R. Ummels

Scheduling Multi-Vessel Placement in Multi-Chamber Inland Waterway Locks Using Switching Max-Plus Algebra

TU Delft

Scheduling Multi-Vessel Placement in Multi-Chamber Inland Waterway Locks Using Switching Max-Plus Algebra

By

R. Ummels

Master Thesis

In partial fulfillment of the requirements for the degrees of

Master of Science
In Mechanical Engineering
Multi-Machine Engineering Track

at the Department Maritime and Transport Technology of Faculty Mechnical, Maritime, and Materials Engineering of Delft University of Technology

Master of Science
In Systems & Control

at the Delft Center for Systems & Control of Delft University of Technology

Student Number: 4669304
Report Number: 2023.MME.8825

Supervisors: Dr. Ir. T. J. J. van den Boom
Dr. V. Reppa
Dr. P. Segovia
Date: July 31st, 2023

An electronic version of this thesis is available at http://repository.tudelft.nl/.

The cover image is from [1].

Contents

Preface 8

Summary 9

1 Introduction 10
1.1 Background . . . 10
1.1.1 IWT Networks . . . 10
1.1.2 Vessels . . . 12
1.1.3 Locks . . . 13
1.1.4 Data and Communication Systems . . . 15
1.1.5 Scheduling with SMPL Systems . . . 15
1.1.6 Problem . . . 16
1.2 Research Scope . . . 16
1.3 Research Questions . . . 17
1.4 Approach . . . 17
1.5 Individual Double Degree . . . 18
1.6 Document Outline . . . 19

2 Literature Review 20
2.1 Lock scheduling and the SPP . . . 20
2.1.1 The lock Scheduling Problem . . . 20
2.1.2 Ship Placement Problem . . . 26
2.1.3 Conclusion . . . 29
2.2 IWT and Max-Plus Algebra . . . 30
2.2.1 The Basics . . . 31
2.2.2 Max-Plus-Linear Systems . . . 32
2.2.3 Switching Max-Plus-Linear Systems . . . 34
2.2.4 IWT Scheduling with SMPL Systems . . . 37
2.2.5 Scheduling MILP . . . 42
2.2.6 Conclusion . . . 45

3 Single-vessel, Single-chamber Scheduling for Arbitrary Network Topologies 47
3.1 Problem Definition . . . 47
3.1.1 Sets . . . 48
3.1.2 Graph . . . 48
3.1.3 Variables . . . 48
3.1.4 Parameters . . . 49
3.1.5 Objective . . . 50
3.1.6 Assumptions . . . 50
3.2 Approach . . . 52
3.2.1 Establishing the Mathematical Model . . . 53
3.2.2 Creating and Solving Scheduling Optimization Problems . . . 53
3.2.3 Novel Contributions . . . 54
3.3 SMPL Constraints . . . 55
3.3.1 Max-Plus Logical Constraints . . . 55

3.3.2 Routing Constraints . . . 56
3.3.3 Ordering Constraints . . . 59
3.4 SMPL System . . . 61
3.4.1 System Equation . . . 61
3.4.2 Allowable Modes . . . 62
3.5 MILP Problem Conversion . . . 63
3.5.1 General MILP Problem Structure . . . 63
3.5.2 Constraints . . . 63
3.5.3 Objectives . . . 65
3.5.4 Performance Indicators . . . 68
3.6 Online Optimization . . . 70
3.7 Conclusion . . . 71

4 Multi-Vessel, Multi-chamber Scheduling with Chamber Capacities 73
4.1 Problem definition . . . 73
4.1.1 Graph . . . 74
4.1.2 Variables . . . 74
4.1.3 Parameters . . . 75
4.1.4 Assumptions . . . 76
4.2 SMPL Constraints . . . 77
4.2.1 Synchronisation Constraints . . . 77
4.2.2 Ordering Constraints . . . 78
4.3 SMPL System . . . 78
4.3.1 System Equation . . . 79
4.3.2 Allowable Modes . . . 79
4.4 MILP Problem Conversion . . . 80
4.4.1 Constraints . . . 80
4.4.2 Number of Lockages Performance Indicator . . . 81
4.5 Conclusion . . . 82

5 Multi-Vessel, Multi-Chamber Scheduling with Ship Placement 83
5.1 Problem Definition . . . 83
5.1.1 Sliding Ship Placement . . . 83
5.1.2 Ship Placement Variables and Parameters . . . 85
5.1.3 Mooring, Safety Distance and Vessels with Dangerous Substances . . . 87
5.1.4 Assumptions . . . 88
5.1.5 Schedule . . . 89
5.2 SMPL Constraints . . . 89
5.2.1 Toxic and Explosive Vessel Synchronisation Constraint . . . 90
5.2.2 Placement Constraints . . . 90
5.2.3 Mooring Constraints . . . 92
5.2.4 Non-SMPL Constraints . . . 95
5.3 SMPL Systems . . . 96
5.3.1 System Equations . . . 96
5.3.2 Allowable Modes . . . 98
5.4 MILP Conversion . . . 98
5.4.1 New Constraints . . . 98
5.4.2 Toxic and Explosive Vessel Synchronisation Constraint . . . 99
5.4.3 Placement Constraints . . . 99
5.4.4 Mooring Constraints . . . 100
5.4.5 Objectives . . . 103
5.5 Practical Considerations . . . 103

5.5.1 Larger-than-inequality Constraints . . . 103
5.5.2 Complexity and Post-Processing . . . 104
5.6 Conclusion . . . 105

6 Verification **106**
6.1 Approach . . . 106
6.1.1 Testing Approach . . . 106
6.1.2 Verification Cases . . . 107
6.2 Verification Case A - Single-Vessel, Single-Chamber Model . . . 108
6.2.1 Tests 1-4 - Ordering and Route Creation . . . 108
6.2.2 Test 5 - Waiting Time Objective . . . 112
6.2.3 Test 6 - Departure Time and Arrival Time Offset . . . 113
6.2.4 Test 7 - Multi-Vessel Stress Test . . . 114
6.3 Verification Case B - Single-Vessel, Single-Chamber Model . . . 115
6.3.1 Test 1 - Route Choice . . . 116
6.3.2 Test 2 - Route Choice . . . 117
6.4 Verification Case C - Multi-Vessel, Multi-Chamber Capacity Model . . . 118
6.4.1 Tests 1-2 - Synchronisation . . . 118
6.5 Verification Case D - Multi-Vessel, Multi-Chamber Capacity Model . . . 120
6.5.1 Test 1 - Chamber Distribution and Capacity Limits . . . 121
6.5.2 Test 2 - Multi-Vessel Stress Test . . . 122
6.6 Verification Case E - Multi-Chamber, Multi-Vessel Placement Model . . . 123
6.6.1 Test 1 - Placement and Mooring . . . 123
6.6.2 Test 2 - Advanced Placement . . . 125
6.7 Verification Case F - Multi-Chamber, Multi-Vessel Placement Model . . . 127
6.7.1 Test 1 - Multiple Locks and Flammable and Explosive Goods . . . 127
6.7.2 Test 2 - Multiple Travel Directions and Intermediate Destinations . . . 131
6.8 Complexity . . . 134
6.8.1 Branch-and-Bound . . . 134
6.8.2 Variables . . . 135
6.8.3 Experimental Comparisons . . . 136
6.9 Comparison to Practice . . . 139
6.9.1 Simulation Description . . . 139
6.9.2 Simulation Model . . . 140
6.9.3 Placement Heuristic . . . 143
6.9.4 Comparisons . . . 144
6.10 Conclusion . . . 148

7 Conclusions and Recommendations **150**
7.1 Conclusion . . . 150
7.2 Recommendations . . . 151
7.2.1 Online Optimization . . . 151
7.2.2 Distributed Optimization . . . 152
7.2.3 Validation . . . 152
7.2.4 Miscellaneous Recommendations . . . 152

List of Symbols **154**

List of Abbreviations **158**

Bibliography **159**

A Research Paper **164**

Preface

This is the report documenting the process and results of my final Thesis project in fulfillment of my master programs Multi-Machine Engineering and Systems & Control.

Those who know me know I like Formula 1 cars, aircraft, and spacecraft. Up until very recently, I knew very little about ships or boats. After my first year of Systems & Control, I chose to also pursue Multi-Machine Engineering, combining the two studies in an Individual Dual Degree, to get a broader engineering education. An Individual Dual Degree requires a shared masters thesis on a topic that has significant overlap with both studies.

When I spoke with Dr. Reppa, it became clear that there was an inland waterway transport scheduling topic that a Systems & Control student had done a report on before me, that I could continue to work on, that had significant overlap with both studies, and that Dr. van den Boom at Systems & Control was also very enthusiastic to continue. So, within the span of a week, I had found and accepted a topic that would fit my Individual Dual Degree, something I had expected would take much longer.

The first goal of a masters thesis is to deliver a scientific contribution. This contribution is explicitly discussed and laid out in the content of this report. In addition to that, I had set myself a secondary goal of delivering models that are practically oriented and scalable. I continually asked myself the question: 'What should these models look like if Rijkswaterstaat or a comparable institution wants to implement this on a large scale in the future?' I recommend the reader keep that question in the back of their mind while reading this report.

First of all, I want to thank my three supervisors Dr. Reppa, Dr. van den Boom, and Dr. Segovia for their generous time, close support, copious enthusiasm, and poignant feedback. I sincerely appreciate them all for being accommodating during the slightly unusual thesis process.

I want to thank my friends, those I still see and those I do not, for bringing color to my time at university and being a part of the person I am today.

Last but not least, I want to thank my family, my mother and father in particular, for always going above and beyond in their support of whatever I set my mind on.

Summary

Inland waterway transport is a low CO_2 emission alternative to road transport. A shift towards more inland waterway transport could also help reduce road congestion and noise pollution. Infrastructure bottlenecking, particularly at locks, is part of the reasons preventing this shift. Congestion is leading to delays.

Locks can be physically improved, or the passage of the vessels through the locks can be optimized through scheduling. Recent work introduced a novel switching-max-plus-linear system approach to scheduling vessels passing through networks of waterways and locks, also introducing a novel routing component to the scheduling problem. Switching max-plus-linear systems are a convenient way to model scheduling systems using max-plus-linear algebra.

The switching-max-plus-linear model only considered locks with a single chamber that can only process one vessel at a time. Additionally, it only considered four specific waterway network configurations, rather than any arbitrary network configuration. Real locks can have multiple chambers, and they can process multiple vessels at the same time if they are placed according to regulations in the two-dimensional space of the chamber.

The scope of this report was then to build upon this switching max-plus-linear model by adding support for arbitrary network configurations, and multi-chamber, multi-vessel locks with proper two-dimensional ship placement, to answer the main research question: *How can multi-vessel, multi-chamber locks with ship placement be integrated into the SMPL IWT scheduling model?*

Three mathematical scheduling models formulated as SMPL systems were introduced, each subsequent model building on the previous one. The first introduced support for arbitrary network configurations. The second introduced support for multi-chamber locks and allowed vessels to pass through lock chambers at the same time, provided that their assigned one-dimensional sizes fit into the assigned one-dimensional capacity of the chamber. The third introduced support for two-dimensional ship placement through a Tetris-like placement sequence also modeled as a switching max-plus-linear system.

The models were translated to mixed integer linear programming models, and arrival time and arrival time offset objectives were added, so they could be used as the rules by which a scheduler would build and solve offline scheduling optimization models on a case-by-case basis. Auxiliary objectives to promote vessels slowing down, rather than waiting stationary in the waiting areas, were also added. The models were all found to be working as intended and implemented correctly through the use of a number of verification cases and tests.

Complexity tests showed that the solution times for all three models already became larger than a practical limit of 15-20 minutes for simple scenarios with 10-15 vessels. A heuristic model that mimics how vessels are assigned to lockages in practice was built. Comparisons to the scheduling models on the verification cases showed negligible differences for the models in single-lock cases, but it indicated that the multi-vessel, multi-chamber models may outperform practice in multi-lock cases.

Future research is recommended to focus on online optimization to account for disturbances, distributed optimization to reduce calculation times, and validation of the model's performance with real data.

1

Introduction

This chapter introduces the inland waterway transport (IWT) and switching max-plus-linear (SMPL) algebra subject matter of this thesis, the problem, the scope, the research questions, and the approach that will be taken.

Background on IWT, SMPL systems, and the problem of lock congestion is given in Section 1.1. The research scope is outlined in Section 1.2, with the corresponding research questions listed in Section 1.3. The approach to answering the research questions is described in Section 1.4. As this is a thesis for an individual double degree, clarification on the division of the subject matter between the two studies involved is given in Section 1.5. Finally, the structure of the rest of this report is outlined in Section 1.6.

1.1 Background

This section provides background on IWT and the use of SMPL systems, and it introduces the problem this thesis works towards solving.

Subsection 1.1.1 discusses the Dutch and European IWT networks. Background information on waterway locks is given in Subsection 1.1.3, followed by information on the available data and communication systems in IWT in Subsection 1.1.4. IWT vessels are discussed in Subsection 1.1.2. Scheduling with SMPL systems is introduced in Subsection 1.1.5. Finally, the problem to be solved is defined in Subsection 1.1.6.

1.1.1 IWT Networks

Figure 1.1 gives an overview of the Dutch main IWT network. The full network totals 4400 km in length [2]. Minor locks are indicated with green circles, and major locks are indicated with triangles and squares. The colors designate the maximum CEMT class the waterways can accommodate, starting at CEMT class II. Waterways that have smaller size limits are colored light blue.

The port of Rotterdam, the biggest port in Europe, lies at the heart of this network. 50% of the goods transported to and from the port of Rotterdam arrive or depart on an inland waterway vessel [3]. The distribution of transported goods by weight over the different modes of transport in the Netherlands is shown in Figure 1.2. The modal share of IWT by tonne-kilometers was about 12.5% in 2021 [4].

The Dutch waterway network is part of the European inland waterway network, which is, in turn, part of the larger Trans-European Transport Network (TEN-T), creating an international IWT network spanning from the Netherlands to the Black Sea, as depicted in Figure 1.3.

The waterways are classified by the largest CEMT vessel class they can accommodate. A simplified overview of vessel and convoy sizes per CEMT class for Western Europe is given in Table 1.1. A convoy is the combination of a push boat and several standardized pushed barges in different possible configurations. The designs of new waterways and waterway infrastructure, like locks and

Figure 1.1: *Main Inland Waterways of the Netherlands. Image from: [2].*

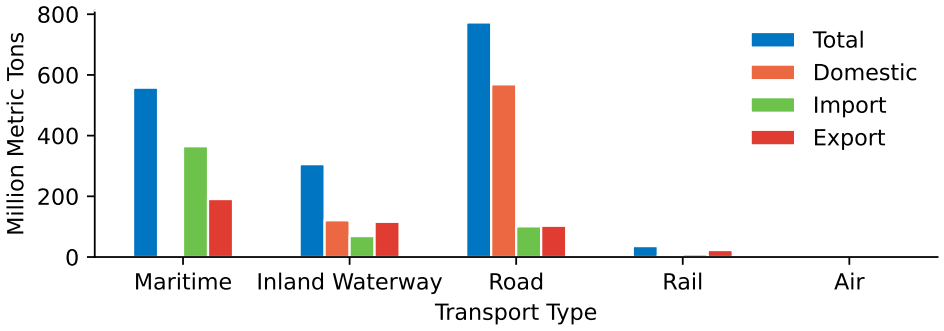


Figure 1.2: *Transport flows by transport types in the Netherlands in 2020. Data from: [5].*

bridges, are based on these standard CEMT classes, which are used as reference vessels to set infrastructure requirements [7].

The total modal share of IWT by the percentage of tonne-kilometers of freight transport in the European Union (EU) is only 1.8%. Excluding maritime transport, this is bumped up to 6.1%, as opposed to 76.3% for road transport and 17.6% for rail transport [8]. IWT is a great alternative to road transport, reducing emissions, relieving congestion, and producing less noise pollution. However, this comes with several physical and digital infrastructure challenges, that are yet to be solved to achieve the growth of the EU IWT sector that has been envisioned for over 20 years

Figure 1.3: *European TEN-T Inland Waterway Network by CEMT Class. Image from: [6]. The legend has been enlarged.*

Table 1.1: *Western European CEMT Classes. Data from: [7].*

CEMT Class	Motor Vessels and Barges	Pushed Convoy	
	Length [m] / Beam[m] / Draught [m]	Length [m] / Beam [m] / Draught [m]	Minimum Height Under Bridges [m]
I	38.5 / 5.05 / 1.8-2.2	-	4.0
II	50-55 / 6.60 / 2.5	-	4.0
III	67-80 / 8.2 / 2.5	-	5.0
IV	80-85 / 9.5 / 2.5	8.5 / 9.5 / 2.5-2.8	5.25-7
Va	95-110 /11.4 /2.5-2.8	95-110 / 11.4 / 2.5-4.5	5.25-9.1
Vb		172-185 / 11.4 / 2.5-4.5	7.1-9.1
VIa		95-110 / 22.8 / 2.5-4.5	7.1-9.1
VIb		185-195 / 22.8 / 2.5-4.5	7.1-9.1
VIc		193-280 / 22.8-34.2 / 2.5-4.5	9.1
VII		195-285 / 33.0-34.2 / 2.4-4.5	9.1

[9]. Significant parts of waterways and waterway infrastructure are underdeveloped and unreliable, forming bottlenecks [10], droughts and climate change reduce navigability, and (international) digital data, communication, and planning are underutilized [11].

1.1.2 Vessels

The main dimensions and terminology that are used to refer to vessel geometry and vessel features in this report are shown in Figure 1.4. Inland waterway vessels are generally long and slender, as also evident from the vessel classification dimensions in Table 1.1.

High-displacement vessels, which most commercial inland waterway vessels can be categorized as have a maximum achievable velocity that is not only dependent on the vessel's shape and draft but also on the waterway itself. Approaching a certain speed range, called the critical range, the

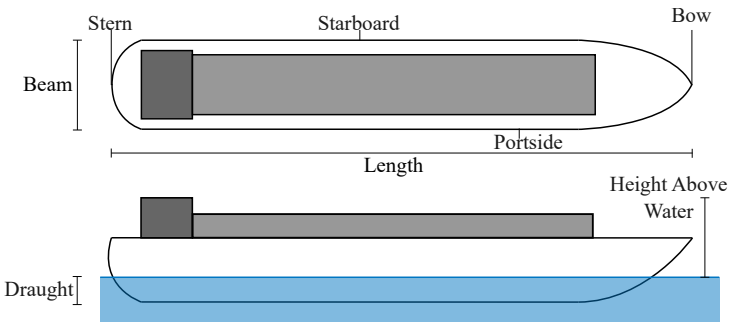


Figure 1.4: *Vessel geometry and features.*

resistance on the vessel rises sharply due to wave effects [12]. The velocity at which this range starts is partly dependent on parameters of the waterway itself, like waterway width and depth [13].

Transporting cargo by inland waterway vessels is a low-emission alternative to road transport. IWT vessels are estimated to only emit 33 grams of CO_2 per tonne kilometer of transport in the EU, compared to 137 grams for heavy goods road vehicles [14]. Fuel consumption is proportional to velocity for IWT vessels [15], as a result, further emission reductions can be achieved by instructing vessels to slow down when approaching a lock if they would have to wait in the waiting area of the lock anyway. Reducing velocity to reduce emissions and conserve fuel is called slow-steaming in the maritime industry [16], and the practice will also be referred to as slow-steaming for IWT vessels in this report.

A shift towards more IWT would, thus, be beneficial from an emission standpoint. Furthermore, it could also reduce noise pollution and road congestion. However, this shift is being held back by the aforementioned bottlenecks at waterway infrastructure, in particular at locks.

1.1.3 Locks

Inland waterway infrastructure includes items like bridges, locks, and fairways. Of these, locks and bridges cause the largest bottlenecks. Optimization of locks is much more prevalent in the literature, indicating that they are a more pressing matter. Furthermore, recent work on lock scheduling in max-plus algebra left a good foundation to build on [17]. Therefore, the focus of this report is on locks.

Locks transfer vessels between upstream and downstream parts of the waterway with different water levels. This height transfer occurs in lock chambers, in which the water height can be raised or lowered to and from the upstream and downstream water levels. The components of one side of a single chamber lock constructed according to Dutch guidelines are depicted in Figure 1.5.

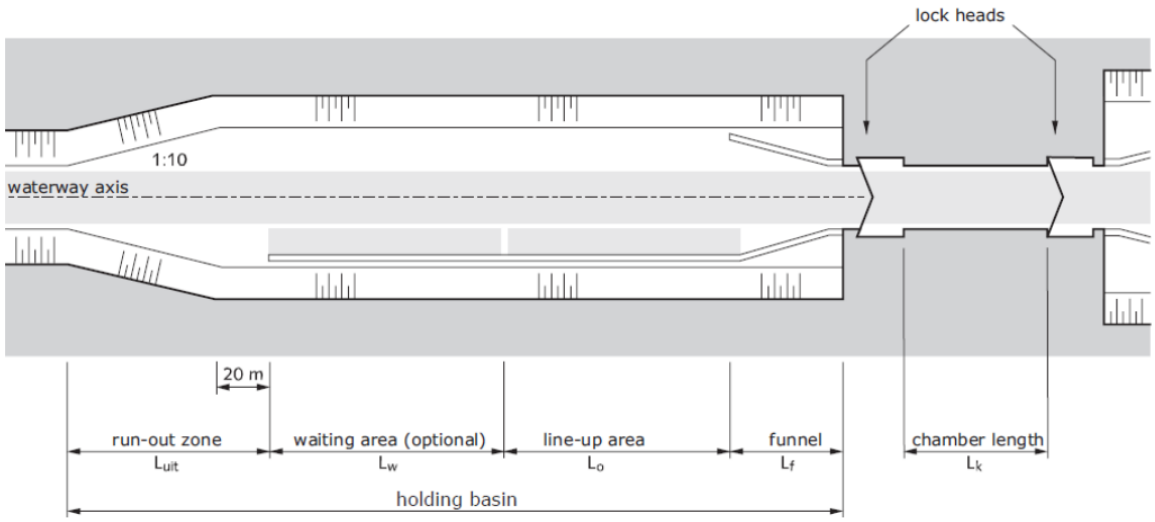


Figure 1.5: *Single-chamber lock layout according to Dutch government guidelines. Image from: [7].*

A vessel being processed by a lock undertakes the following steps. First, the vessel reduces speed upon entering the run-out zone. Locks are operated by lockmasters, who operate the lock and provide instructions to vessels using the lock. A lock movement is also called a lockage. If the approaching vessel is not going to be processed in the upcoming lockage, it waits in the waiting area. Otherwise, it lines up in the line-up area. When given the go-ahead, the vessel enters the chamber and positions itself as far forward as possible while maintaining a safe distance from other vessels unless instructed otherwise [18], mooring to the side of the chamber or another vessel. Once all vessels in the lockage have been positioned and moored, the open lock head is closed, and the water level is appropriately raised or lowered. Finally, the other lock head opens and the vessels can exit.

Commercial vessels always have priority over leisure vessels at a lock, nevertheless, they often have to wait their turn for some time at busy locks, delaying their journey. Commercial vessels are given priority to be processed on a first come first served (FCFS) basis, with overtaking not being permitted in the holding basin, unless a smaller vessel is told to overtake a larger vessel because the larger vessel does not fit in the chamber while the smaller vessel does. Furthermore, a larger distance has to be kept away from vessels transporting hazardous substances, or these vessels have to be processed in entirely separate lockages [18]. The problem of optimally placing vessels inside the locks while obeying regulations is also called the ship placement problem (SPP) in literature [19].

Locks are not limited to a single chamber. They can have multiple dependently or independently operated chambers of different sizes. As an example, the Dutch Volkeraksluis has four independent chambers, a small one for leisure vessels, and three larger ones for commercial vessels [20]. The Chinese Three Gorges Dam (TGD) has two single-directional locks with five dependent chambers in series each [21].

Some locks are equipped for hydrodynamic energy generation, harvesting energy from the water used in lock movements [22]. On the other hand, too many lock movements can negatively impact the water levels of the waterway the lock is a part of.

1.1.4 Data and Communication Systems

Several data and communication systems, like marine radio, Inland Shipping Information and Communication Systems (Binnenvaart Informatie en Communicatie Systeem, BICS), Automatic Identification Systems (AIS), and River Information Systems (RIS) are used in Dutch or international IWT, or are planned to be implemented in the future.

A marine radio is used for short-range communication between vessels, infrastructure, and waterway traffic operators. Vessels are given instructions at locks through marine radio. They can also let lockmasters and other infrastructure operators know they are coming by the same means.

Information on departure location, destination, cargo, draught, and amount of people on board can be electronically reported through the BICS system at the start of a journey [23]. Some commercial vessels are always required to report their information. Different traffic and third-party systems can interface with this data. The BICS system is available in the Netherlands and Flanders.

Real-time information on position, velocity, and course can be tracked with the AIS system [24]. Vessels equipped with AIS send out their data and receive all of the data being sent out by vessels in the vicinity. This data can be used to make informed traffic decisions, or it can be used in centralized traffic monitoring systems. Dutch lockmasters already use AIS data to see which vessels are approaching the locks, to allow for better decision-making [25].

In the Netherlands, the majority of locks and bridges are centrally managed by human operators in regional control centers, aided by cameras, radar, marine radio, AIS, and BICS [24] [26]. This centralized decision-making allows operators to look at multiple locks and bridges as a whole system, rather than at each lock separately. These control centers, for instance, try to create 'blue waves' for vessels passing through infrastructure, similar to 'green waves' for cars when cascaded traffic lights are tuned correctly [25].

Finally, the EU is working on a standardized data communication system for European IWT through the RIS initiative, with even more data and functionality available than the BICS and AIS systems [27].

1.1.5 Scheduling with SMPL Systems

Together, the waterways, the IWT vessels, and the locks can be combined into a system that can be modeled and optimized, with the underlying assumption that the data and communication systems can supply the information required for such models to run. A modeling language that can be used to compose these models is max-plus algebra, in particular by modeling IWT systems' operation as max-plus-linear (MPL) or SMPL systems.

MPL and SMPL systems can be used to model discrete event systems with max-plus algebra [28]. Max plus algebra is an algebra where the plus $(+)$ and times $(\times)$ operators are replaced by the 'oplus' $(\oplus)$ and 'otimes' $(\otimes)$ operators, respectively [29]. Certain systems that are nonlinear in regular algebra can be linear in max-plus algebra. Imagine, for instance, that several vessels are scheduled to be in the same lockage, but the lockage can only start when all vessels have arrived, then the start time of the lockage is the maximum of the arrival times of all of the vessels in the lockage. This can then be modeled as an MPL system.

The system behavior is the same for each event step in MPL systems. SMPL systems are an extension of MPL systems, where the system behavior can be different from event to event, depending on what mode the system is in.

Scheduling can be defined as: *"the process of deciding how to allocate a set of jobs to limited resources over time in such a way that one or more objectives are optimized"* [30]. Within the context of IWT, the vessels traveling through an IWT system can be seen as jobs, and locks and

lock capacity are limited resources that must be shared between the vessels in a system. The arrival times at different locations, routes, and lock chamber assignments of vessels can be optimized to try to reduce objectives like overall travel time and waiting times at locks. This act of optimization is referred to as scheduling in this thesis.

IWT systems with locks were recently scheduled using the SMPL system formulation in a thesis [31] and follow-up paper [17]. These works examined the scheduling of vessels going through set network layouts with single-chamber, single-vessel locks. Scheduling of IWT systems with locks was already a well-studied problem, known as the lock scheduling problem [32], but [31] introduced the novel approach of the SMPL formulation and added a routing component to the scheduling decisions. More details on the state of the art of IWT scheduling are outlined in Chapter 2.

1.1.6 Problem

Inland waterway transport is a low-emission alternative to road transport. The Dutch and European networks should be more heavily utilized, reducing road transport. As discussed, this has long been EU policy. However, several factors are holding back progress.

One of the factors is insufficient infrastructure [10], in particular locks. The limited capacity of locks can cause congestion, bottle-necking waterways, and leading to significant delays. Locks could be physically improved or expanded, but these projects are often lengthy and costly. Moreover, they regularly fail to achieve their stated goals, because the bottleneck can just move to a nearby part of the waterway that was not included in the improvement project [10].

Another option is to better utilize the infrastructure that is already there through optimization and scheduling, which is what will be done in this thesis. The future implementation of these types of systems can be enabled by the data from the current and upcoming data and communication systems listed in Subsection 1.1.4. NOVIMOVE, a European collaboration between IWT stakeholders, researchers, and developers, is looking to tackle this by, among other things, trying to reduce waiting times at infrastructure through dynamic scheduling systems [33]. The SMPL lock scheduling paper [17] and a paper on the scheduling of bridge openings [34] have both been published as part of the NOVIMOVE project. Outside of this project, lock scheduling is a well-studied problem in the literature, as addressed in Subsubsection 2.1.1.1.

1.2 Research Scope

This thesis can be seen as a follow-up to the work done in [31], which established the basis for SMPL scheduling of IWT systems with locks, also introducing a routing component with vessels' routes being chosen as part of the schedule. However, the work only considered single-vessel, single-chamber locks, thus neglecting the proper placement of vessels in the lock chambers through the SPP. Furthermore, the SMPL model was formulated for vessels traveling on four specific network layouts, rather than for any arbitrary system topology. The main benefit of the model being applicable to arbitrary topologies is that it will significantly speed up the process of making models for large-scale IWT networks such as the one depicted in Figure 1.1.

The goal of this thesis is then to create an SMPL IWT scheduling model that works for arbitrary IWT system topologies, and that integrates multi-vessel, multi-chamber locks with proper ship placement. This is achieved in four steps:

1. Augmenting the single-vessel, single-chamber model from [31] so it can be used for any arbitrary system topology.
2. Integrating support for multi-vessel, multi-chamber locks with simplified one-dimensional capacity constraints to limit the number of ships in the chamber.

3. Integrating support multi-vessel, multi-chamber with proper ship placement according to the ship placement problem.
4. Verifying that the models are correct and that they are correctly implemented.

Step 2 could potentially be left out, but it serves two purposes. First, the jump from step 1 to step 3 is quite large, requiring numerous additions to the model. This jump is reduced by already making part of these additions for multi-vessel support, like vessel synchronisation where multiple vessels can occupy a lock simultaneously, in step 2. Furthermore, the inclusion of the SPP may greatly increase the complexity of the problem, exponentially increasing solution times. The model of step 2 does not suffer from this problem, but it can still process multiple vessels per lockage as long as the sum of the vessels' sizes is below the one-dimensional capacity of the chamber. It can thus be used to perform analysis on the trade-off between simplifying assumptions and solution time.

The routing component of the models will remain. It will rarely be the case that a vessel will have multiple route options available. However, the route creation mechanism itself can be used to distribute vessels across chambers. Furthermore, considering routing gives a different perspective on how to build these types of models for large networks, especially when it comes to constraining how vessels interact with each other on their routes.

The resulting model will be a step towards a system that can perform coordinated scheduling of commercial vessels on complex networks like the Dutch network displayed in Figure 1.1.

1.3 Research Questions

The scope outlined in Section 1.2 can be consolidated into a main research question and four research sub-questions. The main research question is as follows:

> *How can multi-vessel, multi-chamber locks with ship placement be integrated into the SMPL IWT scheduling model?*

The answer to this main research question is found by answering the four sub-questions, each corresponding to the listed goals in Section 1.2:

1. *How can the SMPL IWT scheduling model be formulated for systems with arbitrary network topologies?*
2. *How can multi-chamber locks with one-dimensional chamber capacity constraints be integrated into the SMPL IWT scheduling model?*
3. *How can ship placement be integrated into the multi-chamber SMPL IWT scheduling model?*
4. *How can the performance of the SMPL IWT scheduling model be verified?*

1.4 Approach

The research sub-questions will be answered by creating three scheduling models, as indicated in Figure 1.6, corresponding to the first three sub-questions in Section 1.3. Each scheduling model first requires the definition of an underlying mathematical model that describes the sets, variables, parameters, constraints, and objectives. This mathematical model is then implemented into a scheduler in Python, which automatically creates scheduling optimization problems according to the rules of that mathematical model with the input parameters supplied on a case-by-case basis. Different scenarios run on the scheduler are then used to verify the mathematical model and its implementation. Each of the scheduling models builds on the previous, so the majority of the definition of the mathematical models, the implementation, and verification can carry over to the next scheduling model.

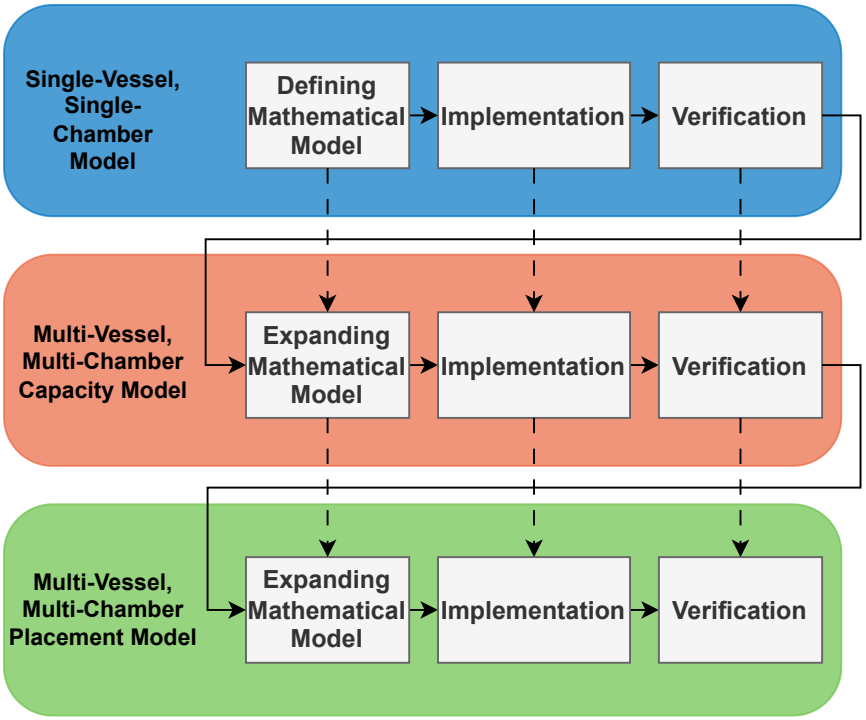


Figure 1.6: *Approach to creating and verifying the scheduling models. Solid arrows indicate the sequence of steps taken. Dashed arrows indicate the fact that work done on that action for a previous model contributes to completing that action for a subsequent model.*

It should be clarified that the optimization problems considered in this thesis are offline optimization problems. This assumes that all information about what is going to happen during the time window the optimization sequence considers is known beforehand. A schedule may prescribe vessels' arrival times for multiple hours. If there is suddenly another vessel that enters the system, if a vessel is delayed on its journey compared to the schedule, or if another disturbance causes reality to differ from the schedule, then this methodology breaks down, as this disturbance was not initially taken into account, so it may propagate through the system and cause unexpected delays.

A way to reduce the effect of disturbances is online optimization, similar to model predictive control (MPC). Online optimization, in essence, relies on solving the offline optimization problem at regular time intervals to be able to adjust the schedule for conditions that are different than predicted at the previous time interval [35]. Online optimization is left outside of the scope of this work, but the offline optimization models introduced here can serve as the basis for a future online implementation. Some considerations and recommendations for online optimization are listed in Section 3.6.

1.5 Individual Double Degree

This thesis is in fulfillment of an individual double degree for MScs in Systems & Control (S&C) and the Mechanical Engineering - Multi-Machine Engineering track (MME). The subject matter has sufficient overlap with both fields of study. For S&C, the emphasis is on the SMPL system approach, but this, of course, requires a proper understanding of the system to be able to model it. For MME, the emphasis is on the underlying understanding and modeling of the IWT system, specifically within the context of Theme 3 - Multi-Machine Coordination and Logistics.

The chapters, sections, and subsections are in general all relevant to both studies, with the exceptions of Section 2.1, Section 2.2, Section 3.4, Section 4.3, and Section 5.3. The first two are the two

literature review sections in Chapter 2. Section 2.1 is mainly focused on MME topics, while Section 2.2 is focused on S&C topics. Both sections are, however, necessary to be able to understand the models that are introduced in the later chapters. The last three are purely focused on SMPL notation with relevancy for S&C.

The research paper in Appendix A is only a requirement for MME, but it also covers the S&C topics.

1.6 Document Outline

The rest of this report's chapters are structured as follows:

- Chapter 2 contains a literature review on lock scheduling problems, the ship placement problem, max-plus algebra, SMPL systems, and SMPL systems in IWT scheduling.
- Chapter 3 answers research sub-question 1, adding support for arbitrary network topologies to the single-vessel, single-chamber model.
- Chapter 4 answers research sub-question 2, integrating multi-vessel, multi-chamber locks with one-dimensional capacities into the SMPL IWT scheduling model.
- Chapter 5 answers research sub-question 3, integrating two-dimensional ship placement into the multi-vessel, multi-chamber SMPL IWT scheduling model.
- Chapter 6 answers research sub-question 4, performing verification tests, a complexity analysis, and comparisons between the models and practice.
- Chapter 7 answers the main research question and lists recommendations for future research.

2

Literature Review

A literature review was performed on the topics of IWT operations research and IWT scheduling with SMPL systems, in preparation for this thesis. This chapter gives a condensed overview of the parts of the literature review that are relevant to this project, namely the lock scheduling problem (LSP), the SPP, max-plus algebra, and SMPL systems and how they can be applied to model IWT scheduling and the SPP.

The literature review is split into two sections. The first section is most relevant to MME, and the second section is most relevant to S&C. Section 2.1 concerns the LSP and the SPP, examining both problems and their literature. Section 2.2 concerns the combination of IWT and SMPL, also giving background on max-plus algebra.

2.1 Lock scheduling and the SPP

Numerous sources have looked at the LSP, which is the problem of scheduling vessels to pass through locks. The lock scheduling problem generally consists of three sub-problems [32]: assigning vessels to lockages, scheduling when the lockages take place, and properly placing the vessels in the lockages, also known as the SPP [19]. The models that are formulated in this thesis can be considered to be implementations of the lock scheduling problem, with the placement model of item 3 in Section 1.2 also incorporating the SPP. It is thus important to investigate the currently available research on those topics and to identify gaps that the research in this thesis can fill. This will be done by answering the following two research questions:

> *"What is the latest research on IWT lock scheduling and what gaps can be identified?"*
>
> *"What is the latest research on the ship placement problem and what gaps can be identified?"*

Subsection 2.1.1 discusses the LSP and its associated literature. The SPP and its literature are outlined in Subsection 2.1.2. Finally, the literature review research questions on the LSP and SPP are answered in Subsection 2.1.3.

2.1.1 The lock Scheduling Problem

The LSP is generally modeled as a mixed integer linear programming (MILP) including the sub-problems listed in the introduction of this section. The general structure, inputs, and outputs of the LSP in the literature are visualized in Figure 2.1. There are exceptions and this structure is often deviated from, with many studies, for instance, not including the SPP or only including a simplified version of the SPP.

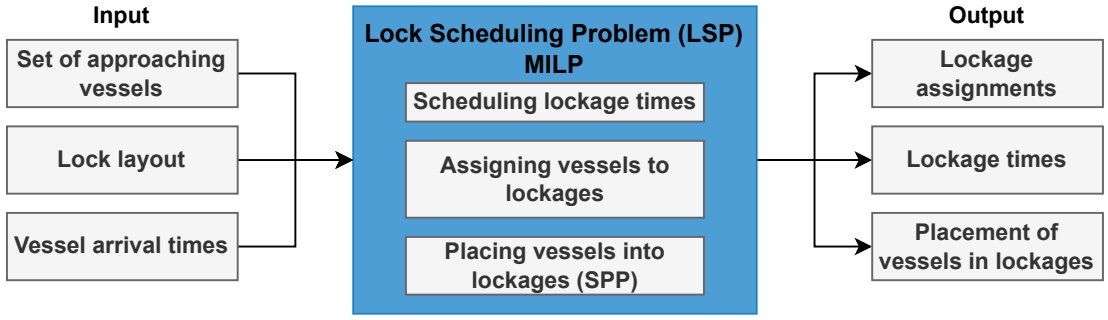


Figure 2.1: *General structure of the lock scheduling problem. Inspired by: [32].*

2.1.1.1 Lock Scheduling Problem Literature

The lockage scheduling and chamber assignment problems are often modeled together as a job shop scheduling (JSS) problem, with vessels or lockages being the jobs, and chambers being the machines. The JSS problem describes the problem of assigning jobs to machines in an order that minimizes a cost function, under constraints like a prescribed job order, or known minimal start times [36]. Furthermore, the SPP is frequently formulated as a two-dimensional bin packing problem [19]. The SPP is treated in more detail in Subsection 2.1.2.

A data-driven solution method for scheduling of the single directional locks next to the TGD was proposed in [37]. It included an SPP with simplified constraints in the model. While the TGD lock has multiple chambers, they are all serial and single-directional, simplifying the problem to a single-chamber problem.

In [38] a polynomial time algorithm was introduced to solve the scheduling problem, modeled as a JSS problem, for a single lock that can handle a single vessel at a time, under the assumption that all vessels have equal processing times. The SPP was not treated.

Polynomial time solvability using a graph-based dynamic programming model of a similar problem with multiple vessels allowed per lockage, without accounting for their placement, dubbed the lockmaster's problem, was shown in [39]. It was also proven that extended versions of the lockmaster's problem, with lockage vessel number capacity constraints, a limit on the number of lockages allowed, or built-in vessel priority achieved through weighted waiting times in the objective function, can all be solved in polynomial time. Furthermore, some NP-hard versions of the problem may become solvable in polynomial time if the FCFS policy is applied.

The authors of [40] solved the lock scheduling problem with all of its sub-problems for multi-chamber locks, using a heuristic for the SPP connected to an MILP that schedules the lockages. This could not result in an optimal solution because of the heuristics, and would still run into issues of computation time on the scheduling sub-problems alone.

In [32], an exact solution to the full lock scheduling problem, including all sub-problems, was examined. An MILP was set up that encompassed the full problem for a single lock of multiple chambers. Computation time was identified as a limiting factor, and combinatorial benders' decomposition and heuristics were introduced as methods to reduce solution complexity. [41] covered the exact MILP model in a shorter paper, focused on exact solution methods only.

A problem formulation for a problem with multiple serial or interdependent multi-chamber locks, known as the serial lock scheduling problem (SLSP), where lockages have to be scheduled for a set of vessels going through a set of consecutive locks, was treated in [42]. A simple length-based capacity constraint was used to replace the SPP, and multiple heuristics were used to find solutions with the model formulated as a graph.

Two exact MILP model formulations, one time-based and one lockage based, for the SLSP with single chambers were described in [43]. A simplified maximum capacity constraint was used instead of the SPP. The velocities of the vessels between locks were added as decision variables, with the objective of reducing emissions, or the time for vessels to move through all of the locks. With a piecewise linear approximated emission model, it was found that, for their model and assumptions, a given percent point increase in time spent going through the serial lock system could result in a comparatively larger percent point decrease in CO_2 emissions.

A layered heuristics solution algorithm for the multi-chamber SLSP with SPP was proposed in [44]. The layered solution consisted of an outer simple heuristic scheduling layer, and an inner generational meta-heuristic SPP layer, which would iterate to achieve a good solution.

MILP for the SLSP with multi-chamber locks were formulated in [45] and [46], respectively from a JSS and a multi-commodity network perspective. The relative performance of both of the proposed MILP model types was tested in [47], and the JSS version outperformed the multi-commodity flow formulation in optimality gap and computation time. However, it should be noted that both model types struggled for exact solutions to larger problems.

Shorter computation times were achieved with an adaptive large neighborhood serach heuristic in [48]. Large neighborhood search was also used in [49], for a formulation with cascaded single chamber locks and vessel velocity control.

Extended problem formulations that consider co-scheduling between locks and transshipment facilities that allow the transshipment of goods to other modes of transport, instead of the vessel having to pass a lock, also exist. An MILP and a heuristics-based solution method were presented for a single lock and chamber problem in [50]. A similar paper for a lock with two unidirectional chambers considered waiting times and transshipment cost as the objectives [51], finding solutions with a genetic algorithm. The combined scheduling of that same lock, the TGD in China, operating in parallel with a ship lift was optimized in [52].

Finally, most of the literature assumes that the route the vessels will take is set from the start. In some cases that may not be true, and there may be an opportunity to take a different route. This problem was considered in [17], for waterway networks with multiple routes and (series) single-chamber, single-vessel capacity locks. The model formulation of the proposed model was done in max-plus algebra for the first time, which will be discussed in more detail in Section 2.2.

Table 2.1 gives an overview of the main components included in the models of each of the pieces of literature on the lock scheduling problem considered in this section.

Table 2.1: *An overview of the different model components present in the literature reviewed in this section.*

	Model Components					
Study	**LSP**	**Multi-chamber**	**Cascaded Locks**	**SPP**	**Transshipment**	**Route Planning**
[38]	✓					
[37]	✓			✓		
[32]	✓			✓		
[42]	✓	✓	✓			
[43]	✓		✓			
[39]	✓					
[41]	✓	✓		✓		
[40]	✓	✓		✓		
[44]	✓	✓	✓	✓	✓	
[45]	✓	✓	✓	✓		
[46]	✓	✓	✓	✓		
[47]	✓	✓	✓			
[48]	✓	✓		✓		
[49]	✓		✓	✓		
[50]	✓	✓[1]		✓	✓	
[51]	✓	✓[1]		✓	✓	
[17]	✓		✓			✓
[52]	✓	✓[2]		✓		

Over time, the formulations of the lock scheduling problem have grown from a simple setup of a single lock with a single chamber to a generalized problem with multiple locks and chambers in a network, with recent papers even adding decision variables on transshipping goods to land-based transport to avoid locks entirely. Beyond the use in very specific cases, this form of transshipment seems to be more of a gimmick than anything else.

More universally applicable improvements could be achieved by further expanding the scope of the problems to include a choice of what infrastructure is used by vessels, as addressed in [31] and [52]. If vessels can be distributed between multiple routes, this could further reduce congestion, however, this would require a problem formulation that can look at a bigger network of diverse locks, which has not been looked at yet. In the future, this network could also include other types of infrastructure.

A novel modeling method with the use of SMPL systems was introduced in [31], but only for single-vessel, single-chamber locks. In lock systems, where a lockage can only start when all vessels of the lockage have arrived making the start time of the lockage equal to the maximum of all vessels' arrival time, or where the time at which a vessel can go through a lock is based on the maximum

[1] Multiple chambers in series.
[2] Lock in parallel with a ship lift, which is essentially a multi-chamber problem.

leaving time of the vessels that are ordered before it, max-plus algebra and SMPL systems are a good modeling tool.

Furthermore, it is not a column in the table, but the papers listed in this section limit themselves to optimizing the situation offline. Coordination systems would also have to be able to handle new ships arriving in the system, or disturbances like lock malfunctions, so it could be valuable to look at the effects of including these factors through online optimization.

Finally, there is little mention of distributed optimization in the literature. Large-scale networks like the one in Figure 1.1 could certainly benefit from some form of distributed optimization to reliably be able to find solutions within an acceptable timeframe.

2.1.1.2 Key Performance Indicators

The studies presented in the previous subsection use a range of different key performance indicators (KPIs). The KPIs are listed per category in Table 2.2. Note that two studies with the same KPI may not have the same mathematical definition for it, but the underlying meaning will be the same.

Table 2.2: *Overview of the KPIs used for the studies examined in this section. A "Max" prefix indicates that the maximum value in the problem of that particular KPI is also used as a KPI.*

	KPIs				
Study	**Time**	**Lockage**	**Cost**	**Emissions**	**Routing**
[38]	B				
[37]	A				
[32]	D, Max D	F			
[42]	C			I	
[43]	B			I	
[39]	A				
[41]	D, Max D	F			
[40]	E, Max E				
[44]	B	G			
[45]	A				
[46]	C	F			
[47]	C	F			
[48]	D, Max D	F			
[49]	C	K			
[50]	A		H		
[51]	A		H		
[52]	A, Max D	F, G			
[17]	C				J

The following list gives a description of each of the KPIs:

- **(A) Vessel Waiting Time:** The amount of time a vessel spends waiting to go through a lock. The waiting time may also be added up for multiple locks.
- **(B) Vessel Processing Time:** The time between the vessel arriving at a lock and leaving the lock (including waiting time). Able to account for lock processing times that vary per vessel or lockage.
- **(C) Vessel Travel Time:** Time it takes for a vessel to get from starting point to destination. Will include waiting times of multiple locks, if multiple locks are traveled through.
- **(D) Vessel Completion Time:** The time at which a vessel exits a lock.
- **(E) Lockage Tardiness:** How late a lockage is for its scheduled timeslot.
- **(F) Number of Lockages:** The number of times that a lock is operated in the problem.
- **(G) Chamber Area Utilization:** The fraction of the area of the lock that is covered by vessels when it is operated.
- **(H) Cost per Vessel:** Monetary cost per vessel. For instance due to waiting times, spoilage of goods on board, fuel, or transshipment costs.
- **(I) Vessel Emissions:** Emissions emitted by the vessel while going through the network. Can be reduced by making velocity a decision variable, as emissions are linked to velocity.
- **(J) Overtaking/Route Selection:** Vessels overtaking one another, or taking an unwanted route, can be prevented by applying a penalty to the decision variables associated with these events.
- **(K) Minimum Number of Ships in Any Lockage:** A way to penalize lockages that are operated with very few ships.

The maximum vessel transit time and maximum lockage tardiness KPIs are an interesting way to enforce a level of fairness in optimization problems. Situations where vessels have to wait a disproportional amount of time can be avoided by adding a penalty for it in the objective function.

Besides that, it is clear that the main focus of these studies is to save time for the vessels. Reducing the amount of lockages is also often taken into account. On the other hand, emissions are rarely looked at, even though ships can be told to reduce their speed instead of going full speed only to wait at a lock [43] [53]. If these models are built out to full networks like suggested in Subsubsection 2.1.1.1, it would be a waste to not try to optimize for emissions as well.

2.1.1.3 Case Studies

Several case studies can be identified across literature, with authors often establishing a general model structure, which is then used for a specific real-life problem.

Data from the Albertkanaal is freely available and often applied in case studies [41]. The Albertkanaal connects the port of Antwerp and Liege and has 6 triple-chamber locks along its length [54].

The TGD and Gezhouba Dam are popular Chinese case studies [44] [51], [52], both situated on the Yangtze River. The TGD has two single-directional locks, with 5 chambers in series for each, as the height difference that has to be traversed is large. A boat lift has recently been installed [21] to help overcome congestion problems. The cooperation of the boat lift and the locks was examined in [52]. The GD has 3 parallel chambers.

Other case studies include the Danube [42] and the Upper Scheldt [43]. An overview of the case studies used in each of the papers discussed in Subsubsection 2.1.1.1 is given in table Table 2.3. Note that some papers do not have case studies based on real life, in which case they usually have a fictitious case study.

Table 2.3: *An overview of case studies applied in lock scheduling problem research. An empty row implies that no case study based on real-life has been performed.*

	Case Study						
Study	**TGD**	**Albertkanaal**	**GD**	**Upper Scheldt**	**Danube**	**Mississippi**	**Ohio**
[38]							
[37]	✓						
[32]		✓					
[42]					✓		
[43]				✓			
[39]							
[41]		✓					
[40]							
[44]	✓		✓				
[45]		✓					
[46]	✓	✓	✓				
[47]	✓	✓	✓				
[48]							
[49]							
[50]	✓						
[51]	✓						
[52]	✓						
[17]							

As shown in the table, there are more than enough case studies available. The most commonly used ones are the TGD and the Albertkanaal because they have comprehensive data available.

2.1.2 Ship Placement Problem

Lock chambers have a limited capacity, as their chambers are constrained by their physical size. Additionally, some rules govern the placement of ships in locks. For instance, ships are required to maintain 10 meters of distance between them and ships with flammable goods in locks in the Netherlands [18]. If the ships and chambers are approximated by rectangles, the problem of fitting as many ships in a chamber as possible becomes a two-dimensional bin-packing problem. In a two-dimensional bin packing problem, rectangular items must be packed into a minimal number of rectangular bins without the items overlapping [55], as portrayed in figure 2.2.

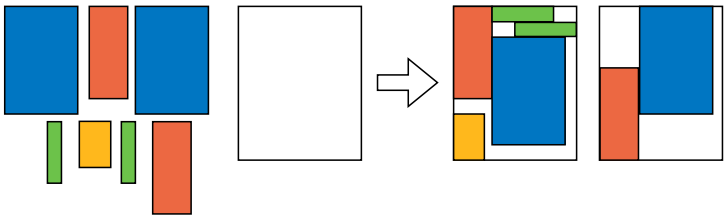


Figure 2.2: *Schematic which illustrates a feasible solution of a bin packing problem.*

The ship-placement problem is the application of the two-dimensional bin packing framework to place ships into lockages, while obeying relevant constraints, as introduced in [32] and [19].

2.1.2.1 Ship Placement Problem Literature

The SPP is a sub-problem of the lock scheduling problem and usually is not the main subject of a piece of research. Furthermore, the two-dimensional bin packing problem is NP-hard [55]. The constraints on the SPP may relax its NP-hardness, similar to the FCFS constraint for the LSP, but the author of this work is not aware of any proof that that may be the case. In short, the SPP adds a lot of complexity to the larger problems it is added to. Other approaches to account for lock chamber capacity aim to reduce the problem's complexity. Examples are simple maximum number of ships per chamber constraints [43], constraints that take into account only one dimension of the chamber and ships, like their length [42], or considering the two-dimensional bin packing problem directly without additional constraints.

The most prevalent and extensive current implementation of the full ship placement problem was described in [32]. It was implemented as an MILP problem, with a representative set of placement constraints for real locks. The constraints that set the ship placement problem apart from a regular 2D bin-packing problem in [32] are: vessels not being allowed to be rotated in the chamber, vessels having to be properly moored to the side of the chamber or another vessel, and safety distance constraints. Mooring of vessels to one another is only allowed if the vessel mooring to another vessel can be fully alongside the vessel it is mooring to. Safety distance constraints between vessels in the chamber must, for instance, be obeyed to allow safe maneuvering, satisfy safety regulations, and allow tug boats to maneuver in the chamber. Other considerations can be placing the vessels a safe distance away from the lock heads and considering the maximum draught of the vessels and the allowable draught in the chamber [32].

Comparisons between an exact MILP solution approach to the SPP and a set of heuristic solution methods were provided in [19]. It was shown that the MILP can be sped up significantly by decomposing it. Additionally, heuristic solution methods proved to be orders of magnitude faster for problems with large amounts of vessels, at the cost of optimality gaps of at most 4% for the best heuristic implementation. The achieved configurations often came very close to how lock operators would have placed them [19].

An SMPL implementation of the lock scheduling problem does not yet exist. However, there are implementations of Tetris and job scheduling systems that can be used as a basement for implementing the sequence of vessels being placed into the chamber as an SMPL system [56]. A different modeling technique could provide new insights or may lead to faster solutions than the traditional MILP modeling technique.

Table 2.1 gives an overview of LSP literature, with one of the columns signifying if the SPP has been applied in some form or not. The column includes cases in which the SPP has been heavily simplified.

2.1.2.2 Placement Problems in Other Modes of Transport

Efficient use of space and smart placement of goods or vehicles are important for any mode of transport. See, for instance, the example in Figure 2.3, where there are two feasible configurations for storing packages in a van. The packages are numbered by delivery order. While the configurations have similar levels of space efficiency, configuration (b) will lead to easier sequential unloading at delivery points. While not always directly analogous, examining the way load placement problems are approached in different modes of transport may give insight into how to improve the implementation of the SPP. Furthermore, fast heuristics developed for one bin packing/placement problem formulation may be portable to other formulations.

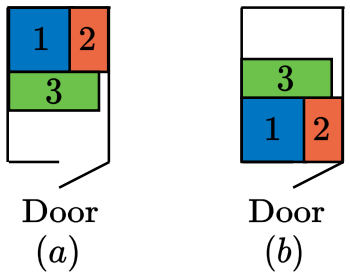


Figure 2.3: *Example of different delivery van packing configurations. The packages are numbered by delivery order.*

In [57], the two-dimensional loading capacitated vehicle routing problem (2L-CVRP) was examined, combining a rear-loaded load packing problem with a vehicle routing problem (VRP) to optimize delivery routing transport costs, while accounting for package ordering and delivery vehicle constraints. Rear loaded means that the space between the delivery item and the door must be empty when it is delivered, so no other items have to be moved around, the benefit of which was demonstrated by the example in Figure 2.3. An ant colony heuristic was used to solve the optimization problem. Extensions of the VRP, like simultaneous pick-ups and deliveries, have also been integrated with 2D package placement problems [58]. Implementations with 3D packing also exist [59].

The constraints introduced by the rear loaded formulation in particular could be used in the SPP, where ordering the ships for departure could help prevent ships having to overtake one another right after going through a lock, or in serial lock configurations ships may be ordered in such a way that the ships scheduled for lockages at the next lock first are placed at the front of the lockage.

Roll-on Roll-off (RoRo) ships face a similar challenge to the one faced by locks, where they have to try and pack as many vehicles as possible into their hold. This problem may be modeled as a 2D bin packing problem, like done in [60] and [61]. [60] introduced the 2D packing version of the problem, and tested MILP implementations with different levels of constraints for a single deck level of a RoRo ship with multiple destinations. The problem was expanded with ship stability constraints, rotated vehicles, and a two-stage solution heuristic for a single destination port in [61].

Multi-deck implementations of the RoRo placement problem could be similar to multi-chamber or multi-lockage SPP formulations, though they would also have to take into account the stability of the RoRo ship.

2.1.2.3 Key Performance Indicators

Since the ship placement problem is often part of a larger overarching problem, the KPIs used in those overarching problems are usually not specific to the SPP, they just indirectly affect the outcome of the SPP to get a better solution to the full problem. The KPIs of lock scheduling research are tabulated in Table 2.2, and they also contain some KPIs related to the SPP. The KPIs that are most

closely related to the SPP are chamber area utilization, number of lockages, and minimum number of ships in any lockage.

At first glance, area utilization seems to be more flexible, as it can account for different-sized chambers, penalizing water usage for locks that are not full. However, the number of lockages KPI can be weighted to create the same effect, where the usage of larger locks can just carry a heavier penalty. The minimum number of ships in any lockage KPI is essentially a different way to maximize area utilization.

The research on placement problems in other modes of transport mainly uses the area utilization factor KPI. A special mention can be made for [60], in which having to shift around the vehicles on RoRo ships to unload other vehicles is penalized in the objective function.

Future LSP research could pay more attention to the quality and behavior of the SPP solutions within the context of LSP solutions when different SPP criteria and criteria weightings are applied alongside the common time-based LSP criteria. An interesting topic could, for instance, be how the weighting of lockage-based criteria compared to waiting-time-based criteria should vary across the seasons to account for seasonal water-level changes.

2.1.3 Conclusion

For the lock scheduling problem, the question to be answered by this literature review was:

> *"What is the latest research on IWT lock scheduling and what gaps can be identified?"*

The most recent literature includes models of the lock scheduling problem that can provide optimal scheduling solutions for multiple, possibly cascaded, locks with multiple heterogeneous chambers, in a single network, simultaneously incorporating the optimal placement of vessels in the chambers through the ship placement problem.

A next step has been taken by including route choice into a simple lock scheduling problem formulation, allowing ships to bypass congested locks by taking a different route entirely. Subsequent research could focus on including route choice into more complex formulations of the lock scheduling problem, in combination with multiple lock chambers and the ship placement problem. Emission objectives could also be more widely included, as vessels can be slowed down when approaching infrastructure they would have to wait for anyway.

IWT lock scheduling systems have also recently been modeled as SMPL systems, though only for a simplified scenario with single-vessel, single-chamber locks. SMPL models of IWT could be further developed to include multi-vessel, multi-chamber locks with proper ship placement.

Some more gaps outside the scope of this thesis can be identified, but they are important to mention. Online optimization has not yet been considered, accounting for disturbances and continuously changing circumstances. A real implementation of a lock scheduling system would have to continuously optimize with regular time intervals. Distributed optimization has also not been considered, while it could prove to be a necessity when trying to implement lock scheduling for larger-scale systems.

For the ship placement problem (SPP), the question to be answered was:

> *What is the latest research on the ship placement problem and what gaps can be identified?*

The SPP is the problem of trying to optimally arrange vessels in lock chambers. The implementation for this problem is analogous to the two-dimensional bin-packing problem with added constraints. It is rarely studied on its own, as it is normally a sub-component of a larger optimization problem.

The majority of recent lock scheduling problem literature has some implementation of the ship placement problem as a sub-problem, and the formulations do not differ much from the one introduced in the seminal research papers, because the original implementation of the problem was already quite comprehensive. The ship placement problem is NP-hard, so it can be a big source of solution complexity. As a result, the sub-problem is often solved through heuristics, which can achieve satisfactory results in less computation time.

The SPP has not been implemented as an SMPL system, but implementations of Tetris do exist. Following this Tetris approach could lead to a new way to formulate the SPP as a SMPL system.

Other modes of transport, like delivery vans and Roll-on Roll-off ships, have their own versions of placement problems. Delivery vans have to ensure that packages that are delivered first can be unloaded easily, without having to shift around other packages. This requirement could be translated to the ship-placement problem in the future, placing faster ships at the front of the chamber so no overtaking is required directly after leaving the lock, or similar modifications based on arrival order could be considered. Roll-on roll-off ships have to pack as many vehicles on a deck as possible. Future applications with multiple decks could be closely analogous to the ship placement problem with multiple chambers.

Lastly, the number of lockages and the area utilization factor are commonly used KPIs to influence the ship placement problem as a sub-problem of the lock scheduling problem. The exact effects of different KPIs on the solutions of the ship placement problem within the context of other problems has not yet been explored in detail.

2.2 IWT and Max-Plus Algebra

The states in max-plus algebra systems generally represent time, rather than quantities like position or velocity. This naturally lends itself well to scheduling formulations that schedule the time at which events take place.

Furthermore, as discussed before in Subsection 1.1.5, the maximization operator is also naturally applicable in scheduling, for instance for scheduling that a lockage can only start once all vessels of that lockage have arrived.

While IWT systems were previously modeled as SMPL systems in [31], it must be investigated how the extensions mentioned in Section 1.2 can be added. This leads to the following research question:

> *How can additions to the existing SMPL IWT scheduling model be made within the SMPL framework?*

Additionally, the research question in Section 1.3 investigates how the SPP could be modeled as an SMPL system:

> *How can the SPP be formulated in the SMPL framework?*

Subsection 2.2.1 gives background on the basics of max-plus-algebra, followed by an explanation of max-plus-linear systems in Subsection 2.2.2. SMPL systems are explained in Subsection 2.2.3, in particular also highlighting how the SPP could be modeled as an SMPL system. How IWT scheduling can be performed with SMPL systems is described in Subsection 2.2.4. Relevant steps for turning SMPL systems into MILPs are listed in Subsection 2.2.5. Finally, the answers to the literature review research questions on SMPL systems are given in Subsection 2.2.6.

2.2.1 The Basics

This section describes the basic definitions, operations, and properties of max-plus algebra. The material of this section is based on [29], which should be assumed to be the source unless stated otherwise.

2.2.1.1 Operations and Definitions

Max-plus algebra is centered around the max plus algebraic operators *oplus*: $\oplus$, and *otimes*: $\otimes$. The following discussion and definitions will be in the max-plus algebraic framework unless specified otherwise. Scalars may take on values in $\mathbb{R}_{\max} = \mathbb{R} \cup \{-\infty\}$, with $\oplus$ and $\otimes$ operations on scalars respectively representing $\max$ and $+$ operations. With $a, b \in \mathbb{R}_{\max}$:

$$a \oplus b = \max(a, b), \qquad a \otimes b = a + b. \tag{2.1}$$

The sum, $\sum$, and product, $\prod$, operators of normal algebra find their equivalents $\bigoplus$ and $\bigotimes$ in max-plus algebra:

$$\bigoplus_{i=1}^{n} a_i = \max(a_1, a_2, ..., a_n), \qquad \bigotimes_{i=1}^{n} a_i = a_1 + a_2 + ... + a_n \tag{2.2}$$

$\oplus$ and $\otimes$ can also be used in matrix computations, where matrix entries may take on values in $\mathbb{R}_{\max}$. With $A, B \in \mathbb{R}_{\max}^{n\times m}$, $C \in \mathbb{R}_{\max}^{m\times n}$, the notation that $[A]_{ij}$ is entry a_{ij} of matrix A, and the notation that $\underline{n} = \{1, 2, ..., n\}$:

$$[A \oplus B]_{ij} = \max(a_{ij}, b_{ij}) \quad \forall (i, j) \in (\underline{n}, \underline{m}), \tag{2.3}$$

$$[A \otimes C]_{ij} = \bigoplus_{l=1}^{n} a_{il} \otimes c_{lj} = \max_{j\in\underline{n}}(a_{il} + c_{lj}) \quad \forall (i, j) \in (\underline{n}, \underline{n}). \tag{2.4}$$

The zero and unit quantities are not equal to 0 and 1, respectively. Instead, the max-plus algebraic zero, ε, and unit, e, quantities are defined as follows, with $a \in \mathbb{R}_{\max}$:

$$\varepsilon = -\infty \qquad a \oplus \varepsilon = \varepsilon \oplus a = a \qquad a \otimes \varepsilon = \varepsilon \otimes a = \varepsilon, \tag{2.5}$$

$$e = 0 \qquad a \otimes e = e \otimes a = a. \tag{2.6}$$

It can be seen that this is how these operations would behave with 0 and 1 in plus-times algebra. Indeed, this is on purpose, as with these properties $\mathbb{R}_{\max}$, $\oplus$, $\otimes$, ϵ and e, form an idempotent, commutative semiring $\mathcal{R}_{\text{max,plus}} = (\mathbb{R}_{\max}, \oplus, \otimes, \epsilon, e)$, also called a dioid. Commutativity is shown, along with other algebraic properties that make max-plus algebra a commutative semiring, in table Table 2.4, and idempotency is shown in Equation 2.14. Max-plus algebra forming a semiring ensures that many statements proven in regular algebra have an equivalent in max-plus algebra.

With its zero and unit matrices, max-plus matrix algebra forms a non-commutative, idempotent semiring. These zero and unit matrices are denoted by $\mathcal{E}(n, m)$ and $E(n, m)$, and are defined as:

$$[\mathcal{E}(n, m)]_{ij} = \varepsilon \quad \forall (i, j) \in (\underline{n}, \underline{m}), \tag{2.7}$$

$$[E(n, m)]_{ij} = \begin{cases} 0 & \text{if} \quad i = j \\ \varepsilon & \text{if} \quad i \neq j \end{cases} \qquad \forall (i, j) \in (\underline{n}, \underline{m}). \tag{2.8}$$

If $n = m$, then $E(n, m)$ is an identity matrix. The following properties hold for these matrices, where $A \in \mathbb{R}_{\max}^{n\times m}$:

$$A \oplus \mathcal{E}(n, m) = \mathcal{E}(n, m) \oplus A, \tag{2.9}$$

$$A \otimes E(m, m) = E(n, n) \otimes A = A. \tag{2.10}$$

$$\mathcal{E}(l,n) \otimes A = \mathcal{E}(l,m), \qquad A \otimes \mathcal{E}(m,l) = \mathcal{E}(n,l). \tag{2.11}$$

Finally, max-plus algebra knows one more operation, the power operation. This operation is defined for scalars and matrices, with $a \in \mathbb{R}_{\max}$ and $A \in \mathbb{R}_{\max}^{n\times n}$:

$$a^{\otimes n} = \underbrace{a \otimes a \otimes \ldots \otimes a}_{n\times\otimes} = n \times a, \qquad a^{\otimes 0} = e, \tag{2.12}$$

$$A^{\otimes n} = \underbrace{A \otimes A \otimes \ldots \otimes A}_{n\times\otimes}, \qquad A^{\otimes 0} = E(n,n) \tag{2.13}$$

2.2.1.2 Properties

Table 2.4 gives an overview of the algebraic properties of scalar operations in max-plus algebra. Properties of normal algebra are given for comparison. Zero and unit element properties were discussed previously, so they are omitted. These properties together make max-plus algebra an algebraic structure called a *commutative semiring*.

Table 2.4: *Algebraic properties of scalars in max-plus (MP) and plus-times (PT) algebra.* $a, b, c \in \mathbb{R}_{\max}$ *for MP, and* $a, b, c \in \mathbb{R}$ *for PT.*

Property	MP	Definition	PT	Definition
Associativity	$\oplus$	$a \oplus (b \oplus c) = (a \oplus b) \oplus c$	$+$	$a + (b + c) = (a + b) + c$
	$\otimes$	$a \otimes (b \otimes c) = (a \otimes b) \otimes c$	$\times$	$a \times (b \times c) = (a \times b) \times c$
Commutativity	$\oplus$	$a \oplus b = b \oplus a$	$+$	$a + b = b + a$
	$\otimes$	$a \otimes b = b \otimes a$	$\times$	$a \times b = b \times a$
Distributivity	$\otimes$ over $\oplus$	$a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$	$\times$ over $+$	$a \times (b + c) = (a \times b) + (a \times c)$

Max-plus algebra is an *idempotent* semiring or *dioid* because of another property, called idempotency [29]. Idempotency is the property that:

$$a \oplus a = a, \tag{2.14}$$

which has no equivalent in conventional algebra, as:

$$a + a \neq a. \tag{2.15}$$

These algebraic properties and idempotency also hold for matrix operations, except for $\otimes$ commutativity. However, $\times$ commutativity does not hold in conventional algebra either:

$$A \otimes B \neq B \otimes A, \qquad AB \neq BA \tag{2.16}$$

2.2.2 Max-Plus-Linear Systems

This section discusses MPL dynamic systems. Theory is based on [29], unless otherwise specified.

2.2.2.1 Autonomous System Description

In standard algebra, discrete-time autonomous systems with state vector $\boldsymbol{x}(k)$ of length n, where $k \in \mathbb{Z}$ is the time/step index, can be described with a state-space equation with $A \in \mathbb{R}^{n\times n}$:

$$\boldsymbol{x}(k+1) = A\boldsymbol{x}(k). \tag{2.17}$$

Similarly, some dynamic systems that are nonlinear in standard algebra, especially a subclass of discrete event systems (DES), can be turned into systems that are linear in max-plus algebra [28], of the form:

$$\boldsymbol{x}(k+1) = A \otimes \boldsymbol{x}(k). \tag{2.18}$$

Starting with $\boldsymbol{x}(0)$, any subsequent state $\boldsymbol{x}(k)$ is determined by:

$$\boldsymbol{x}(k+1) = A^{\otimes k} \otimes \boldsymbol{x}(0). \tag{2.19}$$

2.2.2.2 Autonomous System Example

Take a fictitious production system with 2 workers and 2 workstations, as shown in Figure 2.4. At

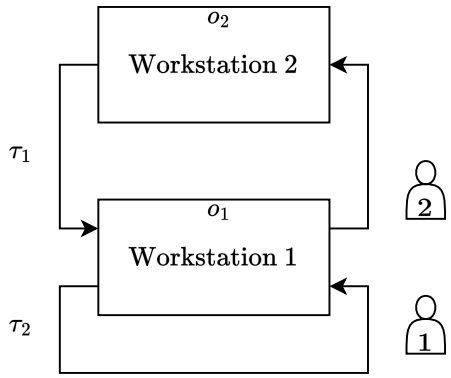


Figure 2.4: *Example DES production system with 2 workers and 2 workstations.*

workstation 1, both workers are required and they perform their task together in o_1 minutes. At workstation 2, worker 2 must perform a specialized task, which takes o_2 minutes. The travel time between workstations for worker 2 is τ_1 minutes. Meanwhile, worker 1 has to take a τ_2 minute rest after the physical labor performed at workstation 1.

The states describe when tasks start, with x_1 being a task on workstation 1, and x_2 being a task on workstation 2. Tasks are performed as soon as possible. For x_1, it is required that:

$$x_1(k+1) \geq x_2(k) + o_2 + \tau_1 \tag{2.20}$$
$$x_1(k+1) \geq x_1(k) + o_1 + \tau_2 \tag{2.21}$$

Which translates to:

$$x_1(k+1) = \max(x_2(k) + o_2 + \tau_1, x_1(k) + o_1 + \tau_2). \tag{2.22}$$

For workstation 2 the relation is simpler:

$$x_2(k+1) = x_1(k) + o_1 + \tau_1. \tag{2.23}$$

Or, equivalently:

$$x_2(k+1) = \max(x_1(k) + o_1 + \tau_1, x_2(k) + \varepsilon). \tag{2.24}$$

Following the system description in Equation 2.18 this becomes:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+2) \end{bmatrix} = \begin{bmatrix} o_1 + \tau_2 & o_2 + \tau_1 \\ o_1 + \tau_1 & \varepsilon \end{bmatrix} \otimes \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \tag{2.25}$$

Note that this system description that would have been non-linear in regular algebra has become linear in max-plus algebra.

2.2.2.3 System with Inputs

Max-plus algebraic systems may also be controlled with external inputs, denoted by $r(k)$. A basic form of such a system, with $A \in \mathbb{R}_{\max}^{n\times n}$, $B \in \mathbb{R}_{\max}^{n\times m}$, $\boldsymbol{r} \in \mathbb{R}_{\max}^{m}$, and $\boldsymbol{x} \in \mathbb{R}_{\max}^{n}$, is given by [62]:

$$\boldsymbol{x}(k) = A \otimes \boldsymbol{x}(k-1) \oplus B \otimes \boldsymbol{r}(k). \tag{2.26}$$

Assume the floor manager of the previously described production system can give worker 1 a break, having another worker take over for worker 1. This worker has an even longer resting time, $\tau_3 > \tau_2$. The input to the system becomes $r(k) = x(k-1) + \tau_3 + o_1$, with $B = \begin{bmatrix} e & \varepsilon \end{bmatrix}^T$. Then:

$$x_1(k) = \max(o_1 + \tau_2 + x_1(k-1), o_2 + \tau_1 + x_2(k-1), x_1(k-1) + o_1 + \tau_3) \tag{2.27}$$
$$x_2(k) = \max(o_1 + \tau_1 + x_1(k-1), \varepsilon, \varepsilon) \tag{2.28}$$

Clearly, this example only makes sense if $\max(o_1+\tau_2+x_1(k-1), o_2+\tau_1+x_2(k-1), o_1+\tau_3+x_1(k-1)) = o_1+\tau_3+x_1(k-1)$, otherwise the floor-manager's actions would have no impact. A formulation that circumvents this problem, and which would allow for situations where $\tau_3 \leq \tau_2$, would be a max-plus-switching system. Those types of systems will be described in Subsection 2.2.3.

State updates of $\boldsymbol{x}(k)$ are not limited to only being dependent on the state $\boldsymbol{x}(k-1)$ and $\boldsymbol{r}(k)$. In a more general form, $\boldsymbol{x}(k)$ may be dependent on $\boldsymbol{x}(k)$, $x(k-\mu)$, with $\mu \in \{\mu_{min}, ..., \mu_{max}\}$, and $\boldsymbol{r}(k)$ [63]:

$$\boldsymbol{x}(k) = \bigoplus_{\mu=\mu_{min}}^{\mu_{max}} A \otimes \boldsymbol{x}(k-\mu) \oplus B \otimes \boldsymbol{r}(k). \tag{2.29}$$

Think, for instance, of a production system where a step spans multiple system cycles. If $0 \in \{\mu_{min}, ..., \mu_{max}\}$, state $\boldsymbol{x}(k)$ depends on itself and the system is in implicit form. Under certain conditions, the system may be rewritten to an explicit form with $0 \notin \{\mu_{min}, ..., \mu_{max}\}$ [64].

SMPL systems with inputs have their own notion of stability and controllability [65].

2.2.2.4 Event and Product Counters

Up until now, k has had the interpretation of an event counter, with $x(k)$ being the state of the dynamic system at event k. However, k may also represent a product counter, in which case $x(k)$ represents the state of product k [30].

For an event counter, causality, in most cases, dictates that $\mu_{min} \geq 0$, as events that occur in the future should not be able to influence events in the past. This does not hold for systems with a product counter, as product $k+1$ may influence the process governing product k without breaking causality.

2.2.3 Switching Max-Plus-Linear Systems

This section treats switching max-plus-linear systems. A general system description is given in Subsubsection 2.2.3.1, with the switching mechanism explained in Subsubsection 2.2.3.2.

