
2					64.113	0.050	57.380	1.140	57.380	302.060	
3					89.373	0.060	74.383	1.220	74.383	301.390	
4					144.857	0.060	144.857	1.210	144.857	303.490	
5					112.513	0.040	99.837	1.220	97.990	303.470	
6					49.390	0.040	49.390	0.850	49.390	303.390	
7					44.910	0.040	44.910	0.960	44.910	302.750	
8					125.437	0.070	124.753	0.780	124.587	303.140	
9					66.820	0.040	45.983	1.050	45.983	301.570	
10					36.590	0.050	34.663	0.580	29.340	67.370	

10.4.2 Complete Results

In this section, all complete results are stated.

Table 26: Total Results for 3 vessels

Instance	Lower Bound	Upper Bound	Benchmark		Basic		Extended	
	Obj.	Obj.	Obj.	Time	Obj.	Time	Obj.	Time
1	0.090	502.060	118.557	5.220	38.590	0.100	0.090	0.270
2	7.080	318.060	67.897	0.160	7.080	0.070	7.080	0.060
3	0.070	147.060	11.453	0.280	0.070	0.040	0.070	0.130
4	2.580	227.060	55.097	0.150	37.533	0.160	2.580	0.250
5	13.423	651.727	92.507	0.250	61.757	0.140	13.423	0.490
6	13.423	318.060	94.797	0.210	75.747	0.330	13.423	0.480
7	26.413	611.727	201.387	5.890	67.747	0.190	30.080	0.820
8	9.413	179.060	45.887	0.190	9.413	0.090	9.413	0.060
9	8.913	390.060	20.963	0.140	67.747	0.160	8.913	6.060
10	31.413	611.727	84.563	0.210	62.223	0.140	31.413	0.350

Table 27: Total Results for 4 vessels

Instance	Lower Bound	Upper Bound	Benchmark		Basic		Extended	
	Obj.	Obj.	Obj.	Time	Obj.	Time	Obj.	Time
1	0.110	953.080	108.493	1.430	24.110	0.410	0.110	1.390
2	0.110	586.413	31.910	0.470	10.610	0.080	10.610	0.700
3	8.120	1,120.747	53.010	0.450	147.423	0.220	8.120	0.430
4	14.453	896.080	114.660	0.950	62.110	0.330	14.453	0.630
5	5.433	380.080	39.667	0.300	5.433	0.080	5.433	0.080
6	19.620	1,661.747	451.567	7.640	67.553	0.180	19.620	0.530
7	64.767	705.080	164.917	0.510	170.757	0.670	64.767	1.140
8	2.100	618.080	232.807	7.860	11.600	0.170	12.600	1.710
9	39.443	667.080	181.400	0.970	98.767	0.310	89.443	2.330
10	5.443	896.080	346.170	7.210	896.080	0.350	201.433	8.740

Table 28: Total Results for 5 vessels

Instance	Lower Bound	Upper Bound	Benchmark		Basic		Extended	
	Obj.	Obj.	Obj.	Time	Obj.	Time	Obj.	Time
1	2.140	1,635.433	171.940	1.070	2.140	0.230	2.160	0.210
2	11.463	594.767	130.003	0.410	11.463	0.160	11.463	0.230
3	38.150	1,951.433	159.617	1.840	86.300	0.300	38.150	1.140
4	52.150	2,035.433	152.293	1.850	2,035.433	0.400	167.040	8.270
5	54.890	658.767	275.753	0.670	56.963	0.680	56.963	0.200
6	47.663	1,219.433	136.770	0.550	251.120	1.060	77.217	5.030
7	23.317	956.767	339.113	2.310	23.317	0.250	23.317	0.150
8	63.983	2,494.100	688.430	8.700	2,494.100	0.860	267.207	53.060
9	37.453	363.767	106.643	0.420	69.610	0.720	37.453	1.050
10	0.140	859.767	191.440	5.680	859.767	0.250	0.140	1

