LOSCO: A Ship Lock Scheduling Model

W.T.C van Adrichem

Delft University of Technology
Systems Navigator

June 2020

TU Delft SYSTEMS NAVIGATOR

LOSCO: A Ship Lock Scheduling Model

by

Wouter Theodorus Cornelis van Adrichem

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on June 10 2020 at 13:00.

Student number:	4215397	
Thesis committee:	Prof. dr. ir. M. van Koningsveld,	TU Delft, Chairman
	Ir. H.J. Verheij,	TU Delft
	Ir. W.F. Molenaar,	TU Delft
	Ir. A.P.G. Macquart,	Systems Navigator

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

Preface

This thesis is about the scheduling of shipping locks, this subject has not always been obvious. The first months of the process was dedicated to find the subject itself. Once the subject got clear, the project became more motivating for me, as I was learning something new. The subject of optimisation and especially in the context of NP problems has been new to me. The phase of programming was most enjoyable to me in the process of this thesis. I realized more that the core of my motivation is about solving some sort of puzzle. Of course there are many things that I would have done differently in hindsight, things that I can now regard as lessons. I would for instance have asked more questions to people instead of trying to solve everything by myself. I would have talked to more and different kinds of people.

I am very thankful to the people of Systems Navigator for giving me the opportunity to experience working at their company and the sharing of their knowledge. In particular I am thankful to Aubin Macquart, who always took the time for me to discuss ideas and helped me advance my thesis. I am also thankful to Henk Verheij, who spend a lot of time reading through my texts that were far from finished. My family and housemates were indispensable as they helped me going through the more emotional aspects of the largest project I have done to date.

Wouter van Adrichem,

Delft, June 2020

Abstract

A ship lock scheduling model called LOSCO was developed in order to decrease passage times at locks with two parallel chambers. Three locks were chosen to model, as they are some of the busiest locks of the Netherlands. The Krammersluizen, the Sluizen Hansweert and the Kreekraksluizen. Passage times are not collected at the locks, therefore the model was compared to SIVAK. SIVAK is the standard model for research on locks at Rijkswaterstaat. To schedule vessels in locks three types of choices can be made. Every vessel needs to be assigned to a chamber, the initiation time of locking should be decided and the order of sailing in a chamber should be defined. By optimising these choices, passage times can be decreased. Optimising is however not straightforward, as the problem is an instance of the job shop scheduling problem. No exact algorithm has been found to solve these problems in a practical amount of time. Therefore the challange is to find a balance in solution quality and the speed of the model when creating a lock scheduling model.

In this thesis, four ideas to improve the lock scheduling model by Verstichel were researched. The first idea is to change the resolution of the timesteps. The gain in performance was however not found to weight up against the loss in solution quality for resolution to be useful. The second idea is to drop the first come, first serve constraint that Verstichel created. This idea was also not found to be effective. The third idea is to divide the scheduling problem up into chunks. This is called cut separation. Chunks of around 25 vessels were found to be effective. The fourth and last idea is that of a maximum waiting time. The maximum waiting time makes the performance of the model better and also makes the scheduling model fairer. Data from the year 2016 was used to schedule all lockings. The LOSCO model is effective for reducing the average passage time per vessel with about 3.9 ± 0.12 SE minutes for the Krammersluizen and 2.0 ± 0.14 SE minutes for the Sluizen Hansweert compared to the SIVAK model. At the Kreekraksluizen, the model could not find a solution, as the Kreekraksluizen are a lot busier than the other locks that were tested. At the Kreekraksluizen on average every 7.8 minutes a vessel arrives, whereas at the Krammersluizen and the Sluizen Hansweert respectively every 14.0 and 13.0 minutes a vessel arrives throughout the year. The LOSCO model is only better than SIVAK if the lock is relatively quiet. At the busiest time of the day, typically in the afternoon, SIVAK performs better. The models perform equal at inter arrival times of around 8 to 10 minutes. Optimisation on economical value of the vessels was found to be less effective than optimisation on time. Optimisation on time was also found to be fairer.