2.2.3.1 System Description

SMPL systems model event systems for which the system dynamics differ for different values of k. This allows for a higher level of modelling flexibility compared to MPL systems, where the dynamics are fixed. They were first introduced in [28].

Max-plus-switching systems have w modes, with the mode being denoted by $\ell(k) \in \{1, 2, ..., w\}$. The general description for such a system, with $A^{(\ell(k))} \in \mathbb{R}_{\max}^{n\times n}$, $B^{(\ell(k))} \in \mathbb{R}_{\max}^{n\times m}$, $r \in \mathbb{R}_{\max}^{m}$, and

$x \in \mathbb{R}^n_{\max}$, is [28], [63]:

$$x(k) = \bigoplus_{\mu=\mu_{min}}^{\mu_{max}} A^{(\ell(k))} \otimes x(k-\mu) \oplus B^{(\ell(k))} \otimes r(k), \tag{2.30}$$

where the superscript $\ell(k)$ describes what mode the matrix is in. Matrices with different $\ell(k)$ may have entirely different entries. The system at the end of Subsubsection 2.2.2.1 may be described as:

$$x(k) = A^{(\ell(k))} \otimes x(k-1), \tag{2.31}$$

with:

$$A^{(1)} = \begin{bmatrix} o_1 + \tau_2 & o_2 + \tau_1 \\ o_1 + \tau_1 & \varepsilon \end{bmatrix}, \quad A^{(2)} = \begin{bmatrix} o_1 + \tau_3 & o_2 + \tau_1 \\ o_1 + \tau_1 & \varepsilon \end{bmatrix}. \tag{2.32}$$

When the floor manager switches worker 1 out for another worker, the system dynamics switch from $A^{(1)}$ to $A^{(2)}$.

2.2.3.2 Switching Mechanism

Switching is controlled by a switching function, $\Phi(k)$, and a switching variable vector $\boldsymbol{\phi}(k)$. This switching mechanism can depend on the states, $\boldsymbol{x}(k)$, the mode, $\ell(k)$, the inputs $\boldsymbol{r}(k)$, and switching control variables $\boldsymbol{v}(k)$. It is denoted by [28], [63]:

$$\boldsymbol{\phi}(k) = \Phi(\boldsymbol{x}(k-\mu)\ \forall \mu \in \{\mu_{min}, ..., \mu_{max}\}, \ell(k), \boldsymbol{r}(k), \boldsymbol{v}(k)), \quad \phi(k) \in \mathbb{R}^{n_\phi}_{\max}, \tag{2.33}$$

where n_ϕ is the amount of entries of $\boldsymbol{\phi}(k)$, and Φ is a mapping function mapping the switching variables to $\mathbb{R}^{n_\phi}_{\max}$. $\mathbb{R}^{n_\phi}_{\max}$ can be divided into n_Φ subsets, with each subset $i \in \{1, 2, ..., n_\Phi\}$ represented by $\mathcal{L}^i$. Then, if $\phi(k) \in \mathcal{L}^i$, the mode switches to $\ell(k) = i$.

Switching may also be partially or entirely stochastic. (Nonlinear) model predictive control (MPC) can be used to stabilize and control SMPL systems under certain conditions, even if there is stochasticity [66].

A simple way to represent the switching mechanism is to simplify it to:

$$\boldsymbol{\phi}(k) = \boldsymbol{v}(k), \tag{2.34}$$

where the values of $\boldsymbol{v}(k)$ directly determine the mode of the system.

2.2.3.3 Max-Plus Representation of the Ship Placement Problem

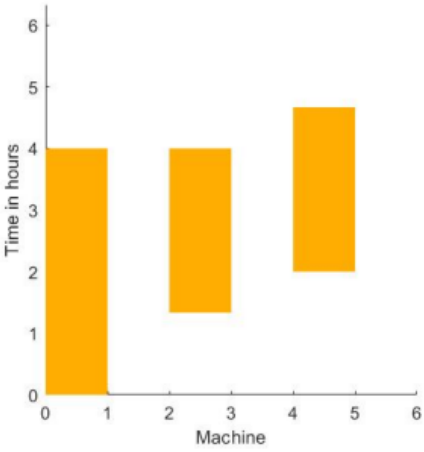
The author is not aware of any max-plus formulations of the SPP. However, the states of MPL models do not necessarily have to represent time, they may also represent space, so it could be possible to formulate the SPP in the max-plus framework. Falling Tetris blocks on a grid and job scheduling in a beer brewery have, for example, been modeled as an SMPL system [56].

Let the states $\boldsymbol{x}(k)$ represent the stacked up heights of each of the n_x columns of the playing field. Let $A^{(\ell(k))}$ represent the possible additions to the height of the columns of the playing field if a given combination of block, rotation, and position represented by mode $\ell(k)$ is allowed to occur. Then the Tetris system can be represented by an SMPL system, updating the column heights every block drop [56]:

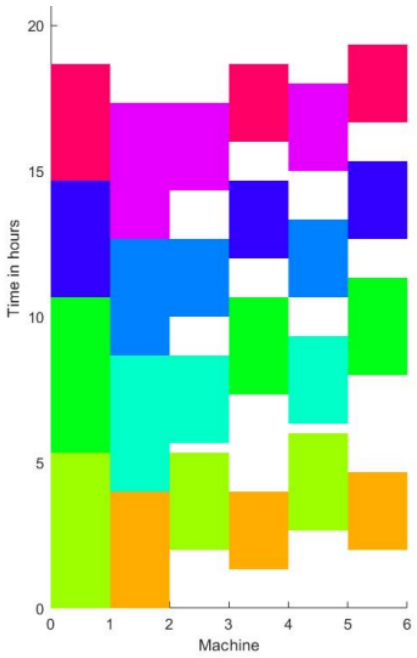
$$\boldsymbol{x}(k) = A^{(\ell(k))} \otimes \boldsymbol{x}(k-1) \tag{2.35}$$

The choice of position and rotation for the sequence of block drops can then be optimised to keep column heights minimal.

This modelling technique was further used in [56] to model the job assignment of different parts of the beer brewery process on the available machines in the brewery. Creating a batch of beer



(a) *Example of a beer brewing recipe block. Image from: [56].*



(b) *Example of a beer brewing schedule. Image from: [56].*

consists of multiple different steps on different machines. The steps have a given duration and sequence, which can be represented as (disconnected) Tetris blocks, see Figure 2.5a as an example. The objective was to create a schedule to complete all of the beer batches in a minimal amount of time. This problem formulation is closely analogous to a JSS problem [36].

A solution to the problem is shown in Figure 2.5b, where every color represents a different batch of beer. When turned on its side, this represents the Gantt charts often used to show JSS problem solutions. If vessels are given their own blocks to represent their size, and the time and machine axes are turned into discrete bins in which the vessels can be placed, this could be used as a start to try to model the SPP in the switching max-plus domain.

This, at least, indicates that it may be possible to model the SPP as a separate problem in the max-plus domain, though it remains to be seen if a connection to the rest of the scheduling model can be made.

2.2.3.4 SMPL Applications in Literature

This subsection will give a small overview of some systems that have been modelled with a SMPL system, other than the previously mentioned thesis and paper on IWT scheduling, and the paper on Tetris blocks.

A model to handle delays in a train network was presented in [63]. Train networks are a good candidate for modelling as SMPL systems, as the trains run on a common schedule, with trains going every set number of minutes, being divisible into cycles corresponding to the event counter. A train that leaves in cycle k, may cause delays for other trains in cycle $k + 2$. The delays can be minimized by taking control actions, like switching trains to parallel tracks or switching train orders. The size of the problem called for distributed optimization, which will be examined further in Subsubsection 2.2.4.5.

Another researched application of SMPL systems is gaits and gait switching of multi-legged walking robots [67]. Walking is essentially cyclical, with the times at which the legs lift off and touch down being events. The rules for the time between touchdowns and lifts of the legs for a given gait can

be registered in a max-plus system matrix, which can be switched when different gaits are required. This was applied successfully in a real quadruped robot.

Predictions on the duration of the transient response when switching between gait modes can be made by determining the coupling time of the gait matrices, which is the number of cycles it takes to reach steady state behaviour [68].

Finally, other work has focused on printers, including stochastic effects in parameters and mode switching, optimally scheduling printer actions [69], and scheduling of equipment interactions in a container terminal [70].

2.2.4 IWT Scheduling with SMPL Systems

The addition of control inputs or control variables SMPL systems allows for control of these systems. A form of control is scheduling, where the states present themselves as a schedule that can be optimised according to a given set of criteria. Production steps in the example in Subsubsection 2.2.3.1 may be scheduled by optimising the switching to minimize $\boldsymbol{x}(k) - \boldsymbol{x}(k-1)$. The example production system may have a simple scheduling law that can be worked out analytically, but more complex systems will require the solution of an optimisation problem, usually in the form of an MILP.

In the general DES framework, explanations are often given from a job-task-resource perspective [30]. Jobs must be processed by the system, each job may consist of multiple different tasks, and tasks may be completed by occupying resources.

However, the explanations in this section will be given from a ship-processing-infrastructure perspective to stay true to the IWT subject matter of the subsequent research, and because of the unorthodox event counter, where k represents the vessel number. A system in this context is a network of waterways and infrastructure, with ships being the jobs, infrastructure, like locks, being the resource, and ships being processed by infrastructure to get to the final destination being the tasks.

The example network that will be used is shown in Figure 2.6. 2 vessels make their way from start to end. The set of vessels is given by $\mathcal{K}$, which consists of $n_{vessels}$ vessels. State $x_i(k)$ represents the arrival time of vessel k at infrastructure item i, $\boldsymbol{x}(k) = [x_1(k), x_2(k), ..., x_{n_I}(k)]^T$, and $\boldsymbol{x} = [x(1), x(2), ..., x(n_{vessels})]^T$. Vessels pass through infrastructure items as soon as they arrive. The set of infrastructure items I consists of n_I items. Lock i has processing time o_i. The waterway leading up to infrastructure item i has travel time $\tau_{k,i}$ for vessel k.

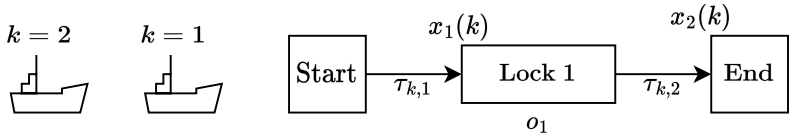


Figure 2.6: *Example IWT network scheduling task, consisting of 2 vessels and 1 lock.*

The control decision constraints can commonly be grouped under the following three categories:

- Routing
- Ordering
- Synchronisation

Routing, ordering, and synchronisation constraints are described with an accompanying example in Subsubsection 2.2.4.1, Subsubsection 2.2.4.2, and Subsubsection 2.2.4.3, respectively. The full

constraint set is described Subsubsection 2.2.4.4. Finally, some comments on the structure of the final scheduling matrix are made in Subsubsection 2.2.4.5. The explanations and mathematical formulations in this section are based on [30], with the extension to a wider range of possible values of μ from [63].

2.2.4.1 Routing

The primary source for the contents of this subsubsection is [30]. Routing is the decision of the sequence in which the job is processed by the resources, or in this case the decision of the sequence in which vessel k will pass through the infrastructure pieces. A routing component can be introduced by adding another route to the system, like depicted in Figure 2.7. Lock 1 is expanded to

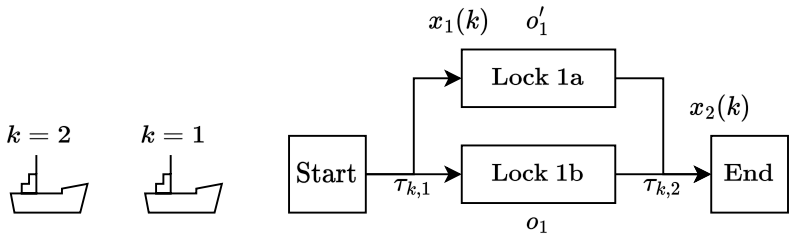


Figure 2.7: *The example IWT system with an additional chamber added to lock 1.*

2 chambers, lock 1a and 1b. To reduce the amount of modes available, the assumption is made that if vessel 1 goes through lock 1a, vessel 2 must pass through lock 1b ($\ell = 1$), and vice versa ($\ell = 2$), where ℓ is the routing mode. The routing constraint matrices, $A^{(\ell)}_{routing}$, for both modes are:

$$A^{(1)}_{routing} = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \tau_{1,1} + o_1 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & & \varepsilon \\ \varepsilon & \varepsilon & \tau_{1,2} + o_1' & \varepsilon \end{bmatrix}, A^{(2)}_{routing} = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \tau_{1,1} + o_1' & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & & \varepsilon \\ \varepsilon & \varepsilon & \tau_{1,2} + o_1 & \varepsilon \end{bmatrix} \tag{2.36}$$

For n_{routes} routes, with mode $\ell \in \{1, 2, ..., n_{routes}\}$, control variable $v^{(\ell)}$ decides if route ℓ is taken. If $v^{(\ell)} = e$, then route ℓ is taken, and $v^{(i)} = \varepsilon \quad \forall i \neq \ell$. $v^{(\ell)}$ is a max-plus algebraic binary variable, if normal binary variables are represented by δ, then [30]:

$$v = \begin{cases} e & \text{Where } \delta \text{ would be 1} \\ \varepsilon & \text{Where } \delta \text{ would be 0} \end{cases}. \tag{2.37}$$

The adjoint of v, denoted by $\overline{v}$, has the following definition:

$$\overline{v} = \begin{cases} \varepsilon & \text{If } v = e \\ e & \text{If } v = \varepsilon. \end{cases} \tag{2.38}$$

Define $\mathbb{B}_{max}$ as the set $\{e, \varepsilon\}$. Which is the set of different values that max-plus binary variables may take on, then let $\boldsymbol{v} = [v^{(1)}, ..., v^{(n_{routes})}]^T$ represent the vector of routing control variables, with $\boldsymbol{v} \in \mathbb{B}^{n_{routes}}_{max}$. The routing constraints matrix is then written as [30]:

$$A_{routing}(\boldsymbol{v}) = \bigoplus_{\ell=1}^{n_r} v^{(\ell)} \otimes A^{(\ell)}_{routing} \tag{2.39}$$

$$\boldsymbol{x} \geq A_{routing}(\boldsymbol{v}) \otimes \boldsymbol{x} \tag{2.40}$$

This example also served to illustrate a way in which multiple chambers could be added to the IWT max-plus scheduling model by adding extra routing constraints and system modes.

In the case of a traditional job-task-resource system, with k being an event counter, the routing constraints are given by:

$$\boldsymbol{x}(k) \geq \bigoplus_{\mu \in \{\mu_{min}, \dots, \mu_{max}\}} A_{routing,\mu}(k) \otimes \boldsymbol{x}(k-\mu), \tag{2.41}$$

where:

$$A_{routing,\mu}(k) = \bigoplus_{\ell=1}^{n_r} v^{(\ell)}(k) \otimes A^{(\ell)}_{routing,\mu}(k), \tag{2.42}$$

$v^{(\ell)}(k)$ are the same type of control variable as in Equation 2.40, the value of which be different for different values of k, provided that if $v^{(\ell)}(k) = e$ then $v^{(i)}(k) = \varepsilon \quad \forall i \neq \ell$, and where $A^{(\ell)}_{routing,\mu}$ is the routing constraint matrix linking event step $k-\mu$ to event step k in system mode ℓ.

2.2.4.2 Ordering

The primary source for the contents of this subsubsection is [30]. Referring back to Figure 2.6, imagine the vessels can only pass through the lock after one another, with a separation of at least o_1. The two possibilities are: vessel 1 passes before vessel 2, or vessel 2 passes before vessel 1. Ordering constraints are required to enforce these possibilities.

Each route ℓ has a resource assignment matrix $P^{(\ell)}$ described by:

$$[P^{(\ell)}]_{ij} = \begin{cases} e & \text{If } x_i \text{ and } x_j \text{ take place on the same infrastructure} \\ \epsilon & \text{If } x_i \text{ and } x_j \text{ take place on different infrastructure} \end{cases} \tag{2.43}$$

Then, as both vessels pass through the lock, so there is only 1 available route:

$$P^{(1)} = \begin{bmatrix} \varepsilon & \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ e & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{bmatrix} \tag{2.44}$$

Then $P(q)$ is given by:

$$P(v) = \bigoplus_{\ell=1}^{1} v^{(\ell)} \otimes P^{(\ell)} \tag{2.45}$$

H is a separation time matrix, with $[H]_{ij}$ equal to the separation time between operations i and j if they are scheduled on the same resource, $[H^{(\ell)}]_{ij} = \varepsilon$ if operations x_i and x_j cannot be scheduled on the same resource. The separation matrix in this case is:

$$H = \begin{bmatrix} \varepsilon & \varepsilon & o_1 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ o_1 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{bmatrix} \tag{2.46}$$

Finally, let the entries of a matrix Z, the order decision matrix, be $[Z]_{ij} = e$ if operation x_i takes place after operation x_j, and $[Z]_{ij} = \varepsilon$ if operation x_j takes place after operation x_i. The entries of Z, z_{ij}, are decision variables in the scheduling problem. Let $\boldsymbol{z} \in \mathbb{B}^{n^2}_{max}$ be the vectorized version of Z. Finally, the constraint matrix $A_{order}(\boldsymbol{v}, \boldsymbol{z})$ becomes:

$$A_{ordering}(\boldsymbol{v}, \boldsymbol{z}) = P(\boldsymbol{v}) + H + Z(\boldsymbol{z}). \tag{2.47}$$

With the ordering constraint given by:

$$\boldsymbol{x} \geq A_{ordering}(\boldsymbol{v}, \boldsymbol{z}) \otimes \boldsymbol{x} \tag{2.48}$$

For example, if $z_{13} = 0$ and $z_{31} = \varepsilon$, so vessel 1 passes lock 1 after vessel 2, then:

$$x_1(1) \geq \max(\varepsilon, x_1(2) + o_1, \varepsilon, \varepsilon) = x_1(2) + o_1 \tag{2.49}$$
$$x_1(2) \geq \max(\varepsilon, \varepsilon, \varepsilon, \varepsilon) = \varepsilon. \tag{2.50}$$

For a general job-task-resource system, with k being an event counter, the routing constraints are the same, but dependency on k and a number of previous cycles must be taken into account. Matrix $P(v(k))$ is then described by:

$$P(\boldsymbol{v}(k)) = \bigoplus_{\ell(k)=0}^{n_{routes}} v^{(\ell(k))} \otimes P^{(\ell)}. \tag{2.51}$$

H may vary with k as $H(k)$. Let $Z_\mu(z_\mu(k))$ be the order decision matrix composed of the decision variables $z_{ij}(k-\mu)$, vectorized in $z_\mu(k)$. Finally, the order constraint matrices are given by:

$$A_{ordering,\mu}(\boldsymbol{v}(k), \boldsymbol{z}_\mu(k)) = P(k) + H(k) + Z_\mu(\boldsymbol{z}_\mu(k)), \tag{2.52}$$

and the ordering constraints are described by:

$$\boldsymbol{x}(k) \geq \bigoplus_{\mu \in \{\mu_{min}, \ldots, \mu_{max}\}} A_{ordering,\mu}(\boldsymbol{v}(k), \boldsymbol{z}_\mu(k)) \otimes \boldsymbol{x}(k-\mu). \tag{2.53}$$

2.2.4.3 Synchronisation

The primary source for the contents of this subsubsection is [30]. Now let both vessels be able to pass through the lock at the same time, or separately. A synchronization constraint is required to enforce this. There are 2 synchronisation modes, either the vessels pass through the lock separately $(\ell = 1)$, or the vessels pass through simultaneously $(\ell = 2)$. Let $A^{(\ell)}_{synch}$ be the synchronization constraint matrix for mode ℓ, then:

$$[A^{(\ell)}_{synch}]_{ij} = \begin{cases} e & \text{If operation } x_j \text{ must be scheduled after operation } x_i. \\ \varepsilon & \text{In other cases} \end{cases} \tag{2.54}$$

The synchronization constraint matrices are:

$$A^{(1)}_{synch} = \mathcal{E}(4,4), \quad A^{(2)}_{synch} = \begin{bmatrix} \varepsilon & \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ e & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{bmatrix}. \tag{2.55}$$

Let binary variable $s^{(\ell)}$ represent if synchronization mode ℓ is active, with there being n_{synch} synchronisation modes in total, and with $\boldsymbol{s} = [s^{(1)}, \ldots, s^{(n_{synch})}]^T$, $s \in \mathbb{B}^{n_{synch}}_{max}$. Then the synchronization constraint matrix $A_{synch}(\boldsymbol{s})$ becomes:

$$A_{synch}(\boldsymbol{s}) = \bigoplus_{\ell=1}^{n_{synch}} s^{(\ell)} \otimes A^{(\ell)}_{synch} \tag{2.56}$$

Finally, the synchronization constraint becomes:

$$\boldsymbol{x} \geq A_{synch}(\boldsymbol{s}) \otimes \boldsymbol{x}. \tag{2.57}$$

If $\ell = 1$, the constraint is trivial. But if $\ell = 2$, then:

$$x_1(1) \geq \max(\varepsilon, \varepsilon, x_1(2), \varepsilon) \tag{2.58}$$
$$x_1(2) \geq \max(x_1(1), \varepsilon, \varepsilon, \varepsilon), \tag{2.59}$$

enforcing that $x_1(1) = x_1(2)$, so both vessels arrive at the lock together, passing through it simultaneously. This example shows that chambers with a capacity for multiple vessels can be modeled with the help of synchronisation constraints.

In the general job-task-resource framework, with k being an event counter, synchronisation at cycle k also depends on different cycles. Let $s_\mu^{(\ell)}(k)$ be max-plus binary variables, let $\boldsymbol{s}_\mu(k) = [s_\mu^{(1)}(k), ..., s_\mu^{(n_{synch})}(k)]^T$ be the vector of those binary variables, and let $A_{synch,\mu}^{(\ell)}(k)$ be a synchronization matrix for mode ℓ, connecting event $k - \mu$ to event k, with:

$$[A_{synch,\mu}^{(\ell)}(k)]_{ij} = \begin{cases} e & \text{If operation } x_j \text{ for event } k \text{ must be scheduled after operation } x_i \text{ in event } k-\mu \\ \varepsilon & \text{In other cases.} \end{cases} \tag{2.60}$$

Then the synchronization constraints are described by:

$$\boldsymbol{x}(k) \bigoplus_{\mu \in \{\mu_{min}, ..., \mu_{max}\}} A_{synch,\mu}(\boldsymbol{s}(k)) \otimes \boldsymbol{x}(k-\mu), \tag{2.61}$$

with:

$$A_{synch,\mu}(\boldsymbol{s}(k)) = \bigoplus_{\ell=0}^{n_{synch}} s_\mu^{(\ell)}(k) \otimes A_{synch,\mu}^{(\ell)}(k). \tag{2.62}$$

2.2.4.4 Full Constraints

The primary source for the contents of this subsubsection is [30]. Once $A_{routing}$, $A_{ordering}$, and A_{synch} have been determined, the full set of scheduling constraints can, in general, be assembled as:

$$A_{scheduling}(\boldsymbol{\phi}) = A_{routing}(\boldsymbol{v}) \oplus A_{ordering}(\boldsymbol{v}, \boldsymbol{z}) \oplus A_{synch}(\boldsymbol{s}), \tag{2.63}$$

where:

$$\boldsymbol{\phi} = \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{z} \\ \boldsymbol{s} \end{bmatrix} \tag{2.64}$$

With the full scheduling constraints being described by:

$$\boldsymbol{x} \geq A_{scheduling}(\boldsymbol{\phi}) \otimes \boldsymbol{x}. \tag{2.65}$$

For the job-task-resource system, the constraint matrix becomes:

$$A_{\mu,scheduling}(\boldsymbol{\phi}(k)) = A_{routing,\mu}(\boldsymbol{v}(k)) \oplus A_{ordering,\mu}(\boldsymbol{v}(k), \boldsymbol{z}_\mu(k)) \oplus A_{synch,\mu}(\boldsymbol{s}_\mu(k)), \tag{2.66}$$

with:

$$\boldsymbol{\phi}(k) = \begin{bmatrix} \boldsymbol{v}(k) \\ \boldsymbol{z}_{\mu_{\min}}(k) \\ \vdots \\ \boldsymbol{z}_{\mu_{\max}}(k) \\ \boldsymbol{s}_{\mu_{\min}}(k) \\ \vdots \\ \boldsymbol{s}_{\mu_{\max}}(k) \end{bmatrix}. \tag{2.67}$$

The final scheduling constraints become:

$$\boldsymbol{x}(k) \geq \bigoplus_{\mu \in \{\mu_{min}, ..., \mu_{\max}\}} A_{scheduling,\mu}(\boldsymbol{\phi}(k)) \otimes \boldsymbol{x}(k-\mu). \tag{2.68}$$

2.2.4.5 Structure of the $A_{scheduling}$ Matrix

The fairly simple examples used in this section would lead to a 4×4 $A_{scheduling}$ matrix. For the rest of this subsection $A_{scheduling}$ will be denoted A. However, once the size of the problem is increased, the number of variables may increase exponentially.

To reduce the computation time, the structure of the A matrix may be exploited for distributed optimization, as was done for a large train scheduling problem in [63]. By shuffling the rows and columns correctly, a close to diagonal matrix structure, like shown in Figure 2.8, can be achieved if the structure of the problem is suitable.

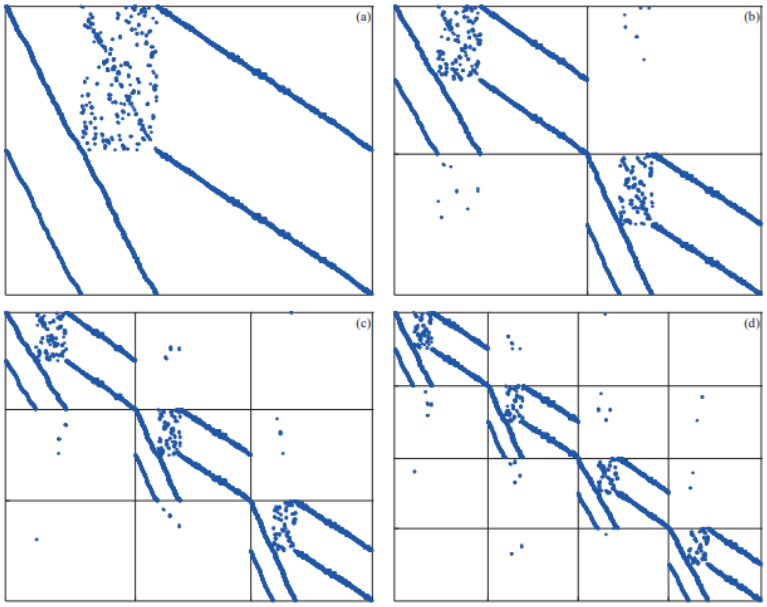


Figure 2.8: *Structure of the constraint matrices of a large train scheduling problem, split into smaller subsystems represented by block matrices on the diagonal, interconnected by block matrices on the off-diagonal. Image from: [63].*

The problem can then be split into several subsystems, each with a block matrix on the diagonal of the larger A matrix, and off-diagonal block matrices connecting them to other subsystems. A distributed optimization problem can then be solved by considering each main subsystem block matrix as a local problem, connecting the solutions of the local problems through the off-diagonal connection matrices.

This structure may also apply to IWT scheduling problems. Vessels traveling the same route or similar routes could be grouped into a subsystem, making the problem solvable in a distributed manner, and the found substructures may reveal important corridors or logical places for regional splits to be made when trying to coordinate IWT traffic.

2.2.5 Scheduling MILP

For the scheduling problem to be solved, it must first be transformed to the domain of regular algebra. The most general structure for this is the MILP, with a mix of scalar, integer, and binary variables [71]. The recasting to an MILP is also used for model predictive control of MPL and SMPL systems [72] [66]. MILP solvers cannot use ε, furthermore, the maximization operator cannot directly be used in a linear programming problem, though it can be implemented through constraints and auxiliary variables.

2.2.5.1 MILP

MILP problems can generally be written in the following form [71] [17]:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{\phi}} \quad & J = \boldsymbol{c_x}^T \boldsymbol{x} + \boldsymbol{c_\phi}^T \boldsymbol{\phi} \\ \text{s.t} \quad & A_x \boldsymbol{x} + A_\phi \boldsymbol{\phi} \leq \boldsymbol{f} \\ & \boldsymbol{x} \geq 0 \\ & \boldsymbol{\phi} \in \mathbb{Z}^{n_\phi} \end{aligned}$$

where $\boldsymbol{x} \in \mathbb{R}^{n_x}$ is the vector of continuous decision variables, $\boldsymbol{\phi} \in \mathbb{Z}^{n_\phi}$ is the vector of integer decision variables, $\boldsymbol{c_x} \in \mathbb{R}^{n_x}$ is the objective function vector of the continuous variables, $\boldsymbol{c_\phi} \in \mathbb{R}^{n_\phi}$ is the objective function vector of the integer decision variables, $A_x \in \mathbb{R}^{m \times n_x}$ is the continuous variable constraint matrix, A_ϕ is the integer variable constraint matrix and $\boldsymbol{f} \in \mathbb{R}^m$ is the scalar inequality constraint vector. Integer decision variables may include binary decision variables, which are in $\mathbb{B} = \{0, 1\}$, with a binary variable with the value 0 being in the false state and a variable with the value 1 being in the true state.

2.2.5.2 Conversion of Binary Variables

Let ϑ be a max-plus binary variable, with $\overline{\vartheta}$ being its max-plus adjoint. Then ϑ can be approximated in conventional algebra for inequality constraints as [30]:

$$\begin{array}{ccc} \text{Max-plus Algebra} & \longleftrightarrow & \text{Conventional Algebra} \\ \vartheta & & \beta(1-\vartheta) \\ \overline{\vartheta} & & \beta\vartheta \\ \vartheta, \overline{\vartheta} \in \mathbb{B}_{max} & & \vartheta \in \mathbb{B} \end{array} \tag{2.69}$$

where β is a large enough negative scalar, such that $\beta \approx \varepsilon$. There are cases where the conventional algebra version of a max-plus-binary variable is used without being multiplied by β, for instance in equality constraints, and when binary variables have to be counted in an inequality constraint.

2.2.5.3 Appropriate β Values

The use of β is a form of big-M notation. An example of this type of notation is the constraint:

$$x \leq M\vartheta, \tag{2.70}$$

where x is a continuous variable, M is a large scalar, and ϑ is a binary variable. The goal of this constraint is to enforce x to be smaller than 0 when ϑ is not true and to let x take on any value when ϑ is true. M is chosen to be a large scalar to enable x to take on any value that it could reasonably need to take. There are computational benefits to reducing the value of M to a reasonable bound, both in avoiding numerical problems where the coefficients of big-M constraints are much larger than other coefficients in the problem [73] and in reducing computation time by allowing for better relaxation of the problem while solving [74].

If a maximum allowable value of x is known through another constraint, then the M bound can be decreased to that. Otherwise, an M value that is small enough, but that does not influence the solution should be chosen. This line of thinking is equivalent when establishing the value of β, but β is now a negative scalar.

2.2.5.4 Maximization Constraint Conversion

Let $a_i, x \in \mathbb{R}_{\max}$. A constraint in the form of:

$$x \geq a_0 \oplus a_1 \oplus ... \oplus a_n = \max(a_0, a_1, ..., a_n) \tag{2.71}$$

Can be described by the following set of constraints [63]:

$$\begin{aligned} x &\geq a_0 \\ x &\geq a_1 \\ &\vdots \\ x &\geq a_n \end{aligned}$$

In words, as long as x is larger than all of the entries in the max operator, it will be larger than the maximum value. A constraint in the form of:

$$x \geq a_0 \otimes a_1 \otimes ... \otimes a_n \tag{2.72}$$

Can simply be described by:

$$x \geq a_0 + a_1 + ... + a_n \tag{2.73}$$

Inequality constraints that are a combination of $\oplus$ and $\otimes$ follow the same rules as established in Equation 2.71 and Equation 2.72, keeping in mind proper order of operations. The constraint:

$$x \geq a_0 \otimes a_1 \oplus a_2 \otimes a_3 \oplus ... \oplus a_{n-1} \otimes a_n, \tag{2.74}$$

converts to [63]:

$$\begin{aligned} x &\geq a_0 + a_1 \\ x &\geq a_2 + a_3 \\ &\vdots \\ x &\geq a_{n-1} + a_n \end{aligned}$$

Note that this is all in $\mathbb{R}_{\max}$. A conversion could be made to $\mathbb{R}$ by setting any ε equal to a sufficiently large negative value.

A constraint of the form:

$$x = a_0 \oplus a_1 \oplus ... \oplus a_n, \tag{2.75}$$

can be linearized by introducing the auxiliary binary variables λ_i with the constraint [75]:

$$\sum_{i=0}^{n} = \lambda_i, \tag{2.76}$$

alongside the following constraints for all $i \in \{0, 1, ..., n\}$:

$$\begin{aligned} x &\geq a_i \\ x &\leq a_i - \beta(1 - \lambda_i). \end{aligned}$$

2.2.5.5 Linearization of Variable Products

Let ϑ be a binary variable, and let x be a continuous variable. Then their product:

$$\theta = \vartheta x, \tag{2.77}$$

can be linearized with the following four constraints [76]:

$$\theta \leq M_{\max}\vartheta \tag{2.78}$$
$$\theta \geq M_{\min}\vartheta \tag{2.79}$$
$$\theta \leq x - M_{\min}(1 - \vartheta) \tag{2.80}$$
$$\theta \geq x - M_{\max}(1 - \vartheta). \tag{2.81}$$

$M_{\max}$ and $M_{\min}$ are maximum and minimum possible values of x, respectively, though they may be taken to be larger bounds.

Let ϑ_1 and ϑ_2 both be binary variables. Then their product:

$$\vartheta_3 = \vartheta_1 \vartheta_2, \tag{2.82}$$

can be linearized with the following three constraints [76]:

$$\begin{aligned} -\vartheta_1 + \vartheta_3 &\leq 0 \\ -\vartheta_2 + \vartheta_3 &\leq 0 \\ \vartheta_1 + \vartheta_2 - \vartheta_3 &\leq 0 \end{aligned}$$

2.2.5.6 Objective Conversion

An objective with $\otimes$ in it can be converted by simply substituting $+$ for $\otimes$. However, conversion for $\oplus$ is more involved. Let the objective be:

$$\min_x x_1 \oplus x_2 \oplus ... \oplus x_n = \min_x \max(x_1, x_2, ..., x_n) \tag{2.83}$$

Then introducing the variable $t = \max(x_1, x_2, ..., x_n)$, a linear representation of this objective is [77]:

$$\begin{aligned} &\min_t t \\ &\text{Subject to:} \\ &t \geq x_1 \\ &t \geq x_2 \\ &\vdots \\ &t \geq x_n. \end{aligned}$$

This substitution technique generalizes to more complex formulations by adding more t variables.

2.2.6 Conclusion

The answers to the two max-plus related research questions are discussed in this section.

> *How can additions to the existing SMPL IWT scheduling model be made within the SMPL framework?*

Max-plus algebra, and specifically SMPL systems, lend themselves well to being used in the problem of IWT scheduling, where the required arrival time at infrastructure is a function of the maximum of a number of values, like travel time. To set up a scheduling model in max-plus, first the routing, ordering, and synchronisation constraints must be defined in max-plus algebra, then an objective must be defined, and lastly, the set of constraints and the objective must be converted to an MILP which can be solved with a solver like Gurobi.

The previous model took into account single chamber locks which can only handle a single vessel at a time. The problem of having multiple chambers could be solved by adding an additional route with proper routing constraints. Additionally, multiple vessels passing through the chambers at the same time could be managed with proper synchronisation and ordering constraints. These two additions would allow for a more realistic representation of real waterways while staying within the SMPL framework. Other constraints on priority at locks, or narrow fairways where overtaking is impossible, could ,for instance, be modelled with ordering constraints.

Furthermore, a subgoal of the subsequent research is to provide a general description of how to convert networks of infrastructure into optimization models. The use of this conversion language will allow for more complex models to be examined, but this will come at a cost of computation time in the eventual MILP. Distributed optimization by exploiting the structure of the constraint matrix, can be employed to try to reduce the complexity of the resulting optimization problems. Moreover, the structure of the constraint matrix could be examined to identify logical groupings for distributed geographical regions in a national, or even international, IWT coordination network.

How can the SPP be formulated in the SMPL framework?

To the best of the author's knowledge, no SMPL formulation of the SPP exists, however, previous formulations of Tetris and beer brewery job scheduling in the max-plus domain give confidence that problems where placement of blocks, or objects that can be approximated by blocks, like vessels, can be formulated as SMPL systems. Bin packing, the less constrained version of the SPP, would be very similar to the beer brewery formulation, but with the axes being composed of height and width bins of lock chambers. Approaching the SPP like the Tetris and beer brewery examples also presents a new approach to looking at the SPP, which may give new insights or improve solution times.

3

Single-vessel, Single-chamber Scheduling for Arbitrary Network Topologies

A scheduling model for a limited number of graph topologies with single-chamber locks that can only process a single vessel at a time was introduced in [31] and [17]. However, a general mathematical model to formulate scheduling optimization problems for networks with arbitrary topologies was not established. Furthermore, these networks were limited to only two origin and destination locations, while larger networks will need to be able to include the effects of vessels traveling to and from many different locations.

The benefit of the model being formulated for arbitrary network topologies is that it will save a significant amount of time when building schedulers for large-scale networks.

The first goal in this chapter is then to generalise the single-vessel, single-chamber lock scheduling problem formulation procedure so it can be used for networks with arbitrary topologies, and so that it can serve as a base to be expanded on with new model features in the models of later chapters.

Thus, the research question of this chapter is:

> *How can the SMPL IWT scheduling model be formulated for networks with arbitrary network topologies?*

Section 3.1 gives a problem definition of the scheduling problem of the single-vessel, single-chamber scheduling model. Subsequently, the approach to establishing the model and scheduling optimization problems is discussed in Section 3.2. Then, the SMPL constraints and the SMPL system of the mathematical model are discussed in Section 3.3 and Section 3.4. The SMPL constraints are converted to MILP constraints in conventional algebra, and objectives are added in Section 3.5. Section 3.6 highlights some recommendations and considerations for a future online optimization implementation of the scheduling models. Finally, the research question is answered in the conclusion in Section 3.7.

3.1 Problem Definition

This section discusses the problem definition according to which the single-vessel, single-chamber mathematical scheduling model and the scheduler will be formulated and built. It should be highlighted that this problem definition is for offline scheduling, meaning that the scheduling problem is solved once at the start of the timeline and the vessels and infrastructure are expected to follow the schedule perfectly.

The relevant sets of objects, graph structure used, variables to be optimized, parameters required as input, objectives used, and assumptions made are listed in Subsection 3.1.1, Subsection 3.1.2, Subsection 3.1.3, Subsection 3.1.4, Subsection 3.1.5, and Subsection 3.1.6, respectively.

3.1.1 Sets

A waterway network is modeled as a directed graph $\mathcal{G}$, with a set of nodes $\mathcal{N} = \{0, 1, ..., n_{nodes} - 1\}$ and a set of directed arcs $\mathcal{D}$ connecting the nodes. An arc may only be traveled in one direction. A special subset of arcs $\mathcal{D}_{locks} \subset \mathcal{D}$ represents the lock chambers in the network. The arrival times at nodes and routes of a set of vessels $\mathcal{K} = \{0, 1, ..., n_{vessels} - 1\}$ have to be optimally scheduled on graph $\mathcal{G}$, according to some objective. $\mathcal{M}(k)$ is the set of values that integer μ can take on for vessel k such that $k - \mu \in \mathcal{K} \setminus \{k\}$. $k - \mu$ is used to refer to vessels other than vessel k in a context where vessel k is the main subject of a variable or constraint. For instance, if the state evolution of a state of vessel k is a function of another vessel $k - \mu$'s states. Finally, $\mathcal{S}(i)$ is the set of nodes connected to node i via any arc.

3.1.2 Graph

The structure of an example graph with two locks is shown in figure Figure 3.1. Regular arcs are shown with solid black arrows. It is defined that if arc (i, j) is an arc in the graph, then (j, i) must also be an arc in the graph. To prevent clutter, the arcs are drawn with double-headed arrows, representing both arc (i, j) and (j, i). Lock chamber arcs are drawn as dashed arrows. A single-chamber lock is represented by a special construction of four nodes, as shown in Figure 3.2. Nodes $\delta(i)$ and $\delta(j)$ are the entry/exit points of the waiting areas, and nodes i and j are the entry/exit points of the lock chamber.

Multi-chamber locks have more nodes and arcs per lock. They are discussed in Chapter 4.

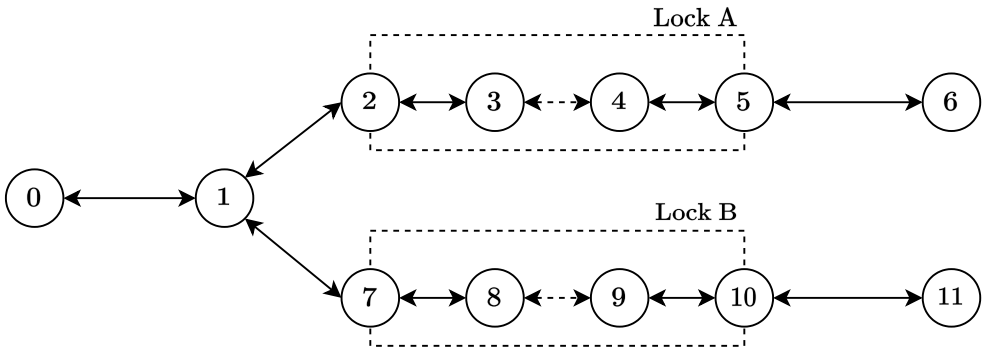


Figure 3.1: *Example graph with two locks.*

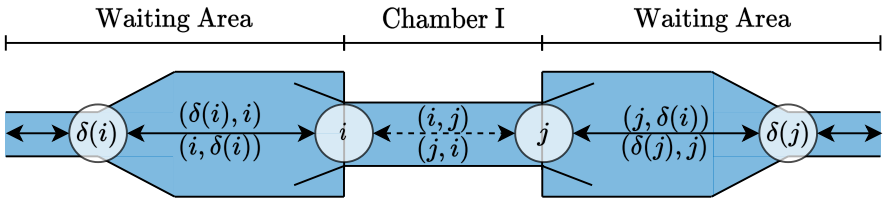


Figure 3.2: *Construction of nodes that represents a lock single-camber lock.*

3.1.3 Variables

The values of the variables are found by the scheduler when it is solving for a schedule. $x_i(k)$ are state variables, also referred to as arrival times, which exist for each combination of node and vessel $(i, k) \in \mathcal{N} \times \mathcal{K}$. $x_i(k)$ is equal to the arrival time of vessel k at node i. $w_{(i,j)}(k)$ are routing decision variables, which exist for each combination of arc and vessel $((i, j), k) \in \mathcal{D} \times \mathcal{K}$. The value of $w_{(i,j)}(k)$ determines whether or not vessel k travels along arc (i, j) on its route.

The interpretations of these arrival time and routing variables are shown in Figure 3.3. Finally, $z_{(i,j)}(k, k-\mu)$ are ordering decision variables, which exist for all combinations of lock chamber arcs (i,j) and combinations of a vessel k with another vessel $k-\mu$, $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$. $z_{(i,j)}(k, k-\mu)$ determines whether or not vessel $k-\mu$ gets to pass lock chamber arc (i,j) before vessel k, as shown in Figure 3.4.

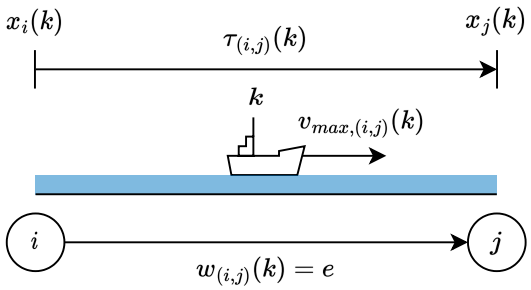


Figure 3.3: *Schematic of a vessel traveling arbitrary non-lock chamber arc* (i,j).

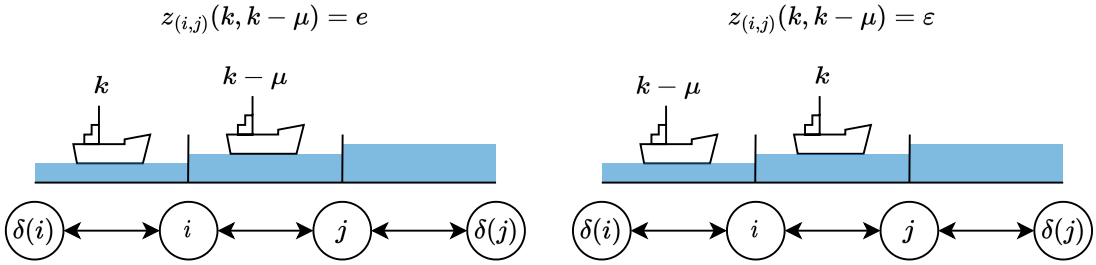


Figure 3.4: *Schematic of ordering between vessels* k *and* $k-\mu$ *on a lock arc* (i,j).

3.1.4 Parameters

Parameters are inputs to the scheduler that are known before solving for a schedule. Each node $i \in \mathcal{N}$ has positions $p_{x,i}$ and $p_{y,i}$ on an $x-y$ plane, which are used to determine the distances between nodes. Nodes that belong to the same lock are each given the same position, neglecting the relatively small distances between different sections of a lock. Each arc $(i,j) \in \mathcal{D}$ has a distance $d_{(i,j)}$, which is the Euclidean distance between the nodes at the ends of the arc, as shown in figure Figure 3.5.

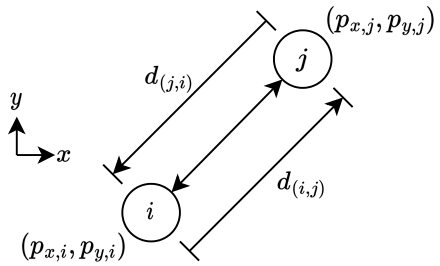


Figure 3.5: *Nodal positions and distance between nodes* i *and* j.

Each lock chamber arc $(i,j) \in \mathcal{D}_{locks}$ has a minimum operation time $\tau_{(i,j)}(k)$, and a corresponding waiting area entry node $\delta(i)$. The minimum operation time is the minimum amount of time it takes

for a lockage operation of that chamber, including the time it takes to position and moor a vessel before the operation and the time it takes for the vessel to exit the chamber after the lock operation. The minimum operation time is assumed to be equal for all vessels, but the notation referring to the vessel index gives room to deviate from that in the future by simply giving different operation time inputs for each vessel. The minimum operation time must be set in such a way that it is always achievable. The waiting area entry node is the node that is the entry point of the waiting area before arc (i, j).

Waiting area arcs have a minimum traversal time $\tau_{(i,j)}(k)$, which is the minimum time it takes for a vessel to pass through the waiting area, even if it does not have to wait there. This is also assumed to not be vessel dependent, but it can be made vessel dependent in the future.

Note that $\tau_{(i,j)}(k)$ is used to refer to regular arc travel time, minimum lock operation time, and minimum traversal time. The general definition of $\tau_{(i,j)}(k)$ can be taken to be the minimum time for vessel k to get from node i to j on arc (i, j). The minimum time is assumed to always be achievable. The way that time is calculated is just different depending on the nature of the system component arc (i, j) represents. This prevents clutter with different symbols and constraint sets having to be introduced for different arc types.

Each vessel $k \in \mathcal{D}$ has a maximum velocity $v_{\max,(i,j)}(k)$ on each arc $(i, j) \in \mathcal{D}$, a departure node $b(k)$, a destination node $d(k)$, a minimum departure time $u(k)$, and objective weights $\sigma(k)$. The maximum velocity is used to determine the minimum travel time $\tau_{(i,j)}(k)$ of vessel k along arc (i, j) as:

$$\tau_{(i,j)}(k) = \frac{d_{(i,j)}}{v_{\max,(i,j)}(k)}. \tag{3.1}$$

$\tau_{(i,j)}(k)$ is overridden by the minimum operation time on lock chamber arcs, and the minimum traversal time on waiting area arcs. The departure and destination nodes are the start and end points of the vessel's route. The minimum departure time is the earliest time at which the vessel can depart. Finally, the weights are used to assign different levels of importance to different vessels in the objectives.

3.1.5 Objective

The goal is to optimize the scheduling variables, which together form the schedule for vessels and locks to follow, such that the objective function is minimized. The main objective is either the cumulative travel time objective, penalizing the weighted sum of the travel times of all vessel, or the arrival time offset objective, penalizing the weighted sum of differences between the vessels' arrival times and their planned arrival times at their final destinations. Vessels can be assigned either one of the objectives. Different vessels are allowed to have different objectives, which are all summed in a multi-criteria objective function.

3.1.6 Assumptions

A number of simplifying assumptions are made to make this a manageable problem to model and solve, as modeling every single detail will result in a very complex model and it may significantly increase the time required to find scheduling solutions.

Some of the assumptions are used in this model because they were present in the original model in [17], and will remain during this first step of writing the model for use on general graphs. Some of them will be removed or relaxed in later chapters. A justification for why an assumption is made, or is necessary, is given in italics.

Vessel-related:

- **Vessels can overtake other vessels on all arcs [17].** *Due to a variety of circumstances, like waterway width, or there being a turn, vessels may not be able to overtake each other on an arc. These cases are not considered to reduce the complexity of the model. They could be added by adding an extra set of ordering constraints for arcs where overtaking is not allowed.*
- **Vessels maintain a constant speed on an arc [17].** *Vessels may have to vary their velocity when travelling on a given arc, because of multiple factors like changing water depth [13], or traffic. The scheduler only prescribes arrival times and no other information on the true velocity profile along the arc is known. Therefore, this assumption is made for visualization of the data.*
- **Vessel departure times are known beforehand [17].** *Vessels can, with some even being required to, register their voyage at the start of their voyage through the Dutch BICS system [23]. This represents a good starting point for the information of the vessel to enter the scheduler. Furthermore, departure times are always uncertain, so if the scheduler specifies them there is a high probability of that time not being achieved.*
- **Vessels are assumed to always be able to achieve their maximum velocity.** *Traffic and changing weather conditions can prevent vessels from being able to achieve their maximum velocity. The a priori model cannot account for this. Traffic could be included, at the cost of complexity, but that is left outside of the scope of this project. This and the previous assumption were derived from the "Sailing times of individual vessels for individual waterways are assumed to be known beforehand" assumption in [17].*
- **Vessels and locks cooperate fully with the scheduler's solution [17].** *Disturbances like delays may prevent the vessels and locks from following the solution schedule exactly. Some vessels may even decide to not cooperate with the schedule if it is not required by the regulations. This may lead to a better outcome for them but it will delay other vessels. However, the offline nature of the scheduler makes it so this cannot be taken into account when generating the solution.*
- **All vessels can traverse all arcs of the network.** *Different waterways are classified for different maximum vessel sizes, see Figure 1.3. Thus, vessels that are large can not necessarily travel on all waterways. For now this is not taken into account. In the future, a vessel's routing variables on arcs it would not be able to travel on can be turned off with constraints.*

Waterway-related:

- **The start and end points of a vessel's journey are nodes on the graph.** *Constraints are required to ensure departure and arrival of each vessel. These locations must be specifically defined as nodes to be able to determine these constraints. In real scenarios, any location a vessel's route may start or end at can just be added to the graph as a node.*
- **Any node may be a departure or destination node.** *This and the previous assumption were derived from the "All vessels navigate between the same two endpoints" assumption in [17].*
- **A route connecting a vessel's departure and destination nodes always exists.** *For the scheduler to be able to find a feasible solution, the destination of the vessel must be reachable by that vessel. In reality, vessels would not plan to have an unreachable destination. Derived from the "There exists more than one waterway connecting the endpoints" assumption in [17].*
- **Maximum velocities for each vessel on each arc are known beforehand.** *Maximum vessel velocities are dependent on many factors, like water depth, waterway width, wind, draft,*

stream velocity and more [13]. In short, they are hard to predict beforehand. However, as this scheduler does a priori offline optimization, it needs this information ahead of time.

Lock-related:

- **All locks are single-vessel, single-chamber locks.** *This was the main simplifying assumption in [17]. Real locks can have multiple independent chambers which can process multiple vessels at the same time. This assumption is kept for the model in this first step of building the model for arbitrary network topologies, and altered in subsequent chapters.*
- **Lock operations can start as soon as a vessel enters a lock.** *This is changed from the "Lock operations start as soon as a vessel enters a lock" assumption in [17]. The lock operation time already includes the vessel positioning, mooring and exiting the chamber, so there is no reason for a vessel to wait longer in the chamber in the single-vessel version of the mathematical model. However, vessels will have to be able to wait for other vessels to arrive in multi-vessel chamber capacity versions of the model, otherwise, all vessels have to arrive at the lock at the same time because the lockage would start as soon as they arrive.*
- **Waiting areas are assumed to have infinite capacity [17].** *In general, if vessels have to wait in the waiting areas for more than 15 minutes, they should moor. There is limited space available for this. However, these waiting areas have been designed to have enough waiting area capacity, even on the busiest days in the year [7]. It is thus unlikely that the waiting area gets too backed up. Furthermore, modelling a capacitated waiting area queue would add unnecessary model complexity.*
- **The run-out zone, waiting area, staging area, and funnel are modelled together as a single waiting area.** *The run-out zone, waiting area, staging area, and funnel, as depicted in Figure 1.5, all have distinct functions. Modelling all of these functions separately would clutter the topology graphs with extra nodes, and the staging area in particular, where vessels line up if they have been selected for the next lockage, would require adding more variables and constraints [7]. Instead, the lengths of all of the separate components are added up when determining the minimum traversal time for the waiting area, and the (minor) differences between the sections are neglected by modelling it all as a single large waiting area. This assumption was already made implicitly in [17], but it was not explicitly mentioned.*

3.2 Approach

This section outlines the approaches used to write the mathematical model and set up the scheduling problem.

Two processes are required to result in a scheduler that can schedule IWT systems with locks in different scenarios. First, the mathematical model with which the scheduling optimization problems are built is defined. The process of defining the mathematical model is outlined in Subsection 3.2.1. This only has to be done once. Once this mathematical model has been established, optimization problems following the rules of that framework are set up and solved by the scheduler to arrive at schedule solutions on a case-by-case basis, as explained in Subsection 3.2.2.

This project builds upon a previous project [31]. As a result, it is important to establish what the differences between the works are and what the novel contributions are. The smaller differences and similarities will be mentioned throughout the model chapters. As this chapter in particular makes the same key assumption of single-vessel, single-chamber locks as in [31], the main novel contributions of this chapter when compared to [31] are specifically highlighted in Subsection 3.2.3.

3.2.1 Establishing the Mathematical Model

A block diagram that describes the process of establishing the mathematical models is depicted in Figure 3.6. Establishing the mathematical model starts with defining the different variables, input parameters, the graph structure, the sets of items involved, and the assumptions made, as described in Section 3.1. Next, these defined components are used to define the different categories of SMPL constraints, as discussed in Section 3.3. The SMPL constraints are then combined into an SMPL system, as discussed in Section 3.4. The SMPL system is not used as input for MILP conversion, the SMPL constraints are sufficient for this. The SMPL system is just composed for completeness and to show that the SMPL constraints do form an SMPL system.

The final mathematical model has to be formulated as an MILP in conventional algebra, so it can be solved by a conventional solver like Gurobi. SMPL systems do not have equivalent solvers. As a result, the SMPL constraints must be converted to MILP constraints. Furthermore, objectives must be added to the optimization problem. The conversion of the constraints and the objectives are both discussed in Section 3.5. Finally, the output is a mathematical model which can be used to define the optimization problems for scheduling IWT systems with locks, given the required input information.

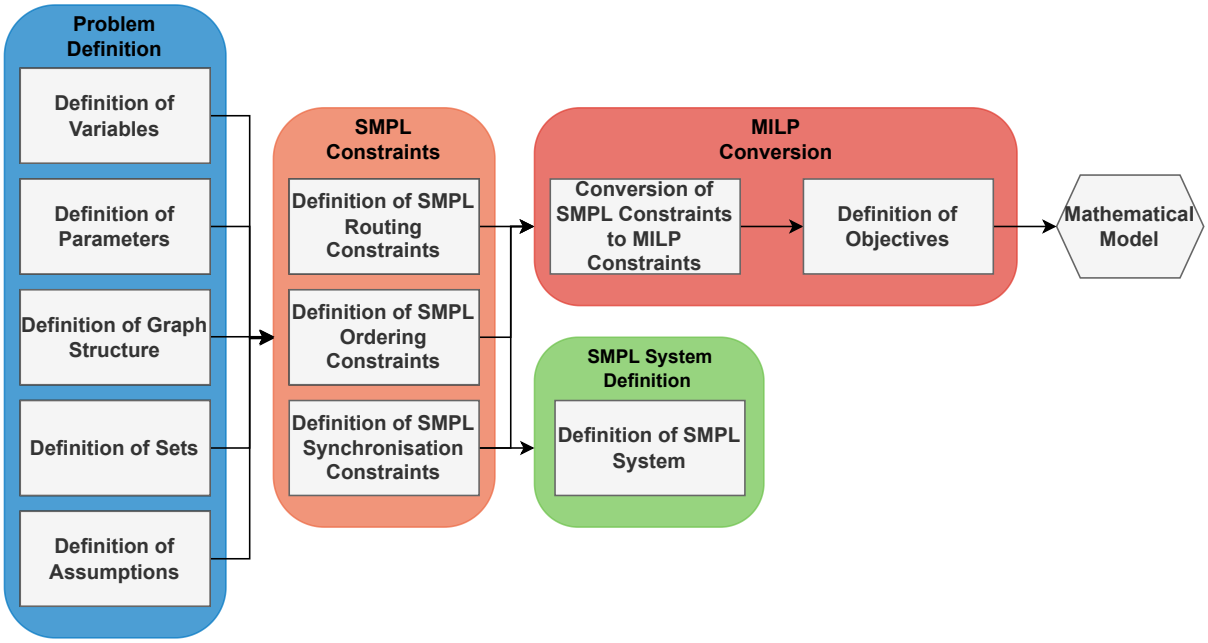


Figure 3.6: *The manually followed process of establishing the mathematical models.*

3.2.2 Creating and Solving Scheduling Optimization Problems

The mathematical model provides the rules for the creation of scheduling optimization problems. The scheduler, which refers to the program used to compose, solve, and post-process each of these scheduling optimization problems, follows a step-by-step process every time a schedule must be found for a given scenario.

This process is described in Figure 3.7. First, the information on vessels, the IWT system, and the objective weights must be input. Then the variables and object sets are generated. The supplied parameter values, graph topology, objective weight values, generated variables, and object sets are used to generate the constraints and objectives. Together all of these components form an optimization problem, for which an optimal solution is found with an MILP problem solver. The solution output is then post-processed to generate human-readable visualizations of the schedule

solution in the form of plots and to find the values of performance indicators that provide information on the quality and behavior of the solution.

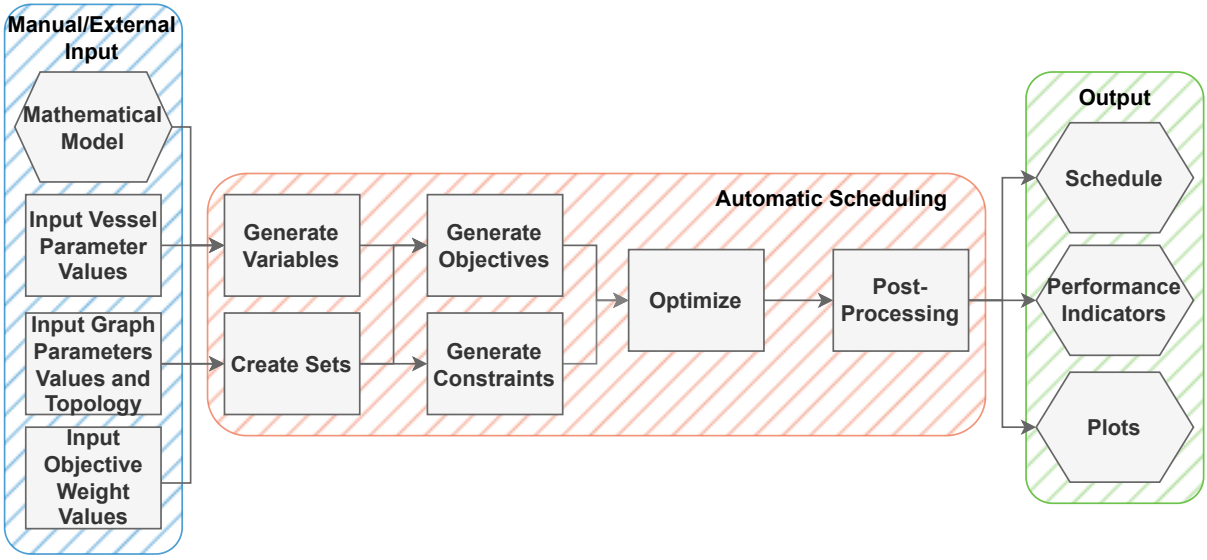


Figure 3.7: *Process the scheduler uses to solve for a schedule for a given scenario.*

The components that make up a schedule are the found decision variable values, as depicted in Figure 3.8. These decision variable values can be used to derive other quantities like prescribed vessel velocities along an arc or the performance indicator values.

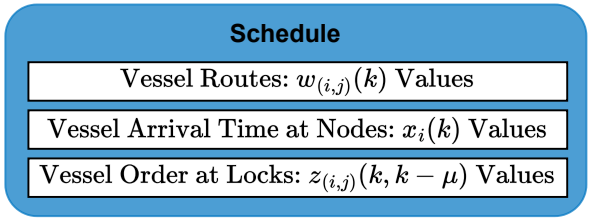


Figure 3.8: *The decision variables that make up a schedule for the single-vessel, single-chamber scheduling model.*

All steps, apart from the manual input step, are automated in Python. Thus, a new scheduling scenario only requires new input, with the scheduler performing all of the other steps. The solver used is the Gurobi 9.1.2 solver. In a real scenario, the input parameters could be automatically retrieved from stored map values for the graph, and real-time information from BICS, AIS, and RIS.

3.2.3 Novel Contributions

There are two main novel contributions for this single-vessel, single-chamber lock model compared to the model in [31]. The first is that the model is formulated for arbitrary network topologies. That means that the mathematical description applies to any graph topology that is used as an input. In [31], the constraints were written specifically for the network topologies that were being considered. As an example, the *travel time constraint* for a network as shown in Figure 6.1 for each vessel k

would have been written as a list of constraints tailored to that network:

$$x_1(k) \geq x_0(k) \otimes w_{(0,1)}(k) \tag{3.2}$$
$$x_0(k) \geq x_1(k) \otimes w_{(1,0)}(k) \tag{3.3}$$
$$\vdots \tag{3.4}$$
$$x_4(k) \geq x_5(k) \otimes w_{(5,4)}(k) \tag{3.5}$$

rather than the general mathematical description that applies to any network that will be given in Equation 3.14. An exception to this is the ordering constraints, those were already given a more general description.

The second main contribution is the fact that the routes the vessels take along the graph are no longer determined by a single variable. Instead, the routes of the vessels are determined by individually adding arcs into the route, activating each arc with an individual variable. This augments virtually every constraint, even requiring some new ones. The reasoning for this choice is given in Subsection 3.3.2.

Smaller novel items are, for instance, different parameter definitions, changed or new assumptions, and additional objectives like the arrival time offset objective.

3.3 SMPL Constraints

This section describes the SMPL constraints of the SMPL single-vessel, single-chamber lock scheduling model for arbitrary network topologies. Constraints of SMPL models can, in general, be split into three categories [30]. Adapted to this problem, these categories have the following interpretations:

- **Routing:** Constraints related to the ordering and occurrence of different events for each of the vessels.
- **Ordering:** Constraints related to the ordering of the use of the lock resources between multiple vessels.
- **Synchronisation:** Constraints related to the simultaneous use of lock resources by multiple vessels.

As this version of the model restricts each lockage to at most one vessel, synchronisation constraints are not necessary. Routing and ordering constraints are necessary, however, and are discussed in Subsection 3.3.2 and Subsection 3.3.3, respectively. Before that, Subsection 3.3.1 explains how max-plus logical constraints are established.

3.3.1 Max-Plus Logical Constraints

The models in this and the following chapters will heavily rely on logical constraints between binary variables. For example, let ξ_a and ξ_b be max-plus binary variables, then a logical constraint may require that at least one of ξ_a and ξ_b is true, or that one and at most one of the two must be true [76]. A general approach can be used for these cases. This approach will be shown by example.

Let ξ_c also be a max-plus binary variable. Let it be required that one, and at most one, of ξ_a, ξ_b, and ξ_c must be true. The following constraint constrains that at least one is true:

$$(\xi_a) \oplus (\xi_b) \oplus (\xi_c) = e. \tag{3.6}$$

However, in this case, all three of the variables may be true at the same time. To add the specification that at most one variable can be true, it must not only be specified that one variable is true in a

given scenario, but that all of the other variables must be false. This can be done as follows:

$$\underbrace{(\xi_a \otimes \overline{\xi}_b \otimes \overline{\xi}_c)}_{\text{Scenario 1}} \oplus \underbrace{(\overline{\xi}_a \otimes \xi_b \otimes \overline{\xi}_c)}_{\text{Scenario 2}} \oplus \underbrace{(\overline{\xi}_a \otimes \overline{\xi}_b \otimes \xi_c)}_{\text{Scenario 3}} = e, \tag{3.7}$$

where $\overline{\xi}_a, \overline{\xi}_b$, and $\overline{\xi}_c$ are the adjoints of their respective max-plus binary variables. Refer to Equation 2.38 for the definition of adjoint variables.

The general approach for a logical constraint is, thus, to define the different allowable combination scenarios of binary variable values and to then maximize over all of the allowable scenarios such that at least one is true. Multiple scenarios can be true at the same time depending on the way the scenarios are written, but in all cases, at least one scenario must be true for the optimization problem to be feasible. If ξ is listed in a scenario, then it must be true for that scenario to hold, if $\overline{\xi}$ is listed in a scenario, then ξ must be false for that scenario to hold, and if ξ is not listed in the scenario, then ξ may take on any value and that scenario will still hold.

3.3.2 Routing Constraints

Each vessel k in the set of vessels $\mathcal{K}$ has a departure node parameter $b(k)$, and a destination node parameter $d(k)$. A route must be established between these nodes and the scheduled arrival times at each of the nodes visited along the route must be constrained. Let the scheduling states variable $x_i(k)$ be the time at which vessel k arrives at node i. Furthermore, let the routing decision variable $w_{(i,j)}(k)$ be a max-plus binary decision variable with the following definition:

$$w_{(i,j)}(k) \begin{cases} e & \text{If vessel } k \text{ travels on arc } (i,j). \\ \varepsilon & \text{If vessel } k \text{ does not travel on arc } (i,j). \end{cases} \tag{3.8}$$

If $w_{(i,j)}(k) = e$, then arc (i,j) is active for vessel k, otherwise it is inactive.

3.3.2.1 Minimum State Value Constraint

By convention, it is chosen that arrival times cannot be negative. This also fulfills the requirement that all continuous variables are larger than or equal to 0 when converting to the MILP formulation according to the structure outlined in Subsection 2.2.5. For all $(i,k) \in \mathcal{N} \times \mathcal{K}$:

$$x_i(k) \geq e. \tag{3.9}$$

3.3.2.2 Departure Time Constraint

Each vessel k may only depart its departure node $b(k)$ after its scheduled minimum departure time parameter value $u(k)$. The state $x_i(k)$ is the time of arrival at node i for vessel k, but for a vessel to depart a node it must first have arrived at that node, and then it can depart any time after. Thus, for all $k \in \mathcal{K}$:

$$x_{b(k)} \geq u(k) \tag{3.10}$$

3.3.2.3 Route Start Constraint

Each vessel k must start its route by departing from its departure node $b(k)$ by traveling an arc $(b(k), i)$, where node i is in the set of nodes connected to node $b(k)$, $\mathcal{S}(b(k))$. The vessel cannot travel back to its departure node, so at most a single $w_{(b(k),i)}(k)$ can be true at a time. Equivalently, if routing variable $w_{(b(k),i)} = e$, then the adjoints of all other routing variables $w_{(b(k),j)}$, with j in the set of nodes connected to $b(k)$ excluding node i, must be equal to e. Thus, for all $k \in \mathcal{K}$:

$$\bigoplus_{i \in \mathcal{S}(b(k))} \left(w_{(b(k),i)}(k) \otimes \bigotimes_{j \in \mathcal{S}(b(k)) \setminus \{i\}} \overline{w}_{(b(k),j)}(k) \right) = e \tag{3.11}$$

3.3.2.4 Route End Constraint

Each vessel k must end its route by arriving at its destination node $d(k)$ by travelling an arc $(i, d(k))$, where node i is in the set of nodes connected to node $d(k)$, $\mathcal{S}(d(k))$. The vessel cannot travel away from node $d(k)$ after arriving at it, so the same procedure as applied to the *Route Start* constraint can be used. For all $k \in \mathcal{K}$:

$$\bigoplus_{i \in \mathcal{S}(d(k))} \left(w_{(i,d(k))}(k) \otimes \bigotimes_{j \in \mathcal{S}(d(k)) \setminus \{i\}} \overline{w}_{(j,d(k))}(k) \right) = e \tag{3.12}$$

3.3.2.5 Transit Constraint

The intermediate nodes that each vessel k visits on its route cannot be the stopping point of the route. Equivalently, if there is an active arc (j, i), with j in $\mathcal{S}(i)$, into node i for every node i in the set of nodes that are not end points of the route for vessel k, $\mathcal{N} \setminus \{b(k), d(k)\}$, then there must also be an active arc (i, l) with l in $\mathcal{S}(i) \setminus \{j\}$ out of node i, because the route cannot stop at a node that is not a destination node. At most one ingoing arc and one outgoing arc may be active at a time, as the node may not be visited twice. If there is no active arc into node i, then there may not be an active arc out of node i and vice versa.

If $w_{(j,i)}(k) = e$ and $w_{(i,l)}(k) = e$ then the routing decision variables for all other arcs with node i must be ε for vessel k, or the adjoints of all other arcs with node i must be e. Finally, there is the case that none of the arcs in or out of node i are active, so all of the adjoints of the routing variables of arcs with node i must be equal to e. Thus, for all $(k, i) \in K \times \mathcal{N} \setminus \{d(k), b(k)\}$:

$$\bigoplus_{j \in \mathcal{S}(i)} \bigoplus_{l \in \mathcal{S}(i) \setminus \{j\}} \left(w_{(j,i)}(k) \otimes w_{(i,l)}(k) \otimes \bigotimes_{p \in \mathcal{S}(i) \setminus \{j\}} \overline{w}_{(p,i)}(k) \otimes \bigotimes_{q \in \mathcal{S}(i) \setminus \{l\}} \overline{w}_{(i,q)}(k) \right) \oplus \left(\bigotimes_{j \in \mathcal{S}(i)} \overline{w}_{(j,i)}(k) \otimes \bigotimes_{l \in \mathcal{S}(i)} \overline{w}_{(i,l)}(k) \right) = e \tag{3.13}$$

3.3.2.6 Travel Time Constraint

If arc (i, j) is active for vessel k, then vessel k cannot instantaneously travel between node i and j, even when travelling at the maximum velocity $v_{\max,(i,j)}(k)$ for that arc. The arrival time $x_j(k)$ at node j must at least be the minimum travel time between the nodes (or the processing time for lock arcs) $\tau_{(i,j)}(k)$ larger than the arrival time $x_i(k)$ at node i, as shown in figure Figure 3.3. For all $(k, (i, j)) \in \mathcal{K} \times \mathcal{D}$:

$$x_j(k) \geq x_i(k) \otimes \tau_{(i,j)}(k) \otimes w_{(i,j)}(k) \tag{3.14}$$

Note that this constraint only actively does something if $w_{(i,j)}(k) = e$, otherwise it trivially defines that $x_j(k) \geq \varepsilon$.

The *Route Start*, *Route End*, and *Transit* constraints were not present like this in [31]. Instead, routing constraints were set up on a case-by-case basis for specific topology graphs on a set number of routes between the endpoints. This is convenient for small networks but may lead to an explosion in the number of variables in large networks. Refer to Figure 3.9 for an illustrative example. Assuming that each of the nodes can always be a start or end point, no matter how extra node 3 is connected, the number of routes always increases (much) faster than the number of arcs, for an already simple network.

In real scenarios, not every route will have to be explored, because only the routes connecting each of the vessels between their actual start and end points have to be considered, so the effect in

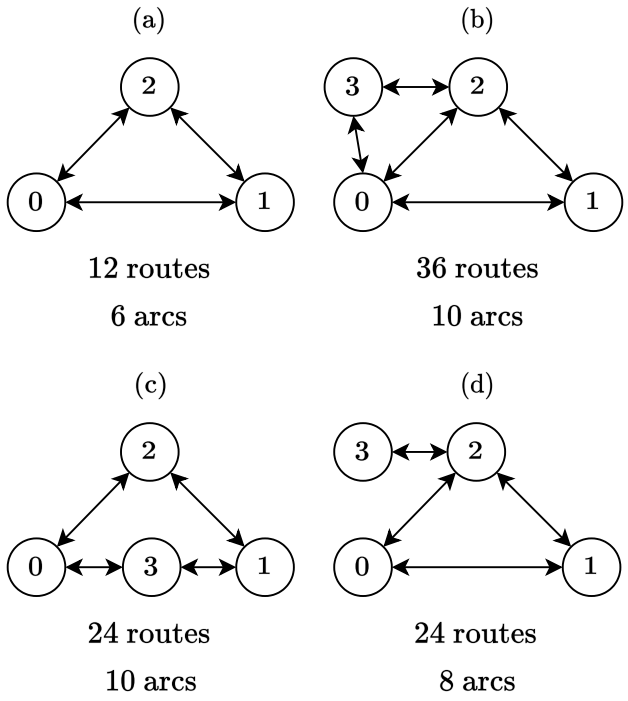


Figure 3.9: *Simple graph configurations with the number of possible routes and the number of arcs.*

Figure 3.9 may be less pronounced. However, as will become clear in the next subsection and later chapters, the ordering and synchronisation constraints have to be turned on and off with the routing variables. Effectively establishing those constraints for arbitrary network topologies becomes much more difficult if routing variables activate entire routes, rather than singular arcs. Furthermore, establishing if vessels are traveling in the same or opposite directions requires much more bookkeeping.

The *travel time* and *departure time* constraints are, however, adaptations of similar constraints in [31].

3.3.2.7 Sub-tour Elimination

The *Route Start*, *Route End*, and *Transit* constraints were inspired by, and are essentially identical in meaning to, the constraints for an MILP model of the classical shortest path problem, which is the problem of finding the shortest path between two nodes on a directed graph [78]. Together they ensure that the path between endpoints is free of loops and gaps and that it starts and ends at the defined endpoints.

However, solutions to the SPP, and thus possibly to this scheduling model, can suffer from unwanted sub-tours. Sub-tours are cycles that are not connected to the route between the vessel's start and endpoints, but that do fulfill the constraints. Figure Figure 3.10 displays a graph and a possible valid routing solution for a vessel traveling from node 0 to node 2.

Sub-tours will only occur if it is possible that the cost of adding an extra arc can be zero or negative in the objective function [78]. For instance, in the case that only the final arrival times of the vessels are penalized, other arcs for a sub-tour can be added without any additional cost in the objective function, which could present a problem.

A common solution for eliminating sub-tours is applying a Miller Tucker Zemlin (MTZ) constraint [79].