Table 29: Total Results for 6 vessels

Instance	Lower Bound	Upper Bound	Benchmark		Basic		Extended	
	Obj.	Obj.	Obj.	Time	Obj.	Time	Obj.	Time
1	2.150	1,017.787	125.533	0.680	18.650	0.220	2.160	0.470
2	36.103	1,074.120	151.427	0.280	1,074.120	1.670	36.103	1.130
3	124.170	1,980.453	356.510	0.790	1,980.453	3.470	124.170	1.060
4	25.493	1,321.787	249.087	2.310	1,321.787	1.050	25.493	1.950
5	5.160	1,140.787	239.767	6.690	1,140.787	0.730	94.160	10.670
6	9.073	368.120	176.130	7.390	20.630	0.230	40.963	3.750
7	11.960	1,158.787	45.717	0.570	126.140	0.250	11.960	1.140
8	10.113	1,690.120	551.537	5.270	21.003	0.410	26.680	2.740
9	55.483	1,277.787	343.183	4.970	261.473	0.340	162.170	15.720
10	35.930	2,854.453	388.740	17.870	2,854.453	1.020	166.643	16.980

Table 30: Total Results for 7 vessels

Instance	Lower Bound	Upper Bound	Benchmark		Basic		Extended	
	Obj.	Obj.	Obj.	Time	Obj.	Time	Obj.	Time
1	29.410	2,864.807	401.380	2.390	2,864.807	2.320	154.023	16.860
2	48.607	2,499.140	232.907	0.740	200.867	3.150	52.107	1.570
3	10.200	2,079.140	278.213	4.660	10.200	0.280	10.200	0.290
4	134.710	3,990.473	516.463	3.530	166.033	1.860	134.710	10.760
5	102.523	4,286.140	649.860	10.760	916.150	0.530	335.337	59.280
6	16.190	1,155.473	101.897	0.550	16.190	0.540	16.190	0.270
7	16.710	2,741.140	630.757	7.590	201.783	2.450	91.960	14.790
8	9.220	2,713.140	93.133	0.610	186.190	0.640	19.850	7.360
9	15.513	1,583.140	376.387	2.860	74.180	2.770	15.513	3
10	14.263	1,033.473	396.230	9.250	76.753	3.420	14.263	0.900

Table 31: Total Results for 8 vessels

Instance	Lower Bound	Upper Bound	Benchmark		Basic		Extended	
	Obj.	Obj.	Obj.	Time	Obj.	Time	Obj.	Time
1	139.247	3,623.493	566.770	1.080	3,623.493	4.500	214.587	33.550
2	10.320	3,016.493	604.570	6.370	3,016.493	2.360	16.653	34.560
3	53.020	4,287.160	663.487	3.160	4,287.160	4.190	376.187	165.790
4	51.687	4,269.827	431.993	5.040	747.513	5.470	143.677	16.880
5	77.597	4,941.827	1,001.337	7.360	4,941.827	3.700	309.377	53.450
6	66.587	3,338.493	1,023.403	7.570	368.553	3.700	66.587	2.540
7	3.710	2,816.160	186.233	4.280	244.523	0.960	3.710	0.790
8	23.720	4,214.827	486.327	2.890	4,214.827	3.450	23.720	118.060
9	24.720	3,278.160	469.477	5.640	3,278.160	4.250	24.720	0.560
10	37.033	1,835.827	545.903	0.980	244.200	1.910	53.210	155.220

Table 32: Total Results for 9 vessels

Instance	Lower Bound	Upper Bound	Benchmark		Basic		Extended	
	Obj.	Obj.	Obj.	Time	Obj.	Time	Obj.	Time
1	14.753	1,804.847	66.507	0.750	1,804.847	0.840	14.753	4.420
2	21.573	2,184.847	179.820	0.720	159.710	1.660	26.163	7.610
3	90.610	7,486.847	1,055.627	3.660	442.937	7.780	90.610	2.220
4	24.663	2,267.847	187.347	1.360	33.323	1.710	24.663	8.730
5	30.260	4,907.513	636.290	9.640	47.760	1.750	47.760	8.420
6	31.350	3,903.847	986.773	7.200	289.427	4.550	62.350	8.670
7	65.773	2,829.847	1,275.737	21.190	537.887	1.510	191.253	24.920
8	1.250	3,958.513	373.557	3.080	9.750	0.350	9.750	5.160
9	54.593	6,958.847	166.543	1.290	6,958.847	0.670	54.593	2.920
10	11.260	3,144.847	333.257	10.190	38.583	2.800	11.260	1.580

Table 33: Total Results for 10 vessels

Instance	Lower Bound	Upper Bound	Benchmark		Basic		Extended	
	Obj.	Obj.	Obj.	Time	Obj.	Time	Obj.	Time
1	20.180	2,066.533	265.077	1.440	20.180	0.990	20.180	0.350
2	16.050	6,156.867	437.053	35.450	384.270	2.250	48.690	13.700
3	25.540	5,760.867	205.323	3.710	580.937	4.230	25.540	1.910
4	20.613	4,088.533	1,092.137	3.700	106.623	6.500	20.613	4.990
5	24.853	5,242.867	269.863	0.900	110.550	4.780	24.853	0.640
6	78.613	5,588.867	856.863	7.350	82.113	1.240	78.613	2.110
7	43.053	4,013.867	722.517	2.640	713.907	8	132.477	14.730
8	38.780	5,056.867	371.580	1.220	38.780	1.260	38.780	1.190
9	60.057	4,213.867	305.080	16.620	749.917	2.390	63.047	31.450
10	5.280	3,852.533	112.280	0.550	45.957	0.860	25.280	7.640

Table 34: Total Results for 15 vessels

Instance	Lower Bound	Upper Bound	Benchmark		Basic		Extended	
	Obj.	Obj.	Obj.	Time	Obj.	Time	Obj.	Time
1	4.920	11,499.300	436.417	13.550	819.057	3.110	4.920	1.860
2	8.670	7,472.967	580.260	8.400	7,472.967	15.930	8.670	4.060
3	28.030	13,036.300	908.050	17.390	13,036.300	13.070		
4	29.280	11,232.970	183.147	1.190	491.253	9.150	29.280	4.480
5	11.193	3,427.967	185.993	1.830	27.093	3.070	11.193	5.290
6	44.150	9,884.300	267.240	5.440	91.067	4.980	44.150	4.890
7	20.600	9,994.300	249.827	35.170	352.880	10.250	50.590	3.630
8	133.110	13,671.970	933.643	21.450	2,353.703	30.610	198.543	16.360
9	4.420	7,614.633	239.290	22.380	117.380	0.660	24.410	16.150
10	25.763	11,926.300	210.197	1.720	70.097	6.030	25.763	4.050

Table 35: Total Results for 20 vessels

Instance	Lower Bound	Upper Bound	Benchmark		Basic		Extended	
	Obj.	Obj.	Obj.	Time	Obj.	Time	Obj.	Time
1	13.620	10,501.400	1,067.870	10.040	18.943	4.650	21.630	41.240
2	9.373	18,773.730	656.057	24.310	18,773.730	9.270	81.997	51.500
3	94.200	18,394.070	1,244.970	22.270	1,539.853	12.900	94.200	7.700
4	30.677	7,907.733	1,343.117	3.090	30.677	46.150	30.677	0.530
5	147.660	22,026.400	2,862.847	18.910	22,026.400	47.970	674.223	706.140
6	4	7,046.067	278.423	3.110	7,046.067	29.890	4	6.290
7	26.113	10,411.070	1,099.590	23.870	10,411.070	36.330	37.813	78.250
8	51.833	10,440.400	514.787	2	129.453	46.770	43.577	312.460
9	51.310	18,878.070	857.573	12.850	18,878.070	54.330	62.510	68.780
10	106.900	18,962.730	772.650	4.380	18,962.730	49.050	232.393	306.920

Table 36: Total Results for 25 vessels

Instance	Lower Bound	Upper Bound	Benchmark		Basic		Extended	
	Obj.	Obj.	Obj.	Time	Obj.	Time	Obj.	Time
1	39.633	27,265.500	1,783.947	20.540	27,265.500	46.860	49.190	113.100
2	39.380	20,025.500	1,698.503	22.050	20,025.500	12.720	47.510	195.950
3	54.883	30,709.500	4,327.867	45.820	4,467.317	35.510	173.863	160.740
4	126.877	29,263.830	1,899.290	18.230	29,263.830	12.980	156.867	104.330
5	83.087	28,005.830	2,049.957	19	28,005.830	13.540	135.067	7.650
6	35.140	20,664.830	557.107	3.780	20,664.830	20.070	69.130	61.720
7	25.270	19,095.830	330.470	2.990	19,095.830	15.920	76.227	84.460
8	106.660	20,686.170	596.123	22.450	1,532.473	61.830	111.503	13.470
9	31.003	16,287.830	582.823	2.720	16,287.830	14.020	80.197	215.230
10	23.913	14,062.170	357.053	5	23.913	5.470	23.913	1.100