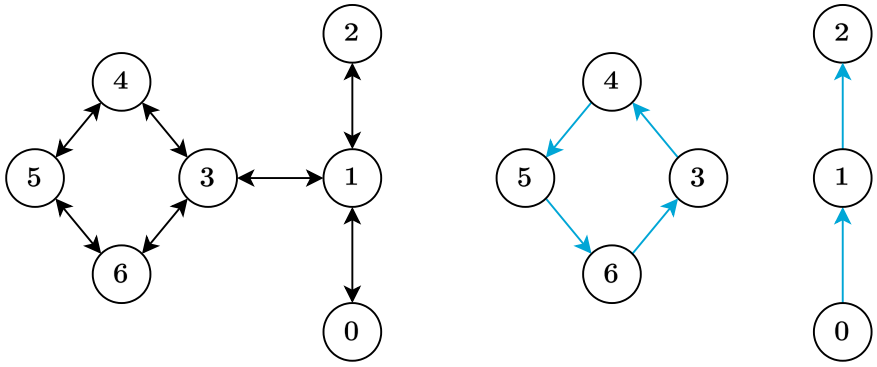


Figure 3.10: *A graph and a routing solution with a sub-tour from node 3 to node 3.*

A counter variable u_i is created for every vessel k in the set of vessels $\mathcal{K}$ for every node i in the set of nodes $\mathcal{N}$. If arc (i,j) is active for vessel k, then the MTZ constraint enforces that $u_j > u_i$. Then for the sub-tour in Figure 3.10, this would result in the impossible inequality $u_3 > u_6 > u_5 > u_4 > u_3$, preventing the sub-tour. This generalizes to all possible sub-tours.

With the knowledge of the MTZ constraint, it can be found that under certain conditions the *Travel Time* constraint will prevent sub-tours. If the minimum travel time $\tau_{(i,j)}(k)$ for all arcs for all vessels is strictly larger than zero, the *Travel Time* constraint does the same thing as an MTZ constraint.

Take the sub-tour in Figure 3.10, this gives $x_3(k) \geq x_6(k) \otimes \tau_{(6,3)}(k) \geq x_5(k) \otimes \tau_{(4,5)}(k) \geq x_4(k) \otimes \tau_{(3,4)} \geq x_3(k)$. If that is true, then $x_3(k) > x_6(k) > x_5(k) > x_4(k) > x_3$ must be true, which is impossible. Therefore, sub-tours are prevented as a consequence of the *Travel Time* constraint, provided that $\tau_{(i,j)}(k) > 0$ for all $((i,j),k) \in \mathcal{D} \times \mathcal{K}$, which will be ensured to always be the case in this report.

3.3.3 Ordering Constraints

Only a single vessel may pass through a lock at a time and each lock has an operation time, so a vessel following another vessel through a lock, or approaching the lock from the opposite direction, will have to wait for the lock to be available again. Ordering constraints are required to resolve these issues.

Let $z_{(i,j)}(k, k-\mu)$ be a max-plus binary ordering variable with the following definition:

$$z_{(i,j)}(k,k-\mu) \begin{cases} e & \text{If vessel } k-\mu \text{ travels lock arc } (i,j) \text{ before vessel } k. \\ \varepsilon & \text{If vessel } k-\mu \text{ travels lock arc } (i,j) \text{ after vessel } k. \end{cases} \tag{3.15}$$

$z_{(i,j)}(k, k-\mu)$ is defined for all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$ (for each vessel combined with any of the other vessels on every lock arc), and can be used to specify the order for different ships using the same infrastructure on arc (i,j), as shown in Figure 3.4.

The ordering variable is not only relevant for ensuring that vessels do not occupy the same lock at the same time but also for ensuring that there is sufficient spacing in time between different vessels. As overtaking is permitted on regular arcs, ordering constraints will not be defined for regular arcs.

Refer to Figure 3.11, which displays the two different ordering scenarios where vessel $k-\mu$ takes priority: one where vessels k and $k-\mu$ travel in the same direction, and one where they travel in opposite directions. In the case where they travel in the same direction, the $k-\mu$ vessel occupies the chamber in step (I), taking $\tau_{(i,j)}(k)$ to be processed and raised to the proper water level. Then once vessel $k-\mu$ has left the chamber in step (II), the water then has to be lowered again to the

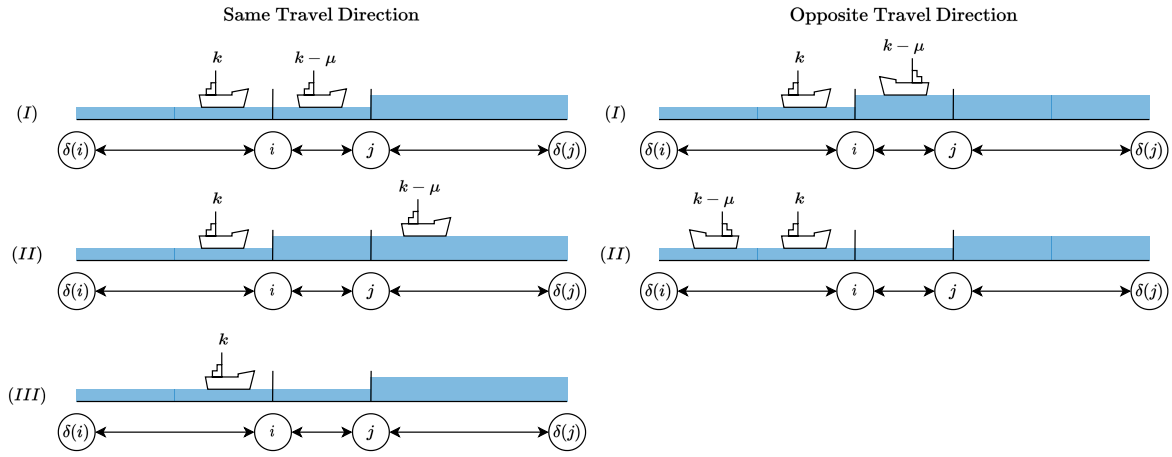


Figure 3.11: *Event sequences for vessels passing through a lock in the same and opposite travel directions.*

water level of vessel k taking another $\tau_{(j,i)}(k)$, vessel k can then start to be processed in step (III). In the case where the vessels travel in opposite directions, vessel $k-\mu$ is processed in step (I), taking $\tau_{(j,i)}$. Then, vessel k can immediately enter the chamber in step (II) after vessel $k-\mu$ has exited, as the water is already on the correct level. The scenarios where vessel k has priority are similar, with the order of vessels passing the lock being flipped.

3.3.3.1 Same Travel Direction Ordering Constraint

For each vessel k in the set of vessels $\mathcal{K}$, if any of the lock arcs (i,j) in the set of lock arcs $\mathcal{D}_{locks}$ are active through variable $w_{(i,j)}(k)$, and if $w_{(i,j)}(k-\mu)$ is active for any other vessel $k-\mu$ in the set of vessels $\mathcal{K}$, excluding vessel k, then if $z_{(i,j)}(k,k-\mu)$ is true, vessel k may only enter lock (i,j) at node i, after vessel $k-\mu$ has left the chamber at node j, and after enough time for at least one lockage process $\tau_{(j,i)}(k)$ has passed. Thus, for all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$x_i(k) \geq x_j(k-\mu) \otimes \tau_{(j,i)}(k) \otimes w_{(i,j)}(k) \otimes w_{(i,j)}(k-\mu) \otimes z_{(i,j)}(k,k-\mu). \tag{3.16}$$

Note that if any of the binary variables in this equation are false, the constraint trivially becomes inactivated, as $x_i(k) \geq \varepsilon$ is always true.

3.3.3.2 Opposite Travel Direction Ordering Constraint

For each vessel k in the set of vessels $\mathcal{K}$, if any of the lock arcs (i,j) in the set of lock arcs $\mathcal{D}_{locks}$ are active through variable $w_{(i,j)}(k)$, and if $w_{(j,i)}(k-\mu)$ is active for any other vessel $k-\mu$ in the set of vessels $\mathcal{K}$ in the set of vessels $\mathcal{K}$, excluding vessel k, then if $z_{(i,j)}(k,k-\mu)$ is true, vessel k may only enter lock (i,j) at node i after vessel $k-\mu$ has left lock (j,i) at node i. Thus, for all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$x_i(k) \geq x_i(k-\mu) \otimes w_{(i,j)}(k) \otimes w_{(j,i)}(k-\mu) \otimes z_{(i,j)}(k,k-\mu). \tag{3.17}$$

3.3.3.3 Ordering Variable Consistency Constraint

For the *Same Travel Direction Ordering* and *Opposite Travel Direction Ordering* constraints to be consistent, the relationship between different $z_{(i,j)}(k,k-\mu)$ must be constrained.

If $z_{(i,j)}(k,k-\mu)$ is true, then vessel $k-\mu$ has priority over vessel k on arc (i,j). The model only remains consistent if $z_{(i,j)}(k-\mu,k)$ is then false. Furthermore, for the opposite travel direction

constraints to remain consistent, if $z_{(i,j)}(k, k-\mu)$ is true, then $z_{(j,i)}(k, k-\mu)$ must be true. Let $\overline{z}_{(i,j)}(k, k-\mu)$ be the adjoint of max-plus binary ordering variable $z_{(i,j)}(k, k-\mu)$, defined as:

$$\overline{z}_{(i,j)}(k, k-\mu) = \begin{cases} \varepsilon & \text{If } z_{(i,j)}(k, k-\mu) = e. \\ e & \text{If } z_{(i,j)}(k, k-\mu) = \varepsilon. \end{cases} \tag{3.18}$$

The consistency relationships can then be defined as:

$$\begin{aligned} &(z_{(i,j)}(k, k-\mu) \otimes z_{(j,i)}(k, k-\mu) \otimes \overline{z}_{(i,j)}(k-\mu, k)) \\ &\qquad \oplus (\overline{z}_{(i,j)}(k, k-\mu) \otimes \overline{z}_{(j,i)}(k, k-\mu) \otimes z_{(i,j)}(k-\mu, k)) = e \end{aligned} \tag{3.19}$$

The *Same Travel Direction Ordering* and *Opposite Travel Direction Ordering* constraints are nearly identical to those in [31]. A key difference, however, is that the ordering variables no longer depend on four ordering variables per lock, corresponding to the different combinations of upstream and downstream movement directions for two vessels, in favor of one ordering variable per lock arc and the *Ordering Variable Constistency* constraint to handle the interaction between ordering variables in different directions. This approach is more general. Furthermore, a specific upstream and downstream direction does not have to be defined for every lock arc.

3.4 SMPL System

The model components and the constraints listed in Section 3.3 are combined into an SMPL model in this section.

Subsection 3.4.1 describes how the inequality constraints are combined into an SMPL system, and Subsection 3.4.2 describes how the logical equality constraints are combined to describe the allowable switching modes of the system.

3.4.1 System Equation

The *Departure Time*, *Travel Time*, *Same Travel Direction Ordering*, and *Opposite Travel Direction Ordering* constraints are all larger than inequality constraints that constrain how the arrival states evolve as function of themselves. In other words, the listed larger than inequality constraints can form an SMPL system for the arrival time states, linking the states of vessel k to the other vessels. Let $\boldsymbol{x}(k)$ be the vector of all arrival time states of vessel k:

$$\boldsymbol{x}(k) = \begin{bmatrix} x_0(k) \\ x_1(k) \\ \vdots \\ x_{|\mathcal{N}|-1}(k) \end{bmatrix}. \tag{3.20}$$

The notation $|\mathcal{N}|$ denotes the cardinality of set $\mathcal{N}$. Let $\boldsymbol{w}(k)$ be the vector of all routing variables of vessel k for all (i,j) in $\mathcal{D}$:

$$\boldsymbol{w}(k) = \left[w_{(i,j)}(k) | \forall\, (i,j) \in \mathcal{D}\right]^T, \tag{3.21}$$

and let $\boldsymbol{z}(k, k-\mu)$ be the vector of all ordering variables connecting vessel k and vessel $k-\mu$, for all (i,j) in $\mathcal{D}_{locks}$:

$$\boldsymbol{z}(k, k-\mu) = \left[z_{(i,j)}(k, k-\mu) | \forall\, (i,j) \in \mathcal{D}_{locks}\right]^T. \tag{3.22}$$

The routing inequalities for vessel k are dependent on $\boldsymbol{w}(k), \boldsymbol{x}(k)$, and $u(k)$. The routing inequalities for vessel k can be gathered together as:

$$\boldsymbol{x}(k) \geq A_{routing,(k)}(\boldsymbol{w}(k)) \otimes \boldsymbol{x}(k) \oplus B_{routing,(k)} \otimes u(k), \tag{3.23}$$

where $A_{routing,(k)}(\boldsymbol{w}(k))$ is the routing inequality constraint matrix for vessel k, and $B_{routing,(k)}$ is the routing reference input matrix for vessel k.

The ordering inequalities for vessel k are dependent on the ordering variables $\boldsymbol{z}(k, k-\mu)$ with all other vessels $k-\mu$, the routing variables $\boldsymbol{w}(k)$, the states $\boldsymbol{x}(k)$, and routing variables $\boldsymbol{w}(k)$ of all other vessels $k-\mu$. The ordering inequalities between vessel k and $k-\mu$ are given by:

$$x(k) \geq A_{ordering,(k,k-\mu)}\left(\boldsymbol{w}(k), \boldsymbol{w}(k-\mu), \boldsymbol{z}(k, k-\mu)\right) \otimes \boldsymbol{x}(k-\mu). \tag{3.24}$$

Where $A_{ordering,(k,k-\mu)}\left(\boldsymbol{w}(k), \boldsymbol{w}(k, k-\mu), \boldsymbol{z}(k, k-\mu)\right)$ is the ordering inequality matrix between vessel k and vessel $k-\mu$. Let $\mathcal{M}(k)$ be the set of possible values of μ such that $\{k-\mu | \mu \in \mathcal{M}(k)\} = \mathcal{K} \setminus \{k\}$. Then the ordering inequalities between vessel k and all other vessels are:

$$\boldsymbol{x}(k) \geq \bigoplus_{\mu \in \mathcal{M}(k)} A_{ordering,(k,k-\mu)}\left(\boldsymbol{w}(k), \boldsymbol{w}(k, k-\mu), \boldsymbol{z}(k, k-\mu)\right) \otimes \boldsymbol{x}(k-\mu). \tag{3.25}$$

Note that, as these are all larger than or equal to constraints, they can be further combined into a single inequality constraint matrix and reference input matrix. Let $\mu_{\min}(k)$ and $\mu_{\max}(k)$ be the minimum and maximum values of $\mathcal{M}(k)$. Let $\boldsymbol{\phi}(k)$ be the combined vector of all binary variables that are related to decisions of vessel k, also referred to as the switching variable vector:

$$\boldsymbol{\phi}(k) = \begin{bmatrix} \boldsymbol{w}(k) \\ \boldsymbol{w}(k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{w}(k-\mu_{\min}(k)) \\ \boldsymbol{z}(k, k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{z}(k, k-\mu_{\min}(k)) \end{bmatrix} \tag{3.26}$$

Let $A_{(k,k-\mu)}(\boldsymbol{\phi}(k))$ be the inequality constraint matrix for vessel k combined with vessel $k-\mu$, defined as:

$$A_{(k,k-\mu)}(\boldsymbol{\phi}(k)) = \begin{cases} A_{routing,(k)}(\boldsymbol{\phi}(k)) & \text{If } \mu = 0 \\ A_{ordering,(k,k-\mu)}(\boldsymbol{\phi}(k)) & \text{If } \mu \in \mathcal{M}(k) \end{cases} \tag{3.27}$$

Let $B_{(k)}$ be the reference input matrix for vessel k, defined as:

$$B_{(k)} = B_{routing,(k)} \tag{3.28}$$

Then, the inequality constraints for vessel k are given by the following switching max-plus-linear system with the switching of its modes governed by switching variable vector $\boldsymbol{\phi}(k)$:

$$\boldsymbol{x}(k) \geq \bigoplus_{\mu \in \mathcal{M}(k) \cup \{0\}} A_{(k,k-\mu)}(\boldsymbol{\phi}(k)) \otimes \boldsymbol{x}(k-\mu) \oplus B_{(k)} \otimes u(k). \tag{3.29}$$

3.4.2 Allowable Modes

The SMPL constraints also have equality constraints. These never include the arrival time states, instead only constraining the max-plus binary switching variables through logical relationships. These equality constraints define the set of allowable switching modes $\mathcal{Q}(k)$ the SMPL system for vessel k may be in. For vessel k they can be written as:

$$\boldsymbol{q}_{(k)}(\boldsymbol{\phi}(k)) = \boldsymbol{y}_{(k)}, \tag{3.30}$$

where $\boldsymbol{q}_{(k)}$ is the equality constraint vector of vessel k, and where the entries of the vector $\boldsymbol{y}_{(k)}$ are max-plus scalar. Any permutation of $\boldsymbol{\phi}(k)$ that is in $\mathcal{Q}(k)$ is an allowable mode of the SMPL system for vessel k. If all vessels k are in an allowable switching mode according to Equation 3.30, then the combined system of all vessels is in an allowable switching mode, meaning that the entire system is compliant with the logical constraints.

3.5 MILP Problem Conversion

This section describes the conversion of the SMPL constraints to MILP constraints in conventional algebra, and the objectives and performance indicators that may be used during scheduling and post-processing.

For the scheduler to be able to find a scheduling solution, the SMPL system constraints must be converted to MILP constraints in conventional algebra, which can then be solved by a solver like Gurobi, which is the solver that will be used in this thesis. Furthermore, an objective function is required to instruct the scheduler what to optimize the scheduling for, like cumulative arrival time, or vessel waiting time. Performance indicators must also be defined, which can be used to analyse the found scheduling solutions past the objective function itself.

The general structure of an MILP problem is described in Subsection 3.5.1. The SMPL constraints are converted to MILP constraints in Subsection 3.5.2. Subsection 3.5.3 discusses the objectives that are added to the optimization problems, and Subsection 3.5.4 lists relevant performance indicators for analysis of the solution.

The mathematical content of this section is written in conventional algebra unless specified otherwise.

3.5.1 General MILP Problem Structure

The general form of MILPs has been discussed in Subsubsection 2.2.5.1. For reference, the version where all integer decision variables are binary is given again here:

$$\begin{aligned} \min_{x,\phi} \quad & J = c_x^T \boldsymbol{x} + c_\phi^T \boldsymbol{\phi} \\ \text{s.t} \quad & A_x \boldsymbol{x} + A_\phi \boldsymbol{\phi} \leq \boldsymbol{f} \\ & \boldsymbol{x} \geq 0 \\ & \boldsymbol{\phi} \in \mathbb{B}^{n_\phi} \end{aligned}$$

Note that the scheduling models only use binary variables, so $\boldsymbol{\phi} \in \mathbb{B}^{n_\phi}$. A max-plus binary variable can take on the values e and ε, where $\varepsilon \notin \mathbb{R}$, thus a conversion is required to convert max-plus-binary variables to scalar binary variables with a similar meaning, as explained in Subsubsection 2.2.5.1. To make it easier to compare the SMPL and MILP models, the different binary variables will maintain their same mathematical symbols, with it being clear from the context if they are in $\mathbb{B}_{\max}$ or in $\mathbb{B}$.

3.5.2 Constraints

Each of the SMPL model constraints is converted to constraints for the MILP in conventional algebra.

Departure Time Constraint

For all $k \in \mathcal{K}$:

$$-x_{b(k)} \leq -u(k) \tag{3.31}$$

3.5.2.1 Route Start Constraint

What Equation 3.11 essentially describes, is that for each vessel k, one arc exiting node $b(k)$ must be active. Thus, for all $k \in \mathcal{K}$:

$$\sum_{i \in \mathcal{S}(b(k))} w_{(b(k),i)}(k) \leq 1 \tag{3.32}$$

$$-\sum_{i \in \mathcal{S}(b(k))} w_{(b(k),i)}(k) \leq -1 \tag{3.33}$$

3.5.2.2 Route End Constraint

Similar to the *route start* constraint, for each vessel k, one arc entering node destination node $d(k)$ must be active. For all $k \in \mathcal{K}$:

$$\sum_{i \in \mathcal{S}(d(k))} w_{(i,d(k))}(k) \leq 1 \tag{3.34}$$

$$-\sum_{i \in \mathcal{S}(d(k))} w_{(i,d(k))}(k) \leq -1 \tag{3.35}$$

3.5.2.3 Transit Constraint

Equation 3.13 can be converted to two MILP constraints, the first constraining that the number of activated arcs going into a node that is not a departure or destination node i for each vessel k must be equal. For all $(k,i) \in \mathcal{K} \times \mathcal{N} \setminus \{b(k), d(k)\}$:

$$\sum_{j \in \mathcal{S}(i)} w_{(i,j)}(k) - \sum_{j \in \mathcal{S}(i)} w_{(j,i)}(k) \leq 0 \tag{3.36}$$

$$\sum_{j \in \mathcal{S}(i)} w_{(j,i)}(k) - \sum_{j \in \mathcal{S}(i)} w_{(i,j)}(k) \leq 0 \tag{3.37}$$

The second constraining that at most one ingoing arc may be active at a time for that node i. For all $(k,i) \in \mathcal{K} \times \mathcal{N} \setminus \{b(k), d(k)\}$:

$$\sum_{j \in \mathcal{S}(i)} w_{(j,i)}(k) \leq 1 \tag{3.38}$$

$$\tag{3.39}$$

3.5.2.4 Travel Time Constraint

For all $((i,j),k) \in \mathcal{D} \times \mathcal{K}$:

$$-x_j(k) + x_i(k) - \beta w_{(i,j)}(k) \leq -\beta - \tau_{(i,j)}(k) \tag{3.40}$$

3.5.2.5 Same Travel Direction Ordering Constraint

For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$-x_i(k) + x_j(k-\mu) - \beta w_{(i,j)}(k) - \beta w_{(i,j)}(k-\mu) - \beta z_{(i,j)}(k,k-\mu) \leq -3\beta - \tau_{(j,i)}(k) \tag{3.41}$$

3.5.2.6 Opposite Travel Direction Ordering Constraint

For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$-x_i(k) + x_i(k-\mu) - \beta w_{(i,j)}(k) - \beta w_{(j,i)}(k-\mu) - \beta z_{(i,j)}(k,k-\mu) \leq -3\beta \tag{3.42}$$

3.5.2.7 Ordering Variable Consistency Constraint

The ordering variable consistency constraint is split into two constraints. The first defines that $z_{(i,j)}(k, k-\mu)$ and $z_{(j,i)}(k, k-\mu)$ must be equal for all lock arcs and vessels. For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$z_{(i,j)}(k, k-\mu) - z_{(j,i)}(k, k-\mu) \leq 0 \tag{3.43}$$

$$z_{(j,i)}(k, k-\mu) - z_{(i,j)}(k, k-\mu) \leq 0 \tag{3.44}$$

The second defines that $z_{(i,j)}(k, k-\mu)$ and $z_{(i,j)}(k-\mu, k)$ must be opposites of each other, or equivalently that they must add up to 1. For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$z_{(i,j)}(k, k-\mu) + z_{(i,j)}(k-\mu, k) \leq 1 \tag{3.45}$$

$$-z_{(i,j)}(k, k-\mu) - z_{(i,j)}(k-\mu, k) \leq -1 \tag{3.46}$$

3.5.3 Objectives

The objectives drive the scheduling solution decisions. Two different objectives, or the combination of different objectives, can lead to entirely different schedules.

Two main objectives will be considered for this project, combined into a multi-criteria objective function. The first is the objective where vessels want to arrive at their destination as quickly as possible, governed by the cumulative travel time objective. The second is the scenario where vessels have an intended arrival time at their final destination, governed by the arrival time offset objective. Vessels may have already planned and reserved their arrival time at their destination, incurring extra fees to reschedule, or having to wait for a different time slot. If vessels are early, they could even have slow-steamed part of their journey to save on fuel with no other consequences. A vessel cannot be assigned both main objectives at the same time, but different vessels may have different objectives in the same scheduling problem.

As discussed in Subsection 2.1.1, waiting time in the waiting areas and the number of lockages are commonly used as the main objectives for lock scheduling problems. These are not used as the main objectives in this model for two reasons.

First, waiting time only works as an objective if the arrival time of the vessels at the waiting areas is set beforehand, as the schedule will just slow the vessels down on non-waiting area arcs if time spent in waiting areas is penalized, with no improvement to the actual arrival time of the vessel at its destination.

Second, empty lockages in between consecutive lockages in the same direction are difficult to linearly count in the objective function without fully changing the model to be able to do so. If empty lockages were neglected, the number of lockages objective could be off by as much as a factor of two. Furthermore, the two main objectives are currently balanced in that they both represent time and a vessel can either have one or the other objective. The number of lockages has a different unit entirely, so it is harder to balance with the time-based objectives in a multi-criteria objective function.

There are also two auxiliary objectives that are intended to enforce certain beneficial scheduling behavior, that are active at all times.

The first ensures that vessels' departure times are always equal to their minimum departure time. This potentially makes the vessels' travel times longer but does not delay the arrival time at their destination. The same distance is traveled in a longer amount of time, so vessels move at a slower pace on average, reducing emissions. The alternative to this was seen in early experiments without this objective, where the departure time of the vessels just had to be larger than $u(k)$. There,

the vessels would all leave at the latest possible moment for them to arrive on time at their first lockage when traveling at full speed. This did not improve the objective function value compared to situations with this auxiliary constraint, the solver just usually ended up with that scenario. In general, it was assumed that it would be better for vessels to not have to travel at full speed from an emission perspective and this was an easy way to prevent that. Future implementations should look at allowing departure time to be less rigid with some mechanism to reduce velocity if there is room.

The second, the waiting time objective, ensures that a vessel that has to wait for a lockage does so by reducing speed on a non-waiting area arc, rather than by coming to a full stop in a waiting area of a lock. This, again, encourages slow-steaming behavior with no influence on the vessels' final arrival times. Slow-steaming could also be added in the post-processing of the solution, but these auxiliary objectives already take care of that step in the optimization problem itself, without influencing the main objectives. There will be scenarios in which vessels are scheduled to go impractically slow to avoid waiting at a waiting area. Post-processing could then decide on a better velocity for the vessel, with the vessel making up for the gained time in the waiting area.

The objectives will all be required to be linear. This ensures that the MILP problem being solved stays linear. Efficient solution algorithms and solvers exist that are generally applicable to all MILP models and that are guaranteed to converge to the global optimum eventually. It could be possible to solve for schedules with nonlinear objectives (and even constraints). However, this would require another study entirely to investigate which solution method would be best, a solution that is a global optimum is not necessarily guaranteed, and the computational difficulty may increase exponentially.

3.5.3.1 Cumulative Travel Time

The cumulative travel time objective ensures that vessels get to their destinations as quickly as possible, by minimizing the (weighted) travel times for the vessels. Travel time is the time it takes, starting from the minimum departure time $u(k)$, for vessel k to arrive at its final destination node $d(k)$. Let $\sigma_{\mathcal{A}}(k)$ be the cumulative travel time objective weight for each vessel. Through this weight, the vessels can be given different levels of priority, by making a longer travel time for a vessel with a high weight count more towards the objective function. The linear cumulative travel time objective $J_{\mathcal{A}}$ is written as:

$$J_{\mathcal{A}} = \sum_{k \in \mathcal{K}} \sigma_{\mathcal{A}}(k) \left(x_{d(k)}(k) - u(k) \right). \tag{3.47}$$

Since the values of $u(k)$ are constant terms, they do not influence the optimization problem, so it would be equally valid to only penalize the arrival times at the vessels' destinations. However, $u(k)$ is left in to show what the objective is physically meant to represent.

A different approach was taken in [31]. The arrival times at the departure nodes were multiplied by 1, and the arrival times at all other nodes were multiplied by a smaller penalty of 10^{-4}. This intermediate penalty is not necessary to achieve the shortest travel time possible, as penalizing the final arrival time alone already ensures the shortest travel time possible is achieved.

3.5.3.2 Cumulative Departure Time

Equation 3.10 defines that $x_{b(k)}(k) \geq u(k)$ for all vessels. To enforce that $x_{b(k)}(k) = u(k)$, without having to add a smaller than or equality constraint to the max-plus formulation, a cumulative departure time objective is added to the objective function at all times, applying a small penalty to all departure states. This cumulative departure time objective J_u is equal to:

$$J_u = \sum_{k \in \mathcal{K}} x_{b(k)}(k) \cdot 10^{-3}. \tag{3.48}$$

As there is no other objective that would benefit from later departure times, the inclusion of this objective is sufficient to ensure that $x_{b(k)}(k) = u(k)$ for all vessels.

3.5.3.3 Waiting Time

The vessel waiting time objective is the sum of the waiting time spent in the waiting areas at all of the locks for each of the vessels. For each lock arc (i, j) in the set of lock arcs $\mathcal{D}_{locks}$, let $\delta(i)$ be the node of the arc that precedes lock arc (i, j), representing the waiting entrance area before the lock arc. $\delta(i)$ and $(\delta(i), i)$ are highlighted for an arbitrary lock arc (i, j) in Figure 3.12. Then the waiting

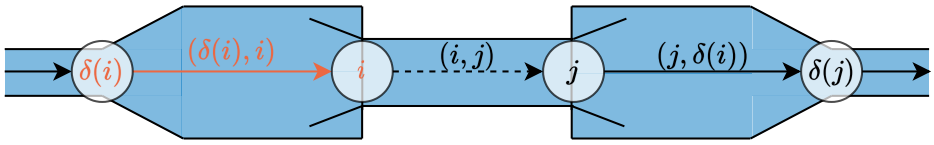


Figure 3.12: *Lock (i, j) with node $\delta(i)$ and arc $(\delta(i), i)$ highlighted.*

time $\pi_{(i,j)}(k)$ is defined as the time spent on $(\delta(i), i)$ for vessel k minus the minimum travel time $\tau_{(\delta(i),i)}(k)$, as that time would be spent on that arc by the vessel even if it did not have to wait:

$$\pi_{(i,j)}(k) = w_{(i,j)}(k)(x_i(k) - x_{\delta(i)}(k) - \tau_{(i,\delta(i))}(k)). \tag{3.49}$$

Note that this is always 0 if vessel k does not travel arc k, and $x_i(k) - x_{\delta(i)}(k) - \tau_{(i,\delta(i))}(k) \geq$ always holds if $w_{(i,j)}(k) = 1$, as by the structure of the lock nodes and by the transit constraints $w_{(\delta(i),i)}(k) = 1$ if $w_{(i,j)}(k) = 1$.

$\pi_{(i,j)}(k)$ is not a linear expression, but a linear objective function that has the same underlying meaning can be built. Define continuous variable $t_{\Pi,(i,j)}(k)$ to be:

$$t_{\Pi,(i,j)}(k) = \max(x_i(k) - x_{\delta(i)}(k) - \tau_{(i,\delta(i))}(k) + \beta(1 - w_{(i,j)}(k)), 0) \tag{3.50}$$

The vessel waiting time objective J_Π can then linearly be defined as:

$$J_\Pi = \sum_{(i,j)\in\mathcal{D}_{locks}} \sum_{k\in\mathcal{K}} t_{\Pi,(i,j)}(k) \cdot 10^{-3}, \tag{3.51}$$

provided that vessel waiting time objective is always a part of a minimization problem, and that the following constraints are added for all $((i, j), k) \in \mathcal{D}_{locks} \times \mathcal{K}$ to enforce the equality in Equation 3.50:

$$t_{\Pi,(i,j)}(k) - x_i(k) + x_{\delta(i)}(k) + \beta(1 - w_{(i,j)}(k)) \leq -\tau_{(i,\delta(i))}(k) \tag{3.52}$$

$$-t_{\Pi,(i,j)}(k) \leq 0 \tag{3.53}$$

This objective will have negligible impact on the optimal values of any of the other objectives, due to its very low weighting of 10^{-3}.

3.5.3.4 Arrival Time Offset

Vessels often may have a scheduled berth or activity at their destination, so arriving early or late can be undesirable. Differences in arrival time can be penalized with the (weighted) arrival time offset objective. Let $\hat{x}_{d(k)}(k)$ be the planned arrival time at node $d(k)$ for vessel k. Let $\sigma_{\mathcal{H}}(k)$ be the arrival time offset weight for vessel k. Then the arrival time offset objective $J_{\mathcal{H}}$ can be written as:

$$J_{\mathcal{H}} = \sum_{k\in\mathcal{K}} \sigma_{\mathcal{H}}(k)|\hat{x}_{d(k)}(k) - x_{d(k)}(k)|. \tag{3.54}$$

This description is non-linear, but it can be linearized by introducing the variable $t_{\mathcal{H}}(k)$:

$$t_{\mathcal{H}}(k) = |\hat{x}_{d(k)} - x_{d(k)}|, \tag{3.55}$$

then the arrival time offset objective $J_{\mathcal{H}}$ can be written as:

$$J_{\mathcal{H}} = \sum_{k \in \mathcal{K}} \sigma_{\mathcal{H}}(k) t_{\mathcal{H}}(k), \tag{3.56}$$

provided that the problem is a minimization problem, and that the following constraints are added to the problem to enforce Equation 3.55. For all $k \in \mathcal{K}$:

$$\hat{x}_{d(k)} - x_{d(k)} - t_{\mathcal{H}}(k) \leq 0 \tag{3.57}$$

$$x_{d(k)} - \hat{x}_{d(k)} - t_{\mathcal{H}}(k) \leq 0 \tag{3.58}$$

The constant term $\hat{x}_{d(k)}$ is not redundant in this case, as it directly impacts the optimization problem through the constraints in Equation 3.57 and Equation 3.58.

3.5.3.5 Overall Objective Function

The full multi-criteria objective function J is a weighted sum of the cumulative travel time and arrival time offset objectives, plus the always present cumulative departure time and waiting time objectives. The cumulative travel time and arrival time offset objectives get their own weights $\omega_{\mathcal{A}}$ and $\omega_{\mathcal{H}}$:

$$J = \omega_{\mathcal{A}} J_{\mathcal{A}} + \omega_{\mathcal{H}} J_{\mathcal{H}} + J_u + J_{\Pi} \tag{3.59}$$

$J_{\mathcal{A}}$ or $J_{\mathcal{H}}$ can be deactivated by setting their respective weights to 0. In the analysis in this thesis in Chapter 6, the travel time and arrival time offset objectives will usually be looked at separately by setting one of the objective weights to zero, but it is also possible to assign different objectives to different vessels through the vessel weights $\sigma_{\mathcal{A}}(k)$ and $\sigma_{\mathcal{H}}(k)$. A vessel cannot have both the arrival time objective and the arrival time offset objectives at the same time because they would interfere with each other.

3.5.4 Performance Indicators

Performance indicators differ from objectives in the fact that they are computed after the scheduling problem has been solved, without influencing the solution, so they can be non-linear without increasing computational difficulty or requiring different solution algorithms as discussed in the introduction to this section. They can be used to analyse the solution.

3.5.4.1 Cumulative Arrival Time

The cumulative arrival time performance indicator $\mathcal{A}$ is computed as the unweighted cumulative arrival time objective:

$$\mathcal{A} = \sum_{\{k \in \mathcal{K} | \sigma_{\mathcal{H}}(k) \neq 0\}} (x_{d(k)} - u(k)). \tag{3.60}$$

3.5.4.2 Arrival Time Offset

The arrival time offset $\mathcal{H}$ is calculated as the unweighted arrival time offset objective:

$$\mathcal{H} = \sum_{\{k \in \mathcal{K} | \sigma_{\mathcal{H}}(k) \neq 0\}} |\hat{x}_{d(k)}(k) - x_{d(k)}(k)|. \tag{3.61}$$

3.5.4.3 Number of Lockages

The number of lockages $\mathcal{L}$ is a common objective in problems involving lock scheduling [32], as opening locks can have an adverse effect on water levels on either side of the lock. On the other hand, lock movements can also be used to generate hydrodynamic energy, which would make it better to have more lock movements [22]. Thus, it is beneficial to be able to keep track of this value.

The number of lock movements is not just equal to the number of vessels passing a lock. When two vessels pass sequentially in the same direction, an empty lockage that is not kept track of by any variable must have taken place in the opposite direction. The true number of lockages can be extracted by for each lock chamber in the graph, described by arcs (i,j) and (j,i), ordering the active routing variables by the value of the state of the vessel and starting node associated with the routing variable, as shown in Equation 3.62 for an arbitrary chamber with arcs (i,j) and (j,i), and three active routing variables in ordered sequence. Any time two subsequent active ordered routing variables are on the same arc, an empty extra lockage should be added to the count in $\mathcal{L}$.

$$\begin{array}{rccc} & \overset{\text{Sequence}}{\longrightarrow} & & \mathcal{L} \mathrel{+}= 1 \\ \text{Routing Variable Sequence:} & w_{(i,j)}(0) = 1 & w_{(j,i)}(1) = 1 & w_{(j,i)}(2) = 1 \\ \text{Associated States:} & x_i(0) = 0.5 & x_j(0) = 1.5 & x_j(2) = 2.5 \end{array} \tag{3.62}$$

In the future, it is important that an objective is added that can penalize the number of lockages. The multi-chamber models will run into scenarios where it is faster to split vessels between chambers in separate lockages, at the cost of just performing another lockage. In times of drought, or for locks that form a division between salt and fresh water, it is paramount that water usage is considered through one of the objective criteria.

3.5.4.4 Average Delay

The cumulative travel time objective $\mathcal{A}$ gives an indication of how long it takes the vessels to complete all of their journeys, but there is no frame of reference for how quickly the vessels could have completed that journey by traveling at their maximum velocity without interference from other vessels.

The average delay KPI $\mathcal{T}$ is introduced to provide this frame of reference. It is given as a percentage of the minimum possible travel time for each of the vessels on their decided routes, without any interference from other vessels. In essence, it is thus the average delay caused to the vessels by the presence of other vessels in the system. For a vessel k this additional travel time is defined as:

$$\mathcal{T}(k) = 100\% \cdot \frac{\sum_{(i,j)\in\mathcal{D}} w_{(i,j)}(k)(x_j(k) - x_i(k))}{\sum_{(i,j)\in\mathcal{D}} w_{(i,j)}(k)\tau_{(i,j)}(k)} - 100\% \tag{3.63}$$

For the full set of vessels the average delay is the average of all of these percentages:

$$\mathcal{T} = \sum_{\{k\in\mathcal{K}|\sigma_{\mathcal{H}}(k)=0\}} \frac{\mathcal{T}(k)}{|\{k \in \mathcal{K}|\sigma_{\mathcal{H}}(k) = 0\}|}. \tag{3.64}$$

The minimum value of $\mathcal{T}$ is 0%, corresponding with a schedule where every vessel spends the minimum amount of time possible on all arcs of their route. Note that vessels that have an arrival offset objective are not counted in this KPI, because they purposefully do not want to travel as fast as possible.

It should be noted that this is not a perfect representation of the delay. The underlying calculation uses the route determined by the optimization problem, which have been scheduled while keeping what other vessels are doing in mind. However, the current KPI is still an acceptable first-order estimate of the delay.

3.6 Online Optimization

As briefly mentioned in Section 1.4, the schedulers in this and the coming chapters perform offline optimization. The assumptions that the future can be predicted perfectly and that all of the components perfectly obey the schedule are made. Those assumptions will not hold up in a real system. Imagine, for instance, that a new vessel declares that it wants to start a journey halfway through a ten hour schedule of the global system, or that a vessel experiences technical difficulties, causing it to deviate from its prescribed arrival times. These disturbances may propagate in the system, causing unintended and unforeseen delays for other vessels.

When a disturbance starts affecting the system, it would be better to take this information into account in a newly computed schedule. This is the basis of online optimization: accounting for imperfect predictions by regularly recomputing the schedule with newly updated information and predictions [35].

A schematic of what an early online implementation for IWT scheduling might look like can be found in Figure 3.13. As will become clear in Section 6.8, a schedule for a large number of vessels (≥ 15) may take a considerable amount of time to compute. The computation time of schedule i is denoted as $\Delta T(i)$. The scheduler uses the most recent system information to start the computation of a new schedule, while the most recently completed schedule is followed by the vessels. Once the computation of the new schedule is done, it is given to the vessels and the scheduler takes in updated system information to start computing a new schedule. It should be noted that the offline problem introduced in this report serves as the starting point for the implementation of the optimization problem run every time a new schedule is required in online optimization.

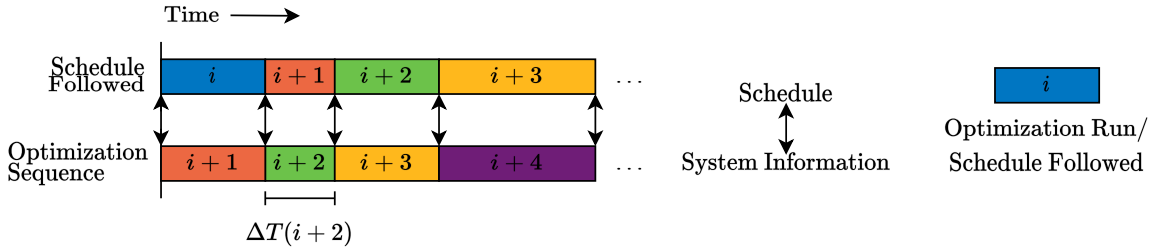


Figure 3.13: *Time-sequence of online scheduling implementation.*

$\Delta T(i+1)$ is different from $\Delta T(i+2)$ in Figure 3.13. This is intentional. The branch-and-bound solution algorithm has variability in solution time for similar problems, so no constant computation time interval can be assured in an initial design. The branch-and-bound algorithm and model complexity are further expanded upon in Section 6.8. As long as the computation times are long, it is important that the new schedule starts computation as soon as possible, to keep the interval between new schedules short, reducing the possible impact of disturbances. If the computation time ever consistently becomes smaller than a few minutes, it may become necessary to place a minimum on the amount of time between new schedules communicated to the vessels, because they cannot reasonably adjust their plans and course every few minutes.

IWT vessels are not very fast, with average velocities of $8-12\,km/h$ being reasonable estimates on some long routes near or in the Netherlands [80]. A look at Figure 1.1 shows that there is, with some exceptions, usually at least $3-4\,km$ between locks. A vessel moving at $12\,km/h$ would require 15-20 minutes to travel between locks. An upper limit on the time interval between finding new schedules could be set to 15-20 minutes, to ensure that a new schedule is generated at least once between lock visits for the vessels. This means that, in the best-case scenario, the optimization problem has to reliably be able to find a solution within that time frame. Of course, lower time intervals would be better. A time limit may also be set to put a hard limit on computation time, after

which the best feasible solution found up until that point is used, instead of continuing computation for the optimal solution.

Long computation times also require careful consideration of what changes can be made by a new schedule compared to an older schedule. A new schedule that took 15 minutes to compute is based on 15 minute old information. For multi-chamber locks, a vessel that was originally scheduled on chamber I of a lock may suddenly be scheduled on chamber II of that same lock. In 15 minutes that same vessel may already have entered chamber I, making it impossible to follow the new schedule. This could have been prevented by analysing the short-term future of the system and constraining those parts of the schedule to be the same for the upcoming schedule.

The disturbance introduced by new vessels entering the system just after a new schedule has been found can be minimized by requiring vessels to announce their departure time at least a scheduling iteration before they leave, through the BICS system for instance, thus they are always already in the schedule when they leave.

The models would have to be changed to be able to handle vessels that are already in transit between nodes, or that are in lock chambers. Currently, the assumption is that vessels are at one of the nodes at the start of the schedule. Travel times for vessels that are already in transit on an arc could be changed based on how far along the arc they are traveling on they are, and the arc they are traveling on can be constrained to always be active so it must be part of the rest of their route.

Finally, the computation time of finding an initial feasible solution to the optimization problem can be reduced by using the results of the previous time step, possibly with a fast heuristic to add a feasible starting schedule for new vessels entering the system. Distributed optimization is also a good option to decrease computation time, as was discussed in Subsection 2.2.4.

3.7 Conclusion

The research question to be answered in this chapter was:

> *How can the SMPL IWT scheduling model be formulated for networks with arbitrary network topologies?*

A single-vessel, single-chamber SMPL lock scheduling model was introduced in [31]. This model was not capable of scheduling vessels on any arbitrary graph representation of an IWT system, which would enable the analysis of large-scale networks.

Representing locations by nodes, waterways by arcs, and locks as special constructions of nodes and arcs, arbitrary waterway network graph topologies with single-vessel, single-chamber locks can be formed.

The mathematical model from [31] was augmented to work with these arbitrary network topologies. In particular, vessel routes are now constructed according to a similar procedure as in the classical shortest path problem, but translated to SMPL algebraic logical constraints.

A set of SMPL routing, and ordering constraints that the scheduling solutions must obey was established. The inequality constraints can be gathered into an SMPL system equation, with the logical equality constraints describing the switching modes the system is allowed to be in.

For scheduling, the SMPL mathematical model is translated into an MILP model. A scheduler uses the rules of this MILP model to automatically generate and solve optimization problems for given input on the topology graph, vessels, and objectives. The scheduled variables are the arrival times of the vessels at the nodes, which arcs the vessels' routes consist of, and the ordering between vessels on lock chambers.

Two main objectives were introduced. The cumulative travel time objective ensures that vessels arrive at their destination as soon as possible and the arrival time offset objective that ensures that vessels arrive at their destination at the correct time. Two auxiliary objectives ensure that vessels depart as early as possible and that vessels slow-steam rather than wait in lock waiting areas, All objectives are combined in a multi-criteria objective function. The auxiliary objectives are given small weights to not impact the main objective values.

Key performance indicators were introduced that keep track of the cumulative arrival time, the arrival time offset, and the average delay of the vessels. The number of lockages that take place is also tracked. A number of lockages objective that is common in the lock scheduling problem literature could not be linearly implemented, though it is recommended to find a way to implement it in the future.

The scheduler built performs offline scheduling. Unforeseen disturbances introduced during the execution of a schedule can significantly disrupt the IWT system. To account for this, online scheduling, rescheduling the system at regular time intervals, should be examined in the future. Important considerations will be the maximum time between new schedules, the effect of computation time on near-future decisions, the scheduling of vessels that are in transit on an arc already, and the reduction of computation time.

4

Multi-Vessel, Multi-chamber Scheduling with Chamber Capacities

The mathematical model in the previous chapter was formulated under the assumption that each lock consists of a single chamber, with a maximum capacity of a single vessel. Real locks can have multiple chambers and they often have the capacity for multiple vessels to pass through at the same time. Therefore, the model would be more widely applicable if multi-vessel, multi-chamber locks were integrated into it.

Integrating support for multi-chamber locks should not present a big challenge. In fact, the model in Chapter 3 was already written with that functionality in mind. Constraints like ordering constraints in Equation 3.16 and Equation 3.17 are not lock specific but lock arc specific, so they should work if additional chambers are added to a lock as additional lock arcs.

However, the addition of the full SPP to accommodate multi-vessel lockages would require big changes to the model and it would lead to a large increase in computational complexity due to the additional variables. A stepping stone to integrating the SPP would be to first create a version of the model which assumes the chambers have a one-dimensional size, or capacity, with the vessels equally having a one-dimensional size, allowing multiple vessels to pass through the chamber simultaneously as long as the sum of their sizes is smaller than the chamber capacity. That still limits computational complexity but does allow for multiple vessels to pass through the chambers at the same time.

The research question of this chapter then becomes:

> *How can multi-chamber locks with one-dimensional chamber capacity constraints be integrated into the SMPL IWT scheduling model?*

Section 4.1 describes the updated problem definition. Section 4.2 lists the additional SMPL constraints that are added. These constraints are combined into an SMPL system in Section 4.3 and converted to MILP constraints in Section 4.4. Finally, a conclusion to the research question is given in Section 4.5.

4.1 Problem definition

The model in this chapter uses the model from Chapter 3 as its starting point. Sets, variables, the graph description, parameters, objectives, assumptions, and constraints remain the same unless a change or addition to them is mentioned in this section.

Additional graph features are given in Subsection 4.1.1. Subsection 4.1.2 and Subsection 4.1.3 list the additional variables and parameters. Finally, Subsection 4.1.4 lists the changed and additional assumptions.

4.1.1 Graph

The single-vessel, single-chamber model was already fully compatible with multi-chamber locks when it comes to constraints and variables. The only change required to integrate multi-chamber locks is to introduce a general nodal structure for multi-chamber locks.

The nodal structure of a multi-chamber lock, with a lock with two chambers being used as an example, is shown in Figure 4.1. The situation for two chambers generalizes to situations with more than two chambers. An IWT network topology graph with a lock with two chambers is shown in Figure 4.2. Chambers are numbered with Roman numerals, starting at I. Adding an extra chamber to a lock adds two nodes and six arcs, two of which are lock arcs representing the chamber. All chambers of a lock share the same two waiting area nodes, so in the example $\delta(i) = \delta(l)$ and $\delta(j) = \delta(p)$. While separate chambers may have separate waiting areas for mooring in reality, the entry point to the run-out zone, and thus the waiting area entry point as it is modeled, is assumed to be the same for each of the chambers in the lock.

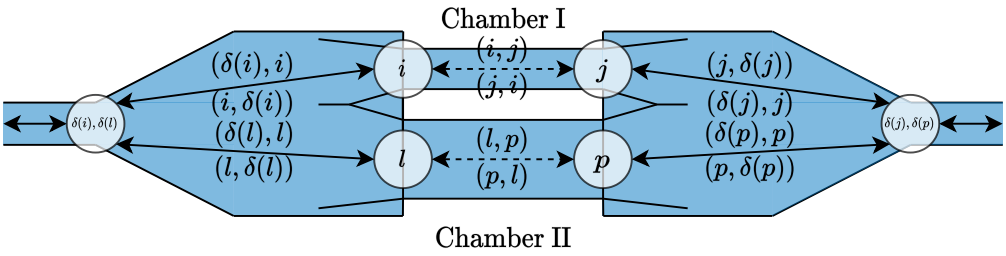


Figure 4.1: *Multi-chamber lock with two chambers.*

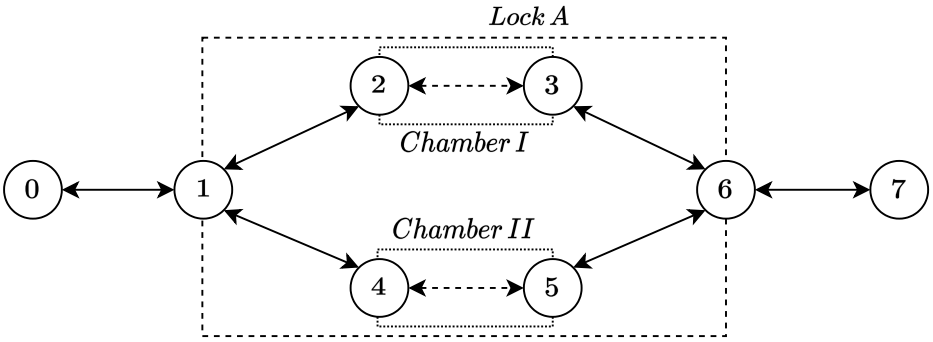


Figure 4.2: *Two-chamber lock in network topology graph.*

4.1.2 Variables

Multiple vessels can now be in the same lockage on a given chamber. When vessels k and $k - \mu$ are in the same lockage in chamber (i, j), they are said to be synchronised on chamber (i, j). $s_{(i,j)}(k, k - \mu)$ are synchronisation decision variables, which exist for each combination of a vessel $k \in \mathcal{K}$, any other vessel $k - \mu \in \mathcal{K} \setminus \{k\}$, and chamber $(i, j) \in \mathcal{D}_{locks}$. $s_{(i,j)}(k, k - \mu)$ determines whether or not vessel k and $k - \mu$ are synchronised on chamber (i, j).

With the added variables, a schedule is now comprised of the components listed in Figure 4.3.

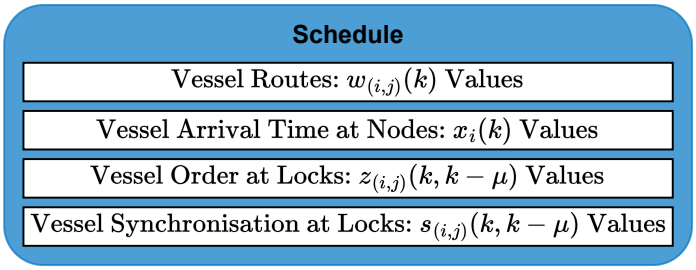


Figure 4.3: *The decision variables that make up a schedule for the multi-vessel, multi-chamber capacity scheduling model.*

4.1.3 Parameters

Chambers can have different shapes and sizes, as illustrated in Figure 4.1 with chamber II being bigger than chamber I. Vessels are also of different shapes and sizes. Let $c_{(i,j)}$ be the one-dimensional capacity of lock chamber arc (i, j), with $c_{(i,j)} = c_{(j,i)}$. Furthermore, let $\rho(k)$ be the one-dimensional size of vessel k. It must hold that $c_{(i,j)} \geq 1$ and $\rho(k) \geq 1$. It does not matter for the mathematical model which dimension of the vessels or the chamber $\rho(k)$ and $c_{(i,j)}$ denote. In Figure 4.4 some possible size dimensions that can be used are shown. The dimension taken could also be based on area, or be another arbitrary metric of size. Note that the shapes of vessels are always approximated by bounding box rectangles which encompass their outer extremities.

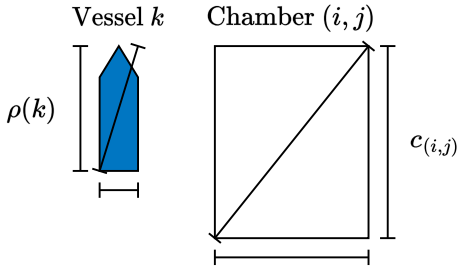


Figure 4.4: *Possible dimensions for the ship size $\rho(k)$ and chamber capacity $c_{(i,j)}$ parameters.*

The dimension and the size scale that are chosen do matter for the quality and validity of the final scheduling results. Refer to Figure 4.5, which gives three examples. Example (a) shows a scenario in which these one-dimensional size parameters work as intended. Examples (b) and (c) show cases in which the one-dimensional size parameters do not work, or do not work optimally. Vessels that fit according to the size of the chamber may unintentionally be required to overlap to truly fit, or vessels that do not fit according to the size of the chamber may actually fit together in some configurations. The values assigned to the size and capacity parameters of the vessels and the chamber must be chosen such that faulty or unnecessarily strict solution scenarios, especially those in example (b) which make the schedule impossible to execute, are avoided as much as possible.

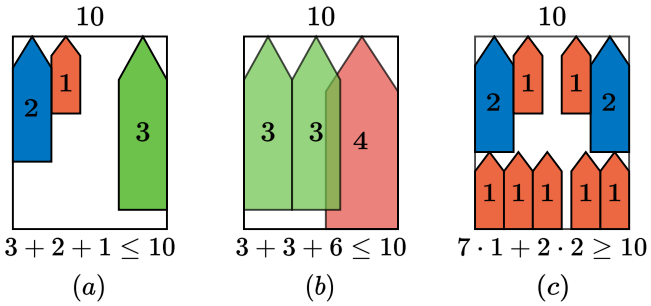


Figure 4.5: *(a) Example of a lockage that is valid in both a real scenario and the model. (b) Example of a lockage that is valid in the model but not in a real scenario. (c) Example of a lockage that is not valid in the model but is valid in a real scenario.*

Chambers of the same lock are assumed to operate independently and can have different minimum operation times $\tau_{(i,j)}(k)$. According to Section 3.1, this minimum operation time includes the worst-case time needed for a single vessel to position, moor, and leave the chamber. However, multiple vessels having to position, moor, and leave will add additional time to a lockage. Let $\overline{\tau}_{(i,j)}$ be the extra vessel processing time of chamber (i,j). This is the worst-case amount of extra time needed to process the lockage with an extra vessel. To reduce the number of inputs of the model, this quantity is independent of the extra vessel itself and the number of vessels already in the chamber. For safety, it can be taken to be the highest known extra processing time of vessels the chamber can process. If a lockage on lock chamber arc (i,j) has three vessels in it, its operation time is then at worst $\tau_{(i,j)}(k) + 2\overline{\tau}_{(i,j)}$, with vessel k being one of the vessels in the lockage.

4.1.4 Assumptions

The *all locks are single-vessel, single-chamber locks* assumption from Chapter 3 is replaced by the following assumption:

- **Locks can have multiple multi-vessel chambers with one-dimensional maximum capacities.**

Some additional assumptions are made for the one-dimensional sizes and capacities:

- **The shape of each vessel is approximated by a bounding rectangle.** *Vessels have a range of shapes and sizes. However, their shape can generally be closely represented by rectangles that form a bounding box around the extremities of the vessel.*
- **The size of a vessel's bounding rectangle can be described by a one-dimensional size parameter which can describe vessel length, width, area, or another arbitrary quantity.** *This assumption is necessary to limit the computational complexity of the model and is used as a first approximation to introduce synchronisation of vessels.*
- **The shape of each chamber can be approximated by a bounding rectangle.** *The majority of lock chambers in the Netherlands is rectangular. The outliers can generally still be approximated by a rectangular shape.*
- **The size of a chamber's bounding rectangle can be described by a one-dimensional capacity parameter.** *This has the same reasoning as for the vessel size parameter.*

Finally, there are additional assumptions for the extra vessel processing time parameter:

- **The extra vessel processing time is independent of the number of other vessels in the lock.** *Maneuvering crowded or empty lock chambers may require different amounts of time.*

Furthermore, vessels may have to moor to each other instead of to the sides of the chamber, requiring a different amount of time. Accounting for this would require complex constraints and would create a large amount of extra input parameter values.

- **The extra vessel processing time is independent of the extra vessel being processed.** *Vessels of different sizes may require different amounts of time to position, moor, and leave the chamber. This is neglected to reduce the number of input parameter values that are required.*

4.2 SMPL Constraints

The starting point of this SMPL model's constraints is the constraints of the previous model. A set of synchronisation constraints will be added to model the behavior of vessels going through lock chambers at the same time. Additionally, some of the ordering constraints have to be augmented to allow for multi-vessel lockages without interference.

Synchronisation constraints are added Subsection 4.2.1 and changed ordering constraints are listed in Subsection 4.2.2.

4.2.1 Synchronisation Constraints

There were no synchronisation constraints in the single-vessel, single-chamber model, so these are all new additions. Let $s_{(i,j)}(k, k-\mu)$ be a max-plus binary synchronisation variable with the following definition:

$$s_{(i,j)}(k,k-\mu)\begin{cases} e & \text{If vessel } k \text{ and } k-\mu \text{ are synchronised on lock chamber arc } (i,j). \\ \varepsilon & \text{If vessel } k \text{ and } k-\mu \text{ are not synchronised on lock chamber arc } (i,j). \end{cases} \tag{4.1}$$

Where synchronised means that they are processed in the same lockage in chamber (i,j). Let $\overline{s}_{(i,j)}(k, k-\mu)$ be the adjoint of $s_{(i,j)}(k, k-\mu)$. If $s_{(i,j)}(k, k-\mu) = \varepsilon$ then that does not prevent vessels k and $k-\mu$ from passing through chamber (i,j) in separate lockages.

4.2.1.1 Lockage Synchronisation Constraint

If vessel k is synchronised with vessel $k-\mu$ on lock chamber (i,j) then one of those vessels may enter the chamber and prepare for the lockage earlier than the other. However, their shared lockage can only start once vessels k and $k-\mu$ have both entered the chamber. Thus, vessel k may only leave chamber (i,j) at $x_j(k)$ once both vessels have entered the chamber at $x_i(k)$ and $x_i(k-\mu)$, and once the minimum operation time $\tau_{(i,j)}(k)$ and one instance of extra vessel processing time $\overline{\tau}_{(i,j)}$ have passed. If vessel k is synchronised with more than one vessel, the time at which vessel k is allowed to leave the chamber is equal to the latest arrival time of any of the vessels in the lockage plus the operation time $\tau_{(i,j)}(k)$ and the extra vessel processing time $\overline{\tau}_{(i,j)}$ times the number of vessels vessel k is synchronised with. For $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\begin{aligned} x_j(k) \geq \left(x_i(k) \oplus \bigoplus_{k-\mu \in \mathcal{K}\setminus\{k\}} x_i(k-\mu) \otimes s_{(i,j)}(k,k-\mu) \right) \\ \otimes \tau_{(i,j)}(k) \otimes \bigotimes_{k-\mu \in \mathcal{K}\setminus\{k\}} \left((s_{(i,j)}(k,k-\mu) \otimes \overline{\tau}_{(i,j)}) \oplus e \right) \otimes w_{(i,j)}(k) \end{aligned} \tag{4.2}$$

Note that this constraint is inactive if $w_{(i,j)}(k) = \varepsilon$.

4.2.1.2 Synchronisation Variable Symmetry Constraint

If vessel k is synchronised with vessel $k-\mu$ on lock chamber arc (i,j) through $s_{(i,j)}(k,k-\mu)=e$, then vessel $k-\mu$ must also be synchronised with vessel k on arc (i,j) through $s_{(i,j)}(k-\mu,k)=e$ Similarly, if $s_{(i,j)}(k,k-\mu)=\varepsilon$, then $s_{(i,j)}(k-\mu,k)=\varepsilon$ must hold. Thus, for all $((i,j),k,k-\mu)\in\mathcal{D}_{locks}\times\mathcal{K}\times\mathcal{K}\setminus\{k\}$:

$$\big(s_{(i,j)}(k,k-\mu)\otimes s_{(i,j)}(k-\mu,k)\big)\oplus\big(\overline{s}_{(i,j)}(k,k-\mu)\otimes\overline{s}_{(i,j)}(k-\mu,k)\big)=e \tag{4.3}$$

4.2.1.3 Synchronisation Capacity Constraint

If vessel k is synchronised with other vessels on lock chamber (i,j), then the sum of their sizes $\rho(k)$ may not exceed the capacity of the chamber $c_{(i,j)}$. For all $((i,j),k)\in\mathcal{D}_{locks}\times\mathcal{K}$:

$$\rho(k)\otimes w_{(i,j)}(k)\otimes\bigotimes_{k-\mu\in\mathcal{K}\setminus\{k\}}(s_{(i,j)}(k,k-\mu)\otimes\rho(k-\mu)\oplus e)\leq c_{(i,j)} \tag{4.4}$$

Note that this constraint is only active when $w_{(i,j)}$ is active and that, even if vessel k is synchronised with no other vessels, its size $\rho(k)$ must still be smaller than the capacity $c_{(i,j)}$ if it passes through chamber (i,j).

4.2.1.4 Synchronisation and Routing Consistency Constraint

If synchronisation variable $s_{(i,j)}(k,k-\mu)$ synchronising vessel k with vessel $k-\mu$ on lock chamber arc (i,j) is true, then $w_{(i,j)}(k)$ must also be true. If $w_{(i,j)}(k)$ is false, then $s_{(i,j)}(k,k-\mu)$ must also be false. Otherwise, synchronisation variables of vessels that do not even use the chamber could interfere with the count of other vessels that do in the *Synchronisation Capacity* constraint. For all $((i,j),k,k-\mu)\in\mathcal{D}_{locks}\times\mathcal{K}\times\mathcal{K}\setminus\{k\}$:

$$(w_{(i,j)}(k)\otimes s_{(i,j)}(k,k-\mu))\oplus(w_{(i,j)}(k)\otimes\overline{s}_{(i,j)}(k,k-\mu))\oplus(\overline{w}_{(i,j)}(k)\otimes\overline{s}_{(i,j)}(k,k-\mu))=0. \tag{4.5}$$

4.2.2 Ordering Constraints

The *Same Travel Direction Ordering* constraint must be changed to prevent it from interfering with the synchronisation constraints.

4.2.2.1 Same Travel Direction Ordering Constraint

The original *Same Travel Direction* ordering constraint in Equation 3.16 is active for vessels k and $k-\mu$ on lock chamber arc (i,j) if $w_{(i,j)}(k)$, $w_{(i,j)}(k-\mu)$, and $z_{(i,j)}(k,k-\mu)$ are true. If $w_{(i,j)}(k)$ and $w_{(i,j)}(k-\mu)$ are true, then $s_{(i,j)}(k,k-\mu)$ may be true. This means that if $z_{(i,j)}(k,k-\mu)$ is then also true, the *Same Travel Direction* and the *Lockage Synchronisation* constraints are both active when only the *Lockage Synchronisation* constraint should be active. This scenario can be prevented by deactivating the *Same Travel Direction* ordering constraint if synchronisation variable $s_{(i,j)}(k,k-\mu)$ is true, or equivalently when its adjoint $\overline{s}_{(i,j)}(k,k-\mu)$ is false. For all $((i,j),k,k-\mu)\in\mathcal{D}_{locks}\times\mathcal{K}\times\mathcal{K}\setminus\{k\}$:

$$x_i(k)\geq x_j(k-\mu)\otimes\tau_{(i,j)}(k)\otimes w_{(i,j)}(k-\mu)\otimes w_{(i,j)}(k)\otimes z_{(i,j)}(k,k-\mu)\otimes\overline{s}_{(i,j)}(k,k-\mu). \tag{4.6}$$

4.3 SMPL System

The constraints can once again be organized into an SMPL system with system equations and a set of constraints constraining the allowable modes of the system.

Subsection 4.3.1 discusses the system equations and Subsection 4.3.2 discusses the allowable modes of the system.

4.3.1 System Equation

The larger-than-inequality constraints that have to do with the evolution of the arrival states are the *Departure Time*, *Travel Time*, *Same Travel Direction Ordering*, and *Opposite Travel Direction Ordering* constraints. These constraints can be assembled into an SMPL system in a procedure similar to that in Subsection 3.4.2.

Let $\boldsymbol{s}(k, k-\mu)$ be the vector of synchronisation variables between vessel k and $k-\mu$ on all lock chamber arcs:

$$\boldsymbol{s}(k, k-\mu) = \left[s_{(i,j)}(k, k-\mu) | \forall\, (i,j) \in \mathcal{D}_{locks}\right]^T. \tag{4.7}$$

Let $\boldsymbol{\phi}(k)$ now include the vectors $\boldsymbol{s}(k, k-\mu)$ between vessel k and all other vessels:

$$\boldsymbol{\phi}(k) = \begin{bmatrix} \boldsymbol{w}(k) \\ \boldsymbol{w}(k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{w}(k-\mu_{\min}(k)) \\ \boldsymbol{z}(k, k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{z}(k, k-\mu_{\min}(k)) \\ \boldsymbol{s}(k, k-\mu_{\min}(k)) \\ \vdots \\ \boldsymbol{s}(k, k-\mu_{\max}(k)) \end{bmatrix} \tag{4.8}$$

The synchronisation inequality constraints between vessel k and $k-\mu$ can be grouped into a synchronisation inequality constraint matrix $A_{synch,(k,k-\mu)}(\boldsymbol{\phi}(k))$. Together with the ordering constraint matrix $A_{ordering,(k,k-\mu)}$, this gives the inequality constraint matrix linking vessel k to vessel $k-\mu$:

$$A_{(k,k-\mu)}(\boldsymbol{\phi}(k)) = A_{ordering,(k,k-\mu)}(\boldsymbol{\phi}(k)) \oplus A_{synch,(k,k-\mu)}(\boldsymbol{\phi}(k)) \qquad \text{If } \mu \in \mathcal{M}(k). \tag{4.9}$$

When $k = k-\mu$, then $A_{(k,k-\mu)}(\boldsymbol{\phi}(k))$ is given by the routing constraint matrix:

$$A_{(k,k-\mu)}(\boldsymbol{\phi}(k)) = A_{routing,(k)}(\boldsymbol{\phi}(k)) \qquad \text{If } k = k-\mu. \tag{4.10}$$

The full set of SMPL system equations is then again given by:

$$\boldsymbol{x}(k) \geq \bigoplus_{\mu \in \mathcal{M}(k) \cup \{0\}} A_{(k,k-\mu)}(\boldsymbol{\phi}(k)) \otimes \boldsymbol{x}(k-\mu) \oplus B_{(k)} \otimes u(k), \tag{4.11}$$

for all vessels k.

4.3.2 Allowable Modes

Other than the *Route Start*, *Route End*, *Transit*, *Ordering Variable Consistency*, *Synchronisation Variable Symmetry*, and *Synchronisation and Routing Consistency* equality constraints, there is now also a smaller than inequality constraint in the *Synchronisation Capacity* constraint. As this inequality constraint does not include the arrival states, it is just another rule that determines the set of allowable modes $\mathcal{Q}(k)$ for each of the vessels.

Let the equality constraints again be grouped in:

$$\boldsymbol{q}_{(k)}(\boldsymbol{\phi}(k)) = \boldsymbol{y}_{(k)}. \tag{4.12}$$

Furthermore, let the *Synchronisation Capacity* constraints be grouped into:

$$\boldsymbol{C}_{(k)}(\boldsymbol{\phi}(k)) \leq \boldsymbol{c_{capacities}}, \tag{4.13}$$

where $\boldsymbol{C}_{(k)}(\boldsymbol{\phi}(k))$ is the smaller than inequality constraint vector for vessel k, and c is the vector of the chamber capacities of all of the chambers in the topology graph:

$$\boldsymbol{c_{capacities}} = \left[c_{(i,j)} | (i,j) \in \mathcal{D}_{locks}\right]^T. \tag{4.14}$$

Then Equation 4.12 and Equation 4.14 together determine the set of allowable switching modes $\mathcal{Q}(k)$ for each of the vessels.

4.4 MILP Problem Conversion

In this section, the new constraints are converted to conventional algebra and added to the MILP problem formulation. Additionally, a number of lockages performance indicator that counts correctly for the multi-vessel model is described.

Conversion of the SMPL constraints to MILP constraints is outlined in Subsection 4.4.1. The number of lockages performance indicator is discussed in Subsection 4.4.2.

4.4.1 Constraints

The new or augmented SMPL constraints are converted to constraints for the MILP in conventional algebra in this subsection.

4.4.1.1 Lockage Synchronisation Constraint

Define the auxiliary variable $\theta_{synch,(i,j)}(k, k-\mu)$ as:

$$\theta_{synch,(i,j)}(k, k-\mu) = x_i(k-\mu) s_{(i,j)}(k, k-\mu), \tag{4.15}$$

for all $((i,j), k, k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$, which can be linearly enforced by adding the following constraints [76]:

$$\theta_{synch,(i,j)}(k, k-\mu) \leq -\beta s_{(i,j)}(k, k-\mu) \tag{4.16}$$
$$\theta_{synch,(i,j)}(k, k-\mu) \geq 0 \tag{4.17}$$
$$\theta_{synch,(i,j)}(k, k-\mu) \leq x_i(k-\mu) \tag{4.18}$$
$$\theta_{synch,(i,j)}(k, k-\mu) \geq x_i(k-\mu) + \beta(1 - s_{(i,j)}(k, k-\mu)). \tag{4.19}$$

Furthermore, introduce the variable $t_{synch,(i,j)}(k)$ for all $((i,j), k) \in \mathcal{D}_{locks} \times \mathcal{K}$ as:

$$t_{synch,(i,j)}(k) = \max\left(x_i(k), \max(\{\theta_{synch,(i,j)}(k, k-\mu) | \forall k-\mu \in \mathcal{K} \setminus \{k\}\})\right), \tag{4.20}$$

which can be linearly enforced by a constraint:

$$t_{synch,(i,j)}(k) \geq x_i(k) \tag{4.21}$$

and a set of constraints for all $k-\mu \in \mathcal{K} \setminus \{k\}$:

$$t_{synch,(i,j)}(k) \geq \theta_{synch,(i,j)}(k, k-\mu). \tag{4.22}$$

Then the *Lockage Synchronisation* constraint can be linearly formulated for all $((i,j), k) \in \mathcal{D}_{locks} \times \mathcal{K}$ as:

$$t_{synch(i,j)}(k) - x_j(k) - \beta w_{(i,j)}(k) + \sum_{k-\mu \in \mathcal{K} \setminus \{k\}} s_{(i,j)}(k, k-\mu)\overline{\tau}_{(i,j)} \leq -\beta - \tau_{(i,j)}(k) \tag{4.23}$$

4.4.1.2 Synchronisation Variable Symmetry Constraint

For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$s_{(i,j)}(k,k-\mu) - s_{(i,j)}(k-\mu,k) \leq 0 \tag{4.24}$$

Note that this will become an equality constraint, as both the equation above and the equation:

$$s_{(i,j)}(k-\mu,k) - s_{(i,j)}(k,k-\mu) \leq 0, \tag{4.25}$$

will be added, enforcing the equality constraint $s_{(i,j)}(k-\mu,k-\mu) = s_{(i,j)}(k,k-\mu)$. This way of writing the constraint prevents duplicate constraint entries from being added to the problem.

4.4.1.3 Synchronisation Capacity Constraint

For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$-\beta w_{(i,j)}(k) + \sum_{k-\mu \in \mathcal{K} \setminus \{k\}} \rho(k-\mu) s_{(i,j)}(k,k-\mu) \leq c_{(i,j)} - \beta - \rho(k). \tag{4.26}$$

4.4.1.4 Synchronisation and Routing Consistency Constraint

Equation 4.5 can be condensed down to the statement that if $w_{(i,j)}$ is true, then $s_{(i,j)}(k,k-\mu)$ must be false. For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$s_{(i,j)}(k,k-\mu) - w_{(i,j)}(k) \leq 0 \tag{4.27}$$

4.4.1.5 Same Travel Direction Ordering Constraint

For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$\begin{aligned} x_j(k-\mu) - x_i(k) - \beta w_{(i,j)}(k) - \beta w_{(i,j)}(k-\mu) - \beta z_{(i,j)}(k,k-\mu) + \beta s_{(i,j)}(k,k-\mu) \\ \leq -3\beta - \tau_{(i,j)}(k) \end{aligned} \tag{4.28}$$

4.4.2 Number of Lockages Performance Indicator

All of the key performance indicators from Subsection 3.5.4, except for the number of lockages indicator, are still functional. The number of lockages indicator is updated in this subsection.

The lockage counting method outlined in Subsection 3.5.4 no longer works when multiple vessels can pass through the chamber in the same lockage. Instead, the ordered sequence of arrival times at the lock must be filtered. This filtering can be done by iterating through the sequence of arrival times. Any time the vessel in the previous index is synchronised with the vessel in the current index, the entry in the current index is not added to the filtered sequence. Once a filtered sequence has been established, the counting method from Subsection 3.5.4 can be used on that sequence to get the true number of lockages for that chamber, and all of the lockages on all chambers can be summed to get the total number of lockages $\mathcal{L}$.

A filtering example for a sequence on a chamber (i,j) with four vessels, where vessels 2 and 3 are synchronised is shown in Equation 4.29. In the sequence, vessels 2 and 3 travel through chamber (j,i) after each other, as indicated by $w_{(j,i)}(2)$ and $w_{(j,i)}(3)$, which would require an empty lockage in between to be counted. However, as the vessels are synchronised through $s_{(j,i)}(2,3)$ and $s_{(j,i)}(3,2)$, their lockage is only counted as a single lockage by removing $w_{(j,i)}(3)$ from the sequence. The filtered sequence is then treated the same as in Equation 3.62.

$$\begin{array}{rcccc} & \xrightarrow{\text{Sequence}} & & & s_{(j,i)}(2,3) = s_{(j,i)}(3,2) = 1 \\ \text{Unfiltered Sequence:} & w_{(i,j)}(0) = 1 & w_{(j,i)}(1) = 1 & w_{(j,i)}(2) = 1 & w_{(j,i)}(3) = 1 \\ \text{Filtered Sequence:} & w_{(i,j)}(0) = 1 & w_{(j,i)}(1) = 1 & w_{(j,i)}(2) = 1 & \end{array} \tag{4.29}$$

4.5 Conclusion

The research question to be answered in this chapter was:

> *How can multi-chamber locks with one-dimensional chamber capacity constraints be integrated into the SMPL IWT scheduling model?*

Real locks can have multiple chambers, that can each process multiple vessels at a time. Accurate modeling of large-scale IWT systems would require the model to be able to include these. Furthermore, the scheduling of vessels passing multi-vessel, multi-chamber locks has not been formulated as an SMPL system before. The single-vessel, single-chamber model from Chapter 3 was used as a starting point to integrate these new features into a multi-vessel, multi-chamber capacity model.

Multi-chamber locks were trivially integrated by adding two nodes for the chamber entry and exit points, two lock arcs for both travel directions of the chamber, and four arcs to and from the waiting area entry and exit points per extra chamber to lock constructions in the graph topology input. The routing system introduced in Chapter 3 was already fully capable of distributing vessels between different chambers of a lock if there had been multiple chambers. Different chambers of the same lock may have different parameters, but they share the same waiting areas.

The state-of-the-art in lock scheduling literature is to implement multi-vessel chambers through the ship placement problem, at the cost of increased computational difficulty. To avoid this computational difficulty, vessels and chambers were given one-dimensional sizes and capacities, as also done in [42]. If the sum of the sizes of the vessels in a lockage is smaller than a chamber's capacity, then the lockage is possible. This comes with the advantage of a less complex model, but with the disadvantage that lockages that are possible according to the one-dimensional system may sometimes not be possible in two-dimensional reality.

SMPL constraints were introduced to allow the synchronization of vessels in the same lockages. Limited changes to the existing ordering constraints were also required. SMPL system equations were formulated based on the larger-than-inequality constraints on the vessel arrival states, with the allowable system modes given by a combination of the logical constraints and the constraint ensuring that vessels in the same lockage stay below the chamber capacity.

The SMPL model was converted to an MILP model in conventional algebra. The scheduler can then use the rules of this model to create scheduling optimization problems to find vessels' schedules based on input graph topology and vessel parameters on a case-by-case basis, similar to the procedure already described in Chapter 3. The scheduled variables are the arrival times of the vessels at the nodes, which arcs the vessels' routes consist of, the ordering between vessels on lock chambers, and if vessels are synchronized on lock chambers.

With the exception of the number of lockages KPI, which had to be augmented to account for the fact that a lockage may now contain multiple vessels, the objectives and KPIs from the single-vessel, single-chamber model are still applicable to this model.

5

Multi-Vessel, Multi-Chamber Scheduling with Ship Placement

A lockmaster plans lockages based on vessel arrival times and the configuration of vessels that actually fits in the chamber while obeying applicable regulations. The state of the art in the lock scheduling problem is to integrate the placement of the vessels into the optimization problem by integrating the ship placement problem [32]. The model developed in Chapter 4 does account for different vessel and chamber sizes, but stops short of ensuring that the vessels that are planned to pass through the chamber together truly fit into it together.

The goal of this chapter is then to truly place the vessels of a lockage into the chamber by integrating the ship placement problem into the multi-vessel, multi-chamber SMPL IWT scheduling model, matching or exceeding the state-of-the-art for lock scheduling problems in modeling complexity, while introducing the novel contributions of an SMPL model formulation, choice of routing, and support for arbitrary network topologies.

The research question for this chapter is then:

> *How can ship placement be integrated into the multi-chamber SMPL IWT scheduling model?*

The changes to the problem definition are listed in Section 5.1. The SMPL constraints and system are derived in Section 5.2 and Section 5.3, respectively. The SMPL constraints are converted to MILP constraints in Section 5.4, followed by some practical optimization considerations in Section 5.5. Finally, a conclusion is given in Section 5.6.

5.1 Problem Definition

The mathematical model developed in this chapter builds on the model developed in Chapter 4, which in turn used large parts of the model developed in Chapter 3. Only the changes in problem definition compared to those models are mentioned in this section.

The sliding procedure with which vessels will be placed into the chambers is introduced in Subsection 5.1.1. The variables and parameters associated with sliding vessel placement are listed in Subsection 5.1.2. Mooring, safety distances, and vessels carrying special goods are discussed in Subsection 5.1.3. New and changed assumptions with respect to earlier models are listed in Subsection 5.1.4. Finally, the decision variables that constitute a schedule are summarized in Subsection 5.1.5.

5.1.1 Sliding Ship Placement

It is common practice in the literature to assign continuous variables to the horizontal (orthogonal to the direction of travel in the chamber) and vertical (parallel to the direction of travel in the

chamber) positions of vessels in the chamber. Placement constraints are then defined with these variables.

In reality, vessels enter the chamber in sequence, vertically slotting into their assigned position in the chamber, which is generally as far forward as possible while being moored to the side of the chamber or another vessel. Every entering vessel is vertically stacked sequentially into the chamber, as shown in Figure 5.1 (a). Vessels that have already entered the chamber may prevent other vessels from moving forward. This is similar to the Tetris system discussed in Subsection 2.1.2. If vessels are all modeled as rectangular Tetris blocks, and if the horizontal direction of the chamber is discretised into several bins which the vessels can slide into, then the ship placement sequence can be modeled like the Tetris system in [56].

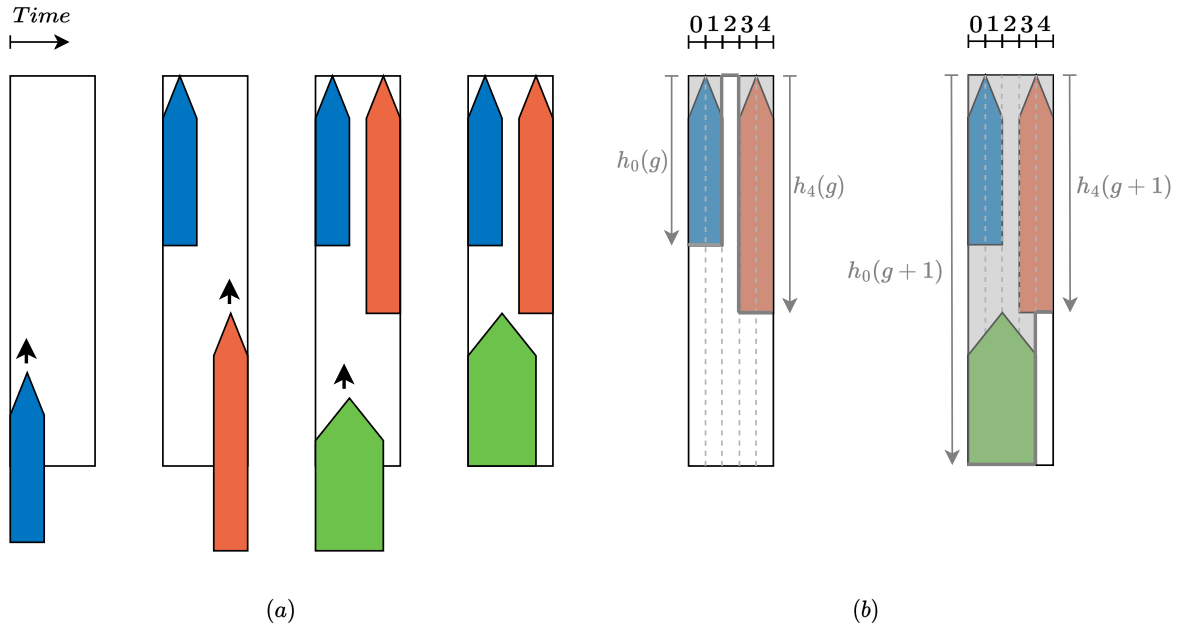


Figure 5.1: (a) : *Example sequence of vessels entering a lock chamber.* (b) *Illustration of bin stack heights evolving from event step* g *to* $g+1$*. The gray area represents the area of the stacks in the bins. Area that is not covered is still available for other vessels to slide into.*

Consider the next example. Let $g = \{0, 1, ..., n_{steps}\}$ be an integer event counter, where a single vessel can be slid into the chamber for each event. Figure 5.1 (b) shows the chamber discretised into 5 equally sized horizontal bins, where the stack height $h_i(g)$ at step g in each of the bins i is a function of the maximum stern position of each of the vessels that has a part of the vessel in bin i. When the green vessel is slid into bins $0-3$ on step g, its furthest forward bow position is equal to $\max(h_0(g), h_1(g), h_2(g), h_3(g))$, and as a result, defining the length of the green vessel as W_γ, the heights of the bins on event step $g+1$ are given by:

$$h_0(g+1) = \max(h_0(g) + W_\gamma, h_1(g) + W_\gamma, h_2(g) + W_\gamma, h_3(g) + W_\gamma) \tag{5.1}$$
$$h_1(g+1) = \max(h_0(g) + W_\gamma, h_1(g) + W_\gamma, h_2(g) + W_\gamma, h_3(g) + W_\gamma) \tag{5.2}$$
$$h_2(g+1) = \max(h_0(g) + W_\gamma, h_1(g) + W_\gamma, h_2(g) + W_\gamma, h_3(g) + W_\gamma) \tag{5.3}$$
$$h_3(g+1) = \max(h_0(g) + W_\gamma, h_1(g) + W_\gamma, h_2(g) + W_\gamma, h_3(g) + W_\gamma) \tag{5.4}$$
$$h_4(g+1) = \max(h_4(g)) \tag{5.5}$$

or equivalently:

$$\begin{bmatrix} h_0(g+1) \\ h_1(g+1) \\ h_2(g+1) \\ h_3(g+1) \\ h_4(g+1) \end{bmatrix} = \begin{bmatrix} W_\gamma & W_\gamma & W_\gamma & W_\gamma & \varepsilon \\ W_\gamma & W_\gamma & W_\gamma & W_\gamma & \varepsilon \\ W_\gamma & W_\gamma & W_\gamma & W_\gamma & \varepsilon \\ W_\gamma & W_\gamma & W_\gamma & W_\gamma & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & e \end{bmatrix} \otimes \begin{bmatrix} h_0(g) \\ h_1(g) \\ h_2(g) \\ h_3(g) \\ h_4(g) \end{bmatrix}. \tag{5.6}$$

With the heights of the bins at each event step g being the states, Equation 5.6 describes part of an SMPL system, which switches based on the vessel that comes sliding into the chamber at a particular event step. The process of vessels being placed into a chamber can thus be modeled as an SMPL system, with the vessels of a lockage sliding into the chamber one by one.

5.1.2 Ship Placement Variables and Parameters

The example introduced in Subsection 5.1.1 must be adapted for use in the wider framework of the SMPL IWT scheduling model.

The geometrical parameters of an arbitrary chamber with a vessel are displayed in Figure 5.2. Each chamber (i, j) is discretised into $N_{\chi,(i,j)}$ equally sized horizontal bins, starting at bin 0. The true width that a bin represents (in length units) is equivalent for all chambers and all vessels in the graph. $W_\chi(k)$ is the width of vessel k in number of bins. $W_\chi(k) = 3$ in Figure 5.2. $W_\gamma(k)$ is the length of vessel k in meters. Finally, $N_{\gamma,(i,j)}$ is the vertical length of chamber (i, j) in meters. Note that there is no need to discretise the length of the chamber and the vessels.

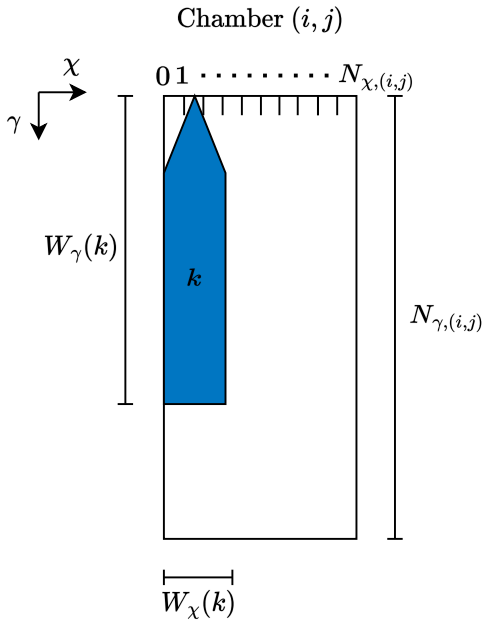


Figure 5.2: *Geometrical parameters of a chamber (i, j) and a vessel k.*

The height of the vessel stack in bin χ of chamber (i, j) at step g for vessel k is denoted as $h_{\chi,(i,j)}(k, g)$. Note that the height of the stack in the bin is dependent on both the vessel and the step. This is so vessels that do not pass through the chamber in the same lockage can have different bin heights in the same chamber on the same event step. Let $\boldsymbol{h}_{(i,j)}(k, g)$ be the ordered

vector of all bin heights of vessel k in chamber (i,j) on step g:

$$\boldsymbol{h}_{(i,j)}(k,g) = \begin{bmatrix} h_{0,(i,j)}(k,g) \\ h_{1,(i,j)}(k,g) \\ \vdots \\ h_{N_{\chi,(i,j)}-1,(i,j)}(k,g) \end{bmatrix}. \tag{5.7}$$

Let $\alpha_{\chi,(i,j)}(k,g)$ be a max-plus binary variable that determines if vessel k is slid into chamber (i,j) in bin χ on event step g. The terminology 'in bin χ' means that the top left corner of the bounding box of the vessel lines up with the left-most point of bin χ. Vessel k has been slid into bin 0 in Figure 5.2.

The number of steps n_{steps} determines the maximum amount of vessels that can be slid into the chamber in the same lockage. More steps also increase the complexity of the problem by increasing the number of decision variables. A safe assumption can be made by setting $n_{steps} = |\mathcal{K}|$, potentially allowing all vessels to be slid into the same chamber on the same lockage, as long as the other constraints of the problem are satisfied. That is what is done for the tests in this thesis. Future work should try to find a tighter value for n_{steps}.

Figure 5.3 shows the step-by-step process of filling a chamber (i,j) according to these variables. The values of the bin heights for each step are given in Table 5.1.

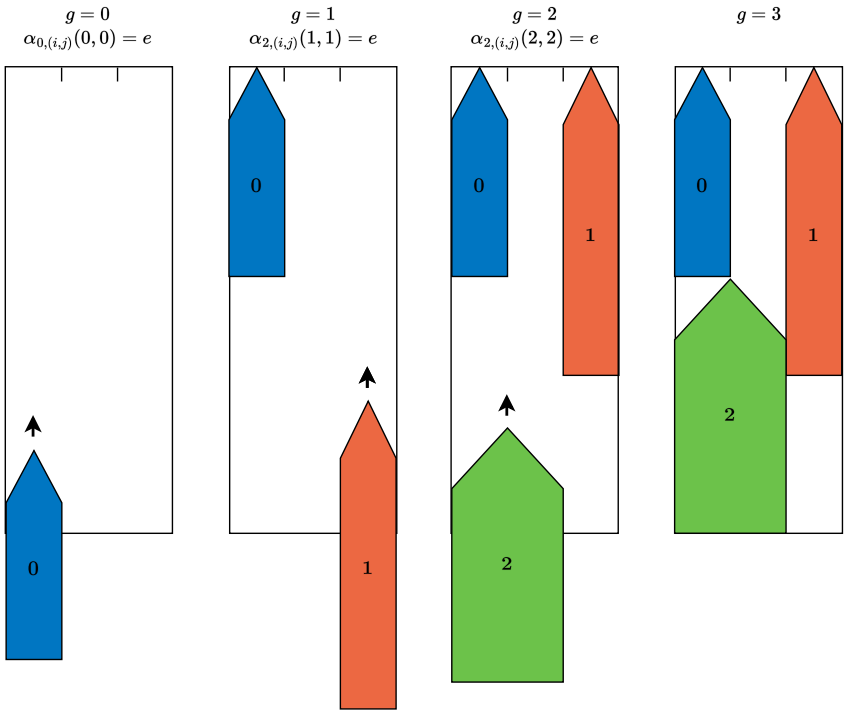


Figure 5.3: *Filling a chamber (i,j) with three bins and three vessels.*

Table 5.1: *Bin height values of the bins in the example in Figure 5.3.*

Bin Height	$g=0$	$g=1$	$g=2$	$g=3$
$h_{0,(i,j)}(k,g)\,\forall\, k \in \{0,1,2\}$	e	$W_\gamma(0)$	$W_\gamma(0)$	$W_\gamma(0) \otimes W_\gamma(2)$
$h_{1,(i,j)}(k,g)\,\forall\, k \in \{0,1,2\}$	e	e	e	$W_\gamma(0) \otimes W_\gamma(2)$
$h_{2,(i,j)}(k,g)\,\forall\, k \in \{0,1,2\}$	e	e	$W_\gamma(1)$	$W_\gamma(1)$

5.1.3 Mooring, Safety Distance and Vessels with Dangerous Substances

The ship placement problem is similar to the 2D bin-packing problem. The point of difference for most cases in the literature is the mooring and safety distance constraints [19].

Vessels are not allowed to use any form of mechanical propulsion while the lockage is taking place [18]. To prevent drifting, the vessels must be moored to the side of a chamber or to the side of another vessel. If vessel k is moored to vessel $k-\mu$, then vessel k is reliant on vessel $k-\mu$ to prevent drifting. Vessel $k-\mu$ can then not also be reliant on vessel k to prevent drifting. A number of rules have been adapted from the mooring constraints in [19]:

- A vessel can only be moored to the sides of the chamber if it is adjacent to them.
- Vessels that are moored to each other must be adjacent.
- A vessel must be moored to at least either a vessel or the sides of the chamber.
- If a vessel k is moored to vessel $k-\mu$, $k-\mu$ cannot be moored to vessel k. Equivalently, this means that a string of vessels moored to each other must eventually have one vessel moored to the sides of the chamber.
- A vessel k that is moored to another vessel $k-\mu$, must be fully alongside vessel $k-\mu$. That is, the bow of vessel k may not be more forward in the chamber than the bow of vessel $k-\mu$, and the stern of vessel k may not be further back in the chamber than the stern of vessel $k-\mu$.

Figure 5.4 shows a number of acceptable and unacceptable configurations. Vessels moored to another vessel are indicated by a black arrow.

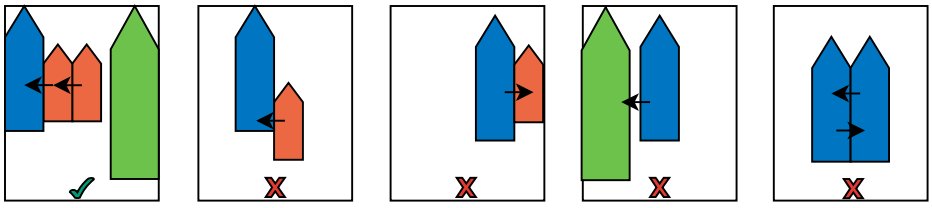


Figure 5.4: *Vessel mooring configurations.* (a)*: Acceptable configuration.* (b)*: The smaller vessel is not fully alongside and the larger vessel is not moored to anything.* (c)*: The larger vessel cannot be fully alongside the smaller vessel.* (d)*: The vessels are not adjacent.* (e)*: The vessels are both fully adjacent to each other, but neither of them are moored to the side of the chamber.*

Mooring will be integrated into the ship placement problem by introducing a mooring decision variable $m_{(i,j)}(k, k-\mu)$, which decides if vessel k is moored to vessel $k-\mu$ on chamber (i,j).

Vessels must keep some distance to each other to allow for safe maneuvering. Some vessels may also require tugs to be positioned in the locks, and these tugs require room to be able to leave the lock. Other vessels carry flammable substances, which by Dutch regulations requires all other vessels to keep at least 10 meters of distance to that vessel [18]. A vessel with flammable substances must display a single inverted blue cone.

Safety distances will, by approximation, be dealt with by including the extra distance into the size of the vessel that is used for the ship placement problem, as displayed for a vessel with flammable cargo in Figure 5.5. Including safety distances through constraints would add a large amount of complexity to the problem. To avoid confusion, visualizations of the solutions of the ship placement

problem will use the size that is used in the placement problem, not the true vessel size. A real scenario may require vessels to shift around in their assigned box including safety distances, to moor to each other for instance.

The additional length added to the vessels should not lead to problems. On the other hand, the discretisation of the chamber into horizontal bins may lead to scenarios where the width assigned to a vessel in the bin placement problem is excessive. If a chamber has 3 bins, and a vessel that has a true width of two bins requires an extra meter of safety distance, then that requires a placement size of three bins, leading to a 50% increase in virtual width. The effects of this discretisation phenomenon can be reduced by adding more bins to the chamber, reducing the space taken up by a horizontal bin. However, this comes at the cost of extra binary variables.

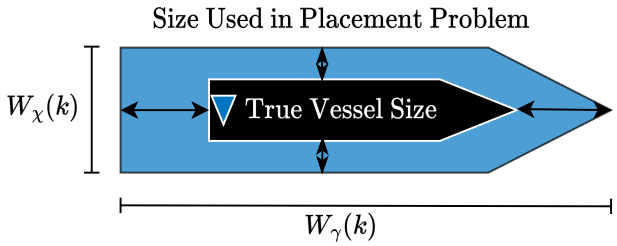


Figure 5.5: *Placement problem vessel shape example with added safety distances for a vessel with flammable goods, as indicated by the upside down blue cone.*

Vessels with toxic goods are required to display a symbol of two inverted blue cones and vessels with explosive goods are required to display three inverted blue cones. Vessels with toxic and explosive goods are required to be processed in separate lockages from other vessels, with some exceptions [18]. Vessels with flammable goods may not be processed in the same lockage as passenger vessels.

Subsets of vessels with flammable goods, $\mathcal{K}_\triangledown \subseteq \mathcal{K}$, and explosive or toxic goods, $\mathcal{K}_{\triangledown\triangledown} \subseteq \mathcal{K}$, are added to the model.

Passenger vessels are not treated as a separate vessel class, so vessels with flammable goods in $\mathcal{K}_\triangledown$ are allowed to be placed with any other vessel in the model. Increased safety distances should be kept in mind when determining the vessel size. Furthermore, as vessels with flammable goods are required to keep 10 meters of distance in reality, they will not be allowed to moor to other vessels in the model, as that distance would be too big for mooring.

Vessels in the $\mathcal{K}_{\triangledown\triangledown}$ subset are not allowed to be processed in the same lockage as any other vessel, neglecting any exceptions to this rule to prevent having to introduce many more vessel subsets. An implementation of this system in practice would have to account for all different vessel groups, and the rules set up for $\mathcal{K}_{\triangledown\triangledown}$ can be used as a template for doing so.

5.1.4 Assumptions

A number of new assumptions are introduced with respect to those already introduced in Subsection 3.1.6 and Subsection 4.1.4:

- **The width of the lock chambers and vessels is discretised based on a set width per section.** *Real vessels and chambers can have any width, but the horizontal direction must be discretised into a set number of bins to make the sliding SMPL model work. The number of bins for chambers and vessels should be rounded up to avoid running into infeasible placement solutions.*

- **All vessels can moor to all other vessels, as long as the vessel that is mooring to the other vessel can be fully alongside the vessel that is being moored to, and as long as the vessel is not carrying flammable goods.** *Significant height above water differences between vessels may prevent mooring. Furthermore, IWT vessels may not be allowed to moor to seagoing vessels [19]. These cases are neglected for this model.*
- **Safety distances that should be maintained between vessels for maneuvering and because of regulations are included in the sizes used for placement of the vessels.** *This is a conservative approximation to avoid the introduction of many complicated constraints in combination with the way the SMPL ship placement sequence is implemented. In some cases, this may lead to there being too much safety distance. Two vessels with flammable goods will, for instance, be placed to keep 20 meters of distance in a real scenario, when only 10 meters of distance is required.*
- **Vessels with toxic and explosive goods may not be synchronised with any other vessels.** *This is according to Dutch regulations, but it does neglect the distinction between toxic and explosive goods and some exceptions that apply to certain vessels with toxic goods [18].*
- **No distinction is made between passenger and cargo vessels.** *Dutch regulations require that vessels with flammable goods do not get processed in the same lockage as passenger vessels [18]. This is neglected to avoid creating more required input parameters.*
- **A vessel must always be moored to the sides of the chamber or another vessel.** *Prevents drifting in the absence of mechanical propulsion.*

5.1.5 Schedule

If the model in this chapter is used in the scheduler, it not only decides the routing, arrival times, ordering, and synchronisation of vessels, but it also decides the placement sequence of the vessels in the chamber. Figure 5.6 gives an overview of the decision variables a schedule consists of.

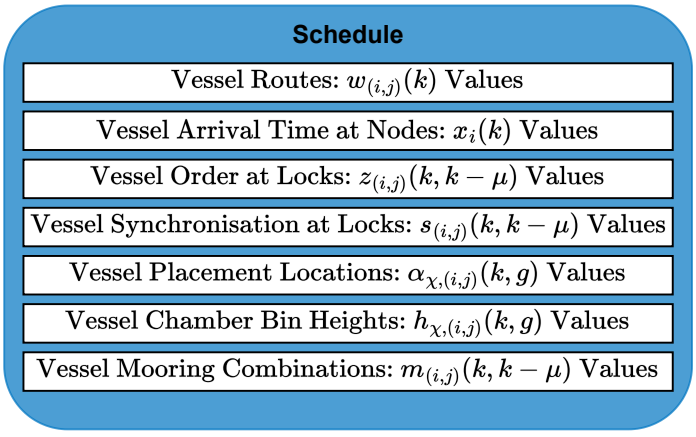


Figure 5.6: *Components of a schedule generated by the multi-vessel, multi-chamber placement model.*

5.2 SMPL Constraints

Up until now, the constraints have been split into routing, ordering, and synchronisation constraints, according to commonly used SMPL system conventions. For clarity, two extra constraint groups are added:

- **Placement Constraints:** Constraints that concern the placement of the vessels in the chambers in the problem.
- **Mooring Constraints:** Constraints that concern the mooring of vessels to one another.

As this model builds on the model introduced in the previous chapters, only the new additions or changes made are mentioned here. This model will be the first model where some of the constraints do not fit into the SMPL system framework. These constraints are discussed in Section 5.4.

A synchronisation constraint for toxic and explosive vessels is introduced in Subsection 5.2.1. The placement and mooring constraints are described in Subsection 5.2.2 and Subsection 5.2.3. Finally, a note on non-SMPL constraints is made in Subsection 5.2.4.

5.2.1 Toxic and Explosive Vessel Synchronisation Constraint

The routing and ordering constraints are unaltered. The *Synchronisation Capacity* constraints introduced to count the vessel sizes are no longer necessary.

A hazardous vessel k in the set of hazardous vessels $\mathcal{K}_{\nabla\nabla}$ may not be processed in the same lockage as any other vessel on any of the chambers (i,j) in the set of chambers $\mathcal{D}_{locks}$. This can be enforced by ensuring that all of the synchronisation variables with vessel $k \in \mathcal{K}_{\nabla\nabla}$ are false. For all $k \in \mathcal{K}_{\nabla\nabla}$:

$$\bigotimes_{(i,j)\in\mathcal{D}_{locks}} \bigotimes_{k-\mu\in\mathcal{K}} \left(\overline{s}_{(i,j)}(k,k-\mu)\right) = e \tag{5.8}$$

5.2.2 Placement Constraints

These constraints dictate the proper placement sequence of the vessels.

5.2.2.1 Initial Condition Constraint

The chamber must be empty at the start of the chamber-filling process. For a vessel k and a chamber (i,j) the bin heights in all bins should be at least e on step $g = 0$. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\boldsymbol{h_{(i,j)}}(k,0) \geq \boldsymbol{e}, \tag{5.9}$$

where $\boldsymbol{e}$ is an $N_{\chi,(i,j)}$ entry long vector of e.

5.2.2.2 System Update Constraint

For a vessel k, if vessel $k-\mu$ is synchronised with vessel k on chamber (i,j), and if vessel $k-\mu$ is slid into bin χ of the chamber on step g according to $\alpha_{\chi,(i,j)}(k-\mu,g)$, then the length of vessel $k-\mu$ will add to the bin height values of vessel k in that chamber for step $g+1$. By the example in Subsection 5.1.1, for all $(k,k-\mu,g,(i,j),\chi) \in \mathcal{K}\times\mathcal{K}\setminus\{k\}\times\{0,1,...,n_{steps}-1\}\times\mathcal{D}_{locks}\times\{0,1,...,N_{\chi,(i,j)}|W_\chi(k-\mu)+\chi \leq N_{\chi,(i,j)}\}$:

$$\boldsymbol{h_{(i,j)}}(k,g+1) \geq s_{(i,j)}(k,k-\mu) \otimes \alpha_{\chi,(i,j)}(k-\mu,g) \otimes A_{\chi,(i,j)}(k-\mu) \otimes \boldsymbol{h_{(i,j)}}(k,g), \tag{5.10}$$

where $\{0,1,...,N_{\chi,(i,j)}|W_{\chi(k-\mu)} \leq N_{\chi,(i,j)}\}$ is the set of values of χ which vessel $k-\mu$ can be placed into without sticking out of the right side of the chamber, as shown in figure Figure 5.7, and where $A_{\chi,(i,j)}(k-\mu)$ is the ship placement matrix for when vessel $k-\mu$ is placed in bin χ of vessel (i,j). $A_{\chi,(i,j)}(k)$ is of shape $N_{\gamma,(i,j)} \times N_{\gamma,(i,j)}$, with the following general form:

$$A_{\chi,(i,j)}(k) = \begin{bmatrix} E(\chi,\chi) & \mathcal{E}(\chi,W_\chi(k)) & \mathcal{E}(\chi,N_{\gamma,(i,j)}-W_\chi(k)-\chi) \\ \mathcal{E}(W_\chi(k),\chi) & \overline{W}_\gamma(k) & \mathcal{E}(W_\chi(k),N_{\gamma,(i,j)}-W_\chi(k)-\chi) \\ \mathcal{E}(N_{\gamma,(i,j)}-W_\chi(k)-\chi,\chi) & \mathcal{E}(N_{\gamma,(i,j)}-W_\chi(k)-\chi,W_\chi(k)) & E(N_{\gamma,(i,j)}-W_\chi(k)-\chi,N_{\gamma,(i,j)}-W_\chi(k)-\chi)) \end{bmatrix}, \tag{5.11}$$

where $\overline{W}_\gamma(k)$ is a full $W_\chi(k) \times W_\chi(k)$ matrix of $W_\gamma(k)$:

$$\overline{W}_\gamma(k) = \begin{bmatrix} W_\gamma(k) & \dots & W_\gamma(k) \\ \vdots & \ddots & \vdots \\ W_\gamma(k) & \dots & W_\gamma(k) \end{bmatrix}. \tag{5.12}$$

The ship placement matrix does not exist when the vessel is too large to be placed in that particular bin, either in width or in length.

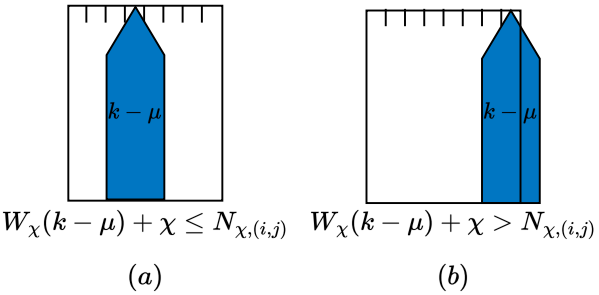


Figure 5.7: (a)*: Vessel is not too wide to be placed in that bin.* (b)*: Vessel is too wide to be placed in that bin.*

If vessel k itself is placed into the chamber, it should also update its own bin height states. For all $(k, g, (i, j), \chi) \in \mathcal{K} \times \{0, 1, n_{steps} - 1\} \times \mathcal{D}_{locks} \times \{0, 1, ..., N_{\chi,(i,j)} | W_\chi(k) + \chi \leq N_{\chi,(i,j)}\}$:

$$h_{(i,j)}(k, g + 1) \geq \alpha_{\chi,(i,j)}(k, g) \otimes A_{\chi,(i,j)}(k) \otimes h_{(i,j)}(k, g) \tag{5.13}$$

5.2.2.3 Bin Height Synchronisation Constraint

If vessel k is synchronised with vessel $k - \mu$ on chamber (i, j), their bin heights must be the same on that chamber for all event steps g. For all $((i, j), k, k - \mu, g) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\} \times \{1, 2, ..., n_{steps}\}$:

$$\boldsymbol{h_{(i,j)}}(k, g) \geq s_{(i,j)}(k, k - \mu) \otimes \boldsymbol{h_{(i,j)}}(k - \mu, g). \tag{5.14}$$

Note that, if the constraint in Equation 5.14 is present, then:

$$\boldsymbol{h_{(i,j)}}(k - \mu, g) \geq s_{(i,j)}(k - \mu, k) \otimes \boldsymbol{h_{(i,j)}}(k, g) \tag{5.15}$$

is also present. This enforces equality of the bin height states between vessels k and $k - \mu$ when the two vessels are synchronised on chamber (i, j). If $s_{(i,j)}(k, k - \mu) = s_{(i,j)}(k - \mu, k) = e$, then by Equation 5.14 and Equation 5.15:

$$\boldsymbol{h}_{(i,j)}(k, g) \geq \boldsymbol{h}_{(i,j)}(k - \mu, g) \tag{5.16}$$

$$\boldsymbol{h}_{(i,j)}(k - \mu, g) \geq \boldsymbol{h}_{(i,j)}(k, g), \tag{5.17}$$

so together this becomes:

$$\boldsymbol{h}_{(i,j)}(k, g) = \boldsymbol{h}_{(i,j)}(k - \mu, g). \tag{5.18}$$

5.2.2.4 Bin Height Increase Constraint

The *System Update* constraint is only relevant on step g when a vessel actually enters the chamber on that step, otherwise Equation 5.10 and Equation 5.13 both trivially resolve to:

$$\boldsymbol{h_{(i,j)}}(k, g) \geq \varepsilon, \tag{5.19}$$

losing the built up height. The following constraint can be added to ensure that the progress in building up bin height for vessel k is maintained for steps without a vessel entering the chamber (i, j). For all $((i, j), k, g) \in \mathcal{D}_{locks} \times \mathcal{K} \times \{0, 1, ..., n_{steps} - 1\}$:

$$\boldsymbol{h_{(i,j)}}(k, g + 1) \geq \boldsymbol{h}_{(i,j)}(k, g). \tag{5.20}$$

5.2.2.5 Routing and Placement Consistency Constraint

If a vessel k is routed through chamber (i,j) by routing variable $w_{(i,j)}(k)$, it must be placed into one of the bins χ of the chamber during one of the steps g through placement variable $\alpha_{\chi,(i,j)}(k,g)$. It may be placed into that chamber at most once. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\bigoplus_{g\in\{0,1,...,n_{steps}-1\}} \bigoplus_{\chi\in\{0,1,...,N_{\chi,(i,j)-1}|\chi+W_\chi(k)\leq N_{\chi,(i,j)}\}} \Big(w_{(i,j)}(k) \otimes \alpha_{\chi,(i,j)}(k,g) \otimes \bigotimes_{p\in\{0,1,...,n_{steps}-1\}\setminus\{g\}} \bigotimes_{l\in\{0,1,...,N_{\chi,(i,j)-1}\}\setminus\{\chi\}} \overline{\alpha}_{l,(i,j)}(k,p)\Big) \oplus \Big(\overline{w}_{(i,j)}(k) \otimes \bigotimes_{g\in\{0,1,...,n_{steps}\}} \bigotimes_{\chi\in\{0,1,...,N_{\chi,(i,j)}-1\}} \overline{\alpha}_{\chi,(i,j)}(k,g)\Big) = e \quad (5.21)$$

Note that the use of the set $\{0,1,...,N_{\chi,(i,j)}-1\}$ for the adjoint variables only ensures that the variables $\alpha_{\chi,(i,j)}(k,g)$ for the bins where the width of the vessel would not fit into the width of the chamber are never active.

5.2.2.6 Single Vessel per Step Constraint

Only a single vessel may enter a chamber (i,j) per step g. For a vessel k, that means that of vessel k and the vessels it is synchronised with on the chamber, at most a single vessel may be slid into a bin χ on step g. For all $(k,k-\mu,(i,j),\chi,g) \in \mathcal{K}\times\mathcal{K}\setminus\{k\}\times\mathcal{D}_{locks}\times\{0,1,...,N_{\chi,(i,j)}-1\}\times\{0,1,...,n_{steps}-1\}$:

$$\begin{aligned}&\Big(s_{(i,j)}(k,k-\mu)\otimes\alpha_{\chi,(i,j)}(k,g)\otimes\bigotimes_{p\in\{0,1,...,N_{\chi,(i,j)}-1\}}\overline{\alpha}_{p,(i,j)}(k-\mu,g)\Big)\\ &\oplus\Big(s_{(i,j)}(k,k-\mu)\otimes\bigotimes_{p\in\{0,1,...,N_{\chi,(i,j)}-1\}}\overline{\alpha}_{p,(i,j)}(k,g)\otimes\alpha_{\chi,(i,j)}(k-\mu,g)\Big)\\ &\oplus\Big(s_{(i,j)}(k,k-\mu)\otimes\overline{\alpha}_{\chi,(i,j)}(k,g)\otimes\bigotimes_{p\in\{0,1,...,N_{\chi,(i,j)}-1\}}\overline{\alpha}_{p,(i,j)}(k-\mu,g)\Big)\\ &\oplus\Big(s_{(i,j)}(k,k-\mu)\otimes\bigotimes_{p\in\{0,1,...,N_{\chi,(i,j)}-1\}}\overline{\alpha}_{p,(i,j)}(k,g)\otimes\overline{\alpha}_{\chi,(i,j)}(k-\mu,g)\Big)\\ &\oplus\overline{s}_{(i,j)}(k,k-\mu)=e \quad (5.22)\end{aligned}$$

5.2.3 Mooring Constraints

The mooring constraints define the rules for the proper mooring of vessels to each other or the sides of the chamber. These constraints require the introduction of auxiliary variables that keep track of the position of the bow and the stern of each of the vessels. Let $\hat{h}_{(i,j)}(k)$ be the vertical position of the bow of vessel k in chamber (i,j), defined as:

$$\hat{h}_{(i,j)}(k) = \bigoplus_{g\in\{0,1,...,n_{steps}-1\}} \bigoplus_{\chi\in\{0,1,...,N_{\chi,(i,j)}-1|\chi+W_\chi(k)\leq N_{\chi,(i,j)}\}} \Big(\alpha_{\chi,(i,j)}(k,g)\otimes\bigoplus_{p\in\{0,1,...,W_\chi(k)-1|p+W_\chi(k)\leq N_{\chi,(i,j)}\}} h_{\chi+p,(i,j)}(k,g)\Big), \quad (5.23)$$

Let $\check{h}_{(i,j)}$ be the vertical position of the stern of vessel k in chamber (i,j), defined as:

$$\check{h}_{(i,j)}(k) = \hat{h}_{(i,j)}(k)\otimes W_\gamma(k). \quad (5.24)$$

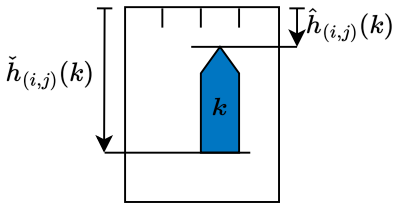


Figure 5.8: *Bow position $\hat{h}_{(i,j)}(k)$ and stern position $\check{h}_{(i,j)}(k)$ of a vessel in chamber (i,j).*

$\hat{h}_{(i,j)}(k)$ and $\check{h}_{(i,j)}(k)$ are shown in Figure 5.8.

Additionally, the mooring decision variable $m_{(i,j)}(k, k-\mu)$ is introduced with the following definition:

$$m_{(i,j)}(k,k-\mu) = \begin{cases} e & \text{If vessel } k \text{ is moored to vessel } k-\mu \text{ in chamber } (i,j). \\ \varepsilon & \text{If vessel } k \text{ is not moored to vessel } k-\mu \text{ in chamber } (i,j). \end{cases} \tag{5.25}$$

5.2.3.1 Mooring Synchronisation Constraint

Vessel k can only be moored to vessel $k-\mu$ on chamber (i,j) if the vessels are synchronised. For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$\begin{aligned} \Big(m_{(i,j)}(k,k-\mu) \otimes s_{(i,j)}(k,k-\mu)\Big) \oplus \Big(s_{(i,j)}(k,k-\mu) \otimes \overline{m}_{(i,j)}(k,k-\mu)\Big) \\ \oplus \Big(\overline{s}_{(i,j)}(k,k-\mu) \otimes \overline{m}_{(i,j)}(k,k-\mu)\Big) = e \end{aligned} \tag{5.26}$$

5.2.3.2 Single Vessel Mooring Constraint

Vessel k can be moored to at most one vessel in chamber (i,j). For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\begin{aligned} \Big(\bigotimes_{k-\mu \in \mathcal{K}\setminus\{k\}} \overline{m}_{(i,j)}(k,k-\mu)\Big) \\ \oplus \bigoplus_{k-\mu \in \mathcal{K}\setminus\{k\}} \Big(m_{(i,j)}(k,k-\mu) \otimes \bigotimes_{p \in \mathcal{K}\setminus\{k,k-\mu\}} \overline{m}_{(i,j)}(k,p)\Big) = e \end{aligned} \tag{5.27}$$

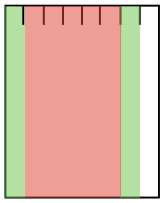
5.2.3.3 Chamber Side Adjacency Constraint

If a vessel k is placed adjacent to the sides of chamber (i,j), it may not be moored to another vessel, so none of its mooring variables may be true. If vessel k is placed in chamber (i,j), but is not adjacent to the sides of the chamber, one of its mooring variables must be true. For all

$((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\begin{aligned}
&\Big(\overline{w}_{(i,j)}(k) \otimes \bigotimes_{k-\mu \in \mathcal{K}\setminus\{k\}} \overline{m}_{(i,j)}(k, k-\mu)\Big) \\
&\qquad \oplus \Big(w_{(i,j)} \otimes \bigoplus_{k-\mu \in \mathcal{K}\setminus\{k\}} \\
&\bigoplus_{g \in \{0,1,\ldots,n_{steps}-1\}} \bigoplus_{\chi \in \{1,2,\ldots,N_{\chi,(i,j)}-2\} | \chi + W_\chi(k) + 1 \leq N_{\chi,(i,j)}\}} \big(\alpha_{\chi,(i,j)}(k,g) \otimes m_{(i,j)}(k, k-\mu)\big)\Big) \\
&\qquad \oplus \Big(w_{(i,j)}(k) \otimes \bigoplus_{k-\mu \in \mathcal{K}\setminus\{k\}} \\
&\bigoplus_{g \in \{0,1,\ldots,n_{steps}-1\}} \bigoplus_{\chi \in \{0, N_{\chi,(i,j)} - W_\chi(k)\}} \alpha_{\chi,(i,j)}(k,g) \otimes \overline{m}_{(i,j)}(k, k-\mu)\Big) = e
\end{aligned} \tag{5.28}$$

Note that the possible values of χ are split into a set in which vessel k is adjacent to the side of the chamber, and a set where it is not. An explanation for these sets is given in with an example in Figure 5.9.



If $W_\chi(k) = 2$:
$\chi \in \{0, N_{\chi,(i,j)} - W_\chi(k)\}$: Vessel k moored to side of chamber.
$\chi \in \{1, 2, \ldots, N_{\chi,(i,j)} - 2\} | \chi + W_\chi(k) + 1 \leq N_{\chi,(i,j)}\}$:
Vessel k moored to other vessel.

Figure 5.9: *Example of the bins in which a vessel k with $W_\chi(k) = 2$ would be moored to the side of the chamber (green), and the bins in which it would have to be moored to another vessel (red). The top left corner of the vessel cannot be placed into the rightmost bin, as the vessel would be too wide for the chamber.*

5.2.3.4 Vessel Adjacency Constraint

A vessel k can only be moored to a vessel $k - \mu$ on chamber (i, j) if they are adjacent. However, it is not required that two vessels be moored to each other if they are adjacent, as shown in the example in Figure 5.10. Vessel 2 is adjacent to vessels 1 and 3, but it is only moored to vessel 3.

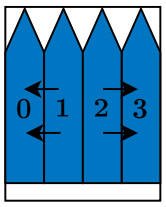


Figure 5.10: *Example of a situation where vessels that are adjacent are not necessarily moored to each other.*

For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$\Big(m_{(i,j)}(k,k-\mu) \otimes \bigoplus_{l\in\{0,1,\dots,n_{steps}-1\}} \bigoplus_{g\in\{0,1,\dots,n_{steps}-1\}} \bigoplus_{\chi\in\{1,2,\dots,N_{\chi,(i,j)}-W_\chi(k)-1\}} \bigoplus_{q\in\{\chi-W_\chi(k-\mu),\chi+W_\chi(k)\}} \alpha_{\chi,(i,j)}(k,g) \otimes \alpha_{q,(i,j)}(k-\mu,l)\Big) \oplus \overline{m}_{(i,j)}(k,k-\mu) = e \quad (5.29)$$

The set $\{1,2,...,N_{\chi,(i,j)} - W_\chi(k) - 1\}$ checks all of the bins χ in which vessel k can be placed where mooring would be required, and the set $\{\chi - W_\chi(k-\mu), \chi + W_\chi(k)\}$ then checks if vessel $k-\mu$ is placed to the left or right of vessel k if it is placed in bin χ.

5.2.3.5 Mooring Anti-symmetry Constraint

If vessel k is moored to vessel $k-\mu$ on chamber (i,j), then vessel $k-\mu$ cannot be moored to vessel k on that same chamber. For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$\big(m_{(i,j)}(k,k-\mu) \otimes \overline{m}_{(i,j)}(k-\mu,k)\big) \oplus \big(\overline{m}_{(i,j)}(k,k-\mu) \otimes m_{(i,j)}(k-\mu,k)\big) \oplus \big(\overline{m}_{(i,j)}(k,k-\mu) \otimes \overline{m}_{(i,j)}(k-\mu,k)\big) = e \quad (5.30)$$

The *Fully Alongside* constraint already mostly prevents two vessels from being moored to each other if they are of different lengths. However, in the specific scenario when two vessels are the same length, they can both be fully alongside each other as shown in Figure 5.4 (e), so this constraint was introduced to prevent the vessels from mooring to each other in that scenario.

5.2.3.6 Flammable Vessel Constraint

Vessels in the set $\mathcal{K}_\triangledown$ must keep 10 meters of distance from all other vessels, so they may not be moored to any other vessels. This indirectly enforces that a flammable vessel must always be moored to the sides of the chamber. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}_\triangledown$:

$$\bigoplus_{k-\mu\in\mathcal{K}\setminus\{k\}} m_{(i,j)}(k,k-\mu) = \varepsilon \quad (5.31)$$

5.2.4 Non-SMPL Constraints

For the mathematical model to properly be able to model reality, two more sets of constraints are required on the bin height states. The first is a constraint on the height of the stacks in the bins not exceeding the chamber length. The second is the constraint to ensure that a vessel that is moored to another vessel is fully alongside the other vessel.

Both constraints require constraining the state variable of the system to be smaller than another quantity, which does not fit into the SMPL framework for state evolution equations as outlined in Subsection 2.2.4. The one-dimensional capacity model in Chapter 4 did contain a smaller-than-inequality constraint in the *Synchronisation Capacity* constraint, but this solely constrained the allowable switching modes through the switching variables, it did not constrain the state, so the state evolution could still be written in the general SMPL system format in Equation 4.11.

It is not required to ensure that every single constraint can be written as a SMPL system constraint, as long as the final MILP implementation that is used in the scheduler is correct. The SMPL system framework has been used as a convenient tool to model the IWT systems in this report, and all constraints have been fully compatible with it up until now. The limitation that smaller than or equal to constraints cannot be implemented can be circumvented by just adding them once the SMPL

model has been converted to a MILP formulation, where smaller than or equal to constraints on the system states are possible, which is what will be done in Subsection 5.4.1.

As a result, the system that will be described in Section 5.3 does not encompass the full mathematical model that is implemented in the scheduler, but the system description is still given.

5.3 SMPL Systems

The SMPL mathematical model is now a combination of two SMPL systems: one to schedule the vessels' arrival time states, and one to place the vessels into the chamber. Both of these systems are linked by the max-plus binary switching variables, like the synchronisation and mooring decision variables in the *Mooring Synchronisation* constraint.

The system equations for both systems are described in Subsection 5.3.1. The allowable modes of the combination of the two systems are described in Subsection 5.3.2.

5.3.1 System Equations

The larger-than-inequality constraints that determine that state evolution equation for the arrival states are the *Departure Time*, *Travel Time*, *Same Travel Direction Ordering*, *Opposite Travel Direction Ordering* and *Chamber Entry Ordering* constraint. For the bin height states, these are the *System Update*, *Bin Height Synchronisation* and *Bin Height Increase* constraints.

Let $\boldsymbol{\alpha_{(i,j)}}(k,g)$ be the vector of placement variables $\alpha_{\chi,(i,j)}(k,g)$ of vessel k on chamber (i,j) at event step g:

$$\boldsymbol{\alpha_{(i,j)}}(k,g) = \begin{bmatrix} \alpha_{0,(i,j)}(k,g) \\ \alpha_{1,(i,j)}(k,g) \\ \vdots \\ \alpha_{N_{\chi,(i,j)}-1,(i,j)}(k,g) \end{bmatrix}. \tag{5.32}$$

Let $\boldsymbol{\alpha_{(i,j)}}(k)$ be the vector of $\boldsymbol{\alpha_{(i,j)}}(k,g)$ for all event steps g:

$$\boldsymbol{\alpha_{(i,j)}}(k) = \begin{bmatrix} \boldsymbol{\alpha_{(i,j)}}(k,0) \\ \boldsymbol{\alpha_{(i,j)}}(k,1) \\ \vdots \\ \boldsymbol{\alpha_{(i,j)}}(k,n_{steps}-1) \end{bmatrix}. \tag{5.33}$$

Let $\boldsymbol{\alpha}(k)$ be the vector of $\boldsymbol{\alpha_{(i,j)}}(k)$ for all chambers (i,j):

$$\boldsymbol{\alpha}(k) = \left[\boldsymbol{\alpha_{(i,j)}}(k) | \forall (i,j) \in \mathcal{D}_{locks}\right]^T. \tag{5.34}$$

Let $\boldsymbol{m}(k,k-\mu)$ be the the vector of $\boldsymbol{m_{(i,j)}}(k,k-\mu)$:

$$\boldsymbol{m}(k,k-\mu) = \left[\boldsymbol{m_{(i,j)}}(k,k-\mu) | \forall (i,j) \in \mathcal{D}_{locks}\right]^T. \tag{5.35}$$

Let the switching variable vector $\phi(k)$ for vessel k then be defined as:

$$\boldsymbol{\phi}(k) = \begin{bmatrix} \boldsymbol{w}(k) \\ \boldsymbol{w}(k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{w}(k-\mu_{\min}(k)) \\ \boldsymbol{z}(k,k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{z}(k,k-\mu_{\min}(k)) \\ \boldsymbol{s}(k,k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{s}(k,k-\mu_{\min}(k)) \\ \boldsymbol{\alpha}(k) \\ \boldsymbol{\alpha}(k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{\alpha}(k-\mu_{\min}(k)) \\ \boldsymbol{m}(k,k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{m}(k,k-\mu_{\min}(k)) \end{bmatrix}. \tag{5.36}$$

The inequality constraint matrix $A_{(k,k-\mu)}(\boldsymbol{\phi}(k))$, similar to the description in Section 4.3, is then defined as:

$$A_{(k,k-\mu)}(\boldsymbol{\phi}(k)) = \begin{cases} A_{routing,(k)}(\boldsymbol{\phi}(k)) & \text{If } \mu = 0 \\ A_{ordering,(k,k-\mu)}(\boldsymbol{\phi}(k)) \oplus A_{synch,(k,k-\mu)}(\boldsymbol{\phi}(k)) & \text{If } \mu \in \mathcal{M}(k) \end{cases} \tag{5.37}$$

The full set of arrival state evolution equations for vessel k then has the same definition as in Equation 4.11.

The arrival state evolution equations only evolve as a function of vessel counter k, whereas the bin height evolution equations evolve as a function of both the vessel counter k and the event counter g. Let $h(k,g)$ be the vector of $\boldsymbol{h}_{(\boldsymbol{i},\boldsymbol{j})}(k,g)$ for all chambers for vessel k at event step g:

$$\boldsymbol{h}(k,g) = \left[\boldsymbol{h}_{(\boldsymbol{i},\boldsymbol{j})}(k,g) | \forall (i,j) \in \mathcal{D}_{locks}\right]^T. \tag{5.38}$$

The *System Update* and *Bin Height Increase* constraints link the entries in the bin height state vector $\boldsymbol{h}(k,g+1)$ at event step $g+1$ for vessel k to entries in the bin height state vector of itself or other vessels at the previous event step g. The *Bin Height Synchronisation* constraint links the bin height state vector $\boldsymbol{h}(k,g)$ at event step g for vessel k to the entries in the bin height state vector of itself or other vessels at the same event step. Furthermore, the *Initial Condition* constraint at step $g=0$ can also be written as an input to the system for every single event step, as the bin heights have to be larger than e on every event step if they start at e at the very first step.

Let $A_{placement,(k,k-\mu)}(\boldsymbol{\phi}(k))$ be a grouping of the *System Update* and *Bin Height Increase* constraints between vessel k and vessel $k-\mu$, with $\mu \in \mathcal{M}(k) \cup \{0\}$, thus including vessel k. Let $A_{binsynch,(k,k-\mu)}(\boldsymbol{\phi}(k))$ be a grouping of the *Bin Height Synchronisation* constraints between vessel k and vessel $k-\mu$, with $\mu \in \mathcal{M}(k)$, excluding vessel k, as a vessel cannot be synchronised with itself. Finally, let $B_{initial}$ be a square max-plus unit matrix of appropriate size for $\boldsymbol{h}(k,g+1)$, and let $\boldsymbol{e}$ be a vector of e of compatible size. Then the system equation for the bin height states of vessel k

at step g can be written as:

$$\boldsymbol{h}(k, g+1) \geq \bigoplus_{\mu \in \mathcal{M}(k) \cup \{0\}} A_{placement,(k,k-\mu)}(\boldsymbol{\phi}(k)) \otimes \boldsymbol{h}(k-\mu, g+1) \oplus \bigoplus_{\mu \in \mathcal{M}(k)} A_{binsynch,(k,k-\mu)}(\boldsymbol{\phi}(k)) \otimes \boldsymbol{h}(k-\mu, g+1) \oplus B_{initial} \otimes \boldsymbol{e}, \quad (5.39)$$

where the *Initial Condition* constraint is given as a constant system input.

5.3.2 Allowable Modes

Similar to the procedures in Chapter 3 and Chapter 4, the logical equality constraints determine the set of allowable modes $\mathcal{Q}(k)$ each of the vessels is allowed to be in for the combination of the SMPL arrival time state and placement systems. Note that the *Synchronisation Capacity Constraint* from the capacity model has been removed, so the grouping of smaller-than-inequality constraints like in Equation 4.13 is no longer required.

Let the logical equality constraints again be grouped in:

$$\boldsymbol{q}_{(k)}(\boldsymbol{\phi}(k)) = \boldsymbol{y}_{(k)}, \quad (5.40)$$

which describes the set of allowable modes $\mathcal{Q}(k)$ of combined arrival time state and placement systems for each vessel k. If every vessel in the system is in an allowable state, then the entire system is in an allowable state.

Note that the constraints given for the auxiliary variables $\hat{h}_{(i,j)}(k)$ and $\hat{h}_{(i,j)}(k)$ are not considered logical constraints, and they are not included in Equation 5.40. These auxiliary variables are defined for constraints that will be added outside of the SMPL framework in Subsection 5.4.1.

5.4 MILP Conversion

The SMPL constraints are converted to MILP constraints in this section. New MILP-only constraints are added in Subsection 5.4.1. The *Toxic and Explosive Vessel Synchronisation* constraint is converted in Subsection 5.4.2. The placement constraints are converted in Subsection 5.4.3. Finally, the mooring constraints are converted in Subsection 5.4.4.

5.4.1 New Constraints

As discussed in Subsection 5.2.4, two additional constraints are required to properly simulate real lock operations, that do not fit into the SMPL system framework. These are the *Chamber Length* and *Fully Alongside* constraints.

5.4.1.1 Chamber Length Constraint

The bin height $h_{(i,j)}(k, g)$ for vessel k on chamber (i, j) at step g may not exceed the chamber length $N_{\gamma,(i,j)}$, as the length of lined up vessels cannot exceed the length of the chamber. For all $((i, j), k, g, \chi) \in \mathcal{D}_{locks} \times \mathcal{K} \times \{1, 2, ..., n_{steps}\} \times \{0, 1, ...N_{\chi,(i,j)} - 1\}$:

$$h_{\chi,(i,j)}(k, g) \leq N_{\gamma,(i,j)}. \quad (5.41)$$

5.4.1.2 Fully Alongside Constraint

If vessel k is moored to vessel $k - \mu$ on chamber (i, j), vessel k must be fully alongside vessel $k - \mu$, which can be constrained through their bow and stern positions. For all $((i, j), k, k - \mu) \in$

$\mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$\hat{h}_{(i,j)}(k) - \hat{h}_{(i,j)}(k-\mu) - \beta m_{(i,j)}(k, k-\mu) \leq -\beta \tag{5.42}$$

$$\check{h}_{(i,j)}(k-\mu) - \check{h}_{(i,j)}(k) - \beta m_{(i,j)}(k, k-\mu) \leq -\beta \tag{5.43}$$

5.4.2 Toxic and Explosive Vessel Synchronisation Constraint

The *Toxic and Explosive Vessel Synchronisation* constraint is the only routing, ordering, or synchronisation constraint that has to be converted. For all $k \in \mathcal{K}_{\nabla\nabla}$:

$$\sum_{(i,j)\in\mathcal{D}_{locks}} \sum_{k-\mu\in\mathcal{K}\setminus\{k\}} s_{(i,j)}(k, k-\mu) \leq 0. \tag{5.44}$$

5.4.3 Placement Constraints

This subsection contains the MILP conversions of the placement constraints.

5.4.3.1 Initial Condition Constraint

For all $((i,j), k, \chi) \in \mathcal{D}_{locks} \times \mathcal{K} \times \{0, 1, ..., N_{\chi,(i,j)} - 1\}$:

$$h_{\chi,(i,j)}(k) \leq 0 \tag{5.45}$$

$$-h_{\chi,(i,j)}(k) \leq 0. \tag{5.46}$$

5.4.3.2 System Update Constraint

It is no longer convenient to write this constraint with vector notation. Instead, it is written on a single-entry basis, enforcing maximization of the height of the individual bins according to all combinations of the bin height values and the $W_\gamma(k)$ entries of the ship placement matrix. e entries of the ship placement matrix, the entries that correspond to the bins where the vessel being placed do not add any height, are accounted for by the bin height increase constraint.

For all $(k, k-\mu, g, (i,j), \chi, p) \in \mathcal{K} \times \mathcal{K} \setminus \{k\} \times \{0, 1, ..., n_{steps} - 1\} \times \mathcal{D}_{locks} \times \{0, 1, ..., N_{\chi,(i,j)} - 1 | \chi + W_\chi(k-\mu) \leq N_{\chi,(i,j)}\} \times \{0, 1, ..., W_\chi(k-\mu) - 1\}$:

$$h_{\chi,(i,j)}(k, g) - h_{\chi+p,(i,j)}(k, g+1) - \beta s_{(i,j)}(k, k-\mu) - \beta\alpha_{\chi,(i,j)}(k-\mu) \leq -2\beta - W_\gamma(k-\mu). \tag{5.47}$$

For all $(k, g, (i,j), \chi, p) \in \mathcal{K} \times \{0, 1, ..., n_{steps} - 1\} \times \mathcal{D}_{locks} \times \{0, 1, ..., N_{\chi,(i,j)} - 1 | \chi + W_\chi(k) \leq N_{\chi,(i,j)}\} \times \{0, 1, ..., W_\chi(k) - 1\}$:

$$h_{\chi,(i,j)}(k, g) - h_{\chi+p,(i,j)}(k, g+1) - \beta\alpha_{\chi,(i,j)}(k-\mu) \leq -\beta - W_\gamma(k). \tag{5.48}$$

Note that if $p + \chi + 1 \leq N_{\chi,(i,j)}$ does not exist for combinations of $\{\chi\} \times \{0, 1, ..., W_\chi(k-\mu) - 1\}$ for Equation 5.47, and $\{\chi\} \times \{0, 1, ..., W_\chi(k) - 1\}$ for Equation 5.48, then vessel $k-\mu$ or k that is being placed is too wide to be placed in the chamber at χ, and no constraints should be added for that combination of χ, (i,j), and $k-\mu$ or k.

5.4.3.3 Bin Height Increase Constraint

For all $((i,j), k, g, \chi) \in \mathcal{D}_{locks} \times \mathcal{K} \times \{0, 1, ..., n_{steps} - 1\} \otimes \{0, 1, ..., N_{\chi,(i,j)} - 1|\}$:

$$h_{\chi,(i,j)}(k, g) - h_{\chi,(i,j)}(k, g+1) \leq 0. \tag{5.49}$$

5.4.3.4 Routing and Placement Consistency Constraint

For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\sum_{\chi \in \{N_{\chi,(i,j)-1}\}} \sum_{g \in \{0,1,...,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g) - w_{(i,j)}(k) \leq 0 \tag{5.50}$$

$$-\sum_{\chi \in \{N_{\chi,(i,j)-1}\}} \sum_{g \in \{0,1,...,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g) + w_{(i,j)}(k) \leq 0 \tag{5.51}$$

5.4.3.5 Single Vessel Per Step Constraint

A single vessel may be slid into the chamber per timestep. If vessel k is slid into the chamber in bin χ at step g, then if it is synchronised with vessel $k-\mu$, that vessel may not be slid into any of the bins of the chamber at step g. On the other hand, if the vessels are not synchronised, then they are not part of the same placement sequences and they may both be placed in the same chamber in the same timestep. For all $(k, k-\mu, (i,j), \chi, g) \in \mathcal{K} \times \mathcal{K} \setminus \{k\} \times \mathcal{D}_{locks} \times \{0,1,...,N_{\chi,(i,j)}\} \times \{0,1,...,n_{steps}-1\}$:

$$\alpha_{\chi,(i,j)}(k,g) + \sum_{p \in \{0,1,...,N_{\chi,(i,j)-1}\}} \alpha_{p,(i,j)}(k-\mu,g) + s_{(i,j)}(k,k-\mu) \leq 2 \tag{5.52}$$

5.4.4 Mooring Constraints

This subsection contains the MILP conversions of the mooring constraints.

5.4.4.1 Bow Position $\hat{h}_{(i,j)}(k)$

Define the auxiliary variable $t_{bow,\chi,(i,j)}(k,g)$ for all $(g,\chi) \in \{0,1,...,n_{steps}-1\} \times \{0,1,...,N_{\chi,(i,j)} - W_\chi(k)\}$:

$$t_{bow,\chi,(i,j)}(k,g) = \max(\{h_{\chi+p,(i,j)}(k,g) | \forall p \in \{0,1,...,W_\chi(k)-1\}\}), \tag{5.53}$$

which can be linearly enforced by introducing the binary variables $\lambda_{\chi+p,(i,j)}(k,g)$ and the constraints that for all $p \in \{0,1,...,W_\chi(k)-1\}$:

$$h_{\chi+p,(i,j)}(k,g) - t_{bow,\chi,(i,j)}(k,g) \leq 0 \tag{5.54}$$

$$t_{bow,(i,j),(i,j)}(k,g) - h_{\chi+p,(i,j)}(k,g) - \beta\lambda_{\chi+p,(i,j)}(k,g) \leq -\beta, \tag{5.55}$$

along with:

$$\sum_{p \in \{0,1,...,W_\chi(k)-1\}} \lambda_{\chi+p,(i,j)}(k,g) \leq 1 \tag{5.56}$$

$$-\sum_{p \in \{0,1,...,W_\chi(k)-1\}} \lambda_{\chi+p,(i,j)}(k,g) \leq -1. \tag{5.57}$$

Then define another auxiliary variable $\theta_{bow,\chi,(i,j)}(k,g)$ for all $(g,\chi) \in \{0,1,...,n_{steps}-1\} \times \{0,1,...,N_{\chi,(i,j)} - W_\chi(k)\}$:

$$\theta_{bow,\chi,(i,j)}(k,g) = \alpha_{\chi,(i,j)}(k,g) t_{bow,\chi,(i,j)}(k,g+1), \tag{5.58}$$

which is enforced through the following constraints:

$$\theta_{bow,\chi,(i,j)}(k,g) + \beta\alpha_{\chi,(i,j)}(k,g) \leq 0 \tag{5.59}$$

$$-\theta_{bow,\chi,(i,j)}(k,g) \leq 0 \tag{5.60}$$

$$\theta_{bow,\chi,(i,j)}(k,g) - t_{bow,\chi,(i,j)}(k,g) \leq 0 \tag{5.61}$$

$$-\theta_{bow,\chi,(i,j)}(k,g) + t_{bow,\chi,(i,j)}(k,g) - \alpha_{\chi,(i,j)}(k,g) \leq -\beta. \tag{5.62}$$

Finally, it can be defined that for all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\hat{h}_{(i,j)}(k) = \max(\{\theta_{bow,\chi,(i,j)}(k,g) | (g,\chi) \in \{0,1,...,n_{steps}-1\} \times \{0,1,...,N_{\chi,(i,j)}-1\}\}). \quad (5.63)$$

This is in turn enforced by introducing binary variable $\lambda_{\chi,(i,j)}(k,g)$ for each $\theta_{bow,\chi,(i,j)}(k,g)$ alongside the following constraints:

$$\theta_{bow,\chi,(i,j)}(k,g) - \hat{h}_{(i,j)}(k) \leq 0 \quad (5.64)$$

$$\hat{h}_{(i,j)}(k) - \theta_{bow,\chi,(i,j)}(k,g) - \beta\lambda_{\chi,(i,j)}(k,g) \leq \beta, \quad (5.65)$$

and:

$$\sum_{g\in\{0,1,...,n_{steps}-1\}} \sum_{\chi\in\{0,1,...,N_{\chi,(i,j)}-1\}} \lambda_{\chi,(i,j)}(k,g) \leq 1 \quad (5.66)$$

$$-\sum_{g\in\{0,1,...,n_{steps}-1\}} \sum_{\chi\in\{0,1,...,N_{\chi,(i,j)}-1\}} \lambda_{\chi,(i,j)}(k,g) \leq -1. \quad (5.67)$$

5.4.4.2 Stern Position $\check{h}_{(i,j)}(k)$

For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\check{h}_{(i,j)}(k) - \hat{h}_{(i,j)}(k) \leq W_\gamma(k) \quad (5.68)$$

$$-\check{h}_{(i,j)}(k) + \hat{h}_{(i,j)}(k) \leq -W_\gamma(k) \quad (5.69)$$

5.4.4.3 Single Vessel Mooring Constraint

For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\sum_{k-\mu\in\mathcal{K}\setminus\{k\}} m_{(i,j)}(k,k-\mu) \leq 1. \quad (5.70)$$

5.4.4.4 Chamber Side Adjacency Constraint

For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\sum_{\chi\in\{1,2,...,N_{\chi,(i,j)}-2|\chi+W_\chi(k)+1\leq N_{\chi,(i,j)}\}} \sum_{g\in\{0,1,...,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g) - \sum_{k-\mu\in\mathcal{K}\setminus\{k\}} m_{(i,j)}(k,k-\mu) \leq 0 \quad (5.71)$$

5.4.4.5 Synchronisation and Mooring Consistency Constraint

For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \setminus \{k\}$:

$$m_{(i,j)}(k,k-\mu) - s_{(i,j)}(k,k-\mu) \leq 0 \quad (5.72)$$

5.4.4.6 Vessel Adjacency Constraint

Define the binary variables $\psi_{\chi-W_\chi(k-\mu),(i,j)}(k,k-\mu)$ and $\psi_{\chi+W_\chi(k),(i,j)}(k,k-\mu)$. $\psi_{\chi-W_\chi(k-\mu),(i,j)}(k,k-\mu)$ only exists if $\chi - W_\chi(k-\mu) \geq 0$ for chamber (i,j). Similarly, $\psi_{\chi+W_\chi(k),(i,j)}(k,k-\mu)$ only exists if $\chi + W_\chi(k) \leq N_{\chi,(i,j)}$ for chamber (i,j).

$$\psi_{\chi-W_\chi(k-\mu),(i,j)}(k,k-\mu) = \left(\sum_{g\in\{0,1,...,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g)\right) \cdot \left(\sum_{g\in\{0,1,...,n_{steps}-1} \alpha_{\chi-W_\chi(k-\mu),(i,j)}(k-\mu,g)\right) \quad (5.73)$$

$$\psi_{\chi+W_\chi(k),(i,j)}(k,k-\mu) = \left(\sum_{g\in\{0,1,\dots,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g)\right) \cdot \left(\sum_{g\in\{0,1,\dots,n_{steps}-1} \alpha_{\chi+W_\chi(k),(i,j)}(k-\mu,g)\right) \quad (5.74)$$

The sums on the right-hand side can only ever be equal to 0 or 1, by virtue of the *Routing and Placement Consistency* constraint, so their entirety can be treated as a binary variable. These relationships can be enforced by the following constraints:

$$-\left(\sum_{g\in\{0,1,\dots,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g)\right) + \psi_{\chi-W_\chi(k-\mu),(i,j)}(k,k-\mu) \leq 0 \quad (5.75)$$

$$-\left(\sum_{g\in\{0,1,\dots,n_{steps}-1} \alpha_{\chi-W_\chi(k-\mu),(i,j)}(k-\mu,g)\right) + \psi_{\chi-W_\chi(k-\mu),(i,j)}(k,k-\mu) \leq 0 \quad (5.76)$$

$$\left(\sum_{g\in\{0,1,\dots,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g)\right) + \left(\sum_{g\in\{0,1,\dots,n_{steps}-1} \alpha_{\chi-W_\chi(k-\mu),(i,j)}(k-\mu,g)\right) - \psi_{\chi-W_\chi(k-\mu),(i,j)}(k,k-\mu) \leq 1 \quad (5.77)$$

$$-\left(\sum_{g\in\{0,1,\dots,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g)\right) + \psi_\chi W_{\chi+W_\chi(k),(i,j)}(k,k-\mu) \leq 0 \quad (5.78)$$

$$-\left(\sum_{g\in\{0,1,\dots,n_{steps}-1} \alpha_{\chi+W_\chi(k),(i,j)}(k-\mu,g)\right) + \psi_\chi W_{\chi+W_\chi(k),(i,j)}(k,k-\mu) \leq 0 \quad (5.79)$$

$$\left(\sum_{g\in\{0,1,\dots,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g)\right) + \left(\sum_{g\in\{0,1,\dots,n_{steps}-1} \alpha_{\chi+W_\chi(k),(i,j)}(k-\mu,g)\right) - \psi_{\chi+W_\chi(k),(i,j)}(k,k-\mu) \leq 1. \quad (5.80)$$

Then the following must hold for the adjacency constraint, for all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$m_{(i,j)}(k,k-\mu) - \sum_{\chi\in\{W_\chi(k-\mu),W_\chi(k-\mu)+1,\dots,N_{\chi,(i,j)}-2 | \chi+W_\chi(k)+1\leq N_{\chi,(i,j)}\}} \Big(\psi_{\chi+W_\chi(k),(i,j)}(k,k-\mu) + \psi_{\chi-W_\chi(k-\mu),(i,j)}(k,k-\mu)\Big) \leq 0. \quad (5.81)$$

5.4.4.7 Fully Alongside Constraint

For all $(k,k-\mu) \in \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$\hat{h}(k-\mu) - \hat{h}(k) - \beta m_{(i,j)}(k,k-\mu) \leq -\beta \quad (5.82)$$

$$\check{h}(k) - \check{h}(k-\mu) - \beta m_{(i,j)}(k,k-\mu) \leq -\beta. \quad (5.83)$$

5.4.4.8 Mooring Anti-Symmetry Constraint

For all $((i,j),k,k-\mu) \in \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$m_{(i,j)}(k,k-\mu) + m_{(i,j)}(k-\mu,k) \leq 1. \quad (5.84)$$

5.4.4.9 Flammable Vessel Constraint

For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}_{\triangledown}$:

$$\sum_{k-\mu \in \mathcal{K} \setminus \{k\}} m_{(i,j)}(k, k-\mu) = 0 \tag{5.85}$$

5.4.5 Objectives

The addition of placement raises the question of whether objectives should be added that specifically target the placement of the vessels, for instance penalizing vessels not being placed as far forward as possible. The choice was made to not add those in this report.

Any lockage generated by the scheduling model will be feasible according to the constraints. If it is beneficial and feasible to the cumulative arrival time or arrival time offset objectives to add a vessel to a lockage, that will be done in the solution.

Additionally, adding a non-time-based placement objective to a multi-criteria objective function that is composed of only time-based objectives requires careful weighting of that objective and makes it harder to explain results in verification cases.

Finally, there is an option to add placement objectives with very low weightings, like was done for the waiting time objective. This led to longer times to solve in the limited testing performed on it, because the solver would get stuck iterating at very low optimality gaps (<0.1%). It should be tested whether this is a problem that regularly occurs or if it was just specific to the cases that were tested.

5.5 Practical Considerations

Several practical considerations have to be taken into account for the current theoretical implementations of the models, or any future practical implementations. The larger-than-inequality constraints of SMPL systems provide extra room for variables to take on larger values than strictly required, as discussed in Subsection 5.5.1. Additionally, extended model functionality can either be added to the models directly, increasing their complexity, or it can be implemented through post-processing of the solution, as outlined in Subsection 5.5.2.

5.5.1 Larger-than-inequality Constraints

A quirk of the SMPL formulation is the fact that state equations are implemented as larger-than-inequality constraints, like in Equation 3.14. Essentially, the states have room to take on any value according to the constraints, as long as it is above some lower bound. For the arrival time states this presents no practical problems, as the objectives that are minimized include the arrival time states at the final location, thus there is no case in which a vessel's final arrival time is negatively impacted by a state being allowed to overshoot its lower bound according to the constraints, as that overshooting behavior is discouraged by the objective function.

However, for the placement problem, there is no mechanism for moving the bin height states to be as small as possible, corresponding to the vessels being placed as far forward as possible, which is what should happen in practice. For the optimization problem, it does not matter if a vessel is placed at the front or the back of the chamber, as long as it does not stick out of it on either end or unless moving it would allow more vessels to fit into the chamber.

An additional objective could force the vessels to be placed as far forward as possible but could lead to the optimization problem becoming harder to solve, because now the solution is not optimal when the vessels fit into the chamber in any configuration, it also has to find the optimal configuration on

top of that. A better candidate for fixing this practicality issue is post-processing, which is outlined in the next subsection.

It should, however, be emphasized that the problem highlighted in this subsection does not have the chance to create an invalid solution. The exact values of the bin height states still always have to obey the constraints, even if they are larger than what they could be, and these same bin height values are used to establish the mooring constraints through the bow and stern positions.

5.5.2 Complexity and Post-Processing

Some additions could be made to this model and the models in Chapter 3 and Chapter 4, to further approximate the true practical considerations of ship placement and the vessels traveling from place to place.

For example, consider the situations in Figure 5.11. Both situations are valid solutions to the placement problem described above. However, in situation (a), vessel 3 arrives in the chamber later than vessels $0, 1$, and 2, making it impossible for it to move into its assigned position. Scenario (b) does present a feasible practical scenario.

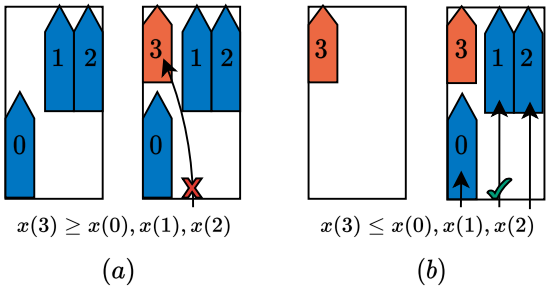


Figure 5.11: (a) : *Impractical but feasible chamber entry ordering scenario.* (b) : *Practical and feasible chamber entry ordering scenario.*

A constraint could be added to prevent the ordering of situations similar to situation (a) in Figure 5.11, but this would further drive up the number of constraints, and could potentially make it take longer to find the optimal solution. This was also empirically observed when trying to implement this constraint.

Another example could be a scenario in which a vessel is told to move very slowly on a given arc to avoid having to wait at a lock coming up, even though it has been moving at its maximum velocity on an arc just before that. A more sensible approach would be to move at a velocity that is averaged on both arcs. This could be added through an objective, but it, once again, would make the optimization problem more complex, with there being many almost optimal velocity distributions for a given problem.

The examples mentioned, and other practical considerations like it, can be handled by post-processing. A lockmaster or a secondary heuristic algorithm could use the schedules that come out of the scheduler as a starting point, making small changes to improve the practicality of the solution, without having to account for minute details during the optimization itself.

This has also been the line of thinking when adding components to the scheduling models. Only the main practical considerations that have a large impact on the scheduling solutions have been added, with the expectation that small practical details can be adjusted by an algorithm or person that runs or acts much faster and more predictably than the exponential complexity of the branch-and-bound algorithm used to find a solution to the MILP.

5.6 Conclusion

The research question to be answered in this chapter was:

> *How can ship placement be integrated into the multi-chamber SMPL IWT scheduling model?*

An important component of lockages is how the vessels are placed in the two-dimensional space of the chamber. Vessels must not overlap and they must obey mooring and safety distance constraints. The problem of placing the ships is usually modeled as the ship placement problem in the context of lock scheduling [32].

By assuming vessels and chambers are both rectangular and by discretising the width of the chamber and the vessels into bins of a set width, the sequence of vessels sailing into the chamber can be modeled similarly to how Tetris has been modeled as an SMPL system [56], with vessels being blocks of different sizes that sequentially slide into the bins of the chamber playing field. The states of this SMPL system are then the height of the vessel stacks in each of the bins of the chamber. This system was integrated into the scheduling model with a new set of SMPL constraints.

Vessels have to maintain safety distances for maneuvering or to obey regulations for certain transported goods. These safety distances are integrated into the model presented in this chapter by including them in the dimensions of the vessels that are placed. Furthermore, special SMPL constraints for vessels with flammable goods, which are not allowed to moor to any other vessel, and vessels with toxic or explosive goods, which are not allowed to be in the same lockages as any other vessels, were added.

Vessels must be moored to the quay or another vessel. A vessel can only moor to another vessel if it is fully alongside that vessel and if that vessel is also moored to another vessel or the quay. Rules governing this behavior were added with an additional mooring variable and additional SMPL constraints.

The full set of SMPL constraints was gathered into two SMPL systems, the first for routing, ordering, and synchronisation and the second for the vessel placement sequence in the chambers. The sets of constraints were translated to MILP problem constraints. Constraints on vessel stack heights staying within the length of the chambers and moored vessels being fully alongside the vessel they are mooring to could not be added as SMPL constraints, so they were added directly as MILP constraints.

The objectives and KPIs from Chapter 4 are still applicable to the model with ship placement introduced in this chapter. A schedule found by the scheduler now consists of vessel arrival time, routing, ordering, synchronisation, mooring, bin placement, and chamber bin stack height variables.

Larger-than-inequality constraints inherent to SMPL system equations may lead to variable values in scheduling solutions not being as tight to the lower bounds as they could be, but this does not jeopardize the validity of the solution. Furthermore, only the most important placement considerations were implemented to reduce the complexity of the model, especially when the minute details of the schedule and placement can be accounted for quickly through post-processing.

6

Verification

It must be verified that the scheduling models introduced in Chapter 3, Chapter 4, and Chapter 5 work as intended and that they have been implemented into the scheduler correctly. Each of the models builds upon the previous models, with their implementations having a large amount of overlap. As a result, it makes sense to verify them all in the same chapter, because parts of the models in later chapters can already be verified by looking at the models in earlier chapters. Furthermore, comparisons can be drawn between the solutions, complexity, and performance of the different schedulers.

The research question that will be answered in this chapter is:

> *How can the performance of the SMPL IWT scheduling models be verified?*

The approach taken to verify the scheduling models is explained in Section 6.1. The verification cases examined are documented in Section 6.2 through Section 6.7. An analysis on the complexity of the models is done in Section 6.8, and the models are compared to a heuristic model that simulates a simplified state of practice in Section 6.9. Finally, the answer to the research question is given in the conclusion in Section 6.10.

6.1 Approach

This section outlines the verification testing approach in Subsection 6.1.1 and the different verification cases and tests that are used in Subsection 6.1.2.

6.1.1 Testing Approach

Within the context of calculation computer models, verification can be defined as: "*Ensuring that the computer program of the computerized model and its implementation are correct*" [81]. In other words, the statements that must be proven for the IWT SMPL scheduling models and schedulers are:

- The mathematical models correctly model what they are intended to model.
- The mathematical models are correctly implemented into the scheduler.

These statements will be proven by setting up a number of verification cases and associated tests. A verification case describes a given combination of graph layout, graph input parameters, and scheduling model that is tested. The tests run on the verification cases each have their own sets of vessels, objectives, and vessel input parameters (and sometimes slightly different graph input parameters). An overview of verification cases and tests is given in Subsection 6.1.2.

The inputs of each of the verification cases and tests are tailored to test a particular part of the model they are run on. The tests are simple and they should give predictable results. By comparing the expected optimization results to the actual results, it can be verified whether the mathematical models are correct and whether the schedulers are correctly implemented. The simple tests should

also give confidence that the schedulers will perform as intended in more complex scenarios. Some higher complexity tests are run to support that assertion.

Furthermore, since the models from Chapter 3, Chapter 4, and Chapter 5 have significant mathematical and implementation overlap, verification of components of a model that are also present in other models is used as proof of verification of those other models. Of course, these components must show that they still work in the tests that are run to verify different parts of the other models.

The scheduling models are implemented in Python and solved using the Gurobi 9.1.2 solver. Sets, variables, constraints, objectives, and plots are all automatically generated by the scheduler based on the given input parameters. A 4-core Intel i7-7700HQ 2.8 GHz processor was used to carry out computations.

It is important to mention that the parameters and scenarios used in the verification tests are not meant to represent real scenarios. While the parameters are in the ballpark of reality, they are set to create specific edge-case scenarios rather than to exactly simulate reality. As a result, the results of the verification scenarios do not validate the model. Studies with real datasets should be performed in the future to properly gauge the efficacy of the scheduling models.

6.1.2 Verification Cases

Table 6.1 gives an overview of the verification cases per model, the associated tests, and their main points of focus. Note that each test still has to show consistent results in areas that the test is not directly targeting. For instance, multi-vessel tests will not specifically target edge cases, but they do test a wider range of different possible scenarios, possibly showing mistakes that have been missed in other tests.

Table 6.1: *Verification cases and tests.*

Test Number	Test Focus	(Sub)section
	Single-Vessel, Single-Chamber Model	
	Verification Case A	Section 6.2
1-4	Basic functioning of routing and ordering. Performance indicators.	Subsection 6.2.1
5	Functioning of the waiting time objective.	Subsection 6.2.2
6	Departure time and arrival time offset objective.	Subsection 6.2.3
7	Multi-vessel test.	Subsection 6.2.4
	Verification Case B	Section 6.3
1	Routing with choice of multiple routes.	Subsection 6.3.1
2	Routing with choice of multiple routes.	Subsection 6.3.2
	Multi-Vessel, Multi-Chamber Capacity Model	
	Verification Case C	Section 6.4
1-2	Basic functioning of synchronisation.	Subsection 6.4.1
	Verification Case D	Section 6.5
1	Distribution of vessels between multiple chambers. Capacity limits of chambers.	Subsection 6.5.1
2	Multi-Vessel Test. Updated lockage counting KPI.	Subsection 6.5.2
	Multi-Vessel, Multi-Chamber Placement Model	
	Verification Case E	Section 6.6
1	Basic placement and mooring.	Subsection 6.6.1
2	Advanced Placement and mooring.	Subsection 6.6.2
	Verification Case F	Section 6.7
1	Multiple locks and vessels with flammable and explosive goods.	Subsection 6.7.1
2	Multiple directions and intermediate destinations. Large vessel variety.	Subsection 6.7.2

6.2 Verification Case A - Single-Vessel, Single-Chamber Model

Verification case A uses one of the simplest graphs possible: a single symmetric waterway with two nodes, 0 and 5, at the ends and a single-chamber lock, lock A, with nodes $1, 2, 3$, and 4, in between, as depicted in Figure 6.1. The distance between node 0 and lock A is equal to the distance between lock A and node 5. As such, if two vessels with the same maximum velocities leave node 0 and node 5 at the same time, they should arrive at the waiting areas of lock A at the same time.

The parameters of the topology graph are shown in table Table 6.2. If $d_{(i,j)}$ is specified, then Equation 3.14 is used to calculate the associated minimum travel times $\tau_{(i,j)}(k)$ for the vessels. If a specified value for $\tau_{(i,j)}(k)$ is entered, like for locks and waiting areas, then that value is used instead.

6.2.1 Tests 1-4 - Ordering and Route Creation

These four tests are intended to show the basic functioning of the routing components of the model, ensuring that vessels start at their departure node and end up at their destination node, not skipping any arcs in between. They also test all four possible ordering scenarios with two vessels. The performance indicators are also tested.

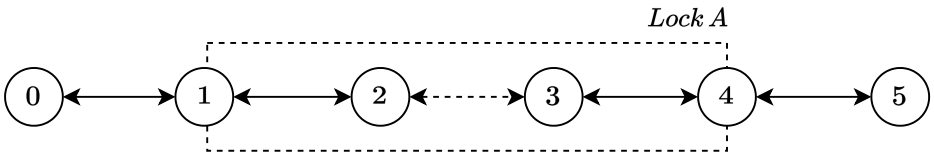


Figure 6.1: *Topology graph for verification Case A.*

Table 6.2: *Node and arc parameters of the graph used in verification case A.*

Node i	$p_{x,i}$ $[km]$	$p_{y,i}$ $[km]$
0	0	0
Lock A $(1, 2, 3, 4)$	10	0
5	20	0

Arc (i, j)	$d_{(i,j)}$ $[km]$	Specified $\tau_{(i,j)}(k)\, \forall k \in \mathcal{K}$ $[h]$
$(0, 1), (1, 0), (4, 5), (5, 4)$	Euclidean	-
Lock A waiting Areas $(1, 2), (2, 1), (3, 4), (4, 3)$	-	0.1
Lock A Chamber $(2, 3), (3, 2)$	-	0.5

The vessel parameters of each of the four tests are given in Table 6.3, Table 6.4, Table 6.5, and Table 6.6, respectively. Vessels 0 and 1 are scheduled, with each test having a different combination of departure and destination nodes for the vessels. The objective that is used is the unweighted cumulative arrival time objective.

Vessel 0 is slightly faster than vessel 1, so it should always arrive at lock A before vessel 1, thus always going through the lock chamber first. When both vessels are moving in the same direction, there should be an empty lockage in between. This should not be the case when the vessels are moving in opposite directions. The vessels' routes should start and end at their departure and destination nodes and should visit all of the nodes in between.

Table 6.3: *Vessel parameters Case A Test 1.*

vessel k	$v_{\max,(i,j)}(k)$ $\forall (i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)$ $[h]$
0	10	0	5	0
1	9	0	5	0

Table 6.4: *Vessel parameters Case A Test 2.*

vessel k	$v_{\max,(i,j)}(k)$ $\forall (i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)$ $[h]$
0	10	5	0	0
1	9	5	0	0

Table 6.5: *Vessel parameters Case A Test 3.*

vessel k	$v_{\max,(i,j)}(k)$ $\forall (i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)$ $[h]$
0	10	5	0	0
1	9	0	5	0

Table 6.6: *Vessel parameters Case A Test 4.*

vessel k	$v_{\max,(i,j)}(k)$ $\forall (i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)$ $[h]$
0	10	0	5	0
1	9	5	0	0

The results of the tests are displayed in journey plots, like in Figure 6.2. The y-axis represents the

possible arrival time states of the nodes in the graph and the x-axis represents time. The vessel lines show the journeys of the vessels, with a marker indicating the time at which a vessel arrives at a node. If a line crosses a state without a marker being placed at the vertical position of that state, the vessel does not actually visit the node associated with that state on its route. A grey dashed line indicates an empty lockage.

Figure 6.2 and Figure 6.3 show the journey plots for tests 1 and 2, where both vessels are moving in the same direction. The results are as expected, vessel 0 is given priority at the lock, as it is slightly faster, an empty lockage takes place between vessels 0 and 1 as they are going through the chamber in the same direction, and the vessels correctly visit all of the nodes in between their departure and destination nodes in the correct order. The values of the performance indicators and objectives for these tests are listed in Table 6.7 and Table 6.8.

How the KPIs are calculated is discussed in Subsection 3.5.4. The lower the objective function J value, the better. Similarly, the lower the cumulative arrival time $\mathcal{A}$ and arrival time offset $\mathcal{H}$ the better. $\mathcal{A}$ can not be zero, as the vessels will always have some travel time, but the lower it is, the faster the vessels arrive at their destination. $\mathcal{H}$ can be zero if all vessels arrive exactly at their planned arrival time, indicating a perfect result.

In general, the lower the number of lockages $\mathcal{L}$, the better in practice. However, the number of lockages is not actively minimized in the objective function. Finally, the average delay $\mathcal{T}$ should also be as low as possible, so each of the vessels experiences as little interference from other vessels on its route as possible.

Figure 6.4 and Figure 6.5 show the results for tests 3 and 4, where the vessels move in opposite directions. In this case, there should not be an empty lockage. The routes of the vessel and the priority of vessel 0 are all in order again. The values of the performance indicators and objectives for these tests are listed in Table 6.9 and Table 6.10. Note that the cumulative arrival time is smaller for tests 3 and 4, because there is no empty lockage required. The average delay is then also smaller because none of the vessels have to wait for an empty lockage.

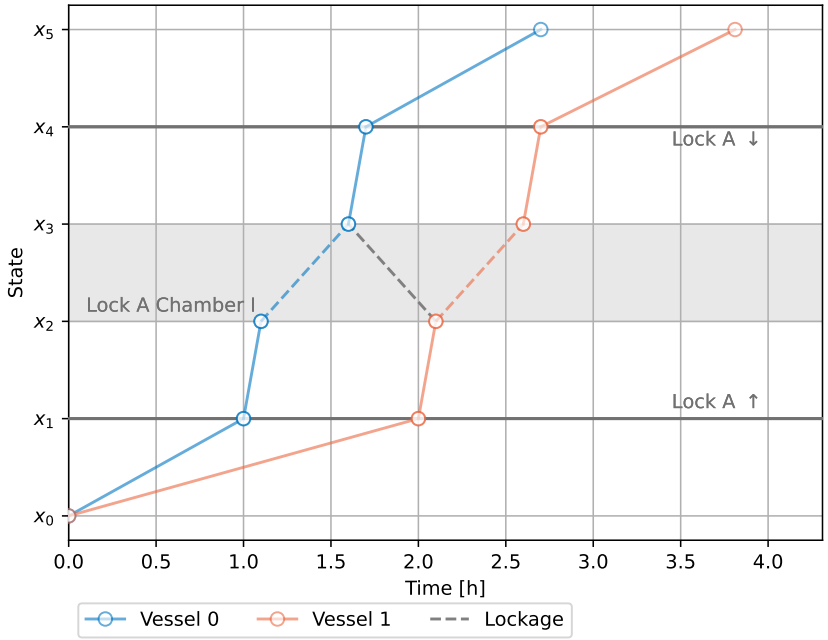


Figure 6.2: *Journey Plot Case A Test 1.*

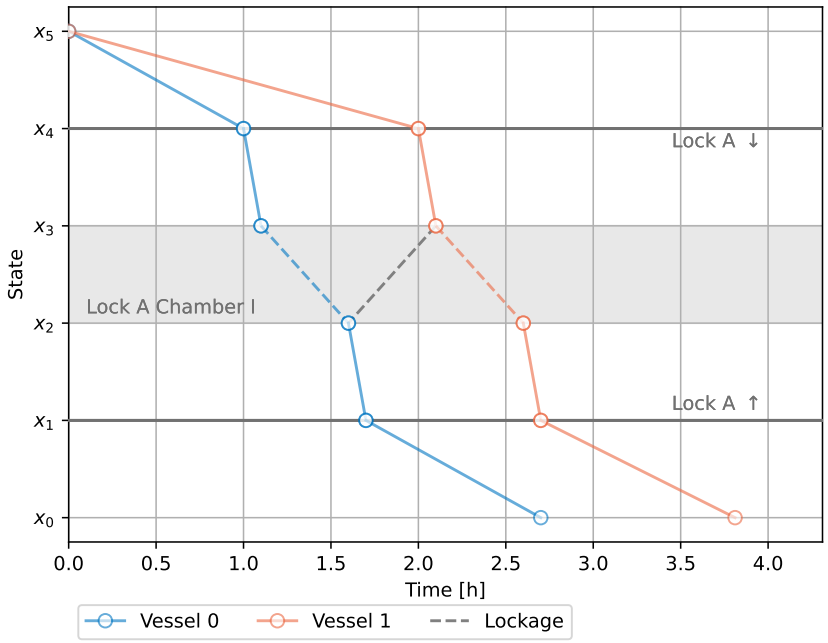


Figure 6.3: *Journey Plot Case A Test 2.*

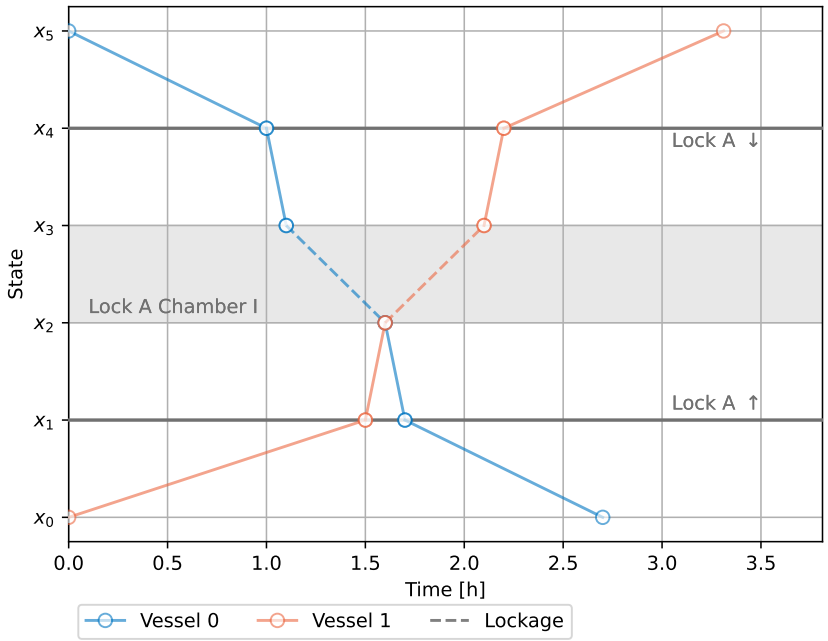


Figure 6.4: *Journey Plot Case A Test 3.*

Table 6.7: *KPIs Case A Test 1.*

Performance Indicator	Value
J	6.51
$\mathcal{A}$	$6.51\,h$
$\mathcal{H}$	-
$\mathcal{L}$	3
$\mathcal{T}$	15.21%

Table 6.8: *KPIs Case A Test 2.*

Performance Indicator	Value
J	6.51
$\mathcal{A}$	$6.51\,h$
$\mathcal{H}$	-
$\mathcal{L}$	3
$\mathcal{T}$	15.21%

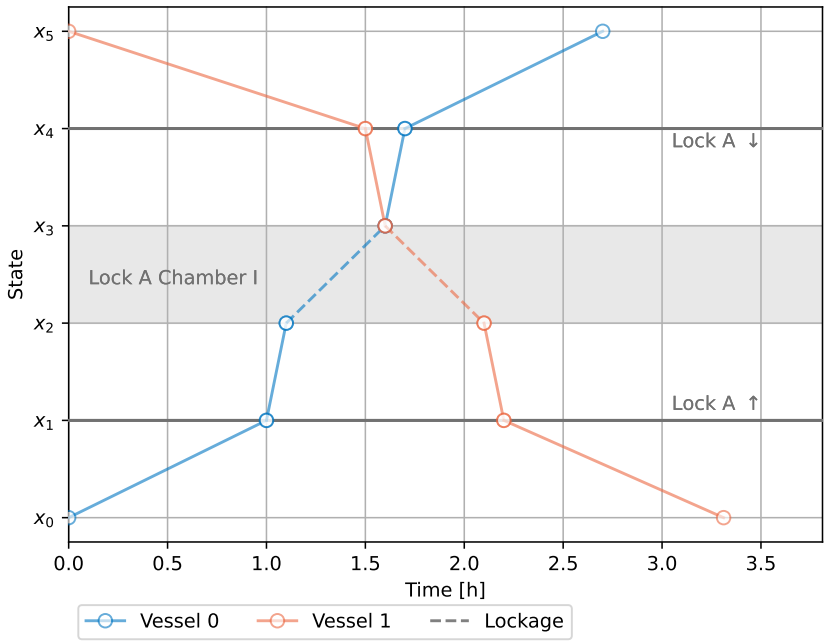


Figure 6.5: *Journey Plot Case A Test 4.*

Table 6.9: *KPIs Case A Test 3.*

Performance Indicator	Value
J	6.01
$\mathcal{A}$	$6.01\,h$
$\mathcal{H}$	-
$\mathcal{L}$	2
$\mathcal{T}$	6.65%

Table 6.10: *KPIs Case A Test 4.*

Performance Indicator	Value
J	6.01
$\mathcal{A}$	$6.01\,h$
$\mathcal{H}$	-
$\mathcal{L}$	2
$\mathcal{T}$	6.65%

6.2.2 Test 5 - Waiting Time Objective

The effect of the waiting time objective can be seen in a test where it is removed entirely. Test 5 has the same node, arc, and vessel parameters as test 1, but the waiting time objective is left out of the objective function.

The results are shown in Figure 6.6. There is one difference between Figure 6.2 and Figure 6.6: in the test without the waiting time objective, the scheduler tells vessel 1 to wait on waiting area arc $(1, 2)$ after moving at full speed on arc $(0, 1)$, while in the test with the waiting time objective vessel 1 is told to slow-steam on arc $(0, 1)$, not spending any time stationary on arc $(1, 2)$. The arrival times of the vessels at their final destinations are identical in both tests, as the waiting time objective should not influence this. The same final arrival times are achieved with the waiting time objective included, but with fewer emissions because of the slow-steaming.

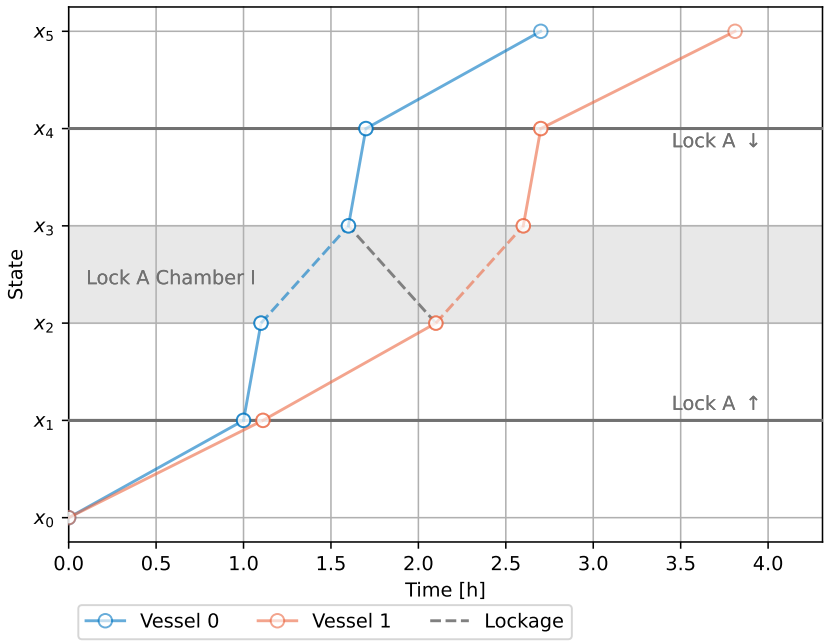


Figure 6.6: *Journey Plot Case A Test 5.*

6.2.3 Test 6 - Departure Time and Arrival Time Offset

In this test, the departure time parameter and arrival time offset are tested. The vessel parameters for this test are given in Table 6.11. The unweighted arrival offset objective is used and the vessels are given non-zero departure times. Vessel 1 is still slower than vessel 0, but it has an earlier departure time and an earlier planned arrival time, so vessel 1 should get priority at the lock, and both vessels should arrive at their destination right on time, if possible.

The journey plot in Figure 6.7 shows that the result aligns with the expectations. The KPI values are listed in Table 6.12. Note that the average delay KPI is not relevant, as there are no vessels with the cumulative arrival time objective. The objective function value is also 0.0, because both vessels arrive exactly on time.

Table 6.11: *Vessel Parameters Case A Test 6.*

vessel k	$v_{\max,(i,j)}(k)$ $\forall(i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)$ $[h]$	$\hat{x}(k)$ $[h]$
0	10	0	5	4	9
1	9	0	5	3	7

Table 6.12: *KPIs Case A Test 6.*

Performance Indicator	Value
J	0.0
$\mathcal{A}$	$9.0\,h$
$\mathcal{H}$	$0.0\,h$
$\mathcal{L}$	3
$\mathcal{T}$	$-$

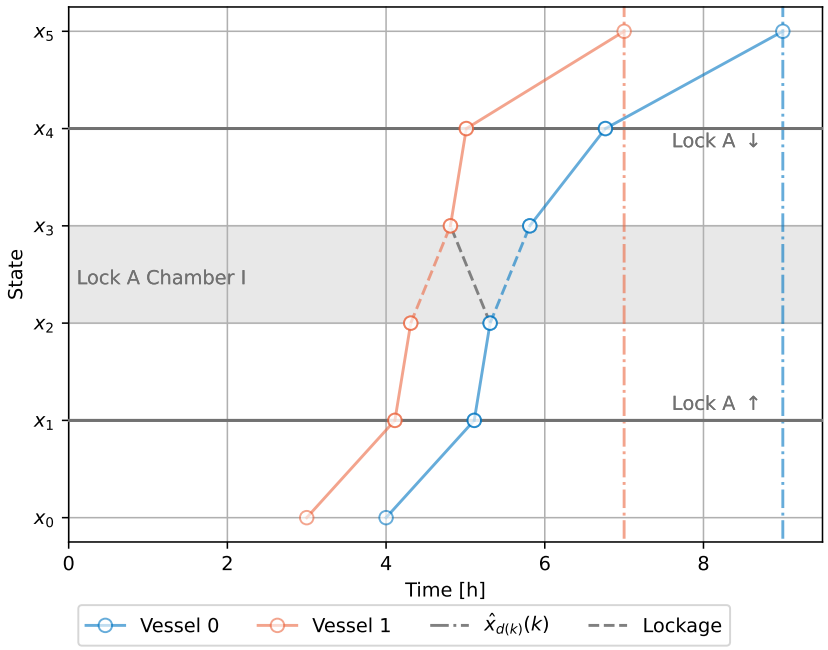


Figure 6.7: *Journey Plot Case A Test 6.*

6.2.4 Test 7 - Multi-Vessel Stress Test

The tests up until now have only involved two vessels. The model might malfunction if more vessels are involved. This test has six vessels, with the parameters given in Table 6.13. All vessels have the same maximum velocity, except for vessel 5 which has double the maximum velocity of the other vessels. The objective is the cumulative travel time objective.

Since all vessels start their journeys at roughly the same time, vessel 5 should be at the lock first, and thus pass the lock before any other vessel. Then, the vessels should go through the lock in alternating directions, avoiding empty lockages. As four vessels are moving in one direction, and two in the other, there has to be an empty lockage. Finally, vessels 3 and 4 have an initial departure time of $1\,h$, so they should depart at $1\,h$. The unweighted cumulative arrival time objective is used.

The results are shown in Figure 6.8. The expectations are met. It is interesting to note that the empty lockage caused by the directional vessel imbalance is moved to the end, that way only one vessel has to wait for an empty lockage, minimizing its impact. The values of the performance indicators and objectives are listed in Table 6.14.

Table 6.13: *Vessel parameters of Case A Test 7.*

vessel k	$v_{\max,(i,j)}(k)$ $\forall(i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)$ $[h]$
0	10	0	5	0
1	10	0	5	0
2	10	5	0	0
3	10	5	0	1
4	10	5	0	1
5	20	5	0	0

Table 6.14: *KPIs Case A Test 7.*

Performance Indicator	Value
J	18.70
$\mathcal{A}$	$18.70\,h$
$\mathcal{H}$	-
$\mathcal{L}$	7
$\mathcal{T}$	21.6%

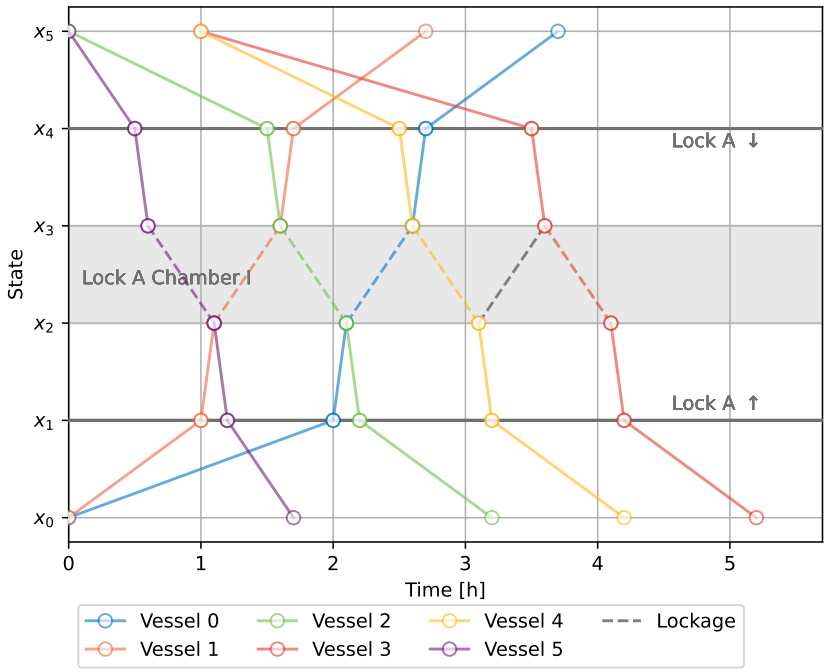


Figure 6.8: *Journey Plot Case A Test 7*

6.3 Verification Case B - Single-Vessel, Single-Chamber Model

Vessels could not be assigned to different possible routes in verification case A, since each movement direction had only one available route. Verification case B introduces a choice of routing into the topology graph, as shown in Figure 6.9. The waterway with the lock is the same as in verification case A, but now there is a detour node, node 6. The detour node results in a further travel distance, but vessels can avoid congestion at lock A by taking the detour. The parameters of the topology graph are listed in Table 6.15.

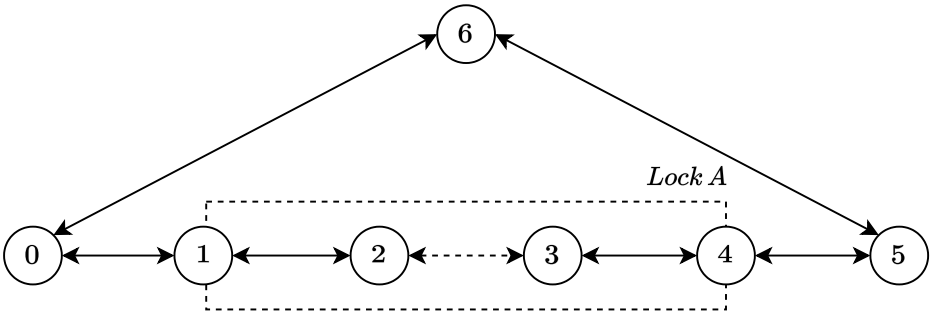


Figure 6.9: *Topology graph for verification Case B.*

Table 6.15: *Node and arc parameters of the graph used in verification Case B.*

Node i	$p_{x,i}$ $[km]$	$p_{y,i}$ $[km]$
0	0	0
Lock A $(1,2,3,4)$	10	0
5	20	0
6	10	15

Arc (i,j)	$d_{(i,j)}$ $[km]$	Specified $\tau_{(i,j)}(k)\,\forall k \in \mathcal{K}$ $[h]$
$(0,1),(1,0),(4,5),(5,4),$ $(0,6),(6,5)$	Euclidean	-
Lock A waiting Areas $(1,2),(2,1),(3,4),(4,3)$	-	0.1
Lock A Chamber $(2,3),(3,2)$	-	0.5

6.3.1 Test 1 - Route Choice

The goal of this test is to see if a vessel will take another feasible route if that is quicker. Three vessels are given the same velocity and they all travel in the same direction. This will create congestion at the lock, which should make it faster for one of the vessels to take a detour through node 6, rather than for it to wait at the lock. The vessel parameters are listed in Table 6.16.

The results are shown in the journey plot in Figure 6.10. The performance indicator values are listed in Table 6.17. It is indeed faster for vessel 1 to take a detour, rather than to wait for its turn at the lock.

Table 6.16: *Vessel parameters Case B Test 1.*

vessel k	$v_{\max,(i,j)}(k)$ $\forall(i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)$ $[h]$
0	8	0	5	0
1	8	0	5	0
2	8	0	5	0

Table 6.17: *KPIs Case B Test 1.*

Performance Indicator	Value
J	11.90
$\mathcal{A}$	$11.90\,h$
$\mathcal{H}$	-
$\mathcal{L}$	3
$\mathcal{T}$	10.42%

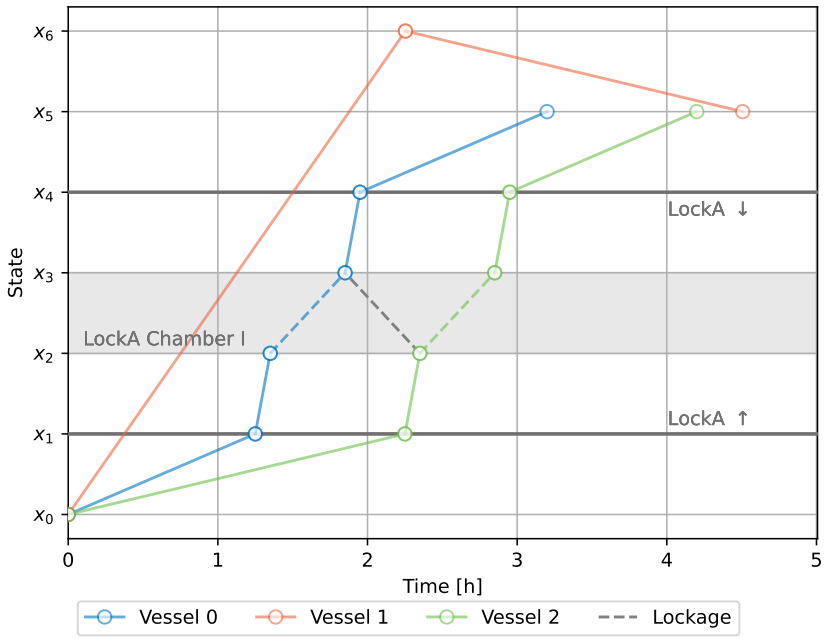


Figure 6.10: *Journey Plot Case B Test 1.*

6.3.2 Test 2 - Route Choice

For completeness and comparison to test 1, a similar test is also run where a detour is available, but where it does not make sense for any vessel to take a detour. This is achieved by using the same vessel parameters as in Table 6.16 while increasing the vertical position of node 6 to $p_{y,6} = 25\ km$, making the detour longer.

All three vessels now avoid the detour, as shown in Figure 6.11. From the KPIs in Table 6.18, it becomes clear that the situation in test 1 is indeed better than having all three vessels pass through the lock, as the cumulative arrival time and average delay are both larger.

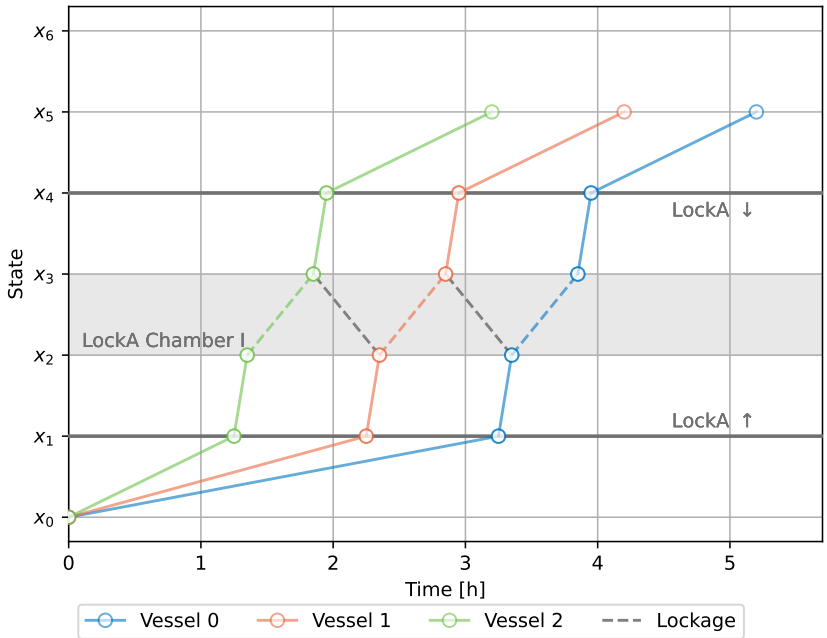


Figure 6.11: *Journey Plot Case B Test 2*

Table 6.18: *KPIs Case B Test 2.*

Performance Indicator	Value
J	12.60
$\mathcal{A}$	$12.60\,h$
$\mathcal{H}$	-
$\mathcal{L}$	5
$\mathcal{T}$	31.25%

6.4 Verification Case C - Multi-Vessel, Multi-Chamber Capacity Model

Verification Case C uses the single chamber topology graph from verification Case A, depicted in Figure 6.1. Note that while the multi-vessel, multi-chamber capacity model is used, the verification case has a single chamber lock. Multi-chamber locks for the same model are tested in Section 6.5.

The single chamber lock now has capacity and additional vessel processing time parameters, as listed in Table 6.19. The tests in Case C will check the functioning of the synchronisation of two vessels in simple scenarios. The tests will include two vessels, so the lock chamber is given a capacity of 2. The additional vessel processing time is set to $0.1\ h$.

Table 6.19: *Node and arc parameters of the graph used in verification case C.*

Node i	$p_{x,i}\ [km]$	$p_{y,i}\ [km]$
0	0	0
Lock A $(1, 2, 3, 4)$	10	0
5	20	0

Arc (i, j)	$d_{(i,j)}\ [km]$	Specified $\tau_{(i,j)}(k)\ \forall k \in \mathcal{K}\ [h]$	$\boldsymbol{c_{(i,j)}}$	$\boldsymbol{\overline{\tau}_{(i,j)}}\ [h]$
$(0, 1), (1, 0), (4, 5), (5, 4)$	Euclidean	-	**-**	**-**
Lock A Waiting Areas $(1, 2), (2, 1), (3, 4), (4, 3)$	-	0.1	**-**	**-**
Lock A Chamber $(2, 3), (3, 2)$	-	0.5	$\mathbf{2}$	$\mathbf{0.1}$

6.4.1 Tests 1-2 - Synchronisation

Tests 1 and 2 test the behavior of synchronised vessels. Test 1 is intended to create a scenario where it is optimal for the two vessels going through the system to synchronise on the chamber, and test 2 is intended to create a scenario where it is optimal for the two vessels to go through the chamber separately, even if they do fit in the chamber together. The unweighted cumulative arrival time objective is used. Vessel 0 departs at $0\ h$ in both tests, and vessel 1 departs at $0.2\ h$ in test 1 and $0.5\ h$ in test 2. In test 1, it makes sense for vessel 0 to wait for vessel 1 so they can have a single lockage together. In test 2, vessel 1 is too far behind vessel 0 for synchronisation to be beneficial, so vessel 0 should have a separate lockage before vessel 1. The vessel parameters for the tests are listed in Table 6.20 and Table 6.21.

The resulting journey plots of the tests are shown in Figure 6.12 for test 1 and Figure 6.13 for test 2. The vessel numbers of synchronised vessels in a lockage are listed in parentheses on top of the

Table 6.20: *Vessel parameters Case C Test 1.*

vessel k	$v_{\max,(i,j)}(k)\ \forall(i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)$ $[h]$	$\boldsymbol{\rho(k)}$
0	10	0	5	0	**1**
1	10	0	5	0.2	**1**

Table 6.21: *Vessel parameters Case C Test 2.*

vessel k	$v_{\max,(i,j)}(k)\ \forall(i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)$ $[h]$	$\boldsymbol{\rho(k)}$
0	10	0	5	0	**1**
1	10	0	5	0.5	**1**

lockage line in the journey plots. Vessels 0 and 1 are synchronised on the lock chamber in test 1, and not synchronised in test 2, conforming to expectations. Furthermore, it shows that the *Lockage Synchronisation Constraint* is working as intended, because vessel 0 can enter the chamber before vessel 1 has arrived, with the lockage starting once vessel 1 has also arrived. The lockage is also slightly longer because of the extra lockage time. Finally, the pre-existing routing and ordering constraints work as they should for these tests, as the routes are complete and the ordering still works in test 2.

For reference, the performance indicator values are listed in Table 6.22 and Table 6.23.

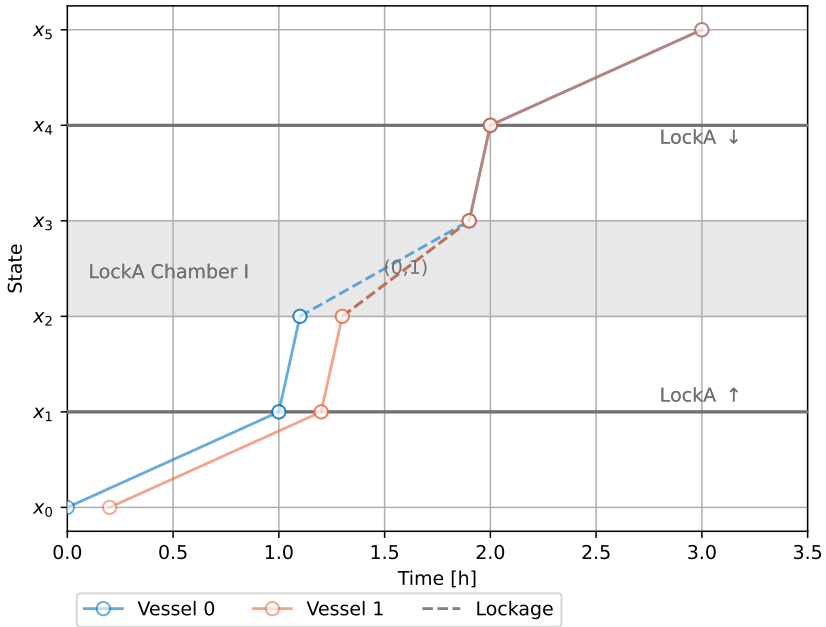


Figure 6.12: *Journey Plot Case C Test 1.*

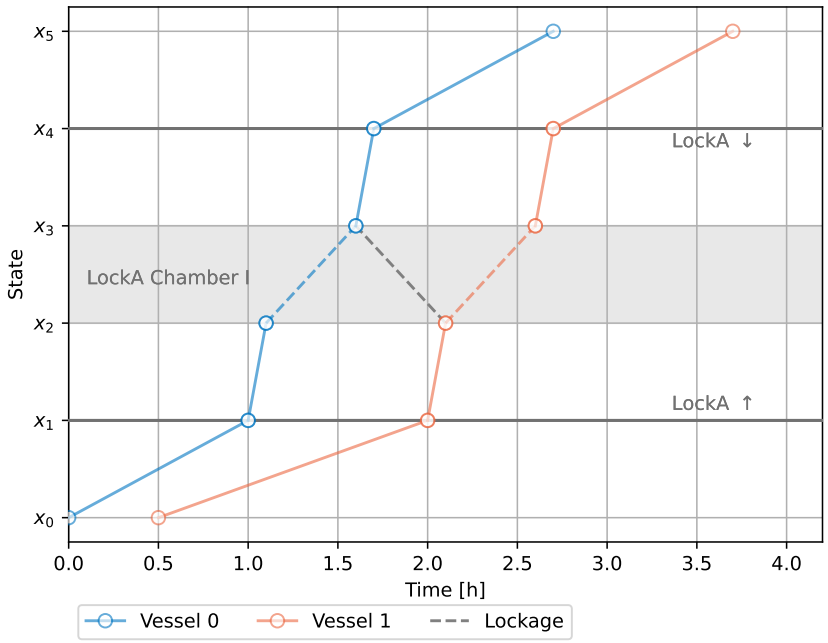


Figure 6.13: *Journey Plot Case C Test 2.*

Table 6.22: *KPIs Case C Test 1.*

Performance Indicator	Value
J	5.80
$\mathcal{A}$	5.80 h
$\mathcal{H}$	-
$\mathcal{L}$	1
$\mathcal{T}$	7.41%

Table 6.23: *KPIs Case C Test 2.*

Performance Indicator	Value
J	5.90
$\mathcal{A}$	5.90 h
$\mathcal{H}$	-
$\mathcal{L}$	3
$\mathcal{T}$	9.26%

6.5 Verification Case D - Multi-Vessel, Multi-Chamber Capacity Model

This verification case is used to test the multi-chamber capabilities of the multi-vessel, multi-chamber capacity model, and to test more of the ship synchronisation behavior. The symmetric waterway from case A is used, but Lock A now has two chambers as shown in Figure 6.14. The nodal and arc parameters are shown in Table 6.24 and Table 6.25. Chamber I is bigger than chamber II, but it has a longer minimum processing time.

Table 6.24: *Node parameters verification Case D.*

Node i	$p_{x,i}$ $[km]$	$p_{y,i}$ $[km]$
0	0	0
Lock A $(1, 2, 3, 4, 5, 6)$	10	0
7	20	0

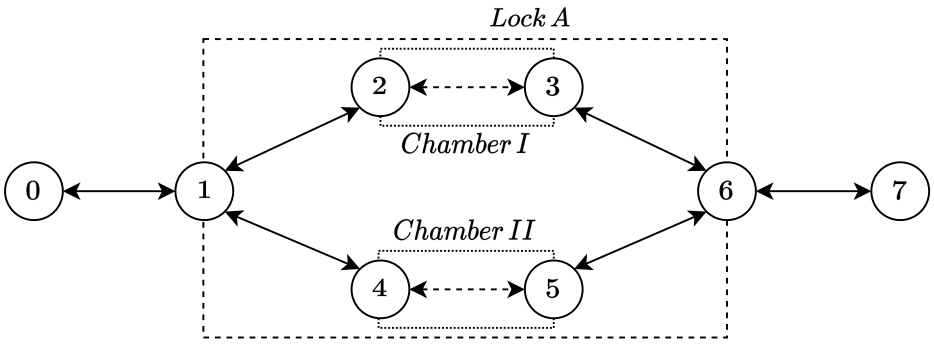


Figure 6.14: *Topology graph for verification Case D.*

Table 6.25: *Arc parameters verification Case D.*

Arc (i,j)	$d_{(i,j)}$ $[km]$	Specified $\tau_{(i,j)}(k)\,\forall k \in \mathcal{K}$ $[h]$	$c_{(i,j)}$	$\overline{\tau}_{(i,j)}$ $[h]$
$(0,1),(1,0),(6,7),(7,6)$	Euclidean	-	-	-
Lock A Waiting Areas $(1,2),(2,1),(4,5),(5,4)$	-	0.1	-	-
Lock A Chamber I $(2,3),(3,2)$	-	0.75	8	0.1
Lock A Chamber II $(4,5),(5,4)$	-	0.5	5	0.1

6.5.1 Test 1 - Chamber Distribution and Capacity Limits

Test 1 is intended to show that the vessels will correctly distribute themselves over the chambers if that is faster. It is also intended to show that a vessel that is too big for a chamber will only use a chamber that can fit it. The unweighted cumulative arrival time objective will be used for this test. Three vessels move in the same direction with the same maximum velocities. They have slightly staggered departure times to make their lines on the journey plot distinct. The exact vessel parameters are listed in Table 6.26. Vessel 0 is so big that it only fits in chamber I. Vessels 1 and 2 are small enough to fit in chamber II, but too big to fit together in either of the chambers. As chamber II has a lower processing time, vessels 1 and 2 should both be using that chamber in separate lockages.

Table 6.26: *Vessel parameters Case D Test 1.*

vessel k	$v_{\max,(i,j)}(k)$ $\forall(i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)$ $[h]$	$\rho(k)$
0	10	0	7	0	8
1	10	0	7	0.05	5
2	10	0	7	0.1	5

Table 6.27: *KPIs Case D Test 1.*

Performance Indicator	Value
J	9.30
$\mathcal{A}$	$9.30\,h$
$\mathcal{H}$	-
$\mathcal{L}$	4
$\mathcal{T}$	11.73%

The journey plot in Figure 6.15 shows that the resulting schedule is according to expectations. Each

of the vessels only visits a single chamber, and the faster chamber is chosen unless there is no other choice due to the capacity limits of the chamber. This indicates that multi-chamber locks and the *Synchronisation Capacity* constraint are working for this test.

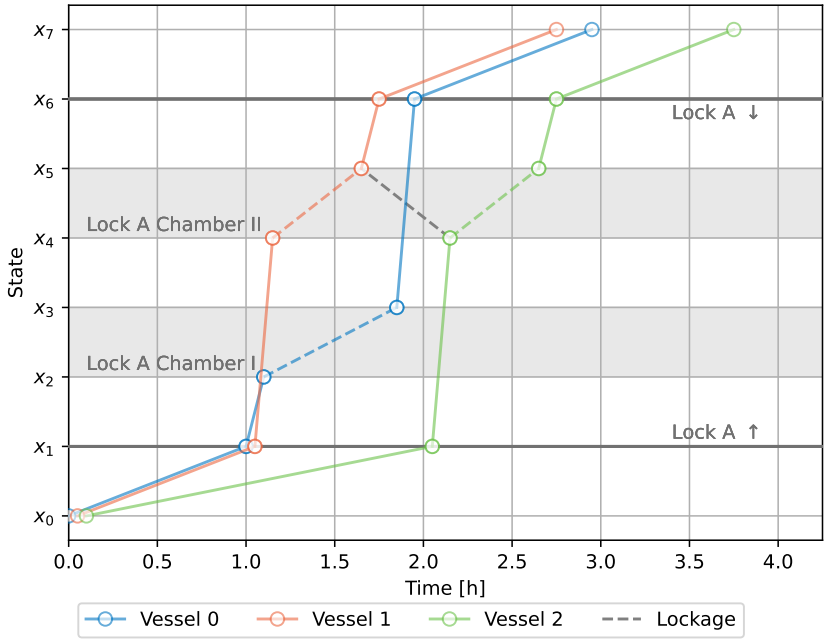


Figure 6.15: *Journey Plot Case D Test 1.*

6.5.2 Test 2 - Multi-Vessel Stress Test

Test 2 is a stress test of the model, with vessels traveling in both directions and multiple synchronised lockages. This test includes seven vessels, four traveling from node 0 to node 7, and three traveling from node 7 to node 0, all with roughly equal departure times to create bi-directional congestion at the lock. The vessel parameters are listed in Table 6.28. Vessel 0 is too big for chamber II, so it should travel through chamber I. Vessels 2, 3, and 4 should be able to fit into chamber II together. In the opposite travel direction, vessels 4, 5, and 6 all have the same size, but only two of them can fit into chamber I at a time and only one of them can fit into chamber II at a time. The objective used is the unweighted cumulative arrival time objective. The lockages with multiple vessels in them should get priority over the other lockages, as fewer vessels will be waiting the sooner the multi-vessel lockages are processed.

Table 6.28: *Vessel parameters Case D Test 2.*

vessel k	$v_{\max,(i,j)}(k)$ $\forall(i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)$ $[h]$	$\rho(k)$
0	10	0	7	0	8
1	10	0	7	0.05	3
2	10	0	7	0.1	1
3	10	0	7	0	1
4	10	7	0	0.05	3
5	10	7	0	0.1	3
6	10	7	0	0	3

Table 6.29: *KPIs Case D Test 2.*

Performance Indicator	Value
J	22.25
$\mathcal{A}$	22.25 h
$\mathcal{H}$	-
$\mathcal{L}$	4
$\mathcal{T}$	13.24%

The results are displayed in the journey plot in Figure 6.16, with the performance indicators listed in Table 6.29. The results are as predicted, with the multi-vessel lockages getting priority over single-vessel lockages. Vessel 0 takes chamber I, as it is too big for chamber II. Similarly, none of the combinations of sizes of the vessels in the lockages exceed the capacities of the chambers. No anomalies that are connected to other constraints, like routing or ordering, are visible.

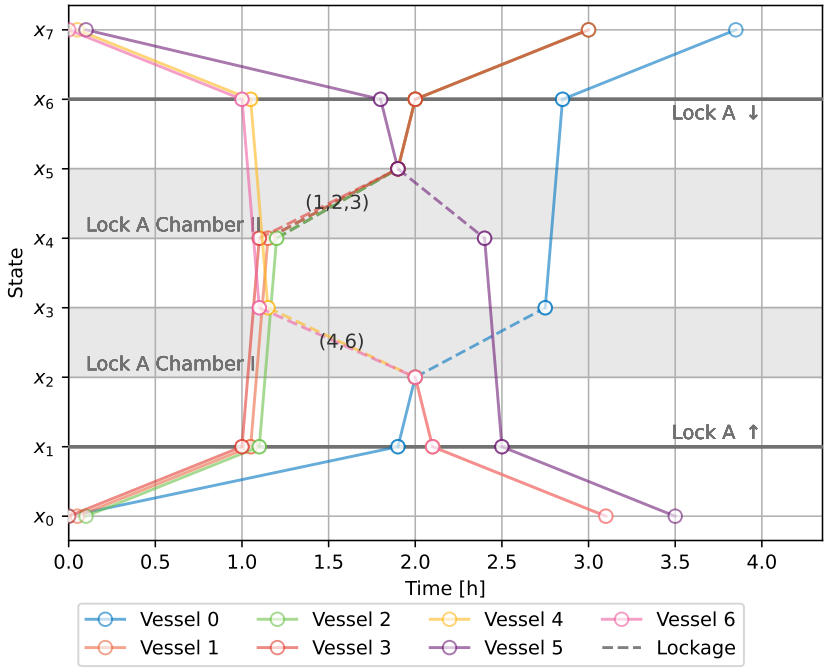


Figure 6.16: *Journey Plot Case D Test 2.*

6.6 Verification Case E - Multi-Chamber, Multi-Vessel Placement Model

This case is used to test the placement of vessels and the mooring constraints. The same single chamber graph topology as in Figure 6.1 is used. The node parameters are the same as in Table 6.2. The arc parameters including the width and length values of the chamber are given in Table 6.30.

The chamber has a length of 200 meters and is divided into 4 bins horizontally. The length is chosen as a simple round number which makes it easy to tailor vessel sizes. This length would correspond with a lock that can accommodate CEMT class V or VI vessels [7]. Chamber lengths are chosen to be arbitrary and easy to work with for these tests, and they will range from 150 to 250 meters in other cases. That is on the larger side, but not as large as chambers like the chambers at the Volkeraksluizen, which are 330 meters long [20]. The widths are chosen to be a middle ground between giving vessels options for horizontal positioning and reducing the number of placement decision variables. In reality, the number of bins will also have to be tailored to what length value can best be used to discretise the chamber width based on the types of vessels that often pass through.

6.6.1 Test 1 - Placement and Mooring

The purpose of this test is to observe if the basic behaviors of vessel placement and mooring are functioning correctly. Three vessels with identical maximum velocities are sent through the system. Their dimensions are listed in Table 6.31. They are dimensioned in such a way that they should all

Table 6.30: *Arc parameters for verification Case E.*

Arc (i,j)	$d_{(i,j)}\ [km]$	Specified $\tau_{(i,j)}(k)\ \forall k \in \mathcal{K}\ [h]$	$N_{\chi,(i,j)}$	$N_{\gamma,(i,j)}\ [m]$	$\overline{\tau}_{(i,j)}\ [h]$
$(0,1),(1,0),$ $(4,5),(5,4)$	Euclidean	-	-	-	-
Lock A Waiting Areas $(1,2),(2,1),$ $(3,4),(4,3)$	-	0.1	-	-	-
Lock A Chamber $(2,3),(3,2)$	-	0.5	4	200	0.0

fit into the chamber together, which they are expected to do. Vessel 0 is too big to moor to the other vessels, and vessel 1 cannot moor to vessel 2 because it is longer than vessel 2.

Table 6.31: *Vessel parameters Case E Test 2.*

vessel k	$v_{\max,(i,j)}(k)\ \forall (i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)\ [h]$	W_χ	$W_\gamma\ [m]$
0	10	0	5	0	2	150
1	10	0	5	0.05	1	110
2	10	0	5	0.1	1	100

The journey plot is shown in Figure 6.17, it shows no anomalies. The placement of the vessels in the lockage is shown in Figure 6.18. Vessels 0 and 2 are both attached to the side of the chamber and vessel 1 is moored to vessel 0. The vessels do not exceed the boundaries of the chamber.

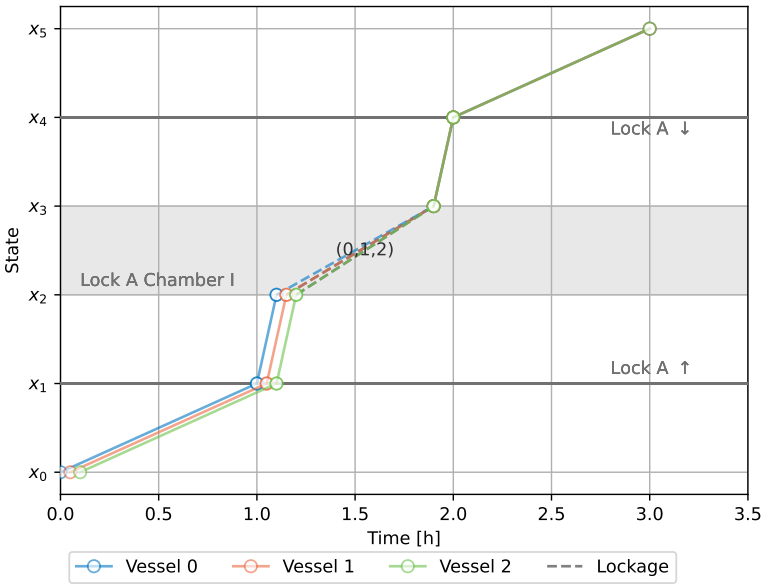


Figure 6.17: *Journey Plot Case E Test 1.*

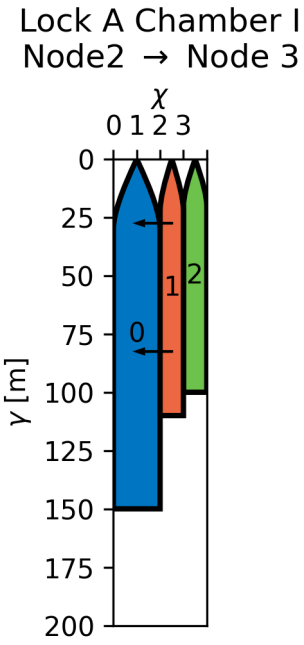


Figure 6.18: *Lockage of Case E Test 1.*

Table 6.32: *KPIs Case E Test 1.*

Performance Indicator	Value
J	8.85
$\mathcal{A}$	$8.85\,h$
$\mathcal{H}$	-
$\mathcal{L}$	1
$\mathcal{T}$	9.26%

6.6.2 Test 2 - Advanced Placement

Test 1 only looked at three vessels with relatively simple placement. This test examines more advanced placement, with multiple vessels moored to other vessels, which should eventually be moored to the side of the chamber. In this test the extra lockage time is removed by setting $\overline{\tau}_{(2,3)} = \overline{\tau}_{(3,2)} = 0\,h$, to encourage all vessels to be processed in the same lockage This test considers seven vessels as listed in Table 6.33. These vessels should all fit into a single lockage together.

Table 6.33: *Vessel parameters Case E Test 2.*

vessel k	$v_{\max,(i,j)}(k)\ \forall(i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)\ [h]$	W_χ	$W_\gamma\ [m]$
0	10	0	5	0	1	150
1	10	0	5	0.05	1	100
2	10	0	5	0.06	1	100
3	10	0	5	0.1	1	150
4	10	0	5	0.09	1	90
5	10	0	5	0.08	1	100
6	10	0	5	0.07	2	50

All vessels are placed into a single lockage, as shown in Figure 6.20. All vessels are properly moored either to the side of the chamber or to another vessel. Vessel 4 in particular is moored to vessel 3, which is moored to vessel 0, which is moored to the side of the chamber. The journey plot is shown in Figure 6.19.

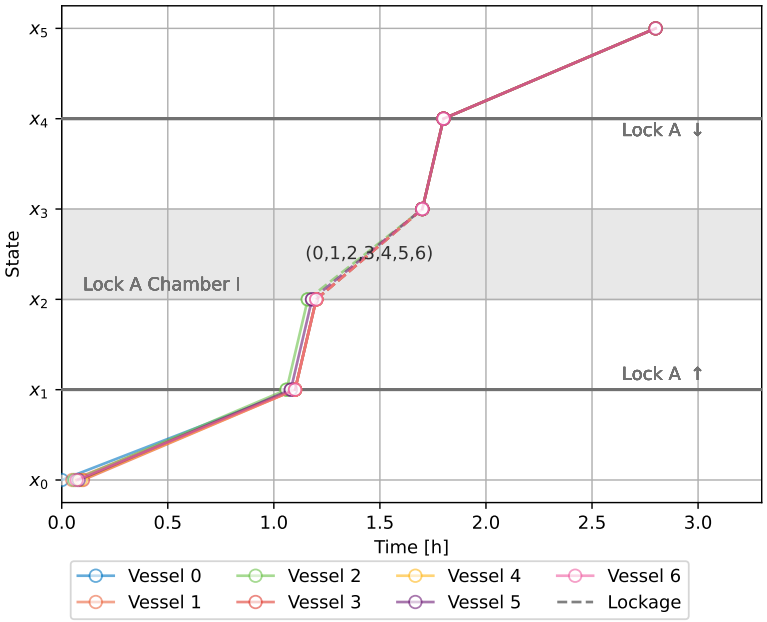


Figure 6.19: *Journey Plot Case E Test 2.*

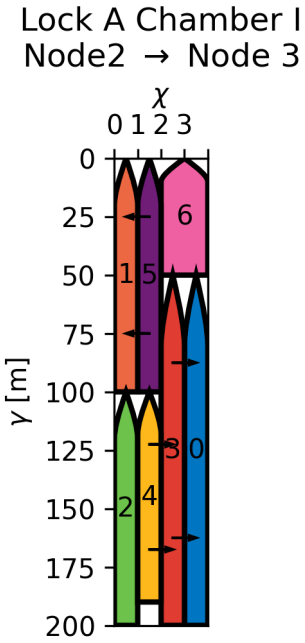


Figure 6.20: *Lockage of Case E Test 2.*

The KPI values are listed in Table 6.34. Interestingly, the average delay is not 0%, even though the vessels can all be processed in the same lockage and the extra lockage time is 0. This is caused by the fact that the vessels have slightly offset departure times, causing earlier vessels to be delayed a few minutes.

Table 6.34: *KPIs Case E Test 2.*

Performance Indicator	Value
J	19.15
$\mathcal{A}$	19.15 h
$\mathcal{H}$	-
$\mathcal{L}$	1
$\mathcal{T}$	1.32%

6.7 Verification Case F - Multi-Chamber, Multi-Vessel Placement Model

Case F is the final verification case, examining a multi-lock case. The graph that is used has one multi-chamber lock and one single chamber lock, as depicted in Figure 6.21. The behavior of vessels with flammable and explosive goods is also tested. The node and arc parameters are listed in Table 6.35 and Table 6.36. Lock A has two chambers of differing sizes and operation times, and Lock B has one large chamber.

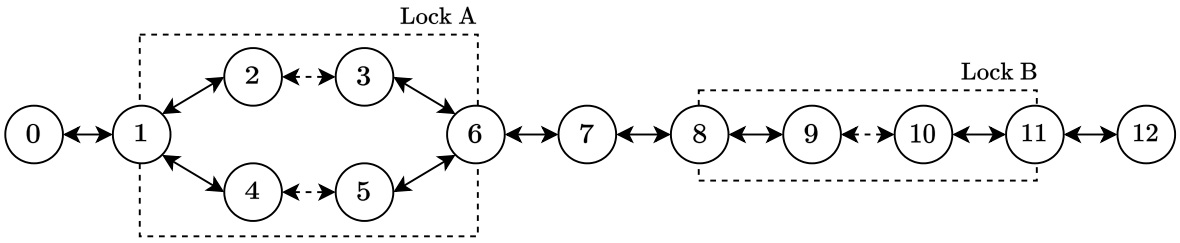


Figure 6.21: *Topology graph for verification Case F.*

Table 6.35: *Node parameters verification Case F.*

Node i	$p_{x,i}$ $[km]$	$p_{y,i}$ $[km]$
0	0	0
Lock A $(1, 2, 3, 4, 5, 6)$	12.5	0
7	25	0
Lock B $(8, 9, 10, 11)$	37.5	0
12	50	0

6.7.1 Test 1 - Multiple Locks and Flammable and Explosive Goods

This test tests the model's behavior with multiple locks and vessels with flammable and explosive goods. The test consists of eight vessels of different sizes moving in the same direction and departing at the same time, as listed in Table 6.37. One vessel is carrying flammable goods, so it should not moor to any other vessel. Another vessel is carrying explosive goods, so it should not be grouped into lockages with any other vessels. The cumulative arrival time objective is used. With the increased complexity of the graph and the set of vessels, it becomes harder to make meaningful predictions. Instead, the solution will be analysed for consistency.

The nine populated lockages of the solution are shown in Figure 6.23. It can be seen that the vessel with flammable goods, vessel 0, is not moored to any other vessel in any of the lockages.

Table 6.36: *Arc parameters for verification Case F.*

Arc (i,j)	$d_{(i,j)}$ $[km]$	Specified $\tau_{(i,j)}(k)\,\forall k \in \mathcal{K}$ $[h]$	$N_{\chi,(i,j)}$	$N_{\gamma,(i,j)}$ $[m]$	$\overline{\tau}_{(i,j)}$ $[h]$
$(0,1),(1,0),(6,7),(7,6),$ $(7,8),(8,7),(11,12),(12,11)$	Euclidean	-	-	-	-
Lock A Waiting Areas $(1,2),(2,1),(1,4),(4,1),$ $(3,6),(6,3),(5,6),(6,5)$	-	0.1	-	-	-
Lock A Chamber I $(2,3),(3,2)$	-	0.5	3	150	0.1
Lock A Chamber II $(4,5),(5,4)$	-	0.75	2	200	0.1
Lock B Waiting Areas $(8,9),(9,8),(10,11),(11,10)$	-	0.1	-	-	-
Lock B Chamber I $(9,10),(10,9)$	-	1	4	250	0.1

Table 6.37: *Vessel parameters Case F Test 1.*

vessel k	$v_{\max,(i,j)}(k)\,\forall(i,j)\in\mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)$ $[h]$	W_χ	W_γ $[m]$	$\triangledown$	$\triangledown\triangledown$
0	10	0	12	0	1	150	✓	
1	10	0	12	0	2	175		
2	10	0	12	0	1	75		✓
3	10	0	12	0	2	200		
4	10	0	12	0	2	100		
5	10	0	12	0	1	125		
6	10	0	12	0	1	100		
7	10	0	12	0	1	100		

Furthermore, vessel 2 with explosive goods is always put into separate lockages. The schedule in Figure 6.22 shows the predictable pattern of prioritizing lockages with multiple vessels on lock A, followed by lockages for vessels that can join grouped lockages on lock B, and then having the vessel with explosive goods go last, so no other vessels are held up by the single-vessel lockages it requires. Lock B has a larger chamber than either of the chambers of lock A, so more vessels can fit into its lockages, causing vessels that passed through lock A separately to be grouped together on lock B. It can be concluded that the model is functioning correctly for multiple locks and vessels with flammable or explosive goods in this test.

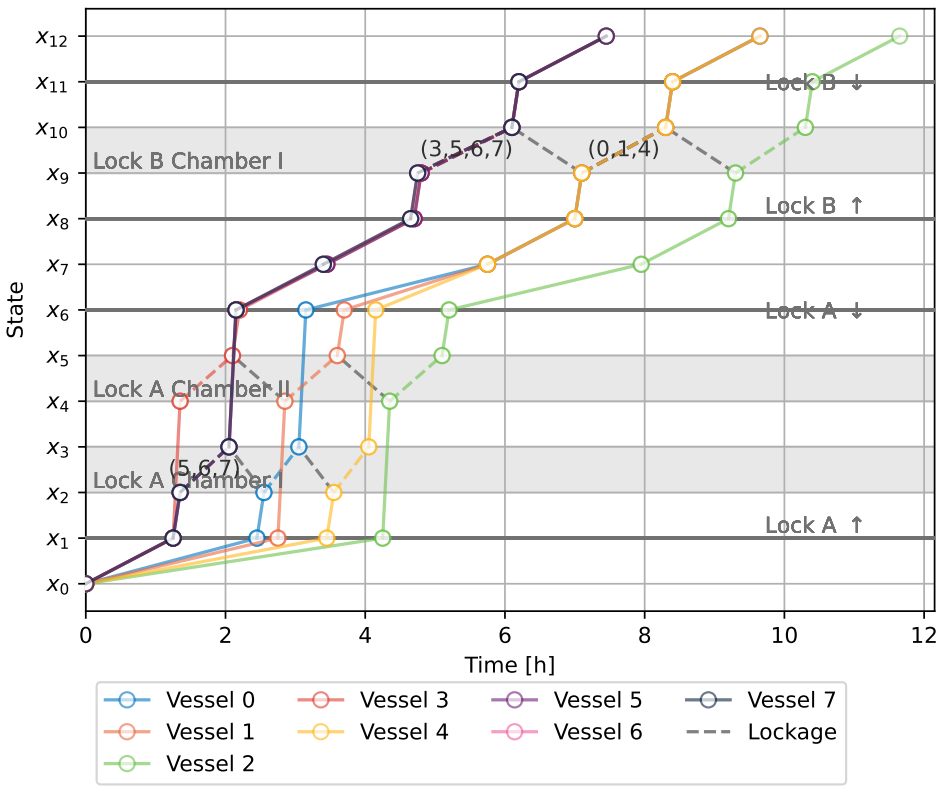


Figure 6.22: *Journey Plot Case F Test 1.*

Table 6.38: *KPIs Case F Test 1.*

Performance Indicator	Value
J	70.4
$\mathcal{A}$	$70.4\,h$
$\mathcal{H}$	-
$\mathcal{L}$	15
$\mathcal{T}$	25.72%

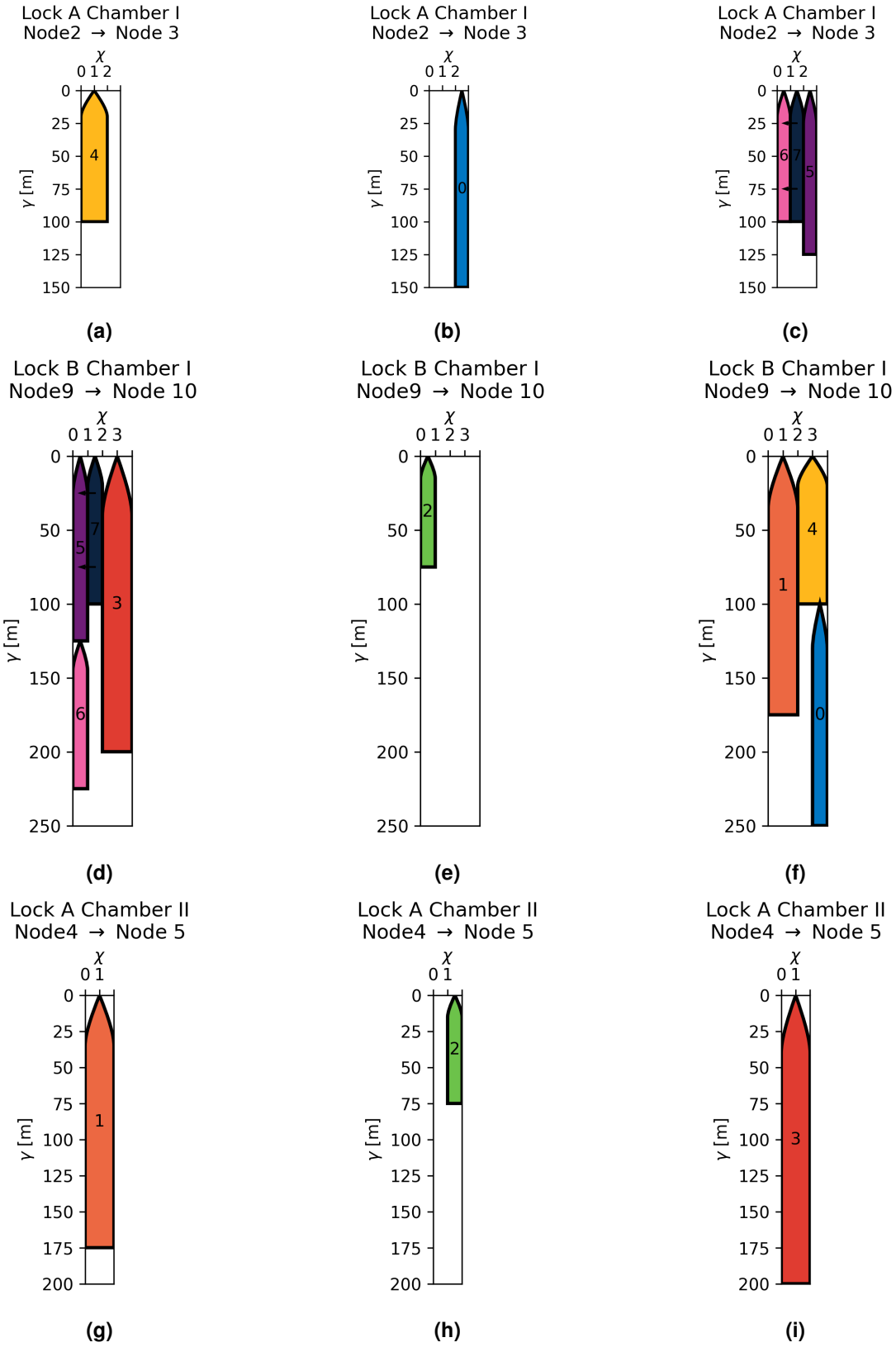


Figure 6.23: *Populated lockages of verification Case F Test 1.*

6.7.2 Test 2 - Multiple Travel Directions and Intermediate Destinations

The tests of the multi-vessel, multi-chamber placement model have all had all of the vessels travel in the same direction. Furthermore, vessels have always had their destinations at one of the ends of the graph, when this is not required. In this test, those features will be examined. It is the final test that will be run, so it will be given a high volume of vessels with more variation in parameters, as shown in Table 6.39. All vessels have the unweighted minimum travel time objective, except for vessel 7 which has a planned arrival time.

For completeness, the vessels' journeys are shown in the journey plot in Figure 6.24, though it becomes difficult to read the journey plots for scenarios with many vessels. Nevertheless, it shows that all vessels, no matter their starting point or destination, have a consistent route. Mixing of objectives also works, as vessel 7 arrives at its destination exactly on time, with all other vessels still being scheduled to be as fast as possible.

The main purpose of the journey plot in this test is so it can be used as a reference for the lockage plots depicted in Figure 6.25. All of the vessels correctly fit into their lockages, properly mooring to other vessels when necessary. The KPI values are listed in Table 6.38.

Table 6.39: *Vessel parameters Case F Test 2.*

vessel k	$v_{\max,(i,j)}(k)\ \forall (i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)\ [h]$	W_χ	$W_\gamma\ [m]$	$\hat{x}_{d(k)}\ [h]$
0	8	0	12	0	1	80	-
1	12	0	12	0.5	2	200	-
2	10	0	7	1	1	150	-
3	10	7	12	2	1	175	-
4	11	0	12	0.4	1	100	-
5	15	7	12	1	1	100	-
6	12	0	12	0	1	150	-
7	15	12	0	0	1	130	10
8	11	12	0	0.5	1	100	-
9	12	12	0	1	1	120	-
10	9	12	7	0.7	2	90	-
11	14	7	0	3	1	150	-
12	13	12	0	0	1	140	-

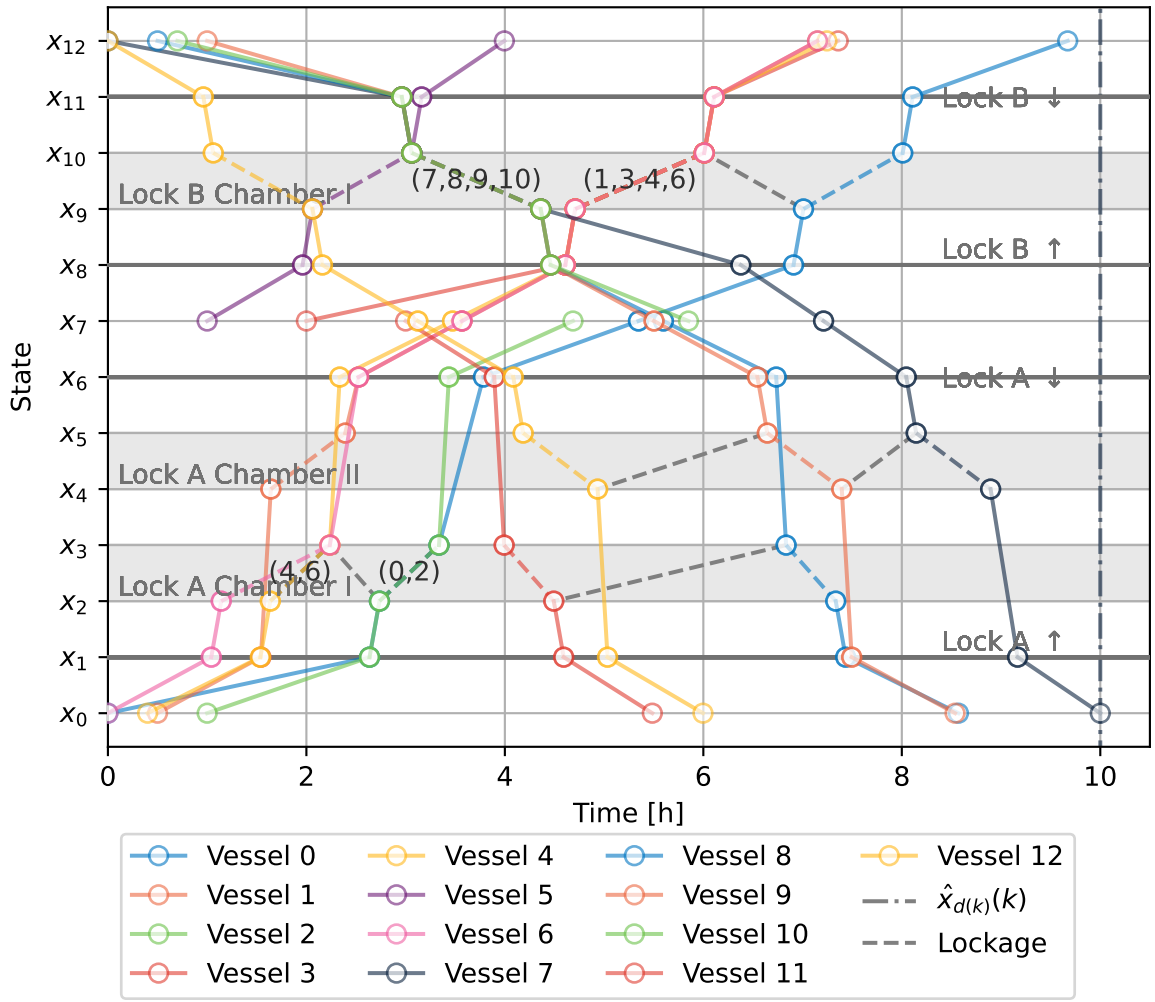


Figure 6.24: *Journey Plot Case F Test 2.*

Table 6.40: *KPIs Case F Test 2.*

Performance Indicator	Value
J	71.60
$\mathcal{A}$	81.60 h
$\mathcal{H}$	-
$\mathcal{L}$	18
$\mathcal{T}$	15.53% (Excluding vessel 7)

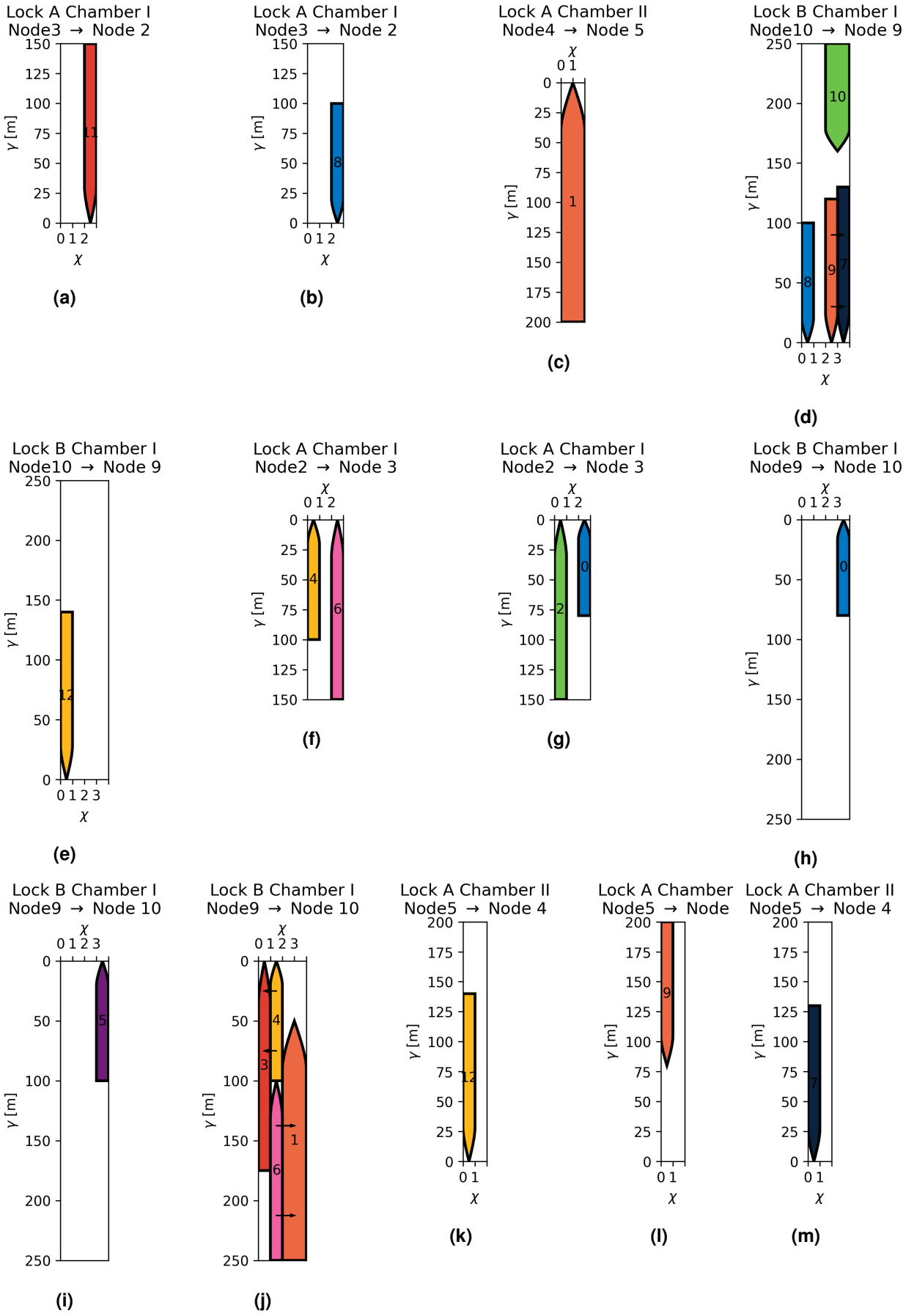


Figure 6.25: *Case F Test 2 Lockages.*

6.8 Complexity

The three models introduced in this thesis have different levels of complexity. Complexity is an indicator of how computationally difficult it will be to find a solution to the MILP problem assembled by the scheduler. The models in this thesis only have binary and continuous variables, but no integer variables. The biggest indicator of complexity for these types of problems is the number of binary variables, followed by the structure of the problem [82]. The number of constraints also increases complexity but to a lesser extent.

It is beyond the scope of this thesis to identify special structures in the optimization problems to then try to exploit them with the solver. Instead, a general analysis of the number of variables and constraints can be performed and related to the computation time through experimentation, allowing for comparison between the models and predictions of if future implementations are required to reduce complexity.

Subsection 6.8.1 explains the important parts of the branch-and-bound algorithm, which is used to solve MILPs. Subsection 6.8.2 is a discussion of the variables of the models and how they scale with the model input. Finally, experimental comparisons between the complexities of the three different scheduling models introduced in this report are made in Subsection 6.8.3.

6.8.1 Branch-and-Bound

The solver used, Gurobi, uses the branch-and-bound solution algorithm to solve MILP problems [83]. The simplest branch-and-bound algorithm works by splitting the problem into branches where binary decision variables have an assigned value [82]. Each binary variable can split the problem into two subproblem branches, one where the variable has a value of 0, and one where the variable has a value of 1, as shown for a two-variable problem in Figure 6.26. This results in 2^{n_b} versions of the problem, where n_b is the number of binary variables.

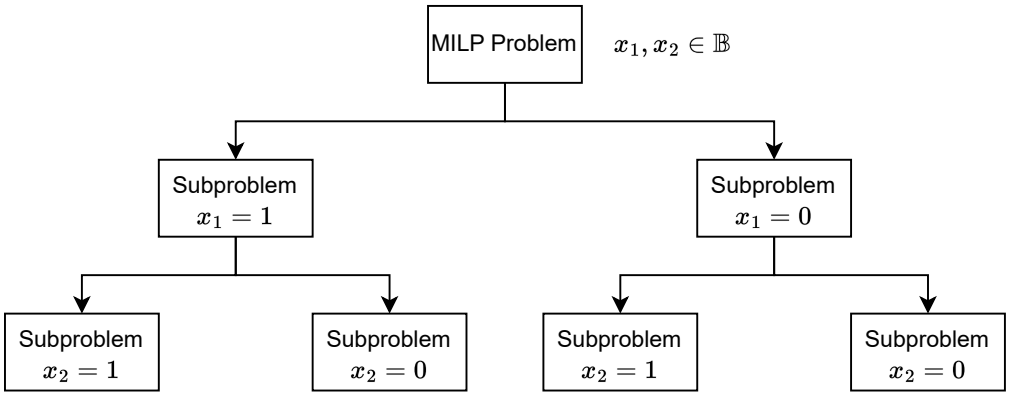


Figure 6.26: *Full subproblem tree of a two binary variable branch and bound problem.*

Instead of exhaustively examining every single solution, tests can be performed to remove entire branches of possible solutions earlier at the top of the tree. This is done by, among other things, comparing LP relaxed solutions of subproblems, where binary variables that do not have a set value according to the subproblem tree already are allowed to have values between 0 and 1, turning the entire problem into a linear programming (LP) problem with only continuous variables which can be solved very quickly, to theoretical solution bounds. This is also where the number of constraints matters for MILP problems, as the number of constraints is a large factor in the time required find a solution to LP relaxations [82].

Numerous things factor into speeding up the branch-and-bound algorithm. The choice of which

variables to branch at first, or which subproblem to evaluate first all contribute to the average time required to find a solution [82]. Solvers use increasingly advanced techniques, like heuristics and cutting planes [83]. It is hard to predict how these will interact with the specific structure of the MILP problems examined in this project. What is predictable, however, is that the size of the search space scales exponentially with the number of binary variables, that LP relaxations take longer to evaluate with more constraints, and that bigger problems with more possible (and more varied) near-optimal solutions make it less likely that branches can be cut higher up in the tree.

6.8.2 Variables

The number of variables introduced can be predicted beforehand and can be assumed to be the biggest factor contributing to the complexity of the problem, thus an overview of the number of variables introduced depending on a given scenario is shown in Table 6.41. The assumption that $n_{steps} = n_{vessels}$ has been used.

Table 6.41: *Number of each variable introduced. C = continuous variable. B = binary variable. If there is a ✓ in the exact column, the number lister is exact. Otherwise, it is an approximation.*

Variable	Type	Number	Highest Order Term	Exact
$x_i(k)$	C	$\|N\|n_{vessels}$	$\|N\|$, $n_{vessels}$	✓
$w_{(i,j)}(k)$	B	$\|\mathcal{D}\|n_{vessels}$	$\|\mathcal{D}\|$,$n_{vessels}$	✓
$z_{(i,j)}(k, k-\mu))$	B	$\|\mathcal{D}_{locks}\|n_{vessels}(n_{vessels}-1)$	$n^2_{vessels}$	✓
$s_{(i,j)}(k, k-\mu)$	B	$\|\mathcal{D}_{locks}\|n_{vessels}(n_{vessels}-1)$	$n^2_{vessels}$	✓
$\theta_{synch,(i,j)}(k, k-\mu)$	C	$\|\mathcal{D}_{locks}\|n_{vessels}(n_{vessels}-1)$	$n^2_{vessels}$	✓
$t_{synch,(i,j)}(k)$	C	$\|\mathcal{D}_{locks}\|n_{vessels}$	$\|\mathcal{D}_{locks}\|$, $n_{vessels}$	✓
$m_{(i,j)}(k, k-\mu)$	B	$\|\mathcal{D}_{locks}\|n_{vessels}(n_{vessels}-1)$	$n^2_{vessels}$	✓
$\alpha_{\chi,(i,j)}(k, g)$	B	$\sum_{(i,j)\in\mathcal{D}_{locks}} N_{\chi,(i,j)} n_{vessels} n_{vessels}$	$n^2_{vessels}$	✓
$h_{\chi,(i,j)}(k, g)$	C	$\sum_{(i,j)\in\mathcal{D}_{locks}} N_{\chi,(i,j)} n_{vessels}(n_{vessels}+1)$	$n^2_{vessels}$	✓
$\hat{h}_{(i,j)}(k)$	C	$\|\mathcal{D}_{locks}\|n_{vessels}$	$\mathcal{D}_{locks}$, $n_{vessels}$	✓
$t_{bow,\chi,(i,j)}(k, g)$	C	$n_{vessels}\sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{D}_{locks}}(N_{\chi,(i,j)} - W_\chi(k)+1)$	$\approx n^2_{vessels}$	✓
$\lambda_{\chi+p,(i,j)}(k, g)$	B	$n_{vessels}\sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{D}_{locks}} W_\chi(k)(N_{\chi,(i,j)} - W_\chi(k)+1)$	$\approx n^2_{vessels}$, $W_\chi(k)^2$	
$\theta_{bow,\chi,(i,j)}(k, g)$	C	$n_{vessels}\sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{D}_{locks}}(N_{\chi,(i,j)} - W_\chi(k)+1)$	$\approx n^2_{vessels}$	
$\lambda_{\chi,(i,j)}(k, g)$	B	$n_{vessels}\sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{D}_{locks}}(N_{\chi,(i,j)} - W_\chi(k)+1)$	$\approx n^2_{vessels}$	
$\check{h}_{(i,j)}(k)$	C	$\|\mathcal{D}_{locks}\|n_{vessels}$	$\mathcal{D}_{locks}$, $n_{vessels}$	✓

Most variables scale linearly with the number of nodes, the number of arcs, or the number of lock arcs, which are graph parameters. The highest order term in the equations giving the number of each of the binary variables is generally $n^2_{vessels}$. In a real application, it is expected that the number of vessels will be large enough to let $n^2_{vessels}$ be the dominant term in determining the number of binary variables.

Going from the single-vessel, single-chamber model to the multi-vessel, multi-chamber capacity model, the introduction of synchronization does not introduce variables that have higher order scaling terms than before, rather there are just more variables scaling with $n^2_{vessels}$. This is the same for the jump from the multi-vessel, multi-chamber capacity model to the multi-vessel, multi-chamber placement model. However, it should be noted that the linear terms of the variables associated with locks are larger for the placement variables, as the widths are summed for each of the chambers and each chamber's width is at least larger than or equal to one.

The single-vessel, single-chamber model has one $n^2_{vessels}$ scaling binary variable, the multi-vessel, multi-chamber capacity model has two, and the multi-vessel, multi-chamber placement model has six. Considering the fact that the number of possible subproblems scales exponentially with the number of binary variables, this means that with four vessels, neglecting binary variable scaling

from all other sources, 2^{16}, 2^{32}, and 2^{96} subproblems are available for the models in order. This example serves to demonstrate that the addition of the placement problem has made the solution space exponentially more complex.

6.8.3 Experimental Comparisons

The previous subsection was merely intended to give an idea of the computational difficulty in finding an optimal solution to the problem when comparing the different models. As explained in Subsection 6.8.1, branch-and-bound solvers are much more sophisticated than just checking all available subproblems.

As the number of vessels was identified as a dominant factor for determining the number of binary variables, experimental simulations are run for all three models with the number of vessels ranging from 2 to 15. The tests are run on the same graph as verification Case A depicted in Figure 6.1. The nodal parameters are the same as in Table 6.2. The arc parameters are given in Table 6.42.

Table 6.42: *Arc parameters for the complexity tests.*

Arc (i,j)	$d_{(i,j)}$ $[km]$	Specified $\tau_{(i,j)}(k)\,\forall k \in \mathcal{K}$ $[h]$	$\rho(k)$	$N_{\chi,(i,j)}$	$N_{\gamma,(i,j)}$ $[m]$	$\overline{\tau}_{(i,j)}$ $[h]$
$(0,1),(1,0),$ $(4,5),(5,4)$	Euclidean	-	-	-	-	-
Lock A Waiting Areas $(1,2),(2,1),$ $(3,4),(4,3)$	-	0.1	-	-	-	-
Lock A Chamber $(2,3),(3,2)$	-	0.5	8	4	200	0.1

Five runs are done for each amount of vessels, to reduce some of the variability of the computation times obtained. The vessels are randomly generated according to the parameters listed in Table 6.43, with each vessel being randomly generated separately. The ranges and sets are set up in such a way that the scenarios are roughly comparable between models. The scenarios do not have to be too similar, this analysis is just done to give an indication of complexity.

Table 6.43: *Randomized parameters for each of the vessels. For a range, the parameter is selected uniformly between the start and endpoints of the range. For sets, one entry of the set is chosen.*

Parameter	Range	Set
$v_{\max,(i,j)}(k) \forall (i,j) \in \mathcal{D}$	-	$\{8,9,10,11,12,13,14,15\}$
$(b(k),d(k))$	-	$\{(0,5),(5,0)\}$
$u(k)$	$[0,2]$	-
$\rho(k)$	-	$\{1,2,3,4,5,6,7,8\}$
$W_\chi(k)$	-	$\{1,2,3,4\}$
$W_\gamma(k)$	-	$\{50,100,150,200\}$

Abbreviations are used to refer to the models. The single-vessel, single-chamber model is abbre-

viated to SVSC, the multi-vessel, multi-chamber capacity model is abbreviated to MVMCC, and the multi-vessel, multi-chamber placement model is abbreviated to MVMCP.

Loglog plots of the numbers of constraints and binary variables as a function of the number of vessels are shown in Figure 6.27. The log slopes of the binary variables are approximately two, indicating quadratic growth as predicted in Table 6.41. They are slightly lower because of the variables that have linear growth with the number of vessels. The jump in complexity between the SVSC model and MVMCC model is small. Meanwhile, the jump from the MVMCC model to the MVMCP model is considerable, so considerably larger calculation times are expected.

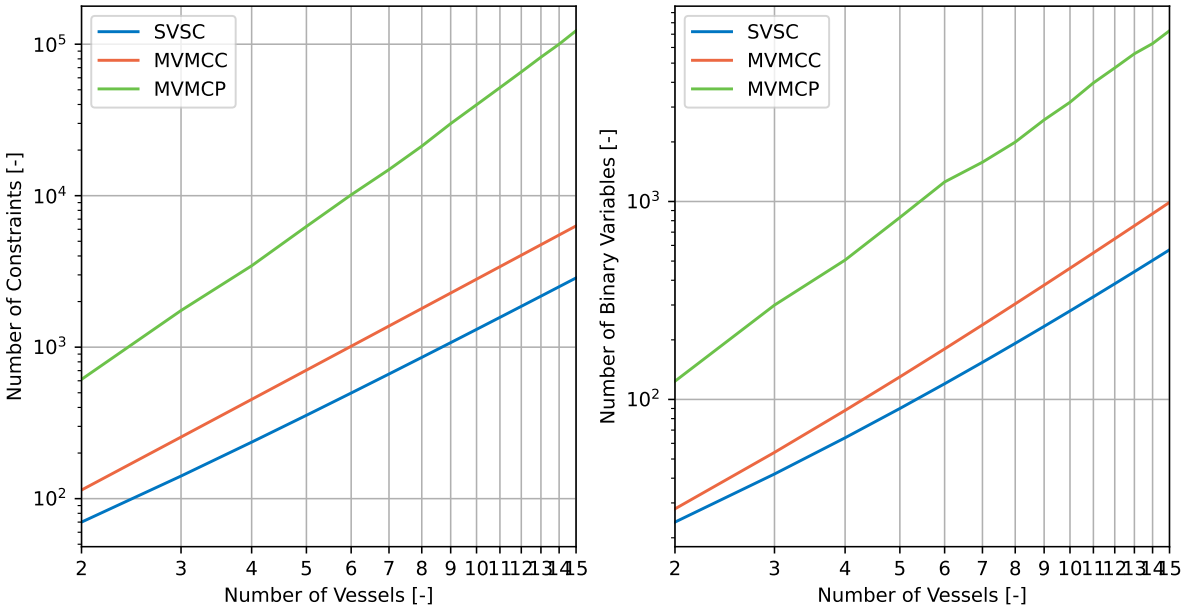


Figure 6.27: *Left: Number of constraints as a function of the number of vessels. Right: Number of binary variables as a function of the number of vessels. Note that the lines drawn show averages over the five runs per number of vessels. This average is only relevant for the MVMCP model, as that is the only model where vessel parameters influence the number of constraints and variables*

Figure 6.28 shows a loglog plot of the computation times of the tests. The average and the value ranges are given. Some minor observations can be made. It should be noted that these observations are only certain to apply to the specific single-lock, single-chamber case being considered with vessel parameter ranges as indicated in Table 6.43.

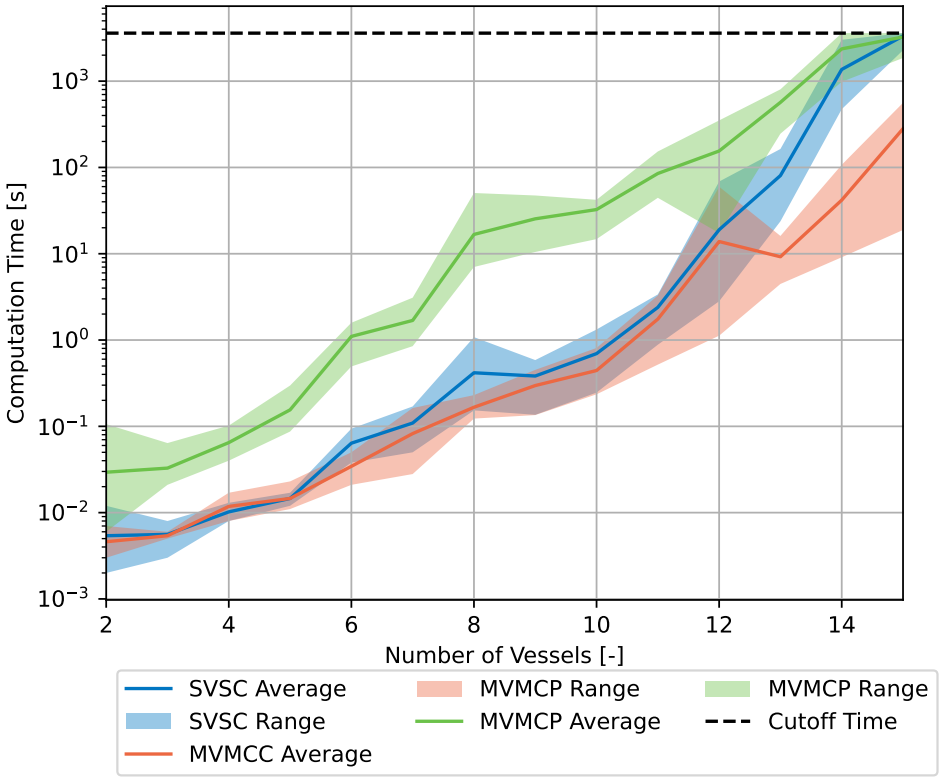


Figure 6.28: *Log plot showing the computation time as a function of the number of vessels in the problem. The solid lines show the averaged values over all runs and the colored areas show the range between the maximum and the minimum computation time value for that number of vessels.*

First, the MVMCC model has computation times that are as low or lower than those of the SVSC model, even though it has more binary variables and constraints according to Figure 6.27. The cause for this could be the way this simulation is set up. It is one lock chamber with a number of vessels all arriving at about the same time. In the SVSC model, only one vessel gets to pass the lock at a time and the vessels are all relatively similar, thus there will be many allowable orderings of the vessels going through one by one that are equivalent to or near the global optimum. This makes it harder to prune branches in the branch-and-bound solution method. On the other hand, the MVMCC model has more binary variables and constraints, but these constraints lead to there being fewer possible optimal or near-optimal scenarios, which can allow the solution algorithm to arrive at a solution faster through efficient fathoming.

Second, the MVMCP model has computation times that are nearly an order of magnitude higher on average. This is not unexpected according to the number of constraints and binary variables in Figure 6.27. The MVMCP model achieves more realistic results by introducing placement, but it comes at the cost of complexity, as also predicted in Chapter 5.

Further testing should be performed to establish if these observations hold for other scenarios.

The major observation that should be made from Figure 6.28 is that none of the models will be able to be directly applied to a global Dutch or European network in practice. Hundreds of vessels may be in transit at any given time and that would lead to impractical computation times. It is likely that this observation also applies to problems with other graph topologies and vessel parameters, just because of the sheer number of binary variables that must be optimized when the number of vessels becomes large.

To arrive at the 15-20 minutes computation time limit established in Section 3.6, different approaches like heuristic solution algorithms or distributed optimization are required in the future. Heuristics like (adaptive) large neighborhood search have previously been applied in lock scheduling to significantly reduce computation times at a small or no cost to optimality [48] [49].

Additionally, in the case of SMPL train scheduling, distributed optimization implemented as outlined in Subsubsection 2.2.4.5 was successfully applied to schedule large railway networks at only a small penalty to the achieved objective function value [63]. This is particularly interesting for the scheduling of vessels, as, just like trains, they each have a nearly set route, so it is possible to group their constraints and variables based on if they might interact with each other along their routes. Two distributed optimization procedures were introduced in [63]. The key mechanism that speeds up computation times is the fact that only a subset of the decision variables of the centralized optimization problem is solved for at one time, keeping the other decision variables constant. If an initial feasible solution is used, continual improvement can be achieved by iterating over the variable groupings while maintaining the feasibility of the central problem.

The models introduced in this project could serve as starting points for future heuristic or distributed implementations. Furthermore, as also concluded in [32], the optimal solutions of the centralized MILP models introduced can be used to benchmark the performance of those future implementations.

6.9 Comparison to Practice

The three scheduling models introduced in this thesis should give optimal solutions for their respective purposes and assumptions. However, there is not yet a frame of reference for how much better the solutions of these scheduling models are compared to practice. This section is intended to provide such a comparison through the use of a simulation model based on Dutch inland waterway practices and regulations. The simulation model in this section is not as sophisticated as the procedures currently used by Rijkswaterstaat, it does not account for interconnectedness between cascaded locks, for instance, but it should provide a first-order estimate of how the scenarios considered in the optimization problems would end up in practice.

Subsection 6.9.1 gives a brief description of the simulation and its assumptions. The full simulation model is outlined in Subsection 6.9.2. The placement heuristic used to place the vessels in the chambers is explained in Subsection 6.9.3. Finally, comparisons between the simulation model and the scheduling models based on the verification cases are made in Subsection 6.9.4.

6.9.1 Simulation Description

The simulation is a simple time-based simulation that checks if certain events, like a lockage starting or ending, take place every time interval dt, updating the components of the system accordingly. It uses the same components as the scheduling models. The only difference in input is that locks are now modeled as lock nodes, with $\mathcal{N}_{locks} = \{0, 1, ..., n_{locks} - 1\}$ being the set of lock nodes. A lock node contains all of the information about the waiting areas and chambers of that lock. This setup requires less bookkeeping when assigning vessels to chambers in the locks.

Unless otherwise mentioned, key operational constraints and assumptions of locks and vessels made in the scheduling models are still applicable. For instance, lockages in the same direction must still have an (empty) lockage in the other direction in between.

The simulation has 2 modes, mode 1 corresponding to the SVSC and MVMCC models, and mode 2 corresponding to the MVMCP model. The capacity of the chambers and sizes of the vessels in the mode 1 simulation model can just be set equal to 1 to get it to behave like the SVSC model.

The following key assumptions apply:

- **Vessels are assigned to lock chambers on an FCFS basis.** *This is how commercial vessels are assigned to chambers in practice, with some exceptions [18].*
- **Vessels that are smaller may overtake larger vessels that arrived before them if the larger vessels do not fit into a lockage, but the smaller vessels do.** *This is also how it is done in practice [18].*
- **Only vessels in the waiting area are considered for placement in the chambers.** *Lockmasters do also keep track of vessels approaching waiting areas from further out, but this is left outside of the scope of this simulation [25].*
- **Vessels in the waiting area of a lock are assigned to the chamber with the lowest processing time among the available chambers.** *Heuristic rule that promotes fairness in assigning vessels to the chambers according to the FCFS principle.*
- **Vessels travel at their maximum velocity on all arcs between nodes.** *Vessels are modeled as selfish actors in the simulation, meaning they will try to get to their destination as fast as possible by traveling at their maximum velocity.*
- **For the placement model, vessels are placed according to a bin-based placement heuristic that obeys mooring constraints.** *Placement could also be done according to an optimization model, but that would increase the run-time and complexity of the simulation.*
- **All vessels have the minimum arrival time objective.** *Implementing rules for vessels with different objectives is too complex for the purposes of this model.*
- **A lockage with a set start time that has not started yet can be delayed to include a new vessel if a vessel that will fit in the lockage arrives in the waiting area of the lock.**
- **Every vessel has at most one path, excluding choice of lock chambers, between departure and destination node.** *This requires no implementation of a decision making mechanism for what route a vessel should follow, other than which chamber they should pass through.*

6.9.2 Simulation Model

The simulation model consists of six blocks that update the vessels and lock components until all vessels have arrived. The model displayed here is the mode 1 simulation model. The mode 2 simulation model has a different block I. The blocks are:

- **Initialization (Figure 6.29):** Instances of simulation components are created according to the given input, global constants are set, routes of arcs between endpoints are selected for all vessels.
- **0 (Figure 6.30):** Checks for departing vessels, vessels transitioning between arcs, vessels arriving in waiting areas before a lockage, and vessels arriving at their destinations.
- **I (Figure 6.31):** Assignment of vessels waiting in waiting areas to chambers.
- **II (Figure 6.32):** Starting and ending chamber processing.
- **III (Figure 6.33):** Assigning vessels that have spent the minimum traversal time in waiting areas after lockages to the next arc on their route and increasing the distance traveled of the vessels currently traveling on an arc.
- **IV (Figure 6.34):** Increasing waiting time counters of vessels in waiting areas and increasing chamber processing counters for currently processing chambers.

- V (Figure 6.35): Increasing time counters and checking if all vessels have arrived.

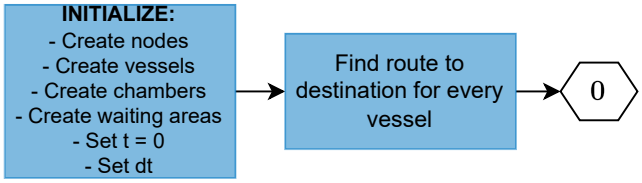


Figure 6.29: *Simulation initialization block.*

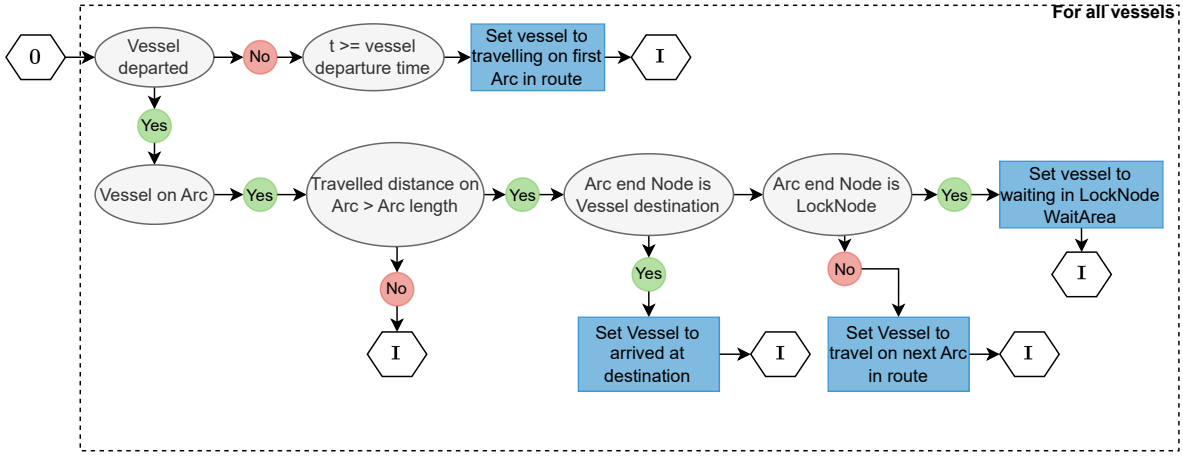


Figure 6.30: *Simulation block 0.*

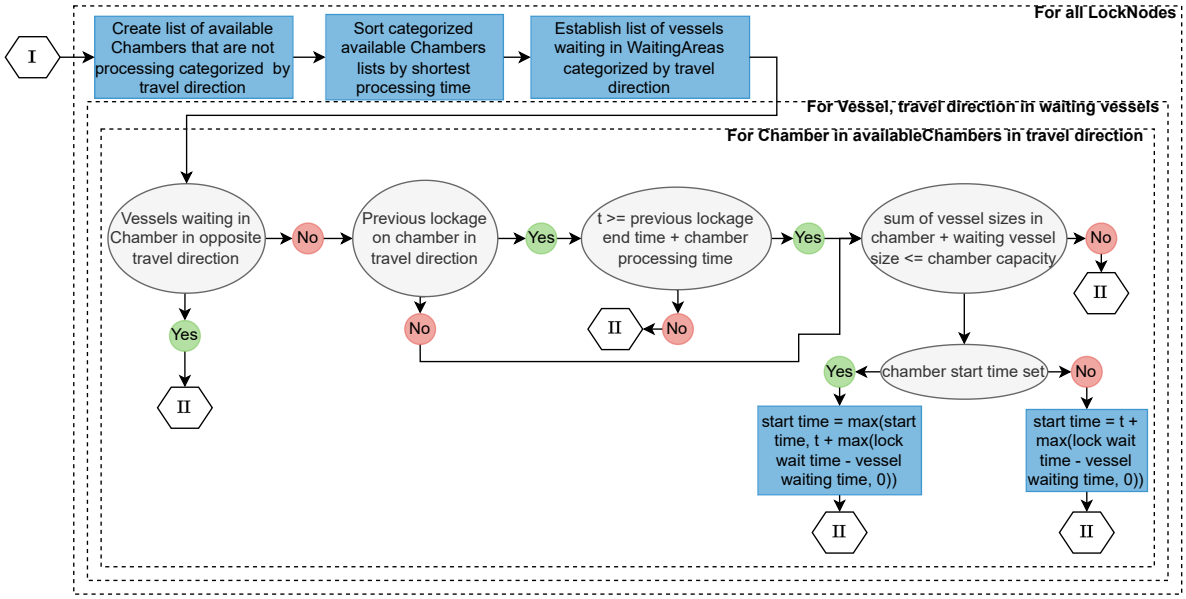


Figure 6.31: *Simulation block I.*

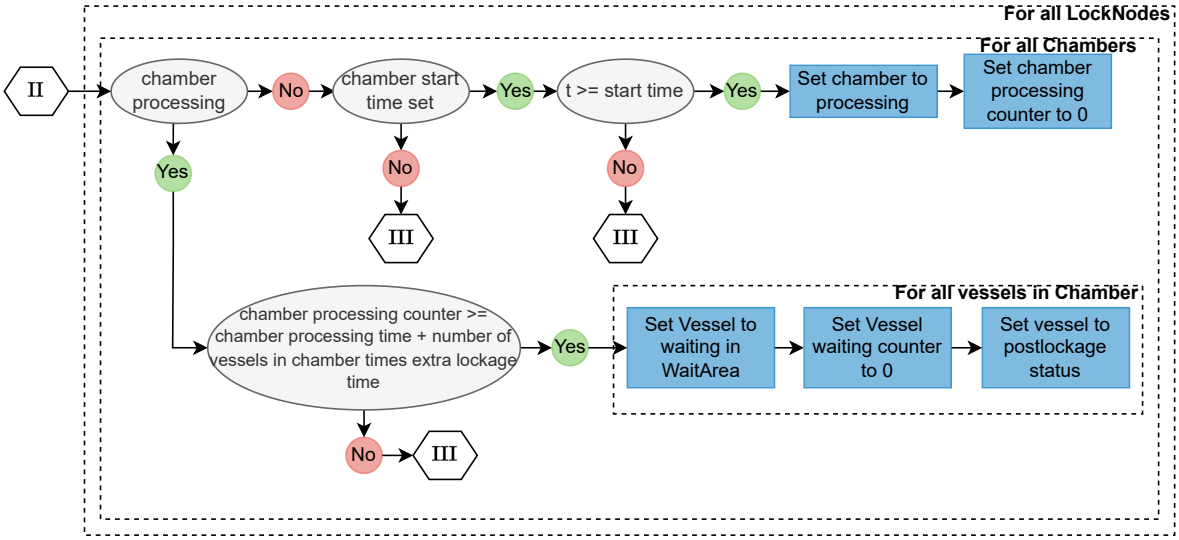


Figure 6.32: *Simulation block II.*

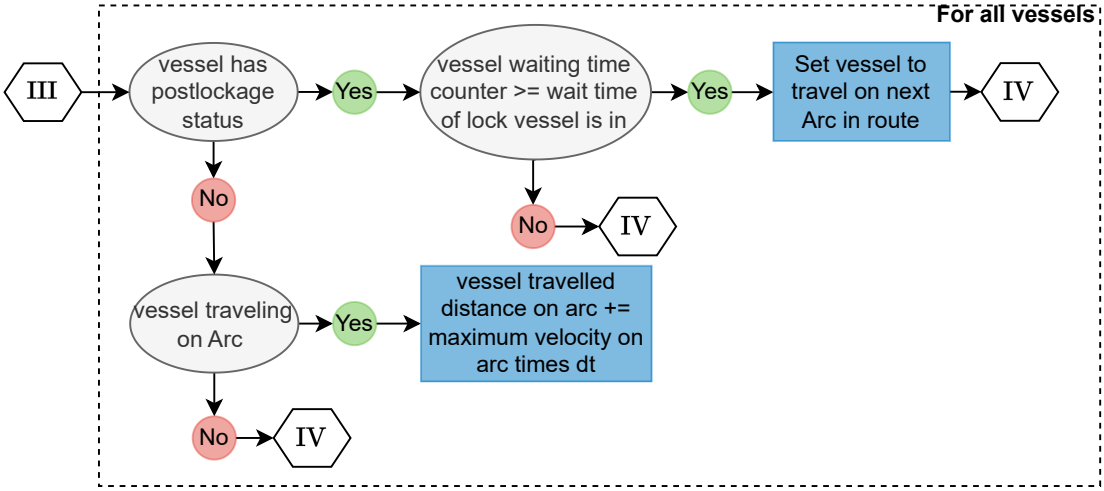


Figure 6.33: *Simulation block III.*

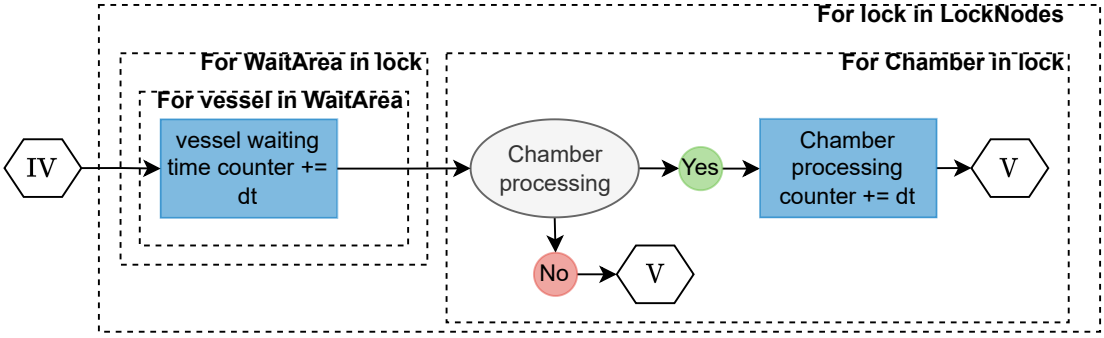


Figure 6.34: *Simulation block IV.*

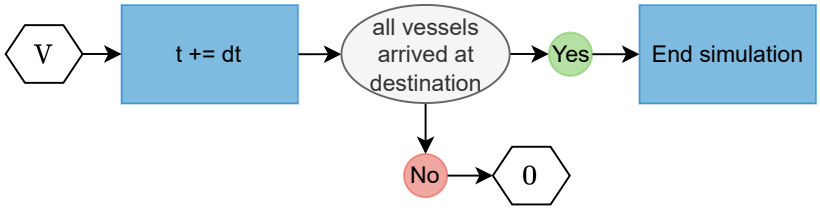


Figure 6.35: *Simulation block V.*

6.9.3 Placement Heuristic

The vessels are placed into the chambers according to a heuristic placement sequence inspired by the placement heuristics introduced in [32].

The goal of the heuristic is to try to place the vessels into the chamber in order of arrival in the waiting areas to obey the FCFS principle, with attachment to the quays being preferred. If a bigger vessel that has a longer waiting time does not fit, then smaller vessels that have been waiting for a shorter amount of time are considered. The widths of the chambers and vessels are still discretised into bins to be consistent with the scheduling models. Similar to the SMPL placement sequence, the height of the vessel stack in each of the bins is kept track of and vessels cannot be placed further forward into the chamber than the height of the bins.

As a first step, vessels in the waiting area are ordered in a sequence based on the highest waiting time count. Then, the following steps are followed for each vessel in the sequence:

1. Check if the vessel fits into the dimensions of the chamber. If not, go to the next vessel in the sequence.
2. Check if the vessel can be moored to the left quay without overlapping with other vessels or sticking out of the chamber. If yes, then place the vessel and go to the next vessel in the sequence.
3. Check if the vessel can be moored to the right quay without overlapping with other vessels or sticking out of the chamber. If yes, then place the vessel there and go to the next vessel in the sequence.
4. For each bin in which the vessel could be placed where it would not be adjacent to the sides of the chambers, check the vessels that the vessel being placed could be moored to if it was placed in that bin. If an acceptable fit is found, then moor the vessel there and go to the next vessel in the sequence.
5. Go to the next vessel in the sequence without placing the vessel.

The placement heuristic is demonstrated with an example in Figure 6.36. Vessel 0 is too long for the chamber so it is not placed. Vessel 1 fits on the left quay, so it is placed there. Vessel 2 does not fit on the left quay, so it is placed on the right quay. Vessel 3 does not fit on any of the quays and it is too big to moor to any of the vessels already placed, so it is not placed. Vessel 4 fits on the left quay. Vessel 5 does not fit on either quay, but it can be moored to vessel 1, so it is placed there. The final lockage then includes vessels 1, 2, 4, and 5.

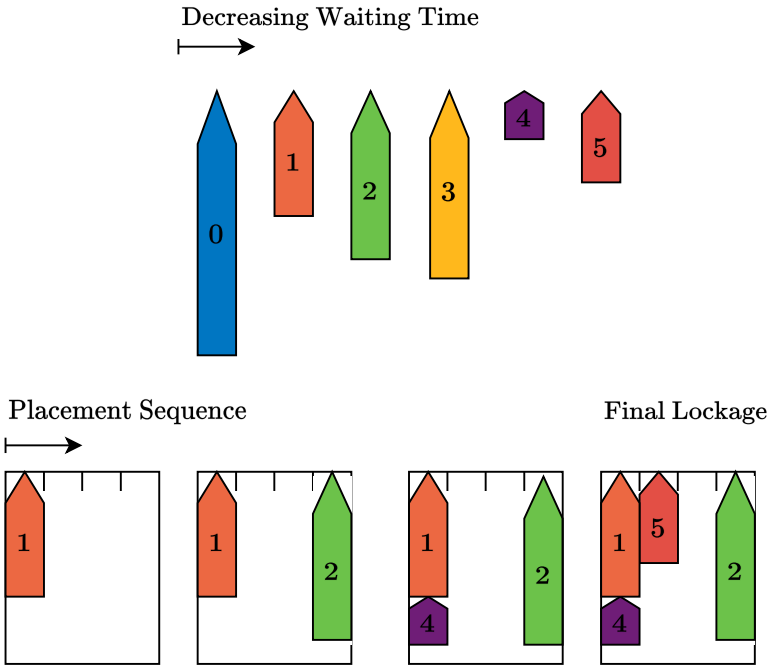


Figure 6.36: *Heuristic placement sequence example.*

Integrating the heuristic placement model into the simulation model requires changes to be made to simulation block I, as depicted in Figure 6.37.

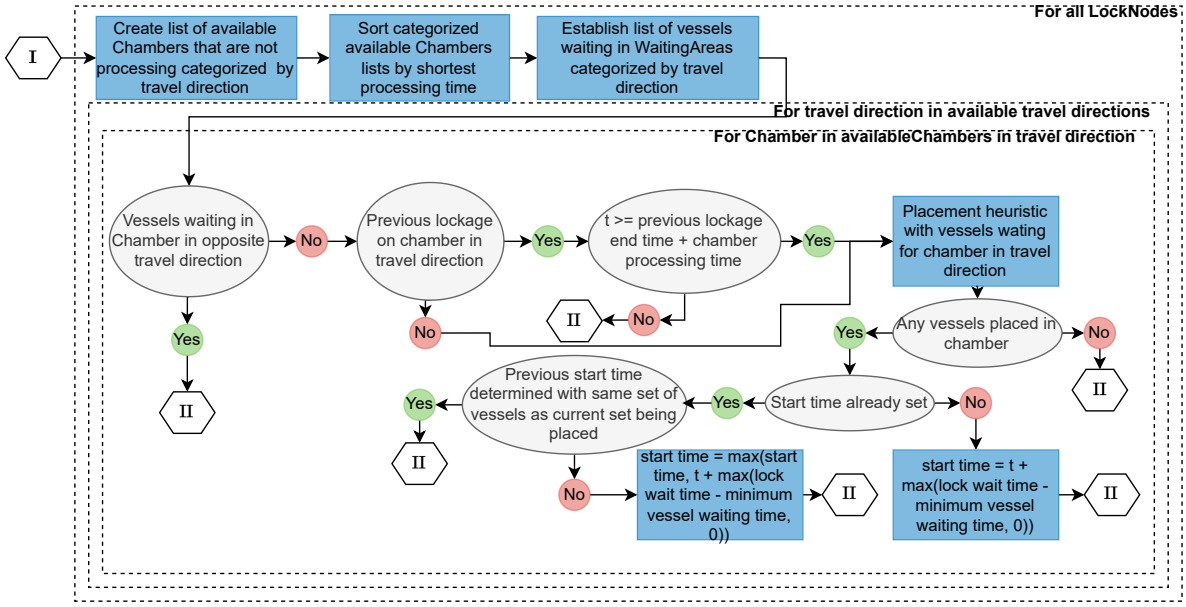


Figure 6.37: *Updated simulation block Id for mode 2 of the simulation model.*

6.9.4 Comparisons

All of the scenarios considered in the verification cases will be run on the simulation model. The simulation model does not have support for vessels being able to take multiple routes, so verification case B is not considered. It also does not support special rules for flammable and toxic vessels, so verification case F is run without giving vessels a flammable or explosive designation. To make a fair comparison, verification case F is also rerun on the optimization models without accounting for

flammable or explosive vessels. Finally, vessels cannot have an arrival time offset objective, so case A test 6 is skipped and vessel 7 gets the cumulative arrival time objective in case F test 2.

The simulations are run with a time interval of $dt = 10^{-4}\ h$. Such a small timestep is used to reduce the effect of time discretisation errors in the simulation process. The simulation is not computationally intensive, so using a small timestep is also not issue for computation times.

As a refresher, case A is run on the SVSC model, cases C and D are run on the MVMCC model, and cases E and F are run on the MVMCP model.

The KPI results are listed in Table 6.44. The first observation that can be made is that the results barely differ between the simulation model and the SVSC and MVMCC models for test cases A-D. These test cases are simple, they have a single lock, and they are all set up in such a way that the vessels all arrive at that lock at around the same time while moving in the same direction. In those scenarios, the benefit of having full information in the scheduling models is lost, as all vessels are going to arrive at the waiting areas at the same time. The heuristic model uses the waiting area queue to make its decisions, so it will have roughly the same amount of information as the global scheduler if there is only one lock and all vessels arrive at the waiting area at the same time.

Table 6.44: *Objective function and KPI values of the verification cases when tested on the scheduling models and the heuristic simulation. $\mathcal{A}$ is not listed, because $\mathcal{A} = J$ for all of the tested cases.*

	Scheduled			**Simulated**			**Δ (Simulated - Optimal)**		
Test	J	$\mathcal{L}$	$\mathcal{T}$ [%]	J	$\mathcal{L}$	$\mathcal{T}$ [%]	J	$\mathcal{L}$	$\mathcal{T}$ [%]
Case A Test 1, 2	6.51	3	15.21	6.51	3	15.22	0	0	0.01
Case A Test 3, 4	6.01	2	6.65	6.01	2	6.67	0	0	0.02
Case A Test 7	18.7	7	21.6	18.7	7	21.63	0	0	0.03
Case C Test 1	5.8	1	7.41	6.20	3	14.83	0.4	2	7.42
Case C Test 2	5.9	3	9.26	5.90	3	9.28	0	0	0.02
Case D Test 1	9.3	4	11.73	9.30	4	11.74	0	0	0.01
Case D Test 2	22.25	4	13.24	22.25	4	13.26	0	0	0.02
Case E Test 1	8.85	1	9.26	8.85	1	9.27	0	0	0.01
Case E Test 2	19.15	1	1.32	20.08	2	6.26	0.93	1	4.94
Case F Test 1	68.00	9	20.37	73.61	13	30.67	5.61	4	10.30
Case F Test 2	77.50	16	15.64	79.91	20	16.37	2.41	4	0.73

An exception to this can be seen in case C test 1, where two vessels arrive at the lock with a gap of $0.2\ h$, which is larger than the waiting area traversal time of the lock. As a result, the heuristic model will decide to start the lockage because it is not anticipating another vessel arriving very soon, while the optimization model does know a vessel will arrive, so it will delay the upcoming lockage so both vessels can go through together.

Furthermore, the SVSC model by nature is not expected to differ much from the heuristic simulation. There are few if no mechanisms at all where the SVSC scheduling model can make a smarter decision than just placing the first vessel that arrives in the chamber first if the chamber can only

hold a single vessel. The author cannot think of a single one when taking into account how the simulation model was built. Where the SVSC model can make a difference is assigning detour routes, like in Section 6.3.

The small increases in delay percentage from the scheduling results to the simulated results are caused by time discretization errors. In practice, those delay percentage differences would be 0.

The verification cases of the MMVCP model do show lower objective function, lockage number, and average delay values for the optimization model. In case E, all vessels arrive in the waiting area at the same time. Test 1 only has three vessels that can easily be placed using the placement heuristic. Test 2 has seven vessels that almost completely fill the lock chamber. This can be handled by the scheduling model, but for the simulation model with a simple placement sequence, this results in one vessel being placed in a separate lockage, as shown in Figure 6.38 and Figure 6.39. Vessel 3 could, in theory, be moored to vessel 0, but vessel 6 is moored to the left quay before vessel 3 is considered, so the heuristic placement algorithm will consider bin 1 to already be full when it is vessel 3's turn to be placed. This could be prevented in the future by using a smarter placement heuristic.

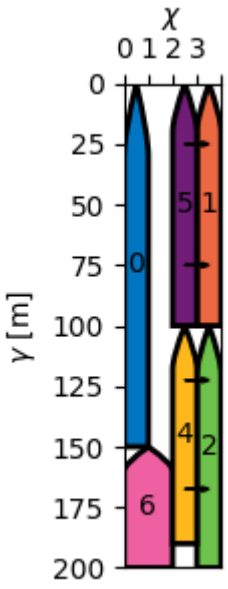


Figure 6.38: *Lockage 1 of Case E Test 2 for the simulated model.*

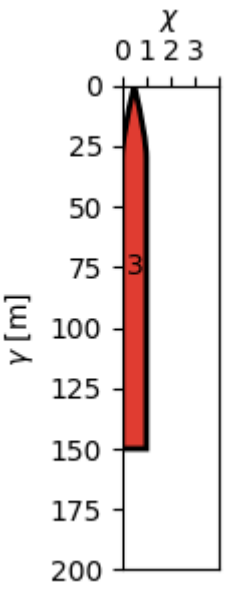


Figure 6.39: *Lockage 2 of Case E Test 2 for the simulated model.*

Case F uses a graph with multiple locks. This is where the scheduling models having full information should lead to better results because vessels can be given priority at one lock so they can be grouped on a subsequent lockage to save time. That is visible when comparing the journey plots of verification case F test 1 in Figure 6.40 and Figure 6.41. The scheduler can arrange the lockages at lock A in such a way that all vessels can pass through lock B in two lockages, saving time compared to the simulated model where there are 3 lockages required on lock B because the simulated model does not have the information to plan ahead.

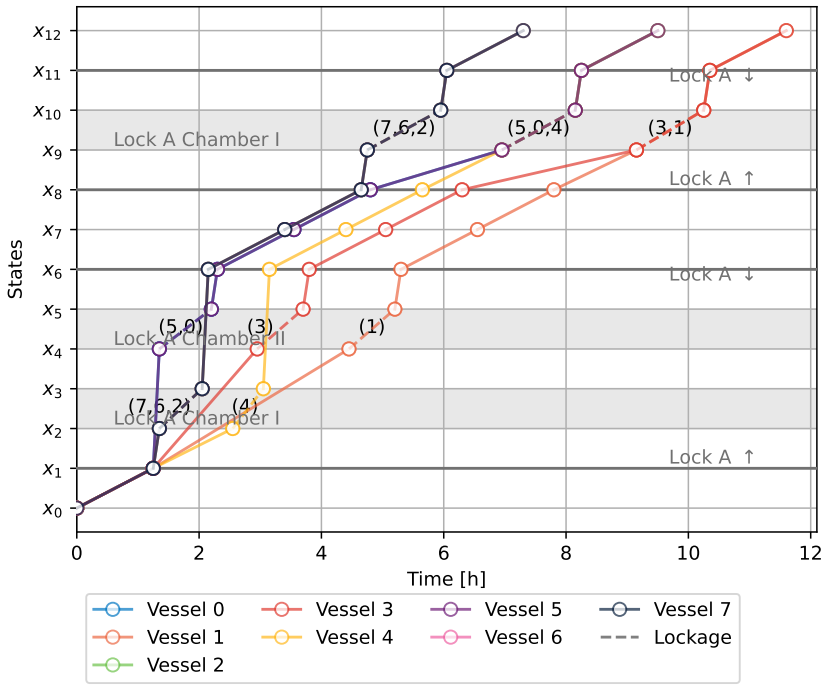


Figure 6.40: *Simulated Case F Test 1 journey plot without flammable or explosive vessels.*

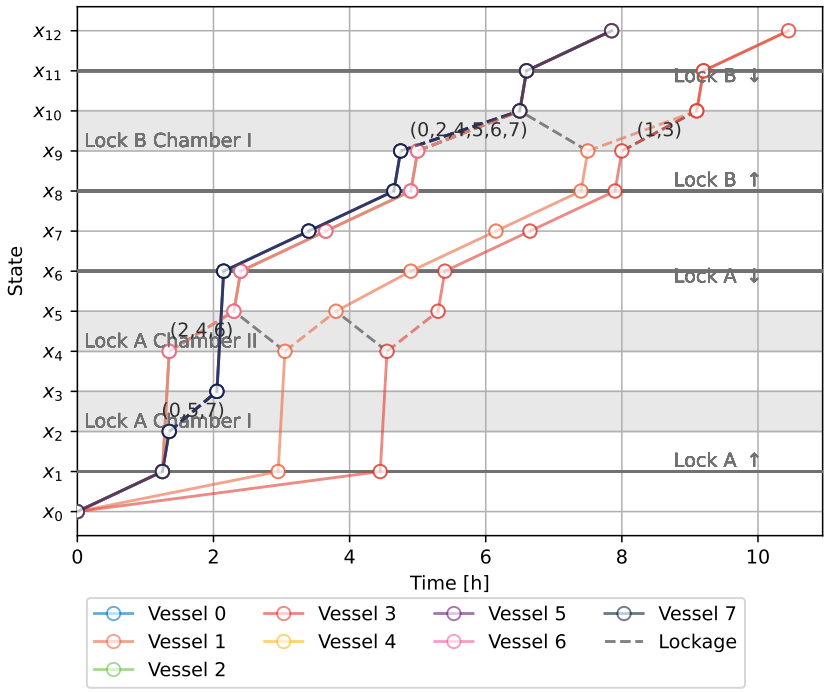


Figure 6.41: *Scheduled Case F Test 1 journey without flammable or explosive vessels.*

In conclusion, the verification cases tested show that there is only a small difference between the heuristic simulation model and the SVSC and MMVCC model in those specific scenarios. The SVSC model in particular does not have many mechanisms to outperform simple FCFS rules. However, the scenarios tested for the SVSC model are simple single-lock situations that are built to verify the model, not to test how it would work in reality.

The multi-lock verification case F does show that the MMVCP model can make significant improvements over the heuristic placement model in that case, because the scheduler makes decisions based on full information of the system. This suggests that the MMVCP model, and possibly the MMVCC model, will outperform current practice for more complex graph topologies with multiple locks, though this should be confirmed by extensive testing and validation with realistic settings in the future. The heuristic model should also be improved to better simulate practice, as current lock decision-making goes further than just looking at which vessels are waiting in the waiting areas.

6.10 Conclusion

The research question to be answered in this chapter was:

How can the performance of the SMPL IWT scheduling models be verified?

The mathematical models and schedulers introduced in earlier chapters have to be verified to ensure that they are modeling what they are intended to model and that they are correctly implemented.

The models were verified with two verification cases per model, with each verification case having several associated tests. Since large parts of the models' mathematics and implementations are shared, tests of those items in one model can be used to verify their functioning in another model.

The single-vessel, single-chamber (SVSC) model was tested in verification cases A and B, verifying the functioning of route creation, vessel ordering, both main objectives, the waiting time objective, and route choice.

The multi-vessel, multi-chamber capacity (MVMCC) model was tested in verification cases C and D, verifying the functioning of synchronisation, vessel distribution between chambers, chamber capacity limits, and the updated lockage counting KPI.

The multi-vessel, multi-chamber placement (MVMCP) model was tested in verification cases E and F, verifying the functioning of vessel placement, mooring, rules for vessels with flammable or explosive goods, multiple travel directions, intermediate destinations, and multi-lock cases. Previously investigated model components, like routing and ordering, also showed to still be working for the multi-vessel, multi-chamber tests.

Altogether, these verification cases and tests, which were designed to push the edge cases of the models, always showed the models to be working correctly, giving strong indications that they model what they are intended to model and that they are correctly implemented. These tests were not based on real data, and they did not validate the model. Validation with realistic scenarios should be performed in the future.

A complexity analysis was performed to compare the complexity of the three models in approximately comparable single-lock scheduling cases. The MVMCP model was found to have far more constraints and variables than the other models in those cases. Furthermore, the calculation times were also 5-7 times higher than those of the SVSC and MVMCC models on average in the single-lock cases.

The main conclusion that can be made on complexity is that no matter the model or graph topology, cases with over 15 vessels being considered, which will be common if the models are used on a regional basis, are likely to result in unacceptable computation times based on the sheer number of binary variables alone. Heuristics, like large neighborhood search, or distributed optimization methods, like those employed in SMPL railway scheduling, that speed up computation times should

be investigated in the future, and the models introduced in this report can be used as benchmarks to gauge their performance.

Finally, a heuristic simulation model was developed to test the scheduling models' performance on the verification tests against a simplified implementation of how they would be assigned to and placed in the lockages in practice. Little difference was found between the performance of the simulation model and the scheduling models for single-lock verification cases.

Multi-lock cases run on the MVMCP model showed where the scheduling models might perform better than practice. The scheduling model can group vessels in lockages based on how they would best pass through the sequence of locks. The simulation model, on the other hand, only assigns vessels to lockages based on what would be best for the lock being considered. The same difference is likely to be seen for the MVMCC model. The SVSC model would be unlikely to be better than the simulation model in practice, as it does not have many mechanisms that allow it to outperform a heuristic that assigns vessels to lockages on a first come first served basis.

More extensive testing of the scheduling models against practice in real data-backed scenarios is required to make concrete assertions about how much better the scheduling models perform. Furthermore, the simulation model should be improved to be more intelligent, for instance also considering vessels that are approaching waiting areas for placement.

7

Conclusions and Recommendations

This chapter contains the answer to the main research question in Section 7.1 and the recommendations for future research in Section 7.2.

7.1 Conclusion

The main research question to be answered in this thesis was:

> *How can multi-vessel, multi-chamber locks with ship placement be integrated into the SMPL IWT scheduling model?*

This thesis introduced and tested three SMPL IWT offline scheduling models, building on a single-vessel, single-chamber SMPL IWT scheduling model introduced in [31], which schedules the times at which vessels arrive at locations and lock components in a system of waterways, locations, and locks. Each new model used the previous model as a starting point, building towards a multi-vessel, multi-chamber model with ship placement according to the rules and constraints of the ship placement problem.

For the first model, the single-vessel, single-chamber model, a generalized graph input language was developed to allow it to work for arbitrary inland waterway networks with single-chamber locks. Furthermore, an arc-based route construction mechanism for vessel route choice based on the shortest path problem was introduced in SMPL algebra.

For the second model, the multi-vessel, multi-chamber capacity model, synchronisation of vessels on lock chambers was added, the graph input language was expanded to include multi-chamber locks, and vessel groupings were fit into the chambers based on one-dimensional vessel sizes and chamber capacities to moderate the jump in complexity from the single-vessel, single-chamber model.

For the third model, the multi-vessel, multi-chamber placement model, two-dimensional placement of the vessels in the chamber was added through the addition of a Tetris-like vertical placement sequence that obeys mooring constraints, safety distances between vessels were accounted for by including them into the placement size of the vessels, and special rules for vessels with flammable or explosive goods were added.

This third SMPL IWT model now included support for chambers that could process multiple vessels at a time, multi-chamber locks, arbitrary waterway and lock network configurations, and placement according to the ship placement problem, fulfilling the objectives set in Chapter 1.

Each of the models first had its constraints fully written in max-plus algebra, with the exception of the multi-vessel, multi-chamber placement model, which required some constraints that did not fit into the SMPL framework, so they were added to the MILP directly. The constraints were gathered into an SMPL model formulated as a set of state evolution equations for the vessels' arrival states and a set of allowable modes for the allowable combinations of binary decision variables. The multi-

vessel, multi-chamber placement model had two sets of state evolution equations, one for the arrival time states and one for the placement sequence bin height states.

The max-plus algebra constraints were converted to MILP constraints, and main objectives minimizing arrival times or arrival time offsets were added, so the scheduling problems could be solved as MILP optimization problems which were automatically built by a scheduler according to manually supplied graph and vessel input. Auxiliary objectives that enforce vessel departure times and that encourage slow steaming were also added.

The models were verified with several verification cases and tests designed to test specific parts and edge cases of the model. Each of the models was verified to be working correctly in the tests, with the results of all tests together indicating that the models were working and implemented correctly.

A preliminary complexity analysis focused on the scaling of the complexity of the models with the number of vessels in the system was performed, and complexity was found to be a limiting factor for all models, with unacceptable computation times arising at scenarios with 10-15 vessels for the given test conditions.

Finally, a time-based simulation model that simulates the passing of vessels through inland waterway networks in practice was built based on simplified guidelines and practices Rijkswaterstaat currently uses. The simulation model considers how the vessels and locks would operate every time step with a limited capacity for predicting future events. Comparison studies were done on the verification cases between the step-by-step simulation model that models current practice and the three offline scheduling optimization models developed in this thesis.

For the verification cases, it was clear that there was a limited benefit to the single-vessel, single-chamber and multi-vessel, multi-chamber capacity models, likely because they were only examined in single-lock cases. The multi-lock cases of the multi-vessel, multi-chamber placement model did indicate that the multi-vessel, multi-chamber models might outperform the simulation model when multiple locks are considered at the same time. More extensive testing of these indications is required, and the intelligent decision-making of the simulation model should be improved.

7.2 Recommendations

Three main recommendations can be made for future research. These are on the topics of online optimization, distributed optimization, and validation, in Subsection 7.2.1, Subsection 7.2.2, and Subsection 7.2.3, respectively. Other miscellaneous recommendations made throughout this project document are listed in Subsection 7.2.4.

7.2.1 Online Optimization

As more elaborately discussed in Section 3.6, the models introduced in this report use offline scheduling, only producing a schedule once under the assumption that all future information is known before optimization starts. Any type of disturbance, like a vessel or lock having mechanical problems, a vessel joining the system in the middle of the schedule, or vessels simply not obeying instructions can disrupt the system and lead to unwanted delays.

To create a scheduling system that can be useful in practice, online optimization will be required, continually recomputing the schedule with new information that becomes available over time.

Important considerations in implementing online optimization for SMPL IWT scheduling will be determining the acceptable minimum and maximum time intervals between schedules, getting the computation time to be consistently low enough to account for most disturbances, avoiding scenarios where the new schedule incorrectly deviates much from the old schedule, and support for

vessels not at a node when a schedule starts. The models introduced in this report can be used as a starting point for online optimization.

7.2.2 Distributed Optimization

As discussed in Subsection 6.8.3, situations with more than 10-15 vessels lead to unacceptable high computation times when compared to the 15-20 minute maximum set in Section 3.6. A national or regional scheduling system may have to schedule hundreds of vessels simultaneously, which will be impractical for all three of the scheduling models introduced here.

A more novel approach to reducing calculation times would be to use distributed optimization, which, to the best of the author's knowledge, has not been done before for lock scheduling. A grouping approach similar to that used for SMPL scheduling of trains could be taken [64], with the scheduling models introduced in this report being used as a starting point.

7.2.3 Validation

The verification tests showed that the models were working as intended in fictional test setups and that they were implemented correctly. The next question that must be answered is whether the models lead to measurable improvements in practice. This can be done through validation testing based on real data.

A good starting dataset for the SMPL IWT lock scheduling models could be the Belgian Albertkanaal dataset, or dataset similar to it, used in [41]. The verification tests indicated that the multi-vessel, multi-chamber models might result in the biggest improvements over practice on cascaded locks, and the Albertkanaal consists of six cascaded locks with three chambers each [54].

It is also important to perform more detailed complexity comparisons based on real input data for the three models. For instance, if it turns out that the MVMCC model is consistently an order of magnitude faster than the MVMCP model, then an additional study can be done on what one-dimensional vessel size and chamber capacity assignments would generally result in the fewest possible infeasible lockage configurations. The initial schedules can then be solved for on the MVMCP model, and the true two-dimensional placement of the vessels in the chambers can be done in post-processing.

7.2.4 Miscellaneous Recommendations

The following miscellaneous smaller recommendations have been made, or can be made based on content presented throughout this report:

- **Overtaking and vessel classes**: Vessels are now allowed to traverse any arc in the system, and they may overtake each other on any arc. Access to certain waterways could be restricted based on vessel class in the future, and there may be arcs where overtaking is not possible, so special rules would have to be implemented.
- **Lockage counting objective**: The currently presented schedulers only account for vessel-based objectives, while it certain circumstances it may also be important that locks limit the number of lock movements they make. The addition of an objective that penalizes lock movements should be investigated in the future.
- **More rules based on vessel type**: Simplified rules were implemented for vessels with flaming, toxic, or explosive goods. Future models should try to include all rules based on vessel type, like vessels with flammable goods not being allowed in the same lockages as passenger vessels.

- **Improve the heuristic simulation model**: If there is not enough (useable) validation data available, then another option could be to validate the scheduling models against the state of practice with the heuristic simulation model in a wide variety of realistic scenarios. The model's intelligence would have to be upgraded for this to be possible, for instance by having it also account for vessels that are approaching the waiting area when composing a lockage, like what is currently done in practice.

List of Symbols

a	Arbitrary scalar
A	Arbitrary matrix or system matrix
$A_{(k,k-\mu)}$	Inequality constraint matrix between vessels k and $k-\mu$
$A^{(\ell(k))}$	System matrix for mode $\ell(k)$
$A^{(\ell)}_{routing}$	Routing constraint matrix for system mode ℓ
$A_{routing}(v)$	Routing constraint matrix as a function of routing control variable vector v
$A_{routing,(k)}$	Routing constraint matrix for vessel k
$A_{routing,\mu}(k)$	Routing constraint matrix linking event step $k-\mu$ to event step k
$A^{(\ell)}_{routing,\mu}(k)$	Routing constraint matrix linking event step $k-\mu$ to event step k in system mode ℓ
$A_{ordering}$	Ordering constraint matrix
$A_{ordering,\mu}$	Ordering constraint matrix linking event step $k-\mu$ to event step k
$A_{ordering,(k,k-\mu)}$	Ordering constraint matrix between vessels k and $k-\mu$
$A_{scheduling}$	Scheduling constraint matrix
$A_{scheduling,\mu}$	Scheduling constraint matrix linking event step $k-\mu$ to event step k
A^{ℓ}_{synch}	Synchronisation constraint matrix for system model ℓ
A_{synch}	Synchronisation constraint matrix
$A^{\ell}_{synch,\mu}$	Synchronisation constraint matrix linking event step $k-\mu$ to event step k in system mode
$A_{synch,\mu}$	Synchronisation constraint matrix linking event step $k-\mu$ to event step k
$A_{synch,(k,k-\mu)}$	Synchronisation constraint matrix between vessels k and $k-\mu$
A_x	Continuous variable constraint matrix
A_ϕ	Integer variable constraint matrix
$\mathcal{A}$	Cumulative arrival time key performance indicator
b	Arbitrary scalar
$b(k)$	Departure node of vessel k
B	Arbitrary matrix
$B^{(\ell(k))}$	System input matrix for mode $\ell(k)$
$B_{(k)}$	Reference input matrix for vessel k
$B_{routing,(k)}$	Routing reference input matrix for vessel k
$\mathbb{B}$	Binary number space
$\mathbb{B}_{\max}$	Max-plus binary number space
c_x	Objective function vector of continuous decision variables
c_ϕ	Objective function vector of integer decision variables
$c_{(i,j)}$	Capacity of chamber (i,j)
$\boldsymbol{c_{capacities}}$	Vector of capacities of all chambers
C	Arbitrary matrix
$C_{(k)}$	Smaller than inequality constraint vector for vessel k
$d(k)$	Destination node of vessel k
$d_{(i,j)}$	Distance of arc (i,j)
$\mathcal{D}$	Set of arcs
$\mathcal{D}_{locks}$	Set of lock or chamber arcs
e	Max-plus unit quantity

$\boldsymbol{e}$	Max-plus unit vector
$E(n,m)$	n by m max-plus unit matrix
$\mathcal{E}(n,m)$	n by m max-plus zero matrix
$\boldsymbol{f}$	Scalar inequality constraint vector
$\mathcal{G}$	Directed waterway network graph
$h_\chi(g)$	Vessel stack height in bin χ at event step g
$h_{\chi,(i,j)}(k,g)$	Vessel stack height in bin χ of chamber (i,j) at event step g for vessel k
$\boldsymbol{h}(k,g)$	Vector of $\boldsymbol{h}_{(i,j)}$ for all chambers for vessel k at step g
$\boldsymbol{h}_{(i,j)}(k,g)$	Vector of vessel stack heights of vessel k on chamber (i,j) at step g
$\hat{h}_{(i,j)}(k)$	Bow position of vessel k in chamber (i,j)
$\check{h}_{(i,j)}(k)$	Stern position of vessel k in chamber (i,j)
H	Separation time matrix
$\mathcal{H}$	Arrival time offset key performance indicator
i	Integer number or index
j	Integer number or index
J	Overall mixed-integer linear programming problem objective function
$J_\mathcal{A}$	Cumulative travel time objective criterion
J_u	Departure time objective criterion
J_Π	Waiting time objective criterion
$J_\mathcal{H}$	Arrival time offset objective criterion
l	Integer number or index
$\ell(k)$	System switching mode at event step k
$\mathcal{K}$	Set of vessels
$\mathcal{K}_\triangledown$	Set of vessels with flammable goods
$\mathcal{K}_{\triangledown\triangledown}$	Set of vessels with toxic or explosive goods
$\mathcal{L}$	Number of lockages key performance indicator
$\mathcal{L}^i$	Subset i of vector space of switching variable vector $\boldsymbol{\phi}$
m	Integer number or index
$m(k,k-\mu)$	Vector of $m_{(i,j)}(k,k-\mu)$ on all chambers for the combination of vessel k and vessel $k-\mu$
$m_{(i,j)}(k,k-\mu)$	Variable that determines if vessel k is moored to vessel $k-\mu$ on chamber (i,j)
M	Large positive scalar
$M_{\max}$	Maximum possible value of some decision variable
$M_{\min}$	Minimum possible value of some decision variable
$\mathcal{M}(k)$	Set of values μ can take on for vessel k such that $k-\mu$ is in the set of vessels excluding vessel k
n	Integer number or index
n_I	Number of infrastructure items
n_k	Number of vessels
n_ϕ	Number of entries in switching variable vector $\boldsymbol{\phi}$
n_{routes}	Number of available routes
n_{steps}	Number of steps in placement sequence
n_{synch}	Number of synchronisation modes
$n_{vessels}$	Number of vessels
$N_{\chi,(i,j)}$	Number of horizontal bins of chamber (i,j)
$N_{\gamma,(i,j)}$	Vertical length of chamber (i,j)
$\mathcal{N}$	Set of nodes
$\mathcal{N}_{locks}$	Set of lock nodes
o_i	Processing time at workstation or lock i
$p_{x,i}$	x location of node i

$p_{y,i}$	y location of node i
$P^{(\ell)}$	Resource assignment matrix of system mode ℓ
$\boldsymbol{q}_{(k)}$	Equality constraint vector of vessel k
$\mathcal{Q}(k)$	Set of allowable switching modes of vessel k
$\boldsymbol{r}(k)$	System input vector at event step k
$\mathbb{R}$	Space of real numbers
$\mathbb{R}_{\max}$	Max-plus space of real numbers
$\mathcal{R}_{\max,\text{plus}}$	Max-plus semiring
$\boldsymbol{s}$	vector of synchronisation variables
$\boldsymbol{s}(k)$	vector of synchronisation variables at event k
$s^{(\ell)}$	Max-plus binary synchronisation variable for synchronisation mode ℓ
$s_{\mu}^{(\ell)}$	Max-plus binary synchronisation variable linking time-step
$s_{(i,j)}(k, k-\mu)$	Synchronisation decision variable between vessels k and $k-\mu$
$\mathcal{S}(i)$	Set of nodes connected to node i via any arc
t	maximization or absolute value objective auxiliary variable
$\mathcal{T}$	Average delay for all vessels
$\mathcal{T}(k)$	Delay percentage for vessel k
u_i	Miller Tucker Zemlin counter variable for node i
$u(k)$	Minimum departure time of vessel k
v	Max-plus binary routing control variable vector
$v^{(\ell(k))}$	Max-plus binary routing control variable deciding if the system is in routing mode $\ell(k)$
$v(k)$	Switching control variable vector at event step k
$v_{\max,(i,j)}(k)$	Maximum velocity of vessel k on arc (i,j)
$\boldsymbol{w}(k)$	Vector of all routing decision variables for vessel k
$w_{(i,j)}(k)$	Routing decision variable for if vessel k travels on arc (i,j)
$W_{\gamma}(k)$	Length of vessel k
$W_{\chi}(k)$	Width of vessel k
$\boldsymbol{x}$	State vector
$\boldsymbol{x}(k)$	State vector at event step k
$x_i(k)$	Arrival time of vessel k at infrastructure item or node i
$\hat{x}_{d(k)}$	Planned arrival time at the destination node of vessel k
$\boldsymbol{y_{(k)}}$	Equality constraint parameter vector for vessel k
$\boldsymbol{z}(k, k-\mu)$	Vector of ordering decision variables deciding the processing order between vessel k and $k-\mu$
z_{ij}	order decision variable between operations i and j
$\boldsymbol{z}_{\mu}(k)$	Order decision variable vector linking time-step k and $k-\mu$
$z_{(i,j)}(k, k-\mu)$	Ordering decision variable deciding the processing order between vessel k and $k-\mu$ on lock (i,j)
Z	Order decision matrix
$\mathbb{Z}$	Integer number space
$\alpha_{\chi,(i,j)}(k, g)$	Decision variable that determines if vessel k is placed into bin χ of chamber (i,j) on step g
$\boldsymbol{\alpha}_{(i,j)}(k, g)$	Vector of $\alpha_{\chi,(i,j)}(k, g)$ on all bins of chamber (i,j) for vessel k at timestep g
$\boldsymbol{\alpha}_{(i,j)}(k)$	Vector $\alpha_{(i,j)}(k, g)$ for all timesteps g on chamber (i,j) for vessel k
β	Large negative scalar
δ	Arbitrary binary variable
$\delta(i)$	Waiting area node attached to lock or chamber node i
$\Delta T(i)$	Computation time of schedule i
ε	Max-plus algebraic zero

ϑ	Arbitrary max-plus binary variable
λ	Auxiliary binary variable for conversion of maximization equality constraints
θ	Binary and continuous variable product auxiliary variable
$\boldsymbol{\phi}(k)$	Switching variable vector at event step k
Φ	Switching mechanism mapping function
ω	Objective criterion weight
μ	Integer time-step or vessel counter offset
$\mu_{\max}$	Maximum value of μ
$\mu_{\min}$	Minimum value of μ
$\rho(k)$	One-dimensional size of vessel k
σ	Vessel objective weight
$\pi_{(i,j)}(k)$	Waiting time of vessel k on waiting area arc (i,j)
ψ	Auxiliary binary variable for products between sums of binary variables
ξ	Arbitrary max-plus-binary variable
τ_i	Travel time between workstations
$\tau_{i,k}$	Travel time on the waterway leading up to infrastructure item i for vessel k
$\tau_{(i,j)}(k)$	Minimum travel time, processing time, or traversal time of vessel k on arc (i,j)
$\overline{\tau}_{(i,j)}$	Extra vessel processing time of chamber (i,j)
$[\cdot]_{ij}$	Entry at row i column j of matrix $cdot$
(i,j)	Waterway arc from node i to node j

List of Abbreviations

2D	Two-Dimensional
3D	Three-Dimensioonal
2L-CVRP	Two-Dimensional Loading Capacitated Vehicle Routing Problem
AIS	Automatic Identification System
BICS	Binnenvaart Informatie en Communicatie Systeem
DES	Discrete Event Systems
EU	European Union
FCFS	First Come First Served
IWT	Inland Waterway Transport
JSS	Job Shop Scheduling
LP	Linear Programming
LSP	Lock Scheduling Problem
MP	Max-Plus
MPC	Model Predictive Control
MVMCC	Multi-Vessel Multi-Chamber Capacity
MVMCP	Multi-vessel Multi-Chamber Placement
MPL	Max-Plus-Linear
MTZ	Miller Tucker Zemlin
PT	Plus-Times
RIS	River Information Systems
RoRo	Roll-on Roll-off
SMPL	Switching Max-Plus-Linear
SPP	Ship Placement Problem
SVSC	Single-Vessel Single Chamber
TEN-T	Trans-European Transport Network
TGD	Three Gorges Dam
SLSP	Serial Lock Scheduling Problem
MILP	Mixed-Integer Linear Programming
KPI	Key Performance Indicators

Bibliography

[1] T. Fisk, *Zonsondergang-water-brug-avond-9587016*, https://www.pexels.com/nl-nl/foto/zonsondergang-water-brug-avond-9587016/, Accessed on 20-06-2023.

[2] Rijkswaterstaat Dienst Verkeer en Scheepvaart, "Scheepvaartinformatie hoofdvaarwegen," 2009.

[3] Port of Rotterdam, *Binnenvaart*, https://www.portofrotterdam.com/nl/logistiek/verbindingen/intermodaal-transport/binnenvaart.

[4] Eurostat, *Modal split of air, sea and inland freight transport*, https://ec.europa.eu/eurostat/databrowser/view/tran_hv_ms_frmod/default/table?lang=en.

[5] Centraal Bureau voor de Statistiek, *Goederenvervoer; vervoerwijzen, vervoerstromen van en naar nederland*, https://www.cbs.nl/nl-nl/cijfers/detail/83101NED?dl=390DC#shortTableDescription.

[6] European Commission, *Tentec interactive map viewer*, https://ec.europa.eu/transport/infrastructure/tentec/tentec-portal/map/maps.html, Accessed on: 15-01-2023.

[7] Rijkswaterstaat Centre for Water, Transport and Environment, "Waterway guidelines 2020," 2020.

[8] Eurostat, *Inland waterways freight transport - quarterly and annual data*, https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Inland_waterways_freight_transport_-_quarterly_and_annual_data, Accessed on: 13-12-2022, 2022.

[9] European Commission, "European transport policy for 2010: Time to decide," European Commission, Brussels, COM(2001) 370, 2001.

[10] E. Soveges, M. Stefanov, P. Puricella, *et al.*, "Inland waterway transport in Europe: No significant improvements in modal share and navigability conditions since 2001," European Court of Auditors, Luxembourg, Luxembourg, 01 /2015, 2015. DOI: doi:10.2865/158305.

[11] European Parliament, *Towards future-proof inland waterway transport in europe*, https://www.europarl.europa.eu/doceo/document/TA-9-2021-0367_EN.html, 2021.

[12] J. P. Hooft, "On the critical speed range of ships in restricted waterways," Netherlands Ship Model Basin, Wageningen, The Netherlands, Tech. Rep. 324, 1969.

[13] D. Spitzer, "Resistance and economic speed of ships and tows in inland waterways," *Journal of Waterway, Port, Coastal, and Ocean Engineering*, vol. 147, 6 2021. DOI: 10.1061/(ASCE)WW.1943-5460.0000672open_in_newPublisher.

[14] European Environment Agency, *Rail and waterborne best for low-carbon motorised transport*, https://www.eea.europa.eu/publications/rail-and-waterborne-transport, 2021. DOI: 10.2800/85117.

[15] E. Backer van Ommeren, *Globale schets gasolieverbruik binnenvaartschepen*, https://www.evofenedex.nl/api/v1/sharepoint/file/Shared%20Documents/Download%20Vervoer/Globale_schets_gasolieverbruik_binnenvaartschepen_06.pdf.

[16] J. Faber, D. Nelissen, G. Hon, H. Wang, and M. Tsimplis, "Regulated slow steaming in maritime transport: An assessment of options, costs and benefits," CE Delft, Delft, 2012.

[17] P. Segovia, M. Pesselse, T. van den Boom, and V. Reppa, "Scheduling inland waterway transport vessels and locks using a switching max-plus-linear systems approach," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 3, pp. 748–762, 2022. DOI: 10.1109/OJITS.2022.3218334.

[18] Dutch Government, *Binnenvaartpolitiereglement*, https://wetten.overheid.nl/BWBR0003628/2017-01-01, 2017.

[19] J. Verstichel, P. De Causmaecker, F. C. R. Spieksma, and G. Vanden Berghe, "Exact and heuristic methods for placing ships in locks," *European Journal of Operational Research*, vol. 235, pp. 387–398, 2 2014. DOI: https://doi.org/10.1016/j.ejor.2013.06.045.

[20] Ministerie van Infrastructuur en Waterstaat, *Volkeraksluizen*, https://www.rijkswaterstaat.nl/water/waterbeheer/bescherming-tegen-het-water/waterkeringen/deltawerken/volkeraksluizen.

[21] D. Stone, *An elevator for 3,000-ton ships, thanks to Archimedes*, https://www.nationalgeographic.com/magazine/article/explore-china-three-gorges-dam-ship-lift, 2017.

[22] F. K. Pour, P. Segovia, L. Etienne, and V. Puig, "A two-layer control architecture for operational management and hydroelectricity production maximization in inland waterways using model predictive control," *Control Engineering Practice*, vol. 124, 2022. DOI: 10.1016/j.conengprac.2022.105172.

[23] Rijkswaterstaat Ministerie van Infrastructuur en Waterstaat, *BICS*, https://www.rijkswaterstaat.nl/zakelijk/verkeersmanagement/scheepvaart/scheepvaartverkeersbegeleiding/bics, Accessed on: 13-05-2023.

[24] Rijkswaterstaat Ministerie van Infrastructuur en Waterstaat, *AIS: Verkeersinformatie scheepvaart*, https://www.rijkswaterstaat.nl/zakelijk/verkeersmanagement/scheepvaart/scheepvaartverkeersbegeleiding/river-information-services/automatic-identification-system, Accessed on: 14-05-2023.

[25] Rijkswaterstaat Ministerie van Infrastructuur en Waterstaat, *Bruggen en sluizen op afstand bediend*, https://www.rijkswaterstaat.nl/water/scheepvaart/bruggen-en-sluizen-op-afstand-bediend, Accessed on: 20-05-2023.

[26] M. Fouraschen and A. de With, "Bediening op afstand sluizen rijkswaterstaat zeeland," 2006.

[27] Rijkswaterstaat Ministerie van Infrastructuur en Waterstaat, *River information services*, https://www.rijkswaterstaat.nl/zakelijk/verkeersmanagement/scheepvaart/scheepvaartverkeersbegeleiding/river-information-services, Accessed on: 13-05-2023.

[28] T. J. J. van den Boom and B. De Schutter, "Modelling and control of discrete event systems using switching max-plus-linear systems," in *Proceedings of the 7th International Workshop on Discrete Event Systems*, Reims, France, 2004, pp. 115–120.

[29] B. Heidergott, G. J. Olsder, and J. van der Woude, *Max Plus at Work*. Oxford, United Kingdom: Princeton University Press, 2005.

[30] T. J. J. van den Boom, M. van den Muijsenberg, and B. De Schutter, "Model predictive scheduling of semi-cyclic discrete-event systems using switching max-plus linear models and dynamic graphs," *Discrete Event Dynamic Systems*, vol. 30, pp. 635–669, 2020. DOI: 10.1007/s10626-020-00318-w.

[31] M. Pesselse, "Modelling and optimal scheduling of inland waterway transport systems," M.S. thesis, Delft University of Technology, Delft, The Netherlands, 2022.

[32] J. Verstichel, "The lock scheduling problem," Ph.D. dissertation, KU Leuven - Faculty of Engineering Science, Heverlee, Belgium, Nov. 2013.

[33] NOVIMOVE, *What we do*, https://novimove.eu/concept/?cn-reloaded=1, Accessed on: 15-01-2023.

[34] P. Segovia, R. R. Negenborn, and V. Reppa, "Vessel passage scheduling through cascaded bridges using mixed-integer programming," *IFAC-PapersOnLine*, vol. 55, pp. 248–253, 2022. DOI: 10.1016/j.ifacol.2022.09.032.

[35] S. V. Raković and W. S. Levine, *Handbook of Model Predictive Control*. Cham: Birkhäuser, 2019. DOI: 10.1007/978-3-319-77489-3.

[36] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan, "Optimization and approximation in deterministic sequencing and scheduling: A survey," *Annals of Discrete Mathematics*, vol. 5, pp. 287–326, 1979. DOI: 10.1016/S0167-5060(08)70356-X.

[37] X. Wang, Y. Zhao, P. Sun, and X. Wang, "An analysis on convergence of data-driven approach to ship-lock scheduling.," *Mathematics and Computers in Simulation*, vol. 88, pp. 31–38, 2013. DOI: 10.1016/j.matcom.2013.03.005.

[38] J. Hermans, "Optimization of inland shipping - a polynomial time algorithm for the single ship single lock optimization problem," *Journal of Scheduling*, vol. 17, pp. 305–319, 4 2014. DOI: 10.1007/s10951-013-0364-7.

[39] W. Passchyn, S. Coene, D. Briskorn, J. L. Hurink, F. C. R. Spieksma, and G. Vanden Berghe, "The lockmaster's problem," *European Journal of Operational Research*, vol. 251, pp. 432–441, 2016. DOI: 10.1016/j.ejor.2015.12.007.

[40] J. Verstichel, P. De Causmaecker, and G. Vanden Berghe, "Scheduling algorithms for the lock scheduling problem," *Procedia - Social and Behavioural Sciences*, vol. 20, pp. 806–815, 2011. DOI: 10.1016/j.sbspro.2011.08.089.

[41] J. Verstichel, P. De Causmaecker, F. C. R. Spieksma, and G. Vanden Berghe, "The generalized lock scheduling problem: An exact approach," *Transportation Research Part E*, vol. 65, pp. 16–34, 2014. DOI: 10.1016/j.tre.2013.12.010.

[42] M. Prandtstetter, U. Ritzinger, P. Schmidt, and M. Ruthmair, "A variable neighborhood search approach for the interdependent lock scheduling problem," in *Evolutionary Computation in Combinatorial Optimization*. Switzerland: Springer, 2015, pp. 36–60.

[43] W. Passchyn, D. Briskorn, and F. C. R. Spieksma, "Mathematical programming models for lock scheduling with an emission objective," *European Journal of Operations Research*, vol. 246, pp. 802–814, 3 2016. DOI: 10.1016/j.ejor.2015.09.012.

[44] B. Ji, X. Yuan, and Y. Yuan, "A hybrid intelligent approach for co-scheduling of cascaded locks with multiple chambers," *IEEE Transactions on Cybernetics*, vol. 49, pp. 1236–1248, 4 2019. DOI: 10.1109/TCYB.2018.2799303.

[45] B. Ji, D. Zhang, S. S. Yu, and X. Fang, "An exact approach to the generalized serial-lock scheduling problem from a flexible job-shop scheduling perspective," *Computers and Operations Research*, vol. 127, 2021. DOI: 10.1016/j.cor.2020.105164.

[46] B. Ji, D. Zhang, S. S. Yu, and B. Zhang, "Optimally solving the generalized serial-lock scheduling problem from a graph-theory-based multi-commodity network perspective," *European Journal of Operational Research*, vol. 288, pp. 47–62, 1 2021. DOI: 10.1016/j.ejor.2020.05.035.

[47] B. Ji, D. Zhang, S. S. Yu, and C. Kang, "Mathematical programming models for scheduling multiple cascaded waterway locks," *Computers & Industrial Engineering*, vol. 156, 2021. DOI: 10.1016/j.cie.2021.107289.

[48] B. Ji, X. Yuan, Y. Yuan, X. Lei, and H. H. C. Iu, "An adaptive large neighborhood search for solving generalized lock scheduling problem: Comparative study with exact methods," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, pp. 3344–3356, 8 2020. DOI: 10.1109/TITS.2019.2926405.

[49] H. Guan, Y. Xu, L. Li, and X. Huang, "Optimizing lock operations and ship arrivals through multiple locks on inland waterways," *Mathematical Problems in Engineering*, vol. 2021, 2021. DOI: 10.1155/2021/6220559.

[50] X. Yuan, B. Ji, X. Wu, and X. Zhang, "Co-scheduling of lock and water-land transshipment for ships passing the dam," *Applied Soft Computing*, vol. 45, Pages 150–162, 2016. DOI: 10.1016/j.asoc.2016.04.019.

[51] B. Ji, H. Sun, X. Yuan, Y. Yuan, and X. Wang, "Coordinated optimized scheduling of locks and transshipment in inland waterway transportation using binary NSGA-II," *International Transactions in Operational Research*, vol. 27, pp. 1501–1525, 2020. DOI: 10.1111/itor.12720.

[52] X. Zhao, Q. Lin, and H. Yu, "A co-scheduling problem of ship lift and ship lock at the Three Gorges Dam," *IEEE Access*, vol. 8, pp. 132 893–132 910, 2020. DOI: 10.1109/ACCESS.2020.3009775.

[53] J. J. S. Hengeveld, "Optimization to reduce waiting times at locks," M.S. thesis, TU Delft, Delft, The Netherlands, 2012.

[54] visuRIS, *Albertkanaal*, https://www.visuris.be/Albertkanaal?KL=nl, Accessed on: 22-01-2023.

[55] N. Bansal, J. R. Correa, C. Kenyon, and M. Sviridenko, "Bin packing in multiple dimensions: Inapproximability results and approximation schemes," *Mathematics of Operations Research*, vol. 31, pp. 31–49, 1 2006. DOI: 10.1287/moor.1050.0168.

[56] L. B. Bartels, S. A. Dambruin, H. M. van der Nagel, and S. W. J. Terwindt, *The relation between the computer game tetris and max-plus scheduling (of industrial and logistic systems)*, Delft Mechanical Engineering BSc End Project, 2021.

[57] G. Fuellerer, K. F. Doerner, R. F. Hartl, and M. Iori, "Ant colony optimization for the two-dimensional loading vehicle routing problem," *Computers and Operations Research*, vol. 36, pp. 655–673, 3 2009. DOI: 10.1016/j.cor.2007.10.021.

[58] E. E. Zachariadis, C. D. Tarantilis, and C. T. Kiranoudis, "The vehicle routing problem with simultaneous pick-ups and deliveries and two-dimensional loading constraints," *European Journal of Operational Research*, vol. 251, pp. 369–386, 2 2016. DOI: https://doi.org/10.1016/j.ejor.2015.11.018.

[59] A. Bortfeldt, T. Hahn, D. Männel, and L. Mönch, "Hybrid algorithms for the vehicle routing problem with clustered backhauls and 3D loading constraints," *European Journal of Operational Research*, vol. 243, pp. 82–96, 1 2015. DOI: https://doi.org/10.1016/j.ejor.2014.12.001.

[60] J. R. Hansen, I. Hukkelberg, K. Fagerholt, M. Stålhane, and J. G. Rakke, "2D-packing with an application to stowage in roll-on roll-off liner shipping," in *Computational Logistics: 7th International Conference*, Lisbon, Portugal, 2016, pp. 35–49. DOI: 10.1007/978-3-319-44896-1_3.

[61] Y. Zhang, H. Tian, L. He, S. Ma, and L. Yang, "A two-phase stowage approach for passenger-carrgo roro ship based on 2D-KP: Coping with complex rotation and safe navigation constraints," *IEEE Access*, vol. 8, pp. 95 807–95 822, 2020. DOI: 10.1109/ACCESS.2020.2995639.

[62] B. De Schutter and T. van den Boom, "Max-plus algebra and max-plus linear discrete event systems: An introduction.," in *Proceedings of the 9th International Workshop on Discrete Event Systems*, Goteborg, Sweden, 2008, pp. 36–42.

[63] B. Kersbergen, "Modeling and control of switching max-plus-linear systems: Rescheduling of railway traffic and changing gaits in legged locomotion," Ph.D. dissertation, TU Delft, Delft, The Netherlands, 2015.

[64] B. Kersbergen, J. Rudan, T. van den Boom, and B. De Schutter, "Towards railway traffic management using switching max-plus-linear systems," *Discrete Event Dynamic Systems*, vol. 26, pp. 183–223, 2014. DOI: 10.1007/s10626-014-0205-7.

[65] A. Gupta, T. van den Boom, J. van der Woude, and B. De Schutter, "Structural controllability of switching max-plus linear systems," *IFAC-PapersOnLine*, vol. 53, pp. 1936–1942, 2 2020. DOI: 10.1016/j.ifacol.2020.12.2587.

[66] T. van den Boom and B. De Schutter, "Model predictive control for switching max-plus-linear systems with random and deterministic switching," *IFAC Proceedings Volumes*, vol. 41, pp. 7660–7665, 2 2008. DOI: 10.3182/20080706-5-KR-1001.01295.

[67] G. A. D. Lopes, B. Kersbergen, T. J. J. van den Boom, B. De Schutter, and R. Babuška, "Modeling and control of legged locomotion via switching max-plus models," *IEEE Transactions on Robotics*, vol. 30, pp. 652–665, 3 2014. DOI: 10.1109/TRO.2013.2296105.

[68] G. A. D. Lopes, B. Kersbergen, T. van den Boom, B. De Schutter, and R. Babuška, "On the eigenstructure of a class of max-plus linear systems," in *2011 50th IEEE Conference on Decision and Control and European Control Conference*, Orlando, Florida, 2011. DOI: 10.1109/CDC.2011.6160904.

[69] S. Zhang, "Analysis of printing process based on a stochastic max-pluslinear approach," M.S. thesis, TU Delft, Delft, The Netherlands, 2021.

[70] F. B. van Boetzelaer, T. J. J. van den Boom, and R. R. Negenborn, "Model predictive scheduling for container terminals," *IFAC Proceedings Volumes*, vol. 47, pp. 5091–5096, 3 2014. DOI: 10.3182/20140824-6-ZA-1003.00134.

[71] B. De Schutter, *Chapter 11: Integer optimization*, https://www.dcsc.tudelft.nl/~bdeschutter/osc/download/osc_slides_opt_integer.pdf, Lecture notes of the course Optimization for Systems and Control, TU Delft, 2022.

[72] B. De Schutter and T. van den Boom, "Model predictive control for max-plus-linear discrete event systems," *Automatica*, vol. 37, 7 2001. DOI: 10.1016/S0005-1098(01)00054-1.

[73] Gurobi, *Dealing with big-M constraints*, https://www.gurobi.com/documentation/9.5/refman/dealing_with_big_m_constra.html, Accessed on: 11-05-2023.

[74] P. A. Rubin, *Choosing "big M" values*, https://orinanobworld.blogspot.com/2018/09/choosing-big-m-values.html, 2018.

[75] Pablo, *Linear programming set a variable the max between two another variables*, https://math.stackexchange.com/a/3568461, 2020.

[76] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, 1999.

[77] B. De Schutter, *Chapter 8: Optimization: Summary; notes made during lecture*, https://www.dcsc.tudelft.nl/~bdeschutter/osc/download/osc_notes_2022_10_11_summary.pdf, Lecture notes of the course Optimization for Systems and Control, TU Delft, 2022.

[78] L. Taccari, "Integer programming formulations for the elementary shortest path problem," *European Journal of Operational Research*, vol. 252, pp. 122–130, 2016. DOI: 10.1016/j.ejor.2016.01.003.

[79] C. E. Miller, A. W. Tucker, and R. A. Zemlin, "Integer programming formulation of traveling salesman problems," *Journal of the ACM*, vol. 7, pp. 326–329, 4 1960. DOI: 10.1145/321043.321046.

[80] Bureau Voorlichting Binnenvaart, *Verladers*, https://www.bureauvoorlichtingbinnenvaart.nl/wp-content/uploads/2021/04/verladers.pdf, Accessed on: 14-05-2023.

[81] R. G. Sargent, "An introduction to verification and validation of simulation models," in *Proceedings of the 2013 Winter Simulation Conference*, Washington, DC, 2013, pp. 321–327. DOI: 10.1109/WSC.2013.6721430.

[82] F. S. Hillier and G. J. Lieberman, *Introduction to Operations Research*. New York: McGraw-Hill, 2021, ch. 12, ISBN: 978-1-260-57587-3.

[83] Gurobi, *Mixed integer programming - a primer on the basics*, https://www.gurobi.com/resources/mixed-integer-programming-mip-a-primer-on-the-basics/, Accessed on: 19-05-2023.

A

Research Paper

Scheduling Multi-Vessel Placement in Multi-Chamber Inland Waterway Locks Using Switching Max-Plus Algebra

R. Ummels[1,2], T. van den Boom[1], V. Reppa[2], P. Segovia[2]

[1] Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands

[2] Department of Maritime and Transport Technology, Delft University of Technology, Delft, The Netherlands

Abstract

This paper discusses the optimal scheduling of inland waterway transport (IWT) vessels through arbitrary waterway systems with multi-vessel, multi-chamber locks, also considering two-dimensional ship placement according to the ship placement problem (SPP). Three switching max-plus-linear (SMPL) scheduling models are developed, each being an incremental step toward the aforementioned goal, using a previously introduced SMPL single-vessel, single-chamber scheduling model as a starting point. These models are first formulated as SMPL systems, which lend themselves well to modeling these types of scheduling systems. The SMPL constraints of these systems are translated to mixed integer linear programming (MILP) constraints and objectives minimizing arrival times or arrival time offsets are added. These MILP constraints and objectives are the rules by which the scheduler builds offline scheduling optimization problems according to the supplied input. The three models are verified through case studies and a complexity study is performed to characterize the computational difficulty of the different models introduced.

1 Introduction

Inland waterway transport (IWT) is a low CO_2 emission alternative to road transport [1]. Other benefits of IWT may include reduced noise pollution and relieved road and railway congestion [2]. Extensive Dutch and European IWT networks should enable large-scale use of IWT. Nevertheless, IWT only makes up a small percentage of the EU modal share [3], despite repeated objectives set for growth.

Deficient infrastructure, particularly locks, has been identified as a factor holding back that growth [4]. Locks are ship-passable divisions between different water levels on a waterway. Locks consist of one or multiple chambers in which the water level can be regulated, so vessels can be brought from one water level to another. These chambers have a limited capacity for vessels and their operation takes time, potentially causing delays for vessels getting backed up.

Expensive and time-consuming physical expansion of locks is not always an option. Additionally, projects that do get executed often miss their targets [4]. Another option is to try to optimize the passage of vessels through the locks or IWT systems with locks, which is what this paper will consider.

Optimization of lock operations, also referred to as lock scheduling, is not a new area of research. [5] used a data-driven approach for lock scheduling of single-directional, single-chamber locks. A polynomial time scheduling algorithm for single-vessel, single-chamber locks was introduced in [6]. Polynomial time solvability for multi-vessel, single-chamber locks was achieved in [7].

Exact and metaheuristic models for Lock scheduling of multi-vessel, multi-chamber locks were presented in [8], using a packing heuristic to achieve proper placement of vessels in the chamber. A full mixed integer programming problem (MILP) formulation for lock scheduling of multi-vessel, multi-chamber locks, in particular including an exact formulation for correct placement of vessels in locks, known as the ship placement problem (SPP), was presented in [9].

Scheduling of multiple multi-vessel, multi-chamber locks in series with a simplified capacity implementation was considered in [10]. Multi-vessel, multi-chamber lock scheduling of locks in series with the SPP was performed with job-shop-scheduling (JSS) and multi-commodity flow (MCF) MILP formulations in [11] and [12].

Other new developments include features like the possibility for transshipment at the locks [13] [14] and scheduling of a lock and a parallel ship lift [15].

Recent work considered lock scheduling of single-vessel, single-chamber locks using switching max-plus-linear (SMPL) systems, a tool of max-plus algebra [16]. Max-plus algebra lends itself well to the scheduling of these types of event-based systems, also having been used in the scheduling of train networks [17], Tetris, and job shops [18].

A routing component was also introduced into the lock scheduling problem in [16], taking a step towards being able to schedule for large-scale inland waterway networks, but only for a given set of networks, not for any arbitrary network configuration.

The novel contributions of this paper will then be building out the SMPL IWT scheduling model by formulating it for use with arbitrary IWT networks, integrating multi-chamber locks, and integrating multi-vessel locks according to the rules of the SPP. This is done by creating three models, each subsequent model building on the previous.

The first model is an adaptation of the single-vessel, single-chamber model introduced in [16], but augmented

for use in arbitrary IWT system layouts.

The second model, the multi-vessel, multi-chamber capacity model, integrates multi-chamber, multi-vessel locks through the use of a one-dimensional size parameter assigned to each of the vessels and the chambers, akin to the approximation used in [10].

Finally, realistic ship placement inspired by the SPP rules outlined in [9] and modeled like the Tetris model in [18] is integrated into the final multi-chamber, multi-vessel placement model.

2 Max-Plus Algebra

The main source for this section is [19].

Max-plus algebra is an algebra where the $+$ operator has been replaced by the maximization operator $\oplus$, and where the $\times$ operator has been replaced by the plus operator $\otimes$.

The unit element is represented by 0 and the zero element is represented by $\varepsilon = -\infty$. The max-plus space of real numbers is defined as $\mathbb{R}_{\max} = \mathbb{R} \cup \{\varepsilon\}$. Let $a, b, c \in \mathbb{R}_{\max}$, then:

$$c = a \oplus b = \max(a, b) \quad c = a \otimes b = a + b. \tag{1}$$

Furthermore, it is defined that for any value of $a \in \mathbb{R}_{\max}$:

$$a \oplus \varepsilon = \varepsilon. \tag{2}$$

Matrix operations in max-plus algebra are equivalent to those in conventional algebra, with the operations replaced by their max-plus counterparts. Let $A, B \in \mathbb{R}_{\max}^{n\times m}, C \in \mathbb{R}_{\max}^{m\times n}$, let $[A]_{ij}$ be entry a_{ij} of matrix A, then:

$$[A \oplus B]_{ij} = \max(a_{ij}, b_{ij}) \quad \forall (i,j) \in (\underline{n}, \underline{m}) \tag{3}$$

$$[A \otimes C]_{ij} = \bigoplus_{l}^{n} a_{il} \otimes c_{lj} = \max_{j\in\underline{n}}(a_{il} + c_{lj}) \forall (i,j) \in (\underline{n}, \underline{n}), \tag{4}$$

where $\underline{n} = \{1, 2, ..., n\}$.

These operations are used in max-plus-linear (MPL) systems to model the behavior of discrete event systems (DES). A general MPL system is defined by [17]:

$$x(k) = \bigoplus_{\mu=\mu_{\min}}^{\mu_{\max}} A \otimes x(k-\mu) \oplus B \otimes r(k), \tag{5}$$

where $x(k) \in \mathbb{R}^n$ is the state vector at event k, $A \in \mathbb{R}^{n\times n}$ is a state matrix, $B \in \mathbb{R}^{n\times m}$ is an input matrix, $r(k) \in \mathbb{R}^m$ is the system input at event k, and $\mu_{\max}$ and $\mu_{\min}$ are the maximum and minimum values of μ.

If the dynamics of the systems change with event counter k, then the system becomes an SMPL system. At event k an SMPL system is in a system mode denoted by $\ell(k)$. The value of $\ell(k)$ is determined by a switching vector $\phi(k)$ and switching function Φ [17]:

$$\phi(k) = \Phi\big(x(k-\mu)\forall\mu \in \{\mu_{\min}, ..., \mu_{\max}\}, \ell(k), r(k), v(k)\big), \quad \phi \in \mathbb{R}_{\max}^{n_\phi}, \tag{6}$$

where $v(k)$ are dedicated switching variables and n_ϕ is the length of the switching vector.

$v(k)$ are max-plus binary variables, which are the max-plus equivalent of binary variables. Let v be a max plus binary variable, and let $\overline{v}$ be its adjoint, then the following definitions apply:

$$v \begin{cases} 0 & \text{True} \\ \varepsilon & \text{False} \end{cases} \qquad \overline{v} \begin{cases} \varepsilon & v = 0 \\ 0 & v = \varepsilon. \end{cases} \tag{7}$$

3 Problem Definition

Since each model builds on the next, the problem definition for the single-vessel, single-chamber model is outlined as the starting point, and changes or additional components are introduced for the multi-vessel, multi-chamber capacity and placement models.

3.1 Single-vessel, Single-Chamber Model

An IWT network is described by a topology graph with a set of nodes $N = \{0, 1, ..., n_{nodes} - 1\}$ representing locations and a set of arcs D representing waterways connecting the nodes, with an arc from node i to node j being denoted by (i, j), as depicted in Figure 1. If (i, j) is in the graph, then (j, i) is in the graph. Each node is given an x and y location with position parameters $p_{x,i}$ and $p_{x,j}$. The travel distance between nodes $d_{(i,j)}$ is taken to be the Euclidian distance.

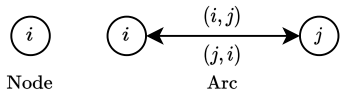


Figure 1: *Nodes and arcs.*

Locks have been reduced to a chamber and two symmetric waiting areas in a four-node construction as shown in Figure 2.

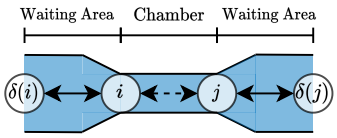


Figure 2: *Construction of nodes representing a lock.*

The minimum travel time of vessel k traveling an arc (i, j) is represented by $\tau_{(i,j)}(k)$. For waiting areas and chambers, $\tau_{(i,j)}(k)$ respectively represents the minimum traversal time of the waiting areas and the operation time of the lock for a single vessel. These times are assumed to be the same for all vessels. $\mathcal{D}_{locks} \subset \mathcal{D}$ is the set of all lock arcs in a graph.

A set of vessels $\mathcal{K} = \{0, 1, ..., n_{vessels} - 1\}$ is scheduled to travel on the graph. Each vessel has a departure node $b(k)$ and a destination node $d(k)$. Vessels also have a minimum departure time $u(k)$. $v_{\max,(i,j)}(k)$ denotes the maximum velocity of vessel k on arc (i, j). The minimum travel time on an arc $\tau_{(i,j)}(k)$ is then calculated as:

$$\tau_{(i,j)}(k) = \frac{d_{(i,j)}}{v_{\max,(i,j)}(k)}, \tag{8}$$

unless it is manually overridden. If a vessel has a scheduled arrival time at its final destination, that is denoted by

$\hat{x}_{d(k)}(k)$.

The following assumptions apply, in large part similar to those in [20]:

- All locks are single-vessel, single-chamber locks.
- Vessels can overtake other vessels on all arcs.
- Vessels maintain a constant speed on an arc.
- The start and end points of a vessel's journey are nodes on the graph.
- Any node may be a departure or destination node.
- A route connecting a vessel's departure and destination nodes always exists.
- Vessel departure times are known beforehand.
- Maximum velocities for each vessel on each arc are known beforehand.
- Vessels are assumed to always be able to achieve their maximum velocities.
- Vessels and locks cooperate fully with the scheduler's solutions.
- Lock operations can start as soon as a vessel enters a lock.
- Waiting areas are assumed to have infinite capacity.
- The different zones leading up to a lock are all modeled as a single waiting area.
- Vessels can traverse all arcs of the network.

The variables to be optimized are the arrival time states $x_i(k)$, routing decision variables $w_{(i,j)}(k)$, and ordering variables $z_{(i,j)}(k, k-\mu)$. Together these form a schedule. Arrival time state $x_i(k)$ denotes the time at which vessel k arrives at node i. Routing decision variable $w_{(i,j)}(k)$ is a max-plus binary variable defined as:

$$w_{(i,j)}(k) = \begin{cases} 0 & \text{Vessel } k \text{ travels arc } (i,j). \\ \varepsilon & \text{Vessel } k \text{ does not travel arc } (i,j). \end{cases} \tag{9}$$

This is used to build routes between $b(k)$ and $d(k)$ for each of the vessels. Ordering variable $z_{(i,j)}(k, k-\mu)$ is a max-plus binary variable defined as:

$$z_{(i,j)}(k, k-\mu) = \begin{cases} 0 & \text{Vessel } k-\mu \text{ uses lock } (i,j) \text{ before vessel } k. \\ \varepsilon & \text{Vessel } k \text{ uses lock } (i,j) \text{ before vessel } k-\mu. \end{cases} \tag{10}$$

The ordering variables are used to order the vessels passing through the locks.

3.2 Multi-Vessel, Multi-Chamber Capacity Model

For multi-chamber locks, two chamber entry and exit nodes are added per lock chamber, as shown for a multi-chamber lock in Figure 3. Chambers are numbered with Roman numerals. Chambers of the same lock may be of different sizes and have different operation times, but they share the same waiting area nodes.

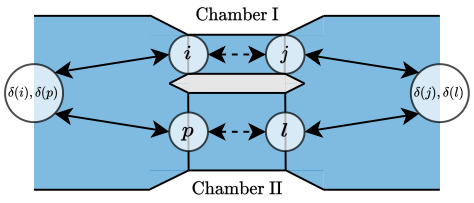


Figure 3: *Two-chamber multi-chamber lock.*

Each vessel is assigned a one-dimensional size $\rho(k)$. Each lock chamber arc $(i,j) \in \mathcal{D}_{locks}$ is assigned a capacity $c_{(i,j)}$. A group of vessels processed together by a lock is called a lockage. A multi-vessel lockage is possible if the sum of the sizes of the vessels in the lockage is smaller than the capacity of the chamber. This simplified scheme avoids the complexity that would be added by fully implementing the SPP.

To coordinate vessels combining into lockages, max-plus binary synchronisation variables $s_{(i,j)}(k, k-\mu)$ are added with the following definition:

$$s_{(i,j)}(k, k-\mu) = \begin{cases} 0 & \text{Vessel } k \text{ is synchronised with vessel } k-\mu \\ & \text{on chamber } (i,j). \\ \varepsilon & \text{Vessel } k \text{ is not synchronised with} \\ & \text{vessel } k-\mu \text{ on chamber } (i,j). \end{cases} \tag{11}$$

Synchronisation between vessels k and $k-\mu$ on chamber (i,j) means that vessels k and $k-\mu$ are processed in the same lockage on chamber (i,j).

Mooring and positioning of each vessel add to the time of the lockage. Each chamber $(i,j) \in \mathcal{D}_{locks}$ is assigned an extra vessel processing time $\overline{\tau}_{(i,j)}$, representing the time an extra vessel being added to a lockage adds to the lockage processing time.

The single-vessel, single-chamber locks assumption is removed. The additional assumptions made are:

- A lock can have multiple multi-vessel chambers with one-dimensional maximum capacities.
- The shape of each vessel is approximated by a bounding rectangle.
- The size of a vessel's bounding rectangle can be described by a one-dimensional size parameter.
- The shape of each chamber can be approximated by a bounding rectangle.
- The size of a chamber's bounding rectangle can be described by a one-dimensional capacity parameter.
- Extra vessel processing time is independent of the number of vessels in the lock.
- Extra vessel processing time is independent of the extra vessel being processed.

3.3 Multi-Vessel, Multi-Chamber Placement model

For the multi-vessel, multi-chamber placement model, the one-dimensional capacity is replaced by a true two-

dimensional ship placement sequence modeled as an SMPL system.

If a lock chamber is the playing field and vessels are considered as blocks, the sequence of vessels filling up the chamber can be modeled similarly to the Tetris system modeled in [18].

For this, the chamber and vessel are dimensioned as shown in Figure 4. The widths of the chambers are discretised into $N_{\chi,(i,j)}$ horizontal bins. The widths of the vessels are similarly discretised into $W_\chi(k)$ horizontal bins. The width of a is bin consistent across all chambers and vessels. This discretisation is necessary to be able to model the placement sequence with discrete states. $W_\gamma(k)$ denotes the length of vessel k and $N_{\gamma,(i,j)}$ is the length of chamber (i,j). These lengths do not have to be discretised.

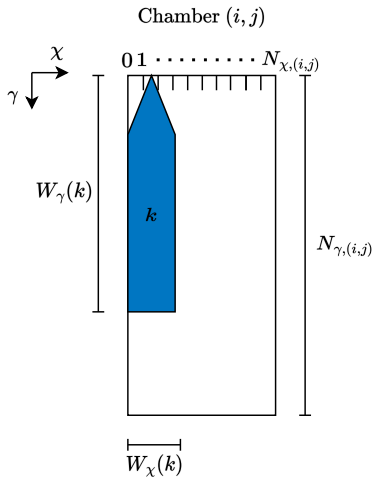


Figure 4: *Dimensions of a chamber and a vessel in the multi-vessel, multi-chamber placement model.*

The vessels are vertically slid into the bins of the chamber according to a placement sequence, simulating how vessels enter the chamber in reality. The step in the sequence is denoted by event counter g. If a vessel is slid into bin χ of chamber (i,j), its top left corner lines up with the leftmost point of bin χ in the chamber. Max-plus binary placement variable $\alpha_{\chi,(i,j)}(k,g)$ determines if vessel k is slid into bin χ of chamber (i,j) on step g.

The states of the sliding vessel placement system are the heights of stacked vessels in each of the bins of the chamber at each of the timesteps, $h_{\chi,(i,j)}(k,g)$. The states are associated with the vessels, so different vessels can have different bin stack height states if they pass through the chamber in different lockages.

Figure 5 shows an example vessel placement sequence. Only one vessel can be placed per event step. Vessel 0 is placed into bin 0 on step 0, vessel 1 is placed into bin 2 on step 1, and vessel 2 is placed into bin 0 on step 2, resulting in the final situation at n_{steps}. As a conservative first estimate, n_{steps} will be set to $n_{vessels}$, making sure that all vessels could be placed into the chamber in the same lockage, if their sizes and the chamber size permit it. This number should be made less conservative in the future, as each extra event step considered creates more binary placement variables. The evolution of the bin height states is listed in Table 1

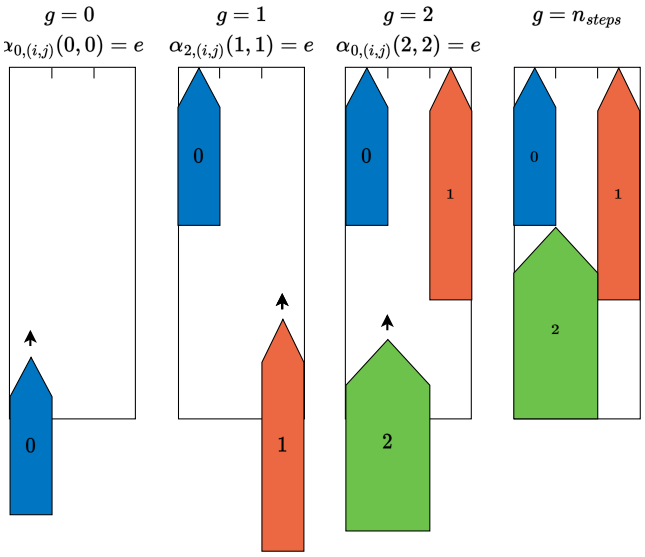


Figure 5: *Sliding vessel placement sequence example.*

Other considerations when placing vessels into the chamber are mooring and safety distances [9]. Furthermore, there are special rules for vessels with certain kinds of goods [21].

Vessels must be moored to the sides of the chamber or to another vessel, so they do not displace during the lockage. For vessel k to moor to another vessel $k-\mu$, the vessel k must be fully adjacent to vessel $k-\mu$, meaning the bow of vessel k may not be further forward in the chamber than the bow of vessel $k-\mu$, the stern of vessel k may not be further back in the chamber than the bow of vessel $k-\mu$, and there may be no horizontal distance between the vessels. Two vessels may not be moored to each other, and a chain of moored vessels must eventually be moored to the quay. Examples of proper and improper mooring situations are indicated in Figure 6.

The max-plus binary mooring variable $m_{(i,j)}(k,k-\mu)$ determines if vessel k is moored to vessel $k-\mu$ on chamber (i,j). $\hat{h}_{(i,j)}(k)$ is the vertical bow position of vessel k in chamber (i,j) and $\check{h}_{(i,j)}(k)$ is the vertical stern position of vessel k in chamber (i,j).

Vessels are required to keep safety distances between each other, for instance to allow for safe maneuvering inside the chamber. Tugs may also have to te able to maneuver in the lock chamber to position vessels. Vessels with flammable goods in particular are required to be at least 10 meters away from other vessels [21]. By approximation, safety distances are included into the placement size of the vessels in these models, as depicted in Figure 7. Including safety distance rules would make placement much more complicated.

Table 1: *Bin height values of the bins in the example in Figure 5.*

Bin Height	$g = 0$	$g = 1$	$g = 2$	$g = 3$
$h_{0,(i,j)}(k, g)\, \forall\, k \in \{0, 1, 2\}$	e	$W_\gamma(0)$	$W_\gamma(0)$	$W_\gamma(0) \otimes W_\gamma(2)$
$h_{1,(i,j)}(k, g)\, \forall\, k \in \{0, 1, 2\}$	e	e	e	$W_\gamma(0) \otimes W_\gamma(2)$
$h_{2,(i,j)}(k, g)\, \forall\, k \in \{0, 1, 2\}$	e	e	$W_\gamma(1)$	$W_\gamma(1)$

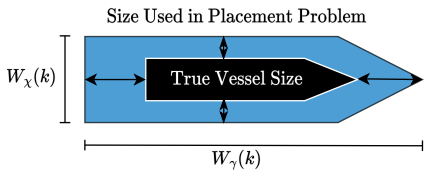


Figure 7: *Vessel placement size including safety distances.*

Dutch regulations have special rules for vessels with flammable goods, explosive goods, and toxic goods. Simplified versions of these rules are implemented.

The set $\mathcal{K}_\triangledown \subset \mathcal{K}$ is the set of vessels with flammable goods. They are required to keep 10 meters of distance to other vessels, translating to them not being able to moor to any other vessel.

$\mathcal{K}_{\triangledown\triangledown} \subset \mathcal{K}$ is the set of vessels with explosive or toxic goods. They may not be in the same lockage as any other vessels, with some exceptions that are neglected for the purposes of this paper, as they would require the creation of more specialized vessel sets.

The assumptions on one-dimensional sizes and capacities from the multi-vessel, multi-chamber capacity model are removed. The additional assumptions made are:

- The width of the lock chambers and vessels is discretised based on a set width per section.
- All vessels can moor to all other vessels, as long as the vessel that is mooring to the other vessels can be fully alongside the vessel it is being moored to, and as long as the vessel is not carrying flammable goods.
- Safety distances that should be maintained between vessels for maneuvering and because of regulations are included in the sizes used for placement of the vessels.
- Vessels with toxic and explosive goods may not be synchronised with any other vessels.
- No distinction is made between passenger and cargo vessels.
- A vessel must always be moored to the sides of the chamber or another vessel.

The assumption for passenger vessels has to do with regulations that vessels with flammable goods cannot be in the same lockages as passenger vessels.

4 Models

The three mathematical models that are outlined in this section serve as the set of rules with which a scheduler can generate optimization problems on a case-by-case basis.

The process of creating a schedule is shown in Figure 8. This sequence was automated in a Python script. It is run every time a schedule is required.

For each model, the constraints are first written as SMPL constraints. These constraints can be combined into an SMPL system. To solve for schedules with a conventional solver like Gurobi, the SMPL constraints are then converted to MILP constraints. Objectives are also added in the MILP modeling stage, and the same objectives can be used for all three models.

4.1 Single-Vessel, Single-Chamber Model

The single-vessel, single-chamber model serves as the basis for the multi-vessel multi-chamber models. It establishes the rules for event ordering and route generation.

4.1.1 SMPL Constraints

Equation 12 sets the minimum arrival time state value to 0. For all $(i, k) \in \mathcal{N} \times \mathcal{K}$:

$$x_i(k) \geq 0. \tag{12}$$

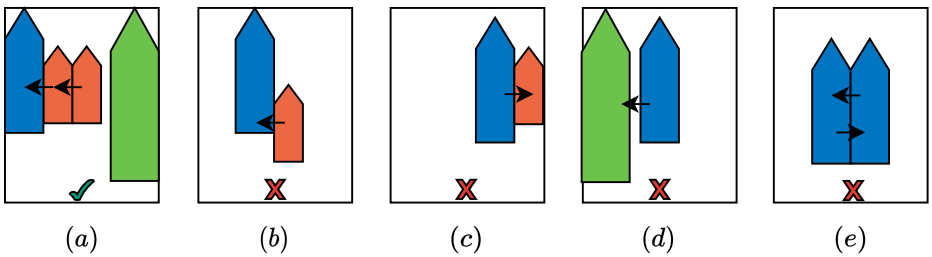


Figure 6: *Vessel mooring configurations.* (a)*: Acceptable configuration.* (b)*: The smaller vessel is not fully alongside and the larger vessel is not moored to anything.* (c)*: The larger vessel cannot be fully alongside the smaller vessel.* (d)*: The vessels are not adjacent.* (e)*: The vessels are both fully adjacent to each other, but neither of them are moored to the side of the chamber.*

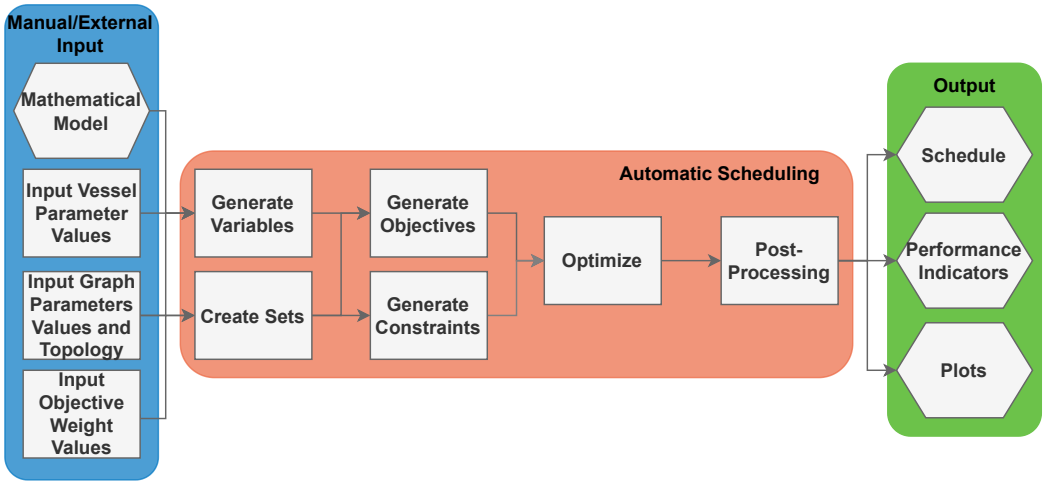


Figure 8: *Process of creating an IWT schedule.*

Equation 13 ensures that vessels depart after their departure time. For all $k \in \mathcal{K}$:

$$x_{b(k)} \geq u(k). \tag{13}$$

Equation 14 ensures that all vessels start their route at their departure node. For all $k \in \mathcal{K}$:

$$\bigoplus_{i \in \mathcal{S}(b(k))} \Big(w_{(b(k),i)}(k) \otimes \bigotimes_{j \in \mathcal{S}(b(k)) \setminus \{i\}} \overline{w}_{(b(k),j)}(k) \Big) = 0. \tag{14}$$

Similarly, Equation 15 ensures that all vessels end their route at their destination node. For all $k \in \mathcal{K}$:

$$\bigoplus_{i \in \mathcal{S}(d(k))} \Big(w_{(i,d(k))}(k) \otimes \bigotimes_{j \in \mathcal{S}(d(k)) \setminus \{i\}} \overline{w}_{(j,d(k))}(k) \Big) = 0. \tag{15}$$

A node that is visited but that is not a destination or departure node, must always have one ingoing arc and one outgoing arc, as per Equation 16. For all $(k,i) \in K \times \mathcal{N} \setminus \{d(k), b(k)\}$:

$$\bigoplus_{j \in \mathcal{S}(i)} \bigoplus_{l \in \mathcal{S}(i) \setminus \{j\}} \Big(w_{(j,i)}(k) \otimes w_{(i,l)}(k) \otimes \bigotimes_{p \in \mathcal{S}(i) \setminus \{j\}} \overline{w}_{(p,i)}(k) \otimes \bigotimes_{q \in \mathcal{S}(i) \setminus \{l\}} \overline{w}_{(i,q)}(k) \Big) \oplus \Big(\bigotimes_{j \in \mathcal{S}(i)} \overline{w}_{(j,i)}(k) \otimes \bigotimes_{l \in \mathcal{S}(i)} \overline{w}_{(i,l)}(k) \Big) = 0. \tag{16}$$

Together, Equation 14, Equation 15, and Equation 16 are the shortest path problem written in max-plus algebra, forming the structure that ensures connected routes are created between nodes $b(k)$ and $d(k)$ for every vessel.

A vessel must at least spend the minimum travel time, processing time, or traversal time on an arc it is assigned in its route. For all $(k,(i,j)) \in \mathcal{K} \times \mathcal{D}$:

$$x_j(k) \geq x_i(k) \otimes \tau_{(i,j)}(k) \otimes w_{(i,j)}(k) \tag{17}$$

Note that the max-plus binary variable $w_{(i,j)}(k)$ is used to turn off this constraint when false, turning it into $x_j(k) \geq \varepsilon$, which is always true. The same principle will be applied with other max-plus binary variables.

Vessels travelling in the same direction are ordered on locks according to Equation 18. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$x_i(k) \geq x_j(k-\mu) \otimes \tau_{(j,i)}(k) \otimes w_{(i,j)}(k) \otimes w_{(i,j)}(k) \otimes z_{(i,j)}(k, k-\mu). \tag{18}$$

Note that vessels following each other must have an empty or occupied lockage in between their passages, as the lock's water level must first return to its original level. This is not the case for vessels travelling in opposite travel direction as per Equation 19. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$x_i(k) \geq x_i(k-\mu) \otimes w_{(i,j)}(k) \otimes w_{(j,i)}(k-\mu) \otimes z_{(i,j)}(k, k-\mu). \tag{19}$$

Finally, Equation 20 ensures symmetry between $z_{(i,j)}(k, k-\mu)$ and $z_{(j,i)}(k, k-\mu)$, and anti-symmetry between $z_{(i,j)}(k, k-\mu)$ and $z_{(i,j)}(k-\mu, k)$. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\Big(z_{(i,j)}(k, k-\mu) \otimes z_{(j,i)}(k, k-\mu) \otimes \overline{z}_{(i,j)}(k-\mu, k) \Big) \oplus \Big(\overline{z}_{(i,j)}(k, k-\mu) \otimes \overline{z}_{(j,i)}(k, k-\mu) \otimes z_{(i,j)}(k-\mu, k) \Big) = 0. \tag{20}$$

4.1.2 SMPL System

The SMPL constraints listed in Equation 12 to Equation 20 are divisible into larger-than-inequality constraints that constrain the states, and logical equality constraints that constrain the possible values of max-plus binary variables.

The larger-than-inequality constraints can be grouped into an SMPL system equation. Let $\boldsymbol{x}(k)$ be the vector of

all arrival time states of vessel k as per Equation 21.

$$\boldsymbol{x}(k) = \begin{bmatrix} x_0(k) \\ x_1(k) \\ \vdots \\ x_{|\mathcal{N}|-1}(k) \end{bmatrix}, \tag{21}$$

where $|\mathcal{N}|$ denotes the cardinality of the set of nodes.

Let $\boldsymbol{w}(k)$ in Equation 22 be the vector of all routing variables of vessel k for all (i,j) in $\mathcal{D}$.

$$\boldsymbol{w}(k) = \left[w_{(i,j)}(k) | \forall\,(i,j) \in \mathcal{D}\right]^T, \tag{22}$$

Let $\boldsymbol{z}(k, k-\mu)$ in Equation 23 be the vector of all ordering variables connecting vessel k and vessel $k-\mu$, for all (i,j) in $\mathcal{D}_{locks}$.

$$\boldsymbol{z}(k,k-\mu) = \left[z_{(i,j)}(k,k-\mu) | \forall\,(i,j) \in \mathcal{D}_{locks}\right]^T. \tag{23}$$

The routing inequalities for vessel k are dependent on $\boldsymbol{w}(k), \boldsymbol{x}(k)$, and $u(k)$. The routing inequalities for vessel k can be gathered together as in Equation 24.

$$\begin{aligned} \boldsymbol{x}(k) \geq\ & A_{routing,(k)}(\boldsymbol{w}(k)) \otimes \boldsymbol{x}(k) \\ & \oplus B_{routing,(k)} \otimes u(k), \end{aligned} \tag{24}$$

where $A_{routing,(k)}(\boldsymbol{w}(k))$ is the routing inequality constraint matrix for vessel k, and $B_{routing,(k)}$ is the routing reference input matrix for vessel k.

The ordering inequalities for vessel k are dependent on the ordering variables $\boldsymbol{z}(k, k-\mu)$ with all other vessels $k-\mu$, the routing variables $\boldsymbol{w}(k)$, the states $\boldsymbol{x}(k)$, and routing variables $\boldsymbol{w}(k)$ of all other vessels $k-\mu$. The ordering inequalities between vessel k and $k-\mu$ are given by Equation 25.

$$\begin{aligned} x(k) \geq\ & A_{ordering,(k,k-\mu)}(\boldsymbol{w}(k), \boldsymbol{w}(k-\mu), \\ & \boldsymbol{z}(k,k-\mu)) \otimes \boldsymbol{x}(k-\mu), \end{aligned} \tag{25}$$

where $A_{ordering,(k,k-\mu)}\left(\boldsymbol{w}(k), \boldsymbol{w}(k,k-\mu), \boldsymbol{z}(k,k-\mu)\right)$ is the ordering inequality matrix between vessel k and vessel $k-\mu$.

Let $\mathcal{M}(k)$ be the set of possible values of μ such that $\{k-\mu | \mu \in \mathcal{M}(k)\} = \mathcal{K} \setminus \{k\}$. Then the ordering inequalities between vessel k and all other vessels can be described as in Equation 26.

$$\begin{aligned} \boldsymbol{x}(k) \geq \bigoplus_{\mu \in \mathcal{M}(k)} & A_{ordering,(k,k-\mu)}(\boldsymbol{w}(k), \\ & \boldsymbol{w}(k,k-\mu), \boldsymbol{z}(k,k-\mu)) \otimes \boldsymbol{x}(k-\mu). \end{aligned} \tag{26}$$

Note that, as these are all larger-than-inequality constraints, they can be further combined into a single inequality constraint matrix and reference input matrix. Let $\mu_{\min}(k)$ and $\mu_{\max}(k)$ be the minimum and maximum values of $\mathcal{M}(k)$.

Let $\boldsymbol{\phi}(k)$ in Equation 27 be the combined vector of all binary variables that are related to decisions of vessel k, also referred to as the switching variable vector.

$$\boldsymbol{\phi}(k) = \begin{bmatrix} \boldsymbol{w}(k) \\ \boldsymbol{w}(k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{w}(k-\mu_{\min}(k)) \\ \boldsymbol{z}(k,k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{z}(k,k-\mu_{\min}(k)) \end{bmatrix} \tag{27}$$

Let $A_{(k,k-\mu)}(\boldsymbol{\phi}(k))$ in Equation 28 be the inequality constraint matrix for vessel k combined with vessel $k-\mu$.

$$A_{(k,k-\mu)}(\boldsymbol{\phi}(k)) = \begin{cases} A_{routing,(k)}(\boldsymbol{\phi}(k)) & \text{If } \mu = 0 \\ A_{ordering,(k,k-\mu)}(\boldsymbol{\phi}(k)) & \text{If } \mu \in \mathcal{M}(k) \end{cases} \tag{28}$$

Let $B_{(k)}$ in Equation 29 be the reference input matrix for vessel k.

$$B_{(k)} = B_{routing,(k)} \tag{29}$$

Then, the inequality constraints for vessel k are given by the SMPL system in Equation 30 with the switching of its modes governed by switching variable vector $\boldsymbol{\phi}(k)$.

$$\boldsymbol{x}(k) \geq \bigoplus_{\mu \in \mathcal{M}(k) \cup \{0\}} A_{(k,k-\mu)}(\boldsymbol{\phi}(k)) \otimes \boldsymbol{x}(k-\mu) \oplus B_{(k)} \otimes u(k). \tag{30}$$

The equality constraints define the set of allowable switching modes $\mathcal{Q}(k)$ the SMPL system for vessel k may be in. For vessel k they are written in Equation 31.

$$\boldsymbol{q}_{(k)}(\boldsymbol{\phi}(k)) = \boldsymbol{y}_{(k)}, \tag{31}$$

where $\boldsymbol{q}_{(k)}$ is the equality constraint vector of vessel k, and where the entries of the vector $\boldsymbol{y}_{(k)}$ are max-plus scalar.

Any permutation of $\boldsymbol{\phi}(k)$ that is in $\mathcal{Q}(k)$ is an allowable mode of the SMPL system for vessel k. If all vessels k are in an allowable switching mode according to Equation 31, then the combined system of all vessels is in an allowable switching mode, meaning that the entire system is compliant with the logical constraints.

4.1.3 MILP Conversion

A general MILP problem where all integer decision variables are binary, can be written as:

$$\begin{aligned} \min_{x,\phi} \quad & J = c_x^T x + c_\phi^T \phi \\ \text{s.t} \quad & A_x x + A_\phi \phi \leq f \\ & x \geq 0 \\ & \phi \in \mathbb{B}^{n_\phi}. \end{aligned}$$

Where J is the objective function, x is a vector of continuous decision variables, ϕ is a vector of binary and integer decision variables, c_x and c_ϕ are objective function cost vectors, A_x and A_ϕ are inequality constraint matrices, f is the scalar inequality ocnstraint vector, and $\mathbb{B}$ is the set $\{0,1\}$.

The SMPL constraints are converted according to this template to be used in a MILP problem that is solvable by a conventional solver.

For inequality constraints, max-plus binary variables can be converted to normal binary variables according to equation Equation 32. Where v is an arbitrary (max-plus) binary variable, and β is a large negative scalar such that $\beta \approx \varepsilon$, thereby approximating the function of the max-plus binary variable.

$$\begin{array}{ccc} \text{Max-plus Algebra} & \longleftrightarrow & \text{Conventional Algebra} \\ v & & \beta(1-v) \\ \overline{v} & & \beta v \\ v, \overline{v} \in \mathbb{B}_{max} & & v \in \mathbb{B} \end{array} \tag{32}$$

Binary conversions of max-plus binary variables are not given different symbols, as it should be clear from the context whether it is binary or max-plus binary.

Equation 12 converts to Equation 33. For all $(i,k) \in \mathcal{N} \times \mathcal{K}$:

$$-x_i(k) \leq 0 \tag{33}$$

Equation 13 converts to Equation 34. For all $k \in \mathcal{K}$:

$$-x_{b(k)} \leq -u(k). \tag{34}$$

The shortest path problem constraints in Equation 14, Equation 15, and Equation 16 convert to Equation 35-Equation 41. For all $k \in \mathcal{K}$:

$$\sum_{i \in \mathcal{S}(b(k))} w_{(b(k),i)}(k) \leq 1 \tag{35}$$

$$-\sum_{i \in \mathcal{S}(b(k))} w_{(b(k),i)}(k) \leq -1 \tag{36}$$

$$\sum_{i \in \mathcal{S}(d(k))} w_{(i,d(k))}(k) \leq 1 \tag{37}$$

$$-\sum_{i \in \mathcal{S}(d(k))} w_{(i,d(k))}(k) \leq -1. \tag{38}$$

For all $(k,i) \in \mathcal{K} \times \mathcal{N} \setminus \{b(k), d(k)\}$:

$$\sum_{j \in \mathcal{S}(i)} w_{(i,j)}(k) - \sum_{j \in \mathcal{S}(i)} w_{(j,i)}(k) \leq 0 \tag{39}$$

$$\sum_{j \in \mathcal{S}(i)} w_{(j,i)}(k) - \sum_{j \in \mathcal{S}(i)} w_{(i,j)}(k) \leq 0 \tag{40}$$

$$\sum_{j \in \mathcal{S}(i)} w_{(j,i)}(k) \leq 1. \tag{41}$$

Equation 17 converts to Equation 42. For all $((i,j),k) \in \mathcal{D} \times \mathcal{K}$:

$$-x_j(k) + x_i(k) - \beta w_{(i,j)}(k) \leq -\beta - \tau_{(i,j)}(k). \tag{42}$$

The ordering constraints in Equation 18 and Equation 19 convert to Equation 43 and Equation 44. For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$\begin{aligned} -x_i(k) + x_j(k-\mu) - \beta w_{(i,j)}(k) - \beta w_{(i,j)}(k-\mu) \\ - \beta z_{(i,j)}(k, k-\mu) \leq -3\beta - \tau_{(j,i)}(k) \end{aligned} \tag{43}$$

$$\begin{aligned} -x_i(k) + x_i(k-\mu) - \beta w_{(i,j)}(k) \\ - \beta w_{(j,i)}(k-\mu) - \beta z_{(i,j)}(k, k-\mu) \leq -3\beta. \end{aligned} \tag{44}$$

Finally, Equation 20 converts to Equation 45-Equation 48. For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$z_{(i,j)}(k, k-\mu) - z_{(j,i)}(k, k-\mu) \leq 0 \tag{45}$$

$$z_{(j,i)}(k, k-\mu) - z_{(i,j)}(k, k-\mu) \leq 0 \tag{46}$$

$$z_{(i,j)}(k, k-\mu) + z_{(i,j)}(k-\mu, k) \leq 1 \tag{47}$$

$$-z_{(i,j)}(k, k-\mu) - z_{(i,j)}(k-\mu, k) \leq -1. \tag{48}$$

4.2 Multi-Vessel, Multi-Chamber Capacity Model

The multi-vessel, multi-chamber capacity model introduces locks with multiple chambers that can process multiple vessels at one time. The model requires no changes from the single-chamber, single-vessel model to accommodate multi-chamber locks, only the graph input described in the problem definition is different. Multi-vessel lockages require synchronisation of vessels, which is what the majority of new or changed constraints are about.

4.2.1 SMPL Constraints

A lockage with multiple vessels may only start once all vessels have entered the chamber, though other vessels may arrive earlier and wait in the chamber, this is constrained in Equation 49. For $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\begin{aligned} x_j(k) \geq \Big(x_i(k) \oplus \bigoplus_{k-\mu \in \mathcal{K} \setminus \{k\}} x_i(k-\mu) \otimes s_{(i,j)}(k, k-\mu) \Big) \\ \otimes \tau_{(i,j)}(k) \\ \otimes \bigotimes_{k-\mu \in \mathcal{K} \setminus \{k\}} \Big((s_{(i,j)}(k, k-\mu) \otimes \overline{\tau}_{(i,j)}) \oplus 0 \Big) \otimes w_{(i,j)}(k) \end{aligned} \tag{49}$$

If vessel k is synchronised with vessel $k-\mu$, vessel $k-\mu$ must be synchronised with vessel k as per Equation 50. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\begin{aligned} \Big(s_{(i,j)}(k, k-\mu) \otimes s_{(i,j)}(k-\mu, k) \Big) \\ \oplus \Big(\overline{s}_{(i,j)}(k, k-\mu) \otimes \overline{s}_{(i,j)}(k-\mu, k) \Big) = 0 \end{aligned} \tag{50}$$

Equation 51 constrains that the sum of the sizes of a group of synchronised vessels must not exceed the capacity of the chamber. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\begin{aligned} \rho(k) \otimes w_{(i,j)}(k) \otimes \\ \bigotimes_{k-\mu \in \mathcal{K} \setminus \{k\}} (s_{(i,j)}(k, k-\mu) \otimes \rho(k-\mu) \oplus 0) \leq c_{(i,j)} \end{aligned} \tag{51}$$

Equation 52 ensures that vessels can only be synchronised on a lock if their route includes that lock. For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$\begin{aligned} \Big(w_{(i,j)}(k) \otimes s_{(i,j)}(k, k-\mu) \Big) \\ \oplus \Big(w_{(i,j)}(k) \otimes \overline{s}_{(i,j)}(k, k-\mu) \Big) \\ \oplus \Big(\overline{w}_{(i,j)}(k) \otimes \overline{s}_{(i,j)}(k, k-\mu) \Big) = 0. \end{aligned} \tag{52}$$

The ordering constraint in Equation 18 would interfere with synchronised vessels. Thus, an $s_{(i,j)}(k, k-\mu)$

term is added to turn it off when vessels are synchronised in a replacement constraint in Equation 53. For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$x_i(k) \geq x_j(k-\mu) \otimes \tau_{(i,j)}(k) \otimes w_{(i,j)}(k-\mu) \otimes w_{(i,j)}(k) \otimes z_{(i,j)}(k,k-\mu) \otimes \overline{s}_{(i,j)}(k,k-\mu). \quad (53)$$

4.2.2 SMPL System

The SMPL system equation and allowable modes can once again be constructed with the SMPL constraints. This model has smaller-than-equality constraints that constrain the max-plus-binary variables, so they can be grouped under the constraints that describe the allowable system modes.

The larger-than-inequality constraints of the single-vessel, single-chamber model and the additional or changed constraints in the multi-vessel, multi-chamber capacity model are grouped into system equations. Let $\boldsymbol{s}(k,k-\mu)$ in Equation 54 be the vector of synchronisation variables between vessel k and $k-\mu$ on all lock chamber arcs.

$$\boldsymbol{s}(k,k-\mu) = \left[s_{(i,j)}(k,k-\mu) | \forall\,(i,j) \in \mathcal{D}_{locks}\right]^T. \quad (54)$$

Let $\boldsymbol{\phi}(k)$ in Equation 55 now include the vectors $\boldsymbol{s}(k,k-\mu)$ between vessel k and all other vessels.

$$\boldsymbol{\phi}(k) = \begin{bmatrix} \boldsymbol{w}(k) \\ \boldsymbol{w}(k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{w}(k-\mu_{\min}(k)) \\ \boldsymbol{z}(k,k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{z}(k,k-\mu_{\min}(k)) \\ \boldsymbol{s}(k,k-\mu_{\min}(k)) \\ \vdots \\ \boldsymbol{s}(k,k-\mu_{\max}(k)) \end{bmatrix} \quad (55)$$

The synchronisation inequality constraints between vessel k and $k-\mu$ can be grouped into a synchronisation inequality constraint matrix $A_{synch,(k,k-\mu)}(\boldsymbol{\phi}(k))$. Together with the ordering constraint matrix $A_{ordering,(k,k-\mu)}$, this gives the inequality constraint matrix linking vessel k to vessel $k-\mu$ in Equation 56.

$$A_{(k,k-\mu)}(\boldsymbol{\phi}(k)) = A_{ordering,(k,k-\mu)}(\boldsymbol{\phi}(k)) \oplus A_{synch,(k,k-\mu)}(\boldsymbol{\phi}(k)) \qquad \text{If } \mu \in \mathcal{M}(k). \quad (56)$$

When $k = k-\mu$, then $A_{(k,k-\mu)}(\boldsymbol{\phi}(k))$ is given by the routing constraint matrix in Equation 57.

$$A_{(k,k-\mu)}(\boldsymbol{\phi}(k)) = A_{routing,(k)}(\boldsymbol{\phi}(k)) \quad (57)$$

The full set of SMPL system equations is then written the same as in Equation 30.

There is now also a smaller than inequality constraint in Equation 51. As this inequality constraint does not include the arrival states, it is just another rule that determines the set of allowable modes $\mathcal{Q}(k)$ for each of the vessels.

The equality constraints are once again grouped as in Equation 31. Furthermore, let the constraint in Equation 51 be grouped as in Equation 58.

$$\boldsymbol{C}_{(k)}(\boldsymbol{\phi}(k)) \leq \boldsymbol{c_{capacities}}, \quad (58)$$

where $\boldsymbol{C}_{(k)}(\boldsymbol{\phi}(k))$ is the smaller than inequality constraint vector for vessel k, and c is the vector of the chamber capacities of all of the chambers in the topology graph as written in Equation 59.

$$\boldsymbol{c_{capacities}} = \left[c_{(i,j)} | (i,j) \in \mathcal{D}_{locks}\right]^T. \quad (59)$$

Then Equation 31 and Equation 59 together determine the set of allowable switching modes $\mathcal{Q}(k)$ for each of the vessels.

4.2.3 MILP Conversion

The MILP constraints are built according to the same principles as the single-vessel, single-chamber model.

The conversion of Equation 49 requires the introduction of auxiliary variables. Define the variable $\theta_{synch,(i,j)}(k,k-\mu)$ as:

$$\theta_{synch,(i,j)}(k,k-\mu) = x_i(k-\mu)s_{(i,j)}(k,k-\mu), \quad (60)$$

for all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$, which can be linearly enforced by adding the following constraints [22]:

$$\theta_{synch,(i,j)}(k,k-\mu) \leq -\beta s_{(i,j)}(k,k-\mu) \quad (61)$$

$$\theta_{synch,(i,j)}(k,k-\mu) \geq 0 \quad (62)$$

$$\theta_{synch,(i,j)}(k,k-\mu) \leq x_i(k-\mu) \quad (63)$$

$$\theta_{synch,(i,j)}(k,k-\mu) \geq x_i(k-\mu) + \beta(1 - s_{(i,j)}(k,k-\mu)). \quad (64)$$

Furthermore, introduce the variable $t_{synch,(i,j)}(k)$ for all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$ as:

$$t_{synch,(i,j)}(k) = \max\Big(x_i(k), \max(\{\theta_{synch,(i,j)}(k,k-\mu) | \forall k-\mu \in \mathcal{K} \setminus \{k\}\})\Big), \quad (65)$$

which can be linearly enforced by a constraint and a set of constraints:

$$t_{synch,(i,j)}(k) \geq x_i(k). \quad (66)$$

For all $k-\mu \in \mathcal{K} \setminus \{k\}$:

$$t_{synch,(i,j)}(k) \geq \theta_{synch,(i,j)}(k,k-\mu). \quad (67)$$

Then the constraint in Equation 49 can be linearly converted as in Equation 69. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$t_{synch(i,j)}(k) - x_j(k) - \beta w_{(i,j)}(k) + \sum_{k-\mu \in \mathcal{K} \setminus \{k\}} s_{(i,j)}(k,k-\mu)\overline{\tau}_{(i,j)} \leq -\beta - \tau_{(i,j)}(k) \quad (68)$$

Equation 51 converts to Equation 70. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$-\beta w_{(i,j)}(k) + \sum_{k-\mu \in \mathcal{K} \setminus \{k\}} \rho(k-\mu)s_{(i,j)}(k,k-\mu) \leq c_{(i,j)} - \beta - \rho(k). \quad (69)$$

Equation 52 converts to Equation 71. For all $((i,j),k,k-$

$\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$s_{(i,j)}(k, k-\mu) - w_{(i,j)}(k) \leq 0. \quad (71)$$

Finally, Equation 53 converts to Equation 72. For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$\begin{aligned} x_j(k-\mu) - x_i(k) - \beta w_{(i,j)}(k) - \beta w_{(i,j)}(k-\mu) \\ - \beta z_{(i,j)}(k, k-\mu) + \beta s_{(i,j)}(k, k-\mu) \\ \leq -3\beta - \tau_{(i,j)}(k) \end{aligned} \quad (72)$$

4.3 Multi-Vessel, Multi-Chamber Placement Model

The multi-vessel, multi-chamber placement model introduces two-dimensional vessel placement through the addition of an SMPL placement sequence model. Furthermore, rules for mooring and vessels carrying special goods are added.

4.3.1 SMPL Constraints

The constraints in Equation 49 and Equation 51 are no longer required.

Each chamber must initially be empty for each vessel and the chamber bin stack heights must be non-negative, as constrained in Equation 73. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\boldsymbol{h}_{(i,j)}(k,0) \geq \mathbf{0}, \quad (73)$$

where $\mathbf{0}$ is an $N_{\chi,(i,j)}$ entry long vector of 0, and where $\boldsymbol{h}_{(i,j)}(k,g)$ is the vector of all bin height states for vessel k in chamber (i,j) at step g:

$$\boldsymbol{h}_{(i,j)}(k,g) = \begin{bmatrix} h_{0,(i,j)}(k,g) \\ h_{1,(i,j)}(k,g) \\ \vdots \\ h_{N_{\chi,(i,j)}-1,(i,j)}(k,g) \end{bmatrix}. \quad (74)$$

For vessel k, if a different vessel $k-\mu$ vessel k is synchronised with is slid into the chamber at step g, the bin height states are updated as in Equation 75. For all $(k, k-\mu, g, (i,j), \chi) \in \mathcal{K} \times \mathcal{K} \setminus \{k\} \times \{0,1,...,n_{steps}-1\} \times \mathcal{D}_{locks} \times \{0,1,...,N_{\chi,(i,j)} | W_\chi(k-\mu)+\chi \leq N_{\chi,(i,j)}\}$:

$$\begin{aligned} &\boldsymbol{h}_{(i,j)}(k,g+1) \geq s_{(i,j)}(k,k-\mu) \otimes \\ &\alpha_{\chi,(i,j)}(k-\mu,g) \otimes A_{\chi,(i,j)}(k-\mu) \otimes \boldsymbol{h}_{(i,j)}(k,g) \end{aligned} \quad (75)$$

Where $A_{\chi,(i,j)}(k)$ is the ship placement matrix for when vessel k is placed into bin χ of chamber (i,j). The general form of $A_{\chi,(i,j)}(k)$ is given in Equation 76.

$$[b]A_{\chi,(i,j)}(k) = \begin{bmatrix} E(\chi,\chi) & \mathcal{E}(\chi, W_\chi(k)) & \mathcal{E}(\chi, N_{\gamma,(i,j)} - W_\chi(k) - \chi) \\ \mathcal{E}(W_\chi(k),\chi) & \overline{W}_\gamma(k) & \mathcal{E}(W_\chi(k), N_{\gamma,(i,j)} - W_\chi(k) - \chi) \\ \mathcal{E}(N_{\gamma,(i,j)} - W_\chi(k) - \chi, \chi) & \mathcal{E}(N_{\gamma,(i,j)} - W_\chi(k) - \chi, W_\chi(k)) & E(N_{\gamma,(i,j)} - W_\chi(k) - \chi, N_{\gamma,(i,j)} - W_\chi(k) - \chi)) \end{bmatrix}, \quad (76)$$

where $\overline{W}_\gamma(k)$ is a full $W_\chi(k) \times W_\chi(k)$ matrix of $W_\gamma(k)$:

$$\overline{W}_\gamma = \begin{bmatrix} W_\gamma(k) & \dots & W_\gamma(k) \\ \vdots & \ddots & \vdots \\ W_\gamma(k) & \dots & W_\gamma(k) \end{bmatrix}. \quad (77)$$

Note that Equation 75 is deactivated if vessels k and $k-\mu$ are not synchronised.

Similarly, Equation 78 gives the bin height updates for vessel k if vessel k itself is placed into the chamber. For all $(k,g,(i,j),\chi) \in \mathcal{K} \times \{0,1,n_{steps}-1\} \times \mathcal{D}_{locks} \times \{0,1,...,N_{\chi,(i,j)} | W_\chi(k)+\chi \leq N_{\chi,(i,j)}\}$:

$$\begin{aligned} h_{(i,j)}(k,g+1) \geq \alpha_{\chi,(i,j)}(k,g) \\ \otimes A_{\chi,(i,j)}(k) \otimes h_{(i,j)}(k,g). \end{aligned} \quad (78)$$

Equation 79 constrains that the bin height states for vessels k and $k-\mu$ are the same on a given chamber if they are synchronised on that chamber. For all $((i,j),k,k-\mu,g) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\} \times \{1,2,...,n_{steps}\}$:

$$\boldsymbol{h}_{(i,j)} \geq s_{(i,j)}(k,k-\mu) \otimes \boldsymbol{h}_{(i,j)}(k-\mu,g). \quad (79)$$

Equation 80 constrains that the bin height states can never decrease from a step g to $g+1$. For all $((i,j),k,g) \in \mathcal{D}_{locks} \times \mathcal{K} \times \{0,1,...,n_{steps}-1\}$:

$$\boldsymbol{h_{(i,j)}}(k,g+1) \geq \boldsymbol{h}_{(i,j)}(k,g). \quad (80)$$

Equation 81 constrains that if a vessel passes through a chamber, it must be placed once and at most once in that chamber. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\begin{aligned} &\bigoplus_{g \in \{0,1,...,n_{steps}-1\}} \bigoplus_{\chi \in \{0,1,...,N_{\chi,(i,j)-1} | \chi + W_\chi(k) \leq N_{\chi,(i,j)}\}} \\ &\Big(w_{(i,j)}(k) \otimes \alpha_{\chi,(i,j)}(k,g) \\ &\otimes \bigotimes_{p \in \{0,1,...,n_{steps}-1\} \setminus \{g\}} \bigotimes_{l \in \{0,1,...,N_{\chi,(i,j)-1}\} \setminus \{\chi\}} \overline{\alpha}_{l,(i,j)}(k,p) \Big) \\ &\oplus \Big(\overline{w}_{(i,j)}(k) \otimes \bigotimes_{g \in \{0,1,...,n_{steps}\}} \\ &\bigotimes_{\chi \in \{0,1,...,N_{\chi,(i,j)}-1\}} \overline{\alpha}_{\chi,(i,j)}(k,g) \Big) = e \end{aligned} \quad (81)$$

Equation 82 constrains that only a single vessel can be placed into a chamber per placement step for synchronised groups of vessels. For all $(k, k-\mu,(i,j),\chi,g) \in \mathcal{K} \times \mathcal{K} \setminus \{k\} \times \mathcal{D}_{locks} \times \{0,1,...,N_{\chi,(i,j)}-$

$1\} \times \{0, 1, ..., n_{steps} - 1\}$:

$$\Big(s_{(i,j)}(k, k-\mu) \otimes \alpha_{\chi,(i,j)}(k, g) \otimes \bigotimes_{p\in\{0,1,...,N_{\chi,(i,j)}-1\}} \overline{\alpha}_{p,(i,j)}(k-\mu, g)\Big) \oplus \Big(s_{(i,j)}(k, k-\mu) \otimes \bigotimes_{p\in\{0,1,...,N_{\chi,(i,j)}-1\}} \overline{\alpha}_{p,(i,j)}(k, g) \otimes \alpha_{\chi,(i,j)}(k-\mu, g)\Big) \oplus \Big(s_{(i,j)}(k, k-\mu) \otimes \overline{\alpha}_{\chi,(i,j)}(k, g) \otimes \bigotimes_{p\in\{0,1,...,N_{\chi,(i,j)}-1\}} \overline{\alpha}_{p,(i,j)}(k-\mu, g)\Big) \oplus \Big(s_{(i,j)}(k, k-\mu) \otimes \bigotimes_{p\in\{0,1,...,N_{\chi,(i,j)}-1\}} \overline{\alpha}_{p,(i,j)}(k, g) \otimes \overline{\alpha}_{\chi,(i,j)}(k-\mu, g)\Big) \oplus \overline{s}_{(i,j)}(k, k-\mu) = e \quad (82)$$

Equation 83 defines how the bow position of a vessel is determined.

$$\hat{h}_{(i,j)}(k) = \bigoplus_{g\in\{0,1,...,n_{steps}-1\}} \bigoplus_{\chi\in\{0,1,...,N_{\chi,(i,j)}-1|\chi+W_\chi(k)\le N_{\chi,(i,j)}\}} \Big(\alpha_{\chi,(i,j)}(k, g) \otimes \bigoplus_{p\in\{0,1,...,W_\chi(k)-1|p+W_\chi(k)\le N_{\chi,(i,j)}\}} h_{\chi+p,(i,j)}(k, g)\Big), \quad (83)$$

The stern position of a vessel is defined by Equation 84.

$$\check{h}_{(i,j)}(k) = \hat{h}_{(i,j)}(k) \otimes W_\gamma(k). \quad (84)$$

Equation 85 constrains that vessels can only be moored to each other if they are synchronised. For all $((i, j), k, k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$\Big(m_{(i,j)}(k, k-\mu) \otimes s_{(i,j)}(k, k-\mu)\Big) \oplus \Big(s_{(i,j)}(k, k-\mu) \otimes \overline{m}_{(i,j)}(k, k-\mu)\Big) \oplus \Big(\overline{s}_{(i,j)}(k, k-\mu) \otimes \overline{m}_{(i,j)}(k, k-\mu)\Big) = e \quad (85)$$

Equation 86 constrains that a vessel can at most moor to one vessel. For all $((i, j), k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\Big(\bigotimes_{k-\mu\in\mathcal{K}\setminus\{k\}} \overline{m}_{(i,j)}(k, k-\mu)\Big) \oplus \bigoplus_{k-\mu\in\mathcal{K}\setminus\{k\}} \Big(m_{(i,j)}(k, k-\mu) \otimes \bigotimes_{p\in\mathcal{K}\setminus\{k,k-\mu\}} \overline{m}_{(i,j)}(k, p)\Big) = e \quad (86)$$

Equation 87 constrains that a vessel that is adjacent to the sides of the chamber cannot be moored to another vessel and that vessels that are not adjacent to the sides of the chamber must be moored to another vessel.For all $((i, j), k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\Big(\overline{w}_{(i,j)}(k) \otimes \bigotimes_{k-\mu\in\mathcal{K}\setminus\{k\}} \overline{m}_{(i,j)}(k, k-\mu)\Big) \oplus \Big(w_{(i,j)} \otimes \bigoplus_{k-\mu\in\mathcal{K}\setminus\{k\}} \bigoplus_{g\in\{0,1,...,n_{steps}-1\}} \bigoplus_{\chi\in\{1,2,...,N_{\chi,(i,j)}-2\}|\chi+W_\chi(k)+1\le N_{\chi,(i,j)}\}} \big(\alpha_{\chi,(i,j)}(k, g) \otimes m_{(i,j)}(k, k-\mu)\big)\Big) \oplus \Big(w_{(i,j)}(k) \otimes \bigoplus_{k-\mu\in\mathcal{K}\setminus\{k\}} \bigoplus_{g\in\{0,1,...,n_{steps}-1\}} \bigoplus_{\chi\in\{0,N_{\chi,(i,j)}-W_\chi(k)\}} \alpha_{\chi,(i,j)}(k, g) \otimes \overline{m}_{(i,j)}(k, k-\mu)\Big) = e \quad (87)$$

Equation 88 constrains that only adjacent vessels can be moored to each other. For all $((i, j), k, k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$\Big(m_{(i,j)}(k, k-\mu) \otimes \bigoplus_{l\in\{0,1,...,n_{steps}-1\}} \bigoplus_{g\in\{0,1,...,n_{steps}-1\}} \bigoplus_{\chi\in\{1,2,...,N_{\chi,(i,j)}-W_\chi(k)-1\}} \bigoplus_{q\in\{\chi-W_\chi(k-\mu),\chi+W_\chi(k)\}} \alpha_{\chi,(i,j)}(k, g) \otimes \alpha_{q,(i,j)}(k-\mu, l)\Big) \oplus \overline{m}_{(i,j)}(k, k-\mu) = e \quad (88)$$

Equation 89 constrains that two vessels cannot be moored to each other. For all $((i, j), k, k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$\big(m_{(i,j)}(k, k-\mu) \otimes \overline{m}_{(i,j)}(k-\mu, k)\big) \oplus \big(\overline{m}_{(i,j)}(k, k-\mu) \otimes m_{(i,j)}(k-\mu, k)\big) \oplus \big(\overline{m}_{(i,j)}(k, k-\mu) \otimes \overline{m}_{(i,j)}(k-\mu, k)\big) = e \quad (89)$$

Equation 90 constrains that vessels with flammable goods may not be moored to any other vessels. For all $((i, j), k) \in \mathcal{D}_{locks} \times \mathcal{K}_\nabla$:

$$\bigoplus_{k-\mu\in\mathcal{K}\setminus\{k\}} m_{(i,j)}(k, k-\mu) = \varepsilon \quad (90)$$

Finally, Equation 91 constrains that vessels with explosive or flammable goods may not be synchronised with any other vessels. For all $k \in \mathcal{K}_{\nabla\nabla}$:

$$\bigotimes_{(i,j)\in\mathcal{D}_{locks}} \bigotimes_{k-\mu\in\mathcal{K}} \big(\overline{s}_{(i,j)}(k, k-\mu)\big) = e \quad (91)$$

4.3.2 SMPL System

The SMPL mathematical model is now a combination of two SMPL systems: one to schedule the vessels' arrival time states, and one to place the vessels into the chamber. Both of these systems are linked by the max-plus binary switching variables.

Let $\boldsymbol{\alpha_{(i,j)}}(k, g)$ in Equation 92 be the vector of placement variables $\alpha_{\chi,(i,j)}(k, g)$ of vessel k on chamber (i, j)

at event step g.

$$\boldsymbol{\alpha_{(i,j)}}(k,g) = \begin{bmatrix} \alpha_{0,(i,j)}(k,g) \\ \alpha_{1,(i,j)}(k,g) \\ \vdots \\ \alpha_{N_{\chi,(i,j)}-1,(i,j)}(k,g) \end{bmatrix}. \quad (92)$$

Let $\boldsymbol{\alpha_{(i,j)}}(k)$ in Equation 93 be the vector of $\boldsymbol{\alpha_{(i,j)}}(k,g)$ for all event steps g.

$$\boldsymbol{\alpha_{(i,j)}}(k) = \begin{bmatrix} \boldsymbol{\alpha_{(i,j)}}(k,0) \\ \boldsymbol{\alpha_{(i,j)}}(k,1) \\ \vdots \\ \boldsymbol{\alpha_{(i,j)}}(k,n_{steps}-1) \end{bmatrix}. \quad (93)$$

Let $\boldsymbol{\alpha}(k)$ be the vector of $\boldsymbol{\alpha_{(i,j)}}(k)$ for all chambers (i,j).

$$\boldsymbol{\alpha}(k) = \left[\boldsymbol{\alpha_{(i,j)}}(k) | \forall (i,j) \in \mathcal{D}_{locks}\right]^T. \quad (94)$$

Let $\boldsymbol{m}(k,k-\mu)$ in Equation 95 be the vector of $\boldsymbol{m_{(i,j)}}(k,k-\mu)$.

$$\boldsymbol{m}(k,k-\mu) = \left[\boldsymbol{m_{(i,j)}}(k,k-\mu) | \forall (i,j) \in \mathcal{D}_{locks}\right]^T. \quad (95)$$

Let the switching variable vector $\phi(k)$ for vessel k then be defined as in Equation 96.

$$\boldsymbol{\phi}(k) = \begin{bmatrix} \boldsymbol{w}(k) \\ \boldsymbol{w}(k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{w}(k-\mu_{\min}(k)) \\ \boldsymbol{z}(k,k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{z}(k,k-\mu_{\min}(k)) \\ \boldsymbol{s}(k,k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{s}(k,k-\mu_{\min}(k)) \\ \boldsymbol{\alpha}(k) \\ \boldsymbol{\alpha}(k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{\alpha}(k-\mu_{\min}(k)) \\ \boldsymbol{m}(k,k-\mu_{\max}(k)) \\ \vdots \\ \boldsymbol{m}(k,k-\mu_{\min}(k)) \end{bmatrix}. \quad (96)$$

The inequality constraint matrix $A_{(k,k-\mu)}(\boldsymbol{\phi}(k))$ is then defined in Equation 97.

$$A_{(k,k-\mu)}(\boldsymbol{\phi}(k)) = \begin{cases} A_{routing,(k)}(\boldsymbol{\phi}(k)) & \text{If } \mu = 0 \\ A_{ordering,(k,k-\mu)}(\boldsymbol{\phi}(k)) \oplus A_{synch,(k,k-\mu)}(\boldsymbol{\phi}(k)) & \text{Else} \end{cases} \quad (97)$$

The full set of arrival state evolution equations for vessel k then has the same definition as in Equation 30.

The arrival state evolution equations only evolve as a function of vessel counter k, whereas the bin height evolution equations evolve as a function of both the vessel counter k and the event counter g. Let $h(k,g)$ in Equation 98 be the vector of $\boldsymbol{h_{(i,j)}}(k,g)$ for all chambers for vessel k at event step g.

$$\boldsymbol{h}(k,g) = \left[\boldsymbol{h_{(i,j)}}(k,g) | \forall (i,j) \in \mathcal{D}_{locks}\right]^T. \quad (98)$$

The constraints in Equation 75, Equation 78, and Equation 80 link the entries in the bin height state vector $\boldsymbol{h}(k,g+1)$ at event step $g+1$ for vessel k to entries in the bin height state vector of itself or other vessels at the previous event step g. The constraint in Equation 79 links the bin height state vector $\boldsymbol{h}(k,g)$ at event step g for vessel k to the entries in the bin height state vector of itself or other vessels at the same event step. Furthermore, the constraint in Equation 73 step $g=0$ can also be written as an input to the system for every single event step, as the bin heights have to be larger than e on every event step if they start at e at the very first step.

Let $A_{placement,(k,k-\mu)}(\boldsymbol{\phi}(k))$ be a grouping of the constraints in Equation 75, Equation 78, and Equation 80 between vessel k and vessel $k-\mu$, with $\mu \in \mathcal{M}(k) \cup \{0\}$, thus including vessel k. Let $A_{binsynch,(k,k-\mu)}(\boldsymbol{\phi}(k))$ be a grouping of the constraint in Equation 79 between vessel k and vessel $k-\mu$, with $\mu \in \mathcal{M}(k)$, excluding vessel k, as a vessel cannot be synchronised with itself. Finally, let $B_{initial}$ be a square max-plus unit matrix of appropriate size for $\boldsymbol{h}(k,g+1)$, and let $\boldsymbol{e}$ be a vector of e of compatible size. Then the system equation for the bin height states of vessel k at step g can be written as in Equation 99.

$$\begin{aligned} \boldsymbol{h}(k,g+1) \geq & \bigoplus_{\mu \in \mathcal{M}(k) \cup \{0\}} A_{placement,(k,k-\mu)}(\boldsymbol{\phi}(k)) \otimes \boldsymbol{h}(k-\mu,g+1) \\ & \oplus \bigoplus_{\mu \in \mathcal{M}(k)} A_{binsynch,(k,k-\mu)}(\boldsymbol{\phi}(k)) \otimes \boldsymbol{h}(k-\mu,g+1) \\ & \oplus B_{initial} \otimes \boldsymbol{e}, \end{aligned} \quad (99)$$

where the constraint in Equation 73 is given as a constant system input.

The smaller-than-inequality constraint of the multi-vessel, multi-chamber placement model has been removed, so the allowable modes of the system can once again be described by gathering all of the equality constraints into Equation 31.

Note that this excludes the $\hat{h}_{(i,j)}(k)$ and $\check{h}_{(i,j)}(k)$ variables, which are auxiliary variables that are not a part of the SMPL system. Additionally, two more constraints will be added outside of the SMPL system framework.

4.3.3 MILP Constraints

Two more constraints that could not be added to the SMPL model because they constrain the height states to be smaller than some quantity must be added. They can just be added directly as MILP constraints.

First, Equation 100 constrains that the vessel stack heights in the bins of the chambers do not exceed the length of the chamber. For all $((i,j),k,g,\chi) \in \mathcal{D}_{locks} \times \mathcal{K} \times \{1,2,...,n_{steps}\} \times \{0,1,...N_{\chi,(i,j)}-1\}$:

$$h_{\chi,(i,j)}(k,g) \leq N_{\gamma,(i,j)}. \quad (100)$$

Second, Equation 101 and Equation 102 constrain that a vessel k that is moored to another vessel $k-\mu$ must be fully alongside vessel $k-\mu$. For all $((i,j),k,k-\mu) \in$

$\mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$\hat{h}_{(i,j)}(k) - \hat{h}_{(i,j)}(k-\mu) - \beta m_{(i,j)}(k, k-\mu) \leq -\beta \quad (101)$$

$$\check{h}_{(i,j)}(k-\mu) - \check{h}_{(i,j)}(k) - \beta m_{(i,j)}(k, k-\mu) \leq -\beta \quad (102)$$

Equation 73 is already useable as an MILP constraint.

Equation 75 is no longer convenient to write as a vector equation. Instead, the constraint is written separately for each entry of the $\boldsymbol{h}_{(i,j)}(k,g)$ vector in Equation 103. For all $(k, k-\mu, g, (i,j), \chi, p) \in \mathcal{K} \times \mathcal{K} \setminus \{k\} \times \{0,1,...,n_{steps}-1\} \times \mathcal{D}_{locks} \times \{0,1,...,N_{\chi,(i,j)}-1 | \chi + W_\chi(k-\mu) \leq N_{\chi,(i,j)}\} \times \{0,1,...,W_\chi(k-\mu)-1\}$:

$$h_{\chi,(i,j)}(k,g) - h_{\chi+p,(i,j)}(k,g+1) - \beta s_{(i,j)}(k,k-\mu) - \beta\alpha_{\chi,(i,j)}(k-\mu) \leq -2\beta - W_\gamma(k-\mu). \quad (103)$$

The same is done for Equation 78 in Equation 104. For all $(k,g,(i,j),\chi,p) \in \mathcal{K} \times \{0,1,...,n_{steps}-1\} \times \mathcal{D}_{locks} \times \{0,1,...,N_{\chi,(i,j)}-1 | \chi + W_\chi(k) \leq N_{\chi,(i,j)}\} \times \{0,1,...,W_\chi(k)-1\}$:

$$h_{\chi,(i,j)}(k,g) - h_{\chi+p,(i,j)}(k,g+1) - \beta\alpha_{\chi,(i,j)}(k-\mu) \leq -\beta - W_\gamma(k). \quad (104)$$

Equation 80 is converted to Equation 105. For all $((i,j),k,g,\chi) \in \mathcal{D}_{locks} \times \mathcal{K} \times \{0,1,...,n_{steps}-1\} \otimes \{0,1,...,N_{\chi,(i,j)}-1|\}$:

$$h_{\chi,(i,j)}(k,g) - h_{\chi,(i,j)}(k,g+1) \leq 0. \quad (105)$$

Equation 81 is converted to Equation 106 and Equation 107. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\sum_{\chi \in \{N_{\chi,(i,j)-1}\}} \sum_{g \in \{0,1,...,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g) - w_{(i,j)}(k) \leq 0 \quad (106)$$

$$-\sum_{\chi \in \{N_{\chi,(i,j)-1}\}} \sum_{g \in \{0,1,...,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g) + w_{(i,j)}(k) \leq 0 \quad (107)$$

Equation 82 is converted to Equation 108. For all $(k, k-\mu, (i,j), \chi, g) \in \mathcal{K} \times \mathcal{K} \setminus \{k\} \times \mathcal{D}_{locks} \times \{0,1,...,N_{\chi,(i,j)}\} \times \{0,1,...,n_{steps}-1\}$:

$$\alpha_{\chi,(i,j)}(k,g) + \sum_{p \in \{0,1,...,N_{\chi,(i,j)-1}\}} \alpha_{p,(i,j)}(k-\mu,g) + s_{(i,j)}(k,k-\mu) \leq 2 \quad (108)$$

The definition of the bow position variables $\hat{h}_{(i,j)}(k)$ as defined in Equation 83 requires a number of auxiliary variables and constraints. Define the auxiliary variable $t_{bow,\chi,(i,j)}(k,g)$ for all $(g,\chi) \in \{0,1,...,n_{steps}-1\} \times \{0,1,...,N_{\chi,(i,j)} - W_\chi(k)\}$:

$$t_{bow,\chi,(i,j)}(k,g) = \max(\{h_{\chi+p,(i,j)}(k,g) | \forall p \in \{0,1,...,W_\chi(k)-1\}\}), \quad (109)$$

which can be linearly enforced by introducing the binary variables $\lambda_{\chi+p,(i,j)}(k,g)$ and the constraints that for all $p \in \{0,1,...,W_\chi(k)-1\}$:

$$h_{\chi+p,(i,j)}(k,g) - t_{bow,\chi,(i,j)}(k,g) \leq 0 \quad (110)$$

$$t_{bow,(i,j),(i,j)}(k,g) - h_{\chi+p,(i,j)}(k,g) - \beta\lambda_{\chi+p,(i,j)}(k,g) \leq -\beta, \quad (111)$$

along with:

$$\sum_{p \in \{0,1,...,W_\chi(k)-1\}} \lambda_{\chi+p,(i,j)}(k,g) \leq 1 \quad (112)$$

$$-\sum_{p \in \{0,1,...,W_\chi(k)-1\}} \lambda_{\chi+p,(i,j)}(k,g) \leq -1. \quad (113)$$

Then define another auxiliary variable $\theta_{bow,\chi,(i,j)}(k,g)$ for all $(g,\chi) \in \{0,1,...,n_{steps}-1\} \times \{0,1,...,N_{\chi,(i,j)} - W_\chi(k)\}$:

$$\theta_{bow,\chi,(i,j)}(k,g) = \alpha_{\chi,(i,j)}(k,g) t_{bow,\chi,(i,j)}(k,g+1), \quad (114)$$

which is enforced through the following constraints:

$$\theta_{bow,\chi,(i,j)}(k,g) + \beta\alpha_{\chi,(i,j)}(k,g) \leq 0 \quad (115)$$

$$-\theta_{bow,\chi,(i,j)}(k,g) \leq 0 \quad (116)$$

$$\theta_{bow,\chi,(i,j)}(k,g) - t_{bow,\chi,(i,j)}(k,g) \leq 0 \quad (117)$$

$$-\theta_{bow,\chi,(i,j)}(k,g) + t_{bow,\chi,(i,j)}(k,g) - \alpha_{\chi,(i,j)}(k,g) \leq -\beta. \quad (118)$$

Finally, it can be defined that for all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\hat{h}_{(i,j)}(k) = \max(\{\theta_{bow,\chi,(i,j)}(k,g) | (g,\chi) \in \{0,1,...,n_{steps}-1\} \times \{0,1,...,N_{\chi,(i,j)}-1\}\}). \quad (119)$$

This is in turn enforced by introducing binary variable $\lambda_{\chi,(i,j)}(k,g)$ for each $\theta_{bow,\chi,(i,j)}(k,g)$ alongside the following constraints:

$$\theta_{bow,\chi,(i,j)}(k,g) - \hat{h}_{(i,j)}(k) \leq 0 \quad (120)$$

$$\hat{h}_{(i,j)}(k) - \theta_{bow,\chi,(i,j)}(k,g) - \beta\lambda_{\chi,(i,j)}(k,g) \leq \beta, \quad (121)$$

and:

$$\sum_{g \in \{0,1,...,n_{steps}-1\}} \sum_{\chi \in \{0,1,...,N_{\chi,(i,j)}-1\}} \lambda_{\chi,(i,j)}(k,g) \leq 1 \quad (122)$$

$$-\sum_{g \in \{0,1,...,n_{steps}-1\}} \sum_{\chi \in \{0,1,...,N_{\chi,(i,j)}-1\}} \lambda_{\chi,(i,j)}(k,g) \leq -1. \quad (123)$$

Equation 84 converts to Equation 124 and Equation 125. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\check{h}_{(i,j)}(k) - \hat{h}_{(i,j)}(k) \leq W_\gamma(k) \quad (124)$$

$$-\check{h}_{(i,j)}(k) + \hat{h}_{(i,j)}(k) \leq -W_\gamma(k) \quad (125)$$

Equation 85 converts to Equation 126. For all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \setminus \{k\}$:

$$m_{(i,j)}(k,k-\mu) - s_{(i,j)}(k,k-\mu) \leq 0 \quad (126)$$

Equation 86 converts to Equation 127. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\sum_{k-\mu \in \mathcal{K} \setminus \{k\}} m_{(i,j)}(k,k-\mu) \leq 1. \quad (127)$$

Equation 87 converts to Equation 128. For all

$((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}$:

$$\sum_{\chi \in \{1,2,\ldots,N_{\chi,(i,j)}-2 | \chi + W_\chi(k)+1 \leq N_{\chi,(i,j)}\}} \sum_{g \in \{0,1,\ldots,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g) - \sum_{k-\mu \in \mathcal{K} \setminus \{k\}} m_{(i,j)}(k,k-\mu) \leq 0 \quad (128)$$

The conversion of Equation 88 requires the introduction of a number of auxiliary variables and constraints. Define the binary variables $\psi_{\chi - W_\chi(k-\mu),(i,j)}(k,k-\mu)$ and $\psi_{\chi+W_\chi(k),(i,j)}(k,k-\mu)$. $\psi_{\chi - W_\chi(k-\mu),(i,j)}(k,k-\mu)$ only exists if $\chi - W_\chi(k-\mu) \geq 0$ for chamber (i,j). Similarly, $\psi_{\chi+W_\chi(k),(i,j)}(k,k-\mu)$ only exists if $\chi+W_\chi(k) \leq N_{\chi,(i,j)}$ for chamber (i,j).

$$\psi_{\chi - W_\chi(k-\mu),(i,j)}(k,k-\mu) = \left(\sum_{g \in \{0,1,\ldots,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g)\right) \cdot \left(\sum_{g \in \{0,1,\ldots,n_{steps}-1} \alpha_{\chi - W_\chi(k-\mu),(i,j)}(k-\mu,g)\right) \quad (129)$$

$$\psi_{\chi+W_\chi(k),(i,j)}(k,k-\mu) = \left(\sum_{g \in \{0,1,\ldots,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g)\right) \cdot \left(\sum_{g \in \{0,1,\ldots,n_{steps}-1} \alpha_{\chi+W_\chi(k),(i,j)}(k-\mu,g)\right) \quad (130)$$

The sums on the right-hand side of Equation 129 and Equation 130 can only ever be equal to 0 or 1, by virtue of Equation 106 and Equation 107, so their entirety can be treated as a binary variable. These relationships can be enforced by the following constraints:

$$-\left(\sum_{g \in \{0,1,\ldots,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g)\right) + \psi_{\chi - W_\chi(k-\mu),(i,j)}(k,k-\mu) \leq 0 \quad (131)$$

$$-\left(\sum_{g \in \{0,1,\ldots,n_{steps}-1} \alpha_{\chi - W_\chi(k-\mu),(i,j)}(k-\mu,g)\right) + \psi_{\chi - W_\chi(k-\mu),(i,j)}(k,k-\mu) \leq 0 \quad (132)$$

$$\left(\sum_{g \in \{0,1,\ldots,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g)\right) + \left(\sum_{g \in \{0,1,\ldots,n_{steps}-1} \alpha_{\chi - W_\chi(k-\mu),(i,j)}(k-\mu,g)\right) - \psi_{\chi - W_\chi(k-\mu),(i,j)}(k,k-\mu) \leq 1 \quad (133)$$

$$-\left(\sum_{g \in \{0,1,\ldots,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g)\right) + \psi_\chi W_{\chi+W_\chi(k),(i,j)}(k,k-\mu) \leq 0 \quad (134)$$

$$-\left(\sum_{g \in \{0,1,\ldots,n_{steps}-1} \alpha_{\chi+W_\chi(k),(i,j)}(k-\mu,g)\right) + \psi_\chi W_{\chi+W_\chi(k),(i,j)}(k,k-\mu) \leq 0 \quad (135)$$

$$\left(\sum_{g \in \{0,1,\ldots,n_{steps}-1\}} \alpha_{\chi,(i,j)}(k,g)\right) + \left(\sum_{g \in \{0,1,\ldots,n_{steps}-1} \alpha_{\chi+W_\chi(k),(i,j)}(k-\mu,g)\right) - \psi_{\chi+W_\chi(k),(i,j)}(k,k-\mu) \leq 1. \quad (136)$$

Then the following must hold for all $((i,j),k,k-\mu) \in \mathcal{D}_{locks} \times \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$m_{(i,j)}(k,k-\mu) - \sum_{\chi \in \{W_\chi(k-\mu),W_\chi(k-\mu)+1,\ldots,N_{\chi,(i,j)}-2 | \chi+W_\chi(k)+1 \leq N_{\chi,(i,j)}\}} \left(\psi_{\chi+W_\chi(k),(i,j)}(k,k-\mu) + \psi_{\chi-W_\chi(k-\mu),(i,j)}(k,k-\mu)\right) \leq 0. \quad (137)$$

Equation 89 converts to Equation 138. For all $((i,j),k,k-\mu) \in \mathcal{K} \times \mathcal{K} \setminus \{k\}$:

$$m_{(i,j)}(k,k-\mu) + m_{(i,j)}(k-\mu,k) \leq 1. \quad (138)$$

Equation 90 converts to Equation 139. For all $((i,j),k) \in \mathcal{D}_{locks} \times \mathcal{K}_\nabla$:

$$\sum_{k-\mu \in \mathcal{K} \setminus \{k\}} m_{(i,j)}(k,k-\mu) = 0 \quad (139)$$

Finally, Equation 91 converts to Equation 140. For all $k \in \mathcal{K}_{\nabla\nabla}$:

$$\sum_{(i,j) \in \mathcal{D}_{locks}} \sum_{k-\mu \in \mathcal{K} \setminus \{k\}} s_{(i,j)}(k,k-\mu) \leq 0. \quad (140)$$

4.4 Objectives

Two main optimization objectives are introduced. The first is the cumulative arrival time objective given in Equation 141. The cumulative travel time objective $J_\mathcal{A}$ penalizes the sum of the total arrival times of each of the vessels, encouraging them to arrive at their destination as quickly as possible.

$$J_\mathcal{A} = \sum_{k \in \mathcal{K}} \sigma_\mathcal{A}(k)(x_{d(k)}(k) - u(k)) \quad (141)$$

The second is the arrival time offset objective $J_\mathcal{H}$. This now penalizes the difference between the planned arrival time of a vessel at its final destination and the true arrival time of the vessel, as per Equation 142.

$$J_\mathcal{H} = \sum_{k \in \mathcal{K}} \sigma_\mathcal{H}(k)|\hat{x}_{d(k)} - x_{d(k)}| \quad (142)$$

This can be linearized by introducing the varaible $t_{\mathcal{H}}(k)$ in Equation 143.

$$t_{\mathcal{H}}(k) = |\hat{x}_{d(k)} - x_{d(k)}| \tag{143}$$

Equation 143 is enforced by Equation 144 and Equation 145.

$$\hat{x}_{d(k)} - x_{d(k)} - t_{\mathcal{H}}(k) \leq 0 \tag{144}$$
$$x_{d(k)} - \hat{x}_{d(k)} - t_{\mathcal{H}}(k) \leq 0 \tag{145}$$

Then the linear arrival time offset objective is written in Equation 146.

$$J_{\mathcal{H}} = \sum_{k \in \mathcal{K}} \sigma_{\mathcal{H}}(k) t_{\mathcal{H}}(k) \tag{146}$$

Different vessels can have different main objectives, but a vessel cannot have both main objectives at the same time. The objectives can be turned on and off for different vessels with the objective weights $\sigma_{\mathcal{A}}(k)$ and $\sigma_{\mathcal{H}}(k)$.

Two auxiliary objectives that are always active are also introduced. The cumulative departure time objective J_u in Equation 147 enforces that all vessels leave at their earliest departure time.

$$J_u = \sum_{k \in \mathcal{K}} x_{b(k)} \cdot 10^{-3}. \tag{147}$$

The waiting time objective J_Π enforces that vessels do not wait in the waiting areas, instead reducing speed on arcs that are not waiting areas in an effort to reduce fuel use. Define the continuous variable $t_{\Pi,(i,j)}(k)$ as in Equation 148.

$$t_{\Pi,(i,j)}(k) = \max(x_i(k) - x_{\delta(i)}(k) - \tau_{(i,\delta(i))}(k) + \beta(1 - w_{(i,j)}(k)), 0), \tag{148}$$

enforced by the constraints in Equation 149 and Equation 150.

$$t_{\Pi,(i,j)}(k) - x_i(k) + x_{\delta(i)}(k) + \beta(1 - w_{(i,j)}(k)) \leq -\tau_{(i,\delta(i))} \tag{149}$$
$$-t_{\Pi,(i,j)}(k) \leq 0 \tag{150}$$

Then the linear waiting time objective is written in Equation 151.

$$J_\Pi = \sum_{(i,j) \in \mathcal{D}_{locks}} \sum_{k \in \mathcal{K}} t_{\Pi,(i,j)}(k) \cdot 10^{-3}. \tag{151}$$

The full multi-criteria objective function J is given in Equation 152. The auxiliary objectives do not impact the main objective values.

$$J = J_{\mathcal{A}} + J_{\mathcal{H}} + J_u + J_\Pi \tag{152}$$

To penalize water usage, objectives penalizing the number of lockages taking place are often introduced in LSP literature. However, it is difficult to add a linear objective that can also take into account empty lockages for the model presented in this paper. A linear objective that does not account for empty lockages may be off by a factor of 2, so a number of lockages objective was not added.

4.5 Key Performance Indicators

A number of key performance indicators (KPIs) are also introduced to be able to analyse the schedule results.

The cumulative arrival time KPI $\mathcal{A}$ written in Equation 153 is similar to the cumulative arrival time objective.

$$\mathcal{A} = \sum_{k \in \mathcal{K}} (x_{d(k)} - u(k)). \tag{153}$$

The arrival time offset KPI $\mathcal{H}$ written in Equation 154 is similar to the arrival time offset objectives. It only considers vessels with an arrival time offset objective.

$$\mathcal{H} = \sum_{\{k \in \mathcal{K} | \sigma_{\mathcal{H}}(k) \neq 0\}} |\hat{x}_{d(k)}(k) - x_{d(k)}(k)|. \tag{154}$$

The number of lockages $\mathcal{L}$ KPI keeps track of the number of lockages that have taken place. Occupied and empty lockages are both counted. Multi-vessel lockages count as a single lockage.

Finally, the average delay KPI $\mathcal{T}$ gives an indication of how much a vessel has been delayed on its route by the presence of other vessels in the system. If a vessel travels as fast as possible on all arcs, its average delay is 0%. Vessels with the arrival time offset objective are not taken into account. For vessel k the delay $\mathcal{T}(k)$ is calculated with Equation 155.

$$\mathcal{T}(k) = 100\% \cdot \frac{\sum_{(i,j) \in \mathcal{D}} w_{(i,j)}(k)(x_j(k) - x_i(k))}{\sum_{(i,j) \in \mathcal{D}} w_{(i,j)}(k) \tau_{(i,j)}(k)} - 100\% \tag{155}$$

Then the average delay for all vessels is calculated with Equation 156.

$$\mathcal{T} = \sum_{\{k \in \mathcal{K} | \sigma_{\mathcal{H}}(k) = 0\}} \frac{\mathcal{T}(k)}{|\{k \in \mathcal{K} | \sigma_{\mathcal{H}}(k) = 0\}|}, \tag{156}$$

where $|\cdot|$ denotes the cardinality of a set.

5 Case Studies

Three case studies are performed to show the functioning of the models. These case studies are fictional and they should not be interpreted as validation for the models.

The case studies were solved with the Gurobi 9.1.2 solver with default settings on an Intel 4-Core i7-7700HQ 2.8GHz CPU.

5.1 Single-Vessel, Single-Chamber Model

The graph network used in the case study for the single-vessel, single-chamber model is depicted in Figure 9. The waterway has a length of 50 km and it has two cascaded single-chamber locks. The node parameters are listed in Table 2 and the arc parameters are listed in Table 3. The waterway is symmetry and the locks and nodes are evenly spaced.

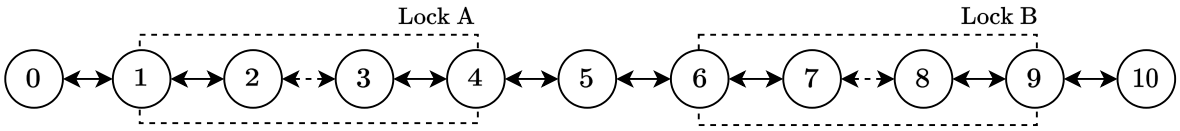


Figure 9: *Graph topology for the case study of the single-vessel, single-chamber model.*

Table 2: *Node parameters of the single-vessel, single chamber case study.*

Node i	$p_{x,i}$ $[km]$	$p_{y,i}$ $[km]$
0	0	0
Lock A $(1, 2, 3, 4)$	12.5	0
5	25	0
Lock B $(6, 7, 8, 9)$	37.5	0
10	50	0

Table 3: *Arc parameters of the single-vessel, single-chamber case study.*

Arc (i, j)	$d_{(i,j)}$ $[km]$	Specified $\tau_{(i,j)}(k)\,\forall k \in \mathcal{K}$ $[h]$
$(0, 1), (1, 0)$, $(4, 5), (5, 4)$	Euclidean	-
Lock A Waiting Areas $(1, 2), (2, 1)$, $(3, 4), (4, 3)$	-	0.1
Lock A Chamber $(2, 3), (3, 2)$	-	0.5
Lock B Waiting Areas $(6, 7), (7, 6)$, $(8, 9), (9, 8)$	-	0.1
Lock B Chamber $(7, 8), (8, 7)$	-	0.5

Eight vessels with a range of different start and destination nodes, maximum velocities, and departure times are scheduled. The parameter values of the vessels are listed in Table 4. The vessels depart at approximately the same times to create congestion at the locks. All vessels have the cumulative arrival time objective with a weight of $\sigma_{\mathcal{A}}(k) = 1$, except for vessel 5, which has the arrival time offset objective with a weight of $\sigma_{\mathcal{H}}(5) = 1$ and an arrival time of $\hat{x}_{d(5)}(5) = 7\ h$.

Table 4: *Vessel parameters of the single-vessel, single-chamber model case study.*

vessel k	$v_{\max,(i,j)}(k)$ $\forall (i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)$ $[h]$	$\hat{x}_{d(k)}$
0	10	0	10	0	-
1	10	10	0	0	-
2	12	0	5	1	-
3	8	5	0	0.5	-
4	8	5	10	1.5	-
5	7	10	5	2	7
6	8	0	10	1	-
7	11	10	0	0.5	-

The results are shown in the form of a journey plot in Figure 10. A journey plot has the possible arrival time states on the y-axis and time on the x-axis. The lines represent the journeys of each of the vessels. If a vessel has a marker at the y-position of a particular arrival time state, it visits the associated node on its route. If a line crosses the y-position of an arrival time state without a marker, the associated vessel does not visit that node.

The journey plot shows that all vessels correctly make their way from their departure node to their destination node, starting their journey at their designated minimum departure time, following connected routes, and being correctly ordered on the locks without any vessels' lockages overlapping.

Vessel 5, the vessel with the departure time offset objective, arrives at its destination at the correct time. Where possible, the locks are told to perform lockages in alternating directions, trying to prevent vessels having to wait for empty lockages, as that is generally faster. Without the ability to group vessels in combined lockages, alternating lockage directions is the main mechanism the scheduler has to reduce vessel travel times.

The KPIs are listed in Table 5. The average delay $\mathcal{T}$ is kept to a minimum because the vessels have diverse routes and because their minimum departure times are distributed over a longer period, so there is less chance for them to interfere with each other. $\mathcal{H}$ is $0\ h$, because vessel 5 arrives at its destination exactly on time.

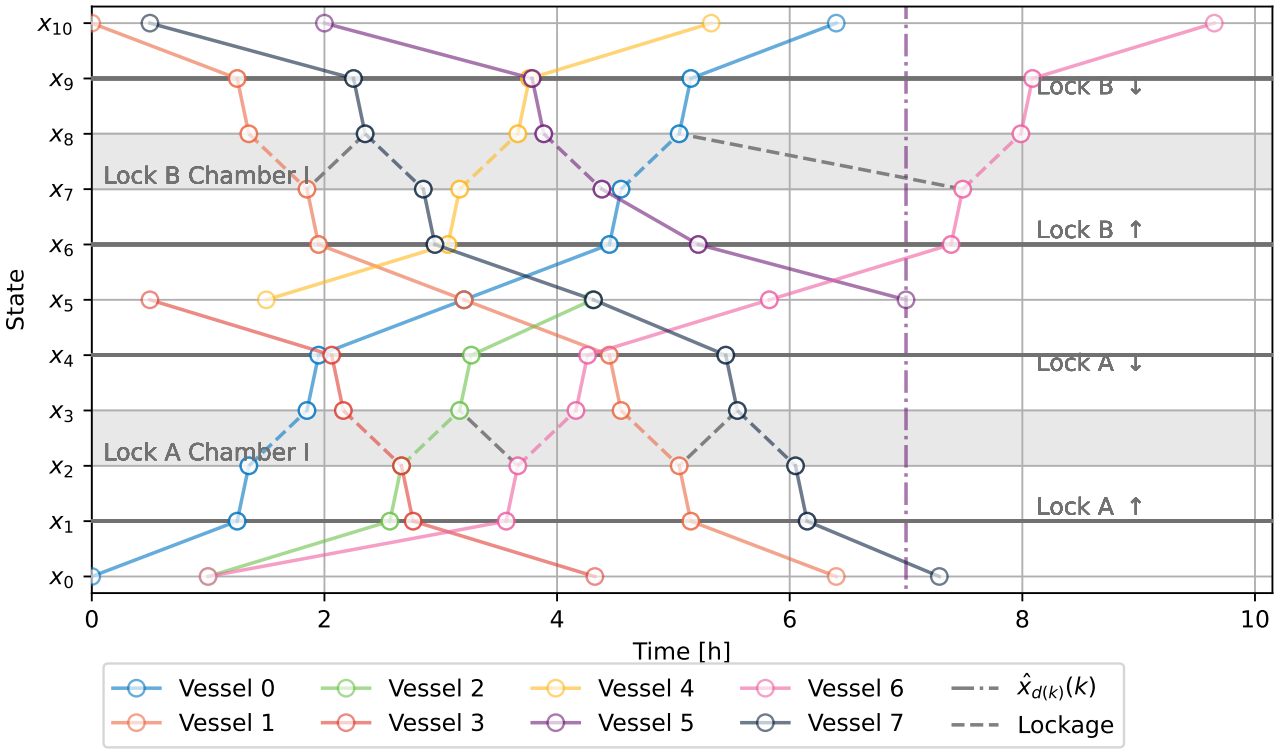


Figure 10: *Journey plot of the single-vessel, single-chamber model case study.*

Table 5: *KPIs single-vessel, single-chamber model case study.*

Performance Indicator	Value
J	39.20
$\mathcal{A}$	44.19 h
$\mathcal{H}$	0 h
$\mathcal{L}$	16
$\mathcal{T}$	6.56%

5.2 Multi-Vessel, Multi-Chamber Capacity Model

The graph network used in the case studies for the multi-vessel, multi-chamber models is depicted in Figure 11. It is similar to the setup used for the single-vessel, single chamber model, but lock A now has two chambers. Chamber I has less capacity than chamber II, but chamber II has a longer operation time. The parameters of the nodes are listed in Table 6 and the parameters of the arcs are listed in Table 7.

Ten vessels are scheduled. Their parameters are listed in Table 8. Their minimum departure times are all set close to 0 to make it more likely for multiple vessels to be grouped together in lockages.

The resulting schedule is shown in the journey plot in Figure 12. Multi-vessel lockages are indicated by overlaying tuples containing the vessel numbers in the lockage.

The journey plot shows that the parts of the model discussed for the single-vessel, single-chamber case study still work for the multi-vessel, multi-chamber capacity model. Additionally, vessels can now be synchronised into the same lockages as shown in the three multi-vessel lockages. These lockages do not exceed the chamber capacities. Furthermore, vessel 3 is correctly assigned to chamber II of lock A, because it would not fit into chamber I. The vessels are generally distributed across the chambers of lock A to prevent waiting time and the same pattern of trying to alternate lockage directions that was seen for the single-vessel, single-chamber case study is also observed.

The KPI values are listed in Table 9. The average delay is higher when compared to the single-vessel, single-chamber model value, even though vessels can now be grouped into lockages, because there are more vessels in the system and because the spread of their minimum departure times is smaller.

Table 6: *Node parameters of the nodes in Figure 11.*

Node i	$p_{x,i}$ $[km]$	$p_{y,i}$ $[km]$
0	0	0
Lock A	12.5	0
$(1, 2, 3, 4, 5, 6)$		
5	25	0
Lock B	37.5	0
$(6, 7, 8, 9)$		
10	50	0

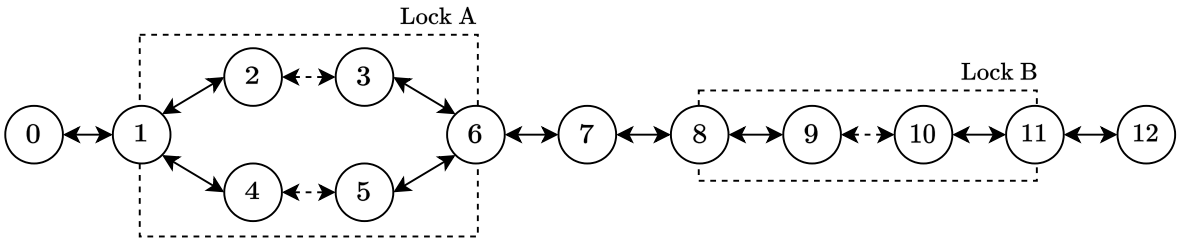


Figure 11: *Graph topology for the case studies of the multi-vessel, multi-chamber models.*

Table 7: *Arc parameters for the multi-vessel, multi-chamber case studies.*

Arc (i,j)	$d_{(i,j)}$ $[km]$	Specified $\tau_{(i,j)}(k)\,\forall k \in \mathcal{K}$ $[h]$	$N_{\chi,(i,j)}$	$N_{\gamma,(i,j)}$ $[m]$	$\overline{\tau}_{(i,j)}$ $[h]$	$c_{(i,j)}$
$(0,1),(1,0),(6,7),(7,6),$ $(7,8),(8,7),(11,12),(12,11)$	Euclidean	-	-	-	-	-
Lock A Waiting Areas $(1,2),(2,1),(1,4),(4,1),$ $(3,6),(6,3),(5,6),(6,5)$	-	0.1	-	-	-	-
Lock A Chamber I $(2,3),(3,2)$	-	0.5	3	150	0.1	5
Lock A Chamber II $(4,5),(5,4)$	-	0.75	2	200	0.1	8
Lock B Waiting Areas $(8,9),(9,8),(10,11),(11,10)$	-	0.1	-	-	-	-
Lock B Chamber I $(9,10),(10,9)$	-	1	4	250	0.1	10

Table 8: *Vessel parameters of the multi-vessel, multi-chamber capacity model case study.*

vessel k	$v_{\max,(i,j)}(k)$ $\forall(i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)$ $[h]$	$\rho(k)$
0	10	0	12	0	2
1	10	12	0	0.1	3
2	12	0	7	0.2	5
3	8	7	0	0.3	8
4	8	7	12	0.4	2
5	7	12	7	0.5	4
6	8	0	12	0.6	3
7	11	12	0	0.7	5
8	10	12	0	0.1	1
9	11	12	0	0.2	1

Table 9: *KPIs multi-vessel, multi-chamber capacity model case study.*

Performance Indicator	Value
J	63.62
$\mathcal{A}$	$63.62\,h$
$\mathcal{H}$	-
$\mathcal{L}$	16
$\mathcal{T}$	9.43%

5.3 Multi-Vessel, Multi-Chamber Placement Model

The graph network and the node and arc parameters used in the case study of the multi-vessel, multi-chamber placement model are listed in Figure 11, Table 6, and Table 7. The number of bins in the chambers has been kept low to keep the number of placement variables down, but the chambers with 3 and 4 bins still have the opportunity for vessel-to-vessel mooring to take place with bins that are not adjacent to the quay.

The vessel parameters are listed in Table 10. The vessels now all have a minimum departure time of 0 and the same travel direction, a range of maximum velocities, widths and lengths. This should create congestion, resulting in difficult placement scenarios. Vessels in the other travel direction are not considered, because the computational burden becomes too high past 10 vessels with these parameters. Vessel 2 is carrying explosive goods, so it may not be in the same lockages as any other vessels. Vessel 9 is carrying flammable goods, so it may not be moored to any other vessels.

The journey plot of the case study is depicted in Figure 13. This case study is designed to simulate a scenario that the multi-vessel, multi-chamber models are expected to be better at than the current FCFS system in place in practice, namely considering groupings of vessels passing through multiple cascaded locks, because the scheduler can take into account all vessels and locks globally, while an operator may only be looking at a single lock, or

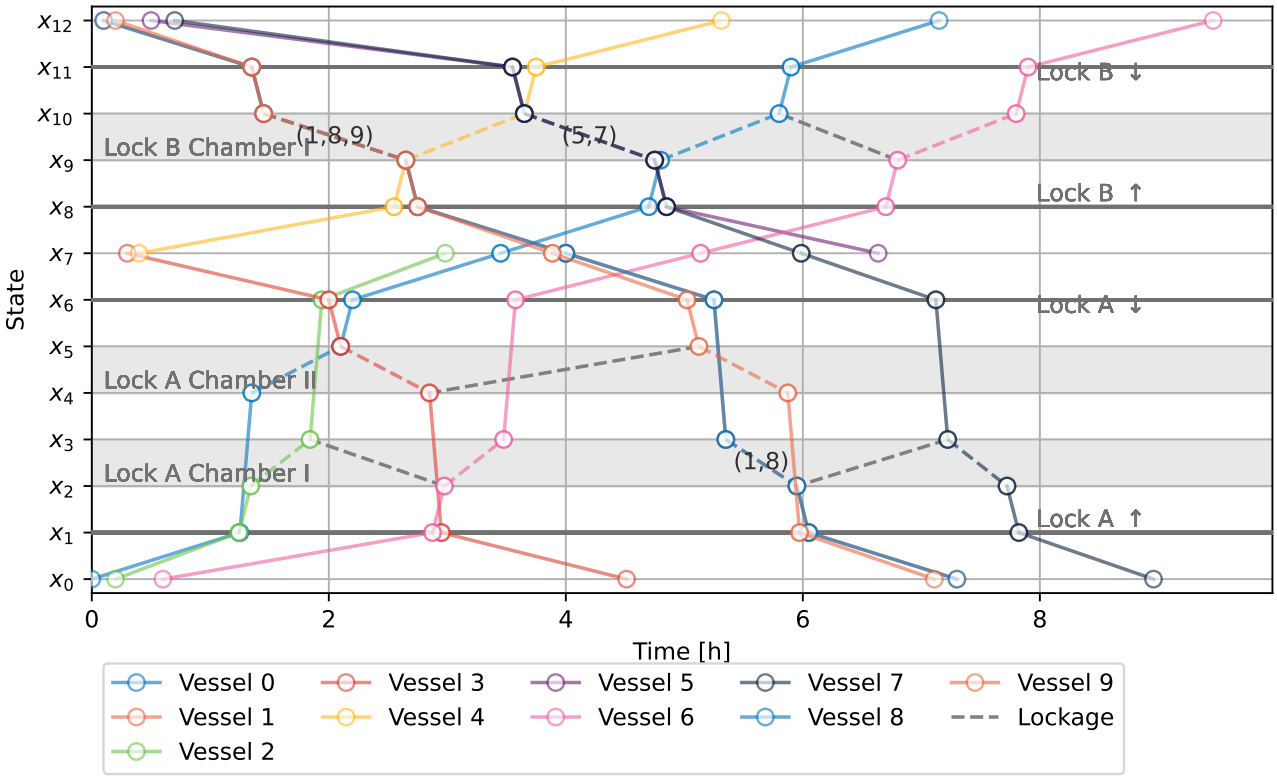


Figure 12: *Journey plot of the multi-vessel, multi-chamber capacity model case study.*

vessels close to the lock.

The vessels are equally distributed over 4 lockages on the chambers of lock A and then the vessels from different lockages come together on two lockages on the single chamber of lock B. Vessel 2 is the outlier, because its explosive goods require it to pass through the lock chambers on its own.

The different lockage configurations are shown in Figure 14. They show consistent behavior with the placement constraints, mooring constraints, and chamber and vessel dimensions. Vessel 8 is correctly moored to vessel 4 in Figure 14d.

Table 11: *KPIs multi-vessel, multi-chamber placement model case study.*

Performance Indicator	Value
J	91.23
$\mathcal{A}$	91.23 h
$\mathcal{H}$	-
$\mathcal{L}$	13
$\mathcal{T}$	23.34%

5.4 Complexity Analysis

As the model progressed from the single-vessel, single-chamber model to the multi-vessel, multi-chamber model, a large number of variables, particularly binary variables, were added.

The main factor in computational difficulty for MILPs is the number of integer or binary variables [23]. A secondary factor is the structure of the underlying problem and model. Finally, the number of constraints impacts how long it takes to solve LP relaxations when investigating nodes in the branch-and-bound method.

It is beyond the scope of this paper to analyse the structure of the problem. However, an analysis can be run on the number of binary variables, constraints, and the solution time of a simple example scenario to analyse the complexity of the different models.

The complexity tests on the different models will use the same simple single-lock, single-chamber symmetric graph topology as depicted in Figure 15. The node parameters are listed in Table 12 and the arc parameters are listed in Table 13.

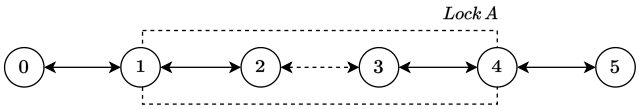


Figure 15: *Topology graph for the tests in the complexity analysis.*

Table 12: *Node parameters of the topology graph used in the complexity analysis.*

Node i	$p_{x,i}$ $[km]$	$p_{y,i}$ $[km]$
0	0	0
Lock A $(1, 2, 3, 4)$	10	0
5	20	0

The vessels are randomly generated, with the ranges

Table 10: *Vessel parameters of the case study for the multi-vessel, multi-chamber placement model.*

vessel k	$v_{\max,(i,j)}(k)\ \forall(i,j) \in \mathcal{D}$ $[km/h]$	$b(k)$	$d(k)$	$u(k)\ [h]$	W_χ	$W_\gamma\ [m]$	$\triangledown$	$\triangledown\triangledown$
0	10	0	12	0	1	100		
1	10	0	12	0	2	150		
2	12	0	12	0	1	100		✓
3	8	0	12	0	1	75		
4	8	0	12	0	1	200		
5	7	0	12	0	1	125		
6	8	0	12	0	1	50		
7	11	0	12	0	1	50		
8	10	0	12	0	2	175		
9	11	0	12	0	1	100	✓	

Table 13: *Arc parameters for the complexity tests.*

Arc (i,j)	$d_{(i,j)}\ [km]$	Specified $\tau_{(i,j)}(k)\,\forall k \in \mathcal{K}$ $[h]$	$\rho(k)$	$N_{\chi,(i,j)}$	$N_{\gamma,(i,j)}\ [m]$	$\overline{\tau}_{(i,j)}\ [h]$
$(0,1),(1,0)$, $(4,5),(5,4)$	Euclidean	-	-	-	-	-
Lock A Waiting Areas $(1,2),(2,1)$, $(3,4),(4,3)$	-	0.1	-	-	-	-
Lock A Chamber $(2,3),(3,2)$	-	0.5	8	4	200	0.1

of possible parameter values listed in Table 14. These parameters are chosen to create roughly equivalent scenarios between the tests of the different models, where possible. Full equivalency is not possible, nor is it required to make a qualitative assessment of the complexity of the models.

Table 14: *Randomized parameters for each of the vessels in the complexity tests. For a range, the parameter is selected uniformly between the start and endpoints of the range. For sets, one entry of the set is chosen.*

Parameter	Range	Set
$v_{\max,(i,j)}(k)$ $\forall(i,j) \in \mathcal{D}$	-	$\{8,9,10,11,12,13,14,15\}$
$(b(k),d(k))$	-	$\{(0,5),(5,0)\}$
$u(k)$	$[0,2]$	-
$\rho(k)$	-	$\{1,2,3,4,5,6,7,8\}$
$W_\chi(k)$	-	$\{1,2,3,4\}$
$W_\gamma(k)$	-	$\{50,100,150,200\}$

The number of vessels will be the dominant factor in determining the number of binary variables, as the majority of the binary variables is expected to scale with $n_{vessels}^2$ if $n_{steps} = n_{vessels}$. Tests are run for cases with $2-15$ vessels for each of the models, with each distinct number of vessels having 5 tests per model to reduce the solution time variability that may be present due to the nature of branch-and-bound MILP solvers.

Log-log plots of the number of constraints and binary variables versus the number of vessels are plotted in Figure 16 and Figure 17, respectively. SVSC stands for single-vessel, single-chamber, MVMCC stands for multi-vessel, multi-chamber capacity, and MVMCP stands for multi-vessel, multi-chamber placement. Note that the MVMCP model is the only model for which the number of constraints and variables depends on the parameters of the vessels, so that is why its lines are not as straight as the lines for the other models.

There is a clear jump of almost an order of magnitude in complexity from the SVSC model to the MVMCP model. The jump from the SVSC model to the MVMCC model is smaller, visualizing the benefit of only considering one-dimensional sizes, simply because fewer new constraints and binary variables were added to create the MVMCC model.

A log plot of the computation time of the tests versus the number of vessels is plotted in Figure 18. A cutoff time of 3600 seconds was set. The solid lines indicate the average values, while the translucent areas indicate the range of achieved values.

The results show that for all three models, in this particular example, the computation times rise exponentially as a function of the number of vessels, with all three models going towards an hour of computation time for 15 vessels. It is expected that, because of the sheer number of

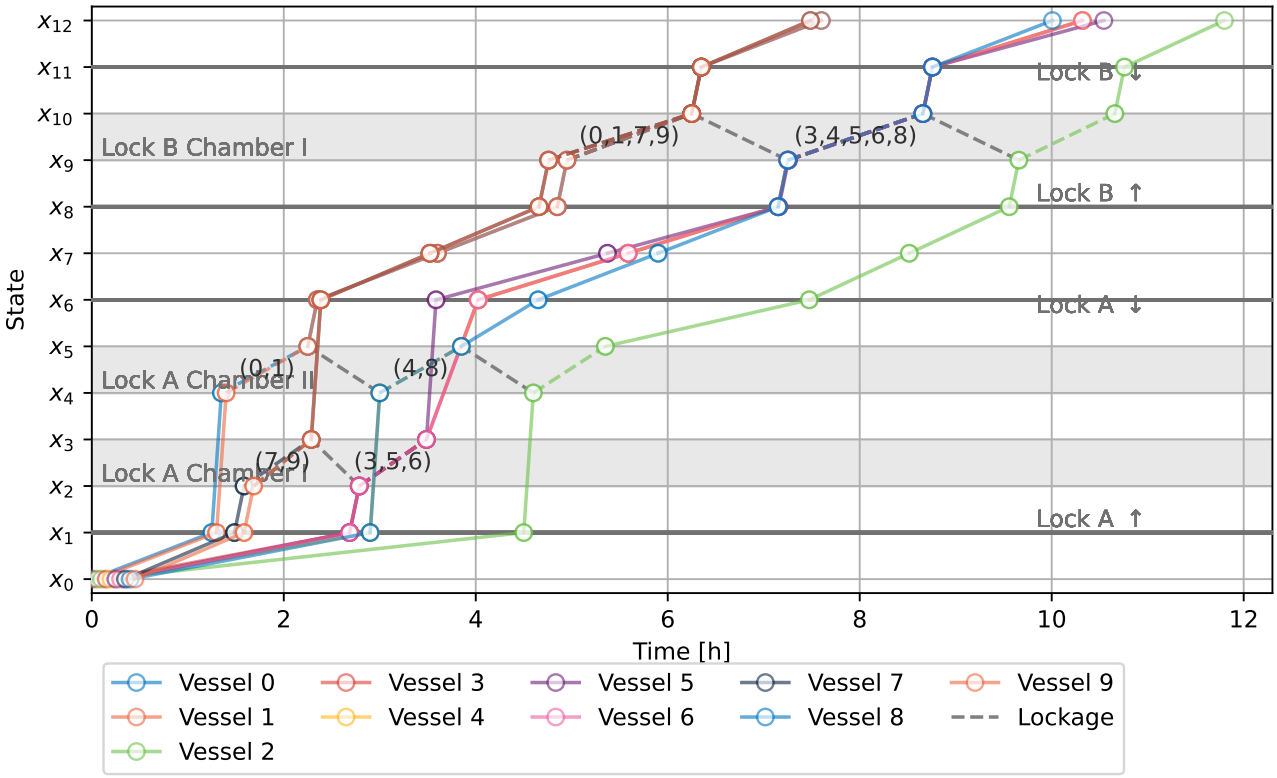


Figure 13: *Journey plot of the multi-vessel, multi-chamber placement model case study.*

binary variables involved with 15 vessels no matter the input, the result will not be different for other topology graphs and vessel parameters.

A real regional or national network may require hundreds of vessels to be scheduled simultaneously, so it will quickly become impractical to employ these models. To achieve an acceptable computation time of less than 15-20 minutes, a different approach like distributed optimization or heuristics will be required.

A minor observation that can be made is that the MVMCP model seems to be as fast or faster than the SVSC model in these test cases. It can be speculated that this is because of the single-lock, single-chamber structure of the problem.

For the SVSC model, the lock will become congested no matter what. When multiple vessels are approaching a congested single-vessel lock, it does not really matter which vessel gets to go first, as it always results in all of the other vessels having to wait. As a result, there are many different ordering scenarios with similar objective near-optimal function values, making it harder the prune solution branches in the branch-and-bound algorithm.

For the MVMCC model, there is more freedom in grouping the vessels on the single lock chamber, but there will be much fewer feasible groupings and only a few will be near-optimal, making it much easier to prune branches and find a solution in a short amount of time.

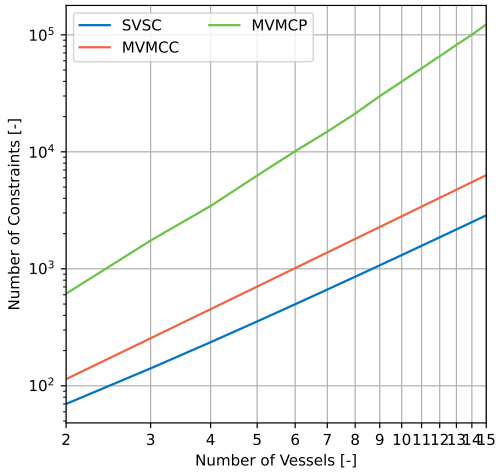


Figure 16: *Average number of constraints for each of the models in the complexity test.*

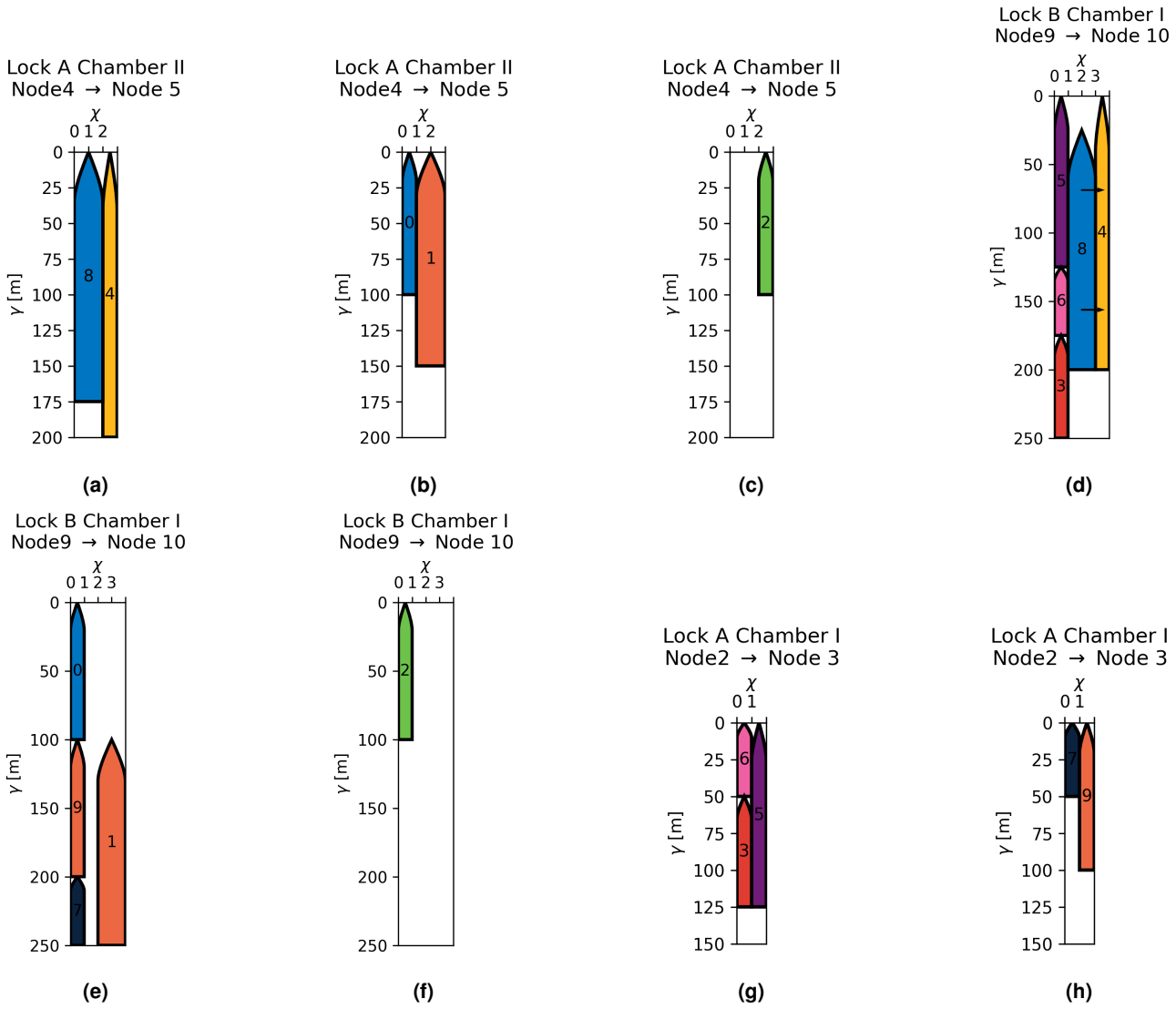


Figure 14: *Lockages of the multi-vessel, multi-chamber placement model case study.*

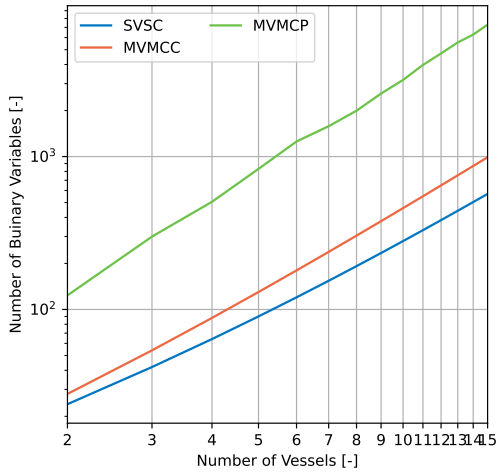


Figure 17: *Average number of binary variables for each of the models in the complexity test.*

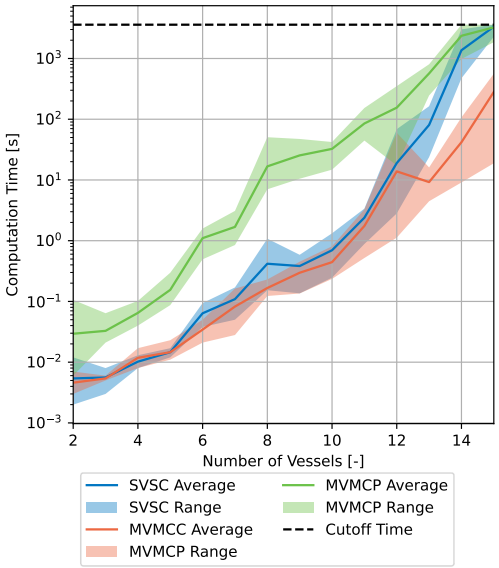


Figure 18: *Computation times of the complexity tests of the models. The cutoff time is 3600 seconds.*

6 Conclusion and Future Research

This paper introduced three SMPL scheduling models for the offline scheduling of IWT vessels passing through arbitrary waterway networks with multi-vessel, multi-chamber locks. Each model was an evolution of the previous model.

The first was built based on a previously established SMPL scheduling model for single-vessel, single-chamber locks [20], adding support for arbitrary network topologies.

The second introduced multi-vessel, multi-chamber locks, by adding multi-chamber support to the graph input structure for locks and simplifying vessel placement by assigning one-dimensional sizes to vessels and one-dimensional capacities to locks.

The third integrated true two-dimensional vessel placement into the multi-vessel, multi-chamber scheduling model by introducing a sliding vessel placement sequence modeled as an SMPL system, analogous to SMPL implementations of Tetris [18]. Mooring constraints, safety distances between vessels, and special rules for vessels with flammable, explosive, or toxic goods were also added.

The system constraints for each of the models were first written as SMPL constraints. These were then converted to MILP constraints, so they can be used by MILP solvers. Main objectives minimizing arrival time or arrival time offset were added in a multi-criteria objective function. The MILP constraints of the different models and the objectives were implemented into a scheduler, that can automatically create and solve optimization problems that schedule vessel arrival times, and placement positions in the chambers for the multi-vessel, multi-chamber placement model, based on the given topology graph and vessel inputs.

Case studies were performed to verify the performance of the models. Each of the models showed to be working correctly.

A complexity analysis was performed on a simple waterway with a one-chamber lock, to observe the growth of model complexity, and thus computation time, as a function of the number of vessels in the system. This showed that the SVSC and MVMCC models were quite close in complexity, while the MVMCP model was almost an order of magnitude higher in complexity.

That same pattern was observed in the computation times. However, even though the MVMCP computation times were higher, it was clear that none of the models would achieve practical computation times at instances of 15+ vessels.

It is recommended that future research explores distributed optimization and online optimization, and performs validation with real data.

The computation times of the global optimization problems presented here must be reduced for practical applications. Distributed optimization has worked for similar SMPL train scheduling systems in the past [17], significantly reducing computation times.

The scheduling performed in this paper was offline scheduling, with the assumption that perfect predictions of the system can be made and that the system will collaborate with the solution. This assumption will not hold in a practical application, where any disturbance or deviation from the schedule may significantly disrupt the system. A solution to this could be to perform online optimization, recalculating the schedule with shorter time intervals.

Finally, the case studies performed in this paper, while using semi-realistic parameter values, were purely fictional and for verification purposes only. Future research will have to validate the performance of the models in real scenarios, comparing the scheduling models to practice.

References

[1] European Environment Agency, *Rail and waterborne best for low-carbon motorised transport*, https://www.eea.europa.eu/publications/rail-and-waterborne-transport, 2021. DOI: 10.2800/85117.

[2] European Commission, "European transport policy for 2010: Time to decide," European Commission, Brussels, COM(2001) 370, 2001.

[3] Eurostat, *Modal split of air, sea and inland freight transport*, https://ec.europa.eu/eurostat/databrowser/view/tran_hv_ms_frmod/default/table?lang=en.

[4] E. Soveges, M. Stefanov, P. Puricella, *et al.*, "Inland waterway transport in Europe: No significant improvements in modal share and navigability conditions since 2001," European Court of Auditors, Luxembourg, Luxembourg, 01 /2015, 2015. DOI: doi:10.2865/158305.

[5] X. Wang, Y. Zhao, P. Sun, and X. Wang, "An analysis on convergence of data-driven approach to ship-lock scheduling.," *Mathematics and Computers in Simulation*, vol. 88, pp. 31–38, 2013. DOI: 10.1016/j.matcom.2013.03.005.

[6] J. Hermans, "Optimization of inland shipping - a polynomial time algorithm for the single ship single lock optimization problem," *Journal of Scheduling*, vol. 17, pp. 305–319, 4 2014. DOI: 10.1007/s10951-013-0364-7.

[7] W. Passchyn, S. Coene, D. Briskorn, J. L. Hurink, F. C. R. Spieksma, and G. Vanden Berghe, "The lockmaster's problem," *European Journal of Operational Research*, vol. 251, pp. 432–441, 2016. DOI: 10.1016/j.ejor.2015.12.007.

[8] J. Verstichel, P. De Causmaecker, and G. Vanden Berghe, "Scheduling algorithms for the lock scheduling problem," *Procedia - Social and Behavioural Sciences*, vol. 20, pp. 806–815, 2011. DOI: 10.1016/j.sbspro.2011.08.089.

[9] J. Verstichel, "The lock scheduling problem," Ph.D. dissertation, KU Leuven - Faculty of Engineering Science, Heverlee, Belgium, Nov. 2013.

[10] M. Prandtstetter, U. Ritzinger, P. Schmidt, and M. Ruthmair, "A variable neighborhood search approach for the interdependent lock scheduling problem," in *Evolutionary Computation in Combinatorial Optimization*. Switzerland: Springer, 2015, pp. 36–60.

[11] B. Ji, D. Zhang, S. S. Yu, and X. Fang, "An exact approach to the generalized serial-lock scheduling problem from a flexible job-shop scheduling perspective," *Computers and Operations Research*, vol. 127, 2021. DOI: 10.1016/j.cor.2020.105164.

[12] B. Ji, D. Zhang, S. S. Yu, and B. Zhang, "Optimally solving the generalized serial-lock scheduling problem from a graph-theory-based multi-commodity network perspective," *European Journal of Operational Research*, vol. 288, pp. 47–62, 1 2021. DOI: 10.1016/j.ejor.2020.05.035.

[13] X. Yuan, B. Ji, X. Wu, and X. Zhang, "Co-scheduling of lock and water-land transshipment for ships passing the dam," *Applied Soft Computing*, vol. 45, Pages 150–162, 2016. DOI: 10.1016/j.asoc.2016.04.019.

[14] B. Ji, H. Sun, X. Yuan, Y. Yuan, and X. Wang, "Coordinated optimized scheduling of locks and transshipment in inland waterway transportation using binary NSGA-II," *International Transactions in Operational Research*, vol. 27, pp. 1501–1525, 2020. DOI: 10.1111/itor.12720.

[15] X. Zhao, Q. Lin, and H. Yu, "A co-scheduling problem of ship lift and ship lock at the Three Gorges Dam," *IEEE Access*, vol. 8, pp. 132 893–132 910, 2020. DOI: 10.1109/ACCESS.2020.3009775.

[16] M. Pesselse, "Modelling and optimal scheduling of inland waterway transport systems," M.S. thesis, Delft University of Technology, Delft, The Netherlands, 2022.

[17] B. Kersbergen, J. Rudan, T. van den Boom, and B. De Schutter, "Towards railway traffic management using switching max-plus-linear systems," *Discrete Event Dynamic Systems*, vol. 26, pp. 183–223, 2014. DOI: 10.1007/s10626-014-0205-7.

[18] L. B. Bartels, S. A. Dambruin, H. M. van der Nagel, and S. W. J. Terwindt, *The relation between the computer game tetris and max-plus scheduling (of industrial and logistic systems)*, Delft Mechanical Engineering BSc End Project, 2021.

[19] B. Heidergott, G. J. Olsder, and J. van der Woude, *Max Plus at Work*. Oxford, United Kingdom: Princeton University Press, 2005.

[20] M. Pesselse, T. van den Boom, and V. Reppa, "Scheduling inland waterway transport vessels and locks using a switching max-plus-linear systems approach," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 3, pp. 748–762, 2022. DOI: 10.1109/OJITS.2022.3218334.

[21] Dutch Government, *Binnenvaartpolitiereglement*, https://wetten.overheid.nl/BWBR0003628/2017-01-01, 2017.

[22] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, 1999.

[23] F. S. Hillier and G. J. Lieberman, *Introduction to Operations Research*. New York: McGraw-Hill, 2021, ch. 12, ISBN: 978-1-260-57587-3.