The LOSCO model is a step ahead towards a practical lock scheduling model. In order to achieve a fully practical model, some simplifications need to be expanded. It is recommended to first improve the model before it is applied in practice, as the model is able to outperform SIVAK in some cases, but not in the busiest cases. After this some extra features can be implemented, such as the model dealing with vessel delays and locks with 3 chambers.

Contents

1

Introduction

1.1. Context

In the last 50 years, the distance between producers and consumers was multifold. This is a result of economies of scale and improved transportation methods. Enormous vessels ship goods overseas. Oil tankers, bulk carriers and container vessels are now bigger than they have ever been. These kinds of vessels cannot sail inland, so the goods are transported to their inland destination by inland vessel, truck or train. The share of the total inland transport by each of these modalities is called the modal split. Governmental institutions try to establish a modal shift towards IWT, because IWT is the cheapest, the most environmentally friendly (Pauli, 2010) and the safest way of inland transport and congestion of roads is a growing concern. The Rotterdam Port Authority set a goal to increase this percentage from 35% to 45% for 2035 (Rotterdam Port Authority, 2014).

Besides navigability for the transport of goods, the waterways also serve other functions. The waterways also serve as a source of fresh water, which requires the water to be clean and protected against salinisation from sea. Flood risk management is an important function of waterways in the Dutch situation, risk of floods from the sea as well as from extreme river discharges. Water quality is important for social and economic functions of the waterways.

Shipping locks are structures that can provide multiple of functions required for waterways. The main function of shipping locks is to help ships overcome water level differences. A lock also has the function of flood defence and is often part of road infrastructure. Another function of shipping locks is water management. For instance to prevent salt intrusion or to minimise the loss of water in a canal (Rijkswaterstaat, 2000). Dutch shipping locks are often low in elevation difference, with the exception of the south of the Netherlands, where more elevation differences exist. Shipping locks exist in many different sizes for all kinds of different ships. The busiest shipping locks are often characterized by multiple chambers for inland vessels and sometimes also separate chambers for recreational vessels.

Decisions at the major locks in the Netherlands are currently made by a human lock operator. This operator operates from a tower next to the lock, from where he can get an overview of approaching vessels. He is also in contact with the vessels

using radio communication. Based on the information available he makes the decisions at the lock. In essence there are three types of decisions that the operator makes. Each vessel should be assigned to one of the chambers. Per chamber the times of initiation of lock should be decided and the order that the vessels sail in should be determined. These decisions are not based on a strict set of rules. Moreover, in practice no mathematics are applied for this mathematical problem.

One way for governments to increase the modal split is by creating an economic incentive through improving the IWT system. This thesis explores an approach to optimise locking operations as a means to provide this economic incentive. This can make IWT cheaper, faster and more reliable. Locks often constraint the capacity of a waterway. Optimization of locking operations is low in cost, compared to system intervention (building or expanding locks or waterways).

The introduction of River Information Services (RIS) gives the opportunity to improve the locking operations. Vessels are now obliged to share information about their route and cargo electronically (IVS Next). With this information lock masters can anticipate. Schedules can be created for locking cycles. Lock scheduling can increase the capacity of shipping locks, and consequently the capacity of the waterways. Scheduling will also make IWT more reliable, as arrival times can be estimated better, so that vessels can in their turn anticipate better on the lock.

1.2. Objective and Research Questions

The objective of this study is to create a model that can minimise the total passage time by scheduling vessels at shipping locks with parallel chambers. Models of this type are known to be very computationally expensive. Therefore the objective can only be obtained by balancing the runtime of the model with the solution quality. This can be achieved with heuristic search methods. Last, an optimisation on economical value is executed, as larger vessels are at a potential disadvantage if passage time is minimised and because this might support the economy better.

The objective can be translated in the following main research question:

How can a model be made that schedules vessels in locks with parallel chambers with the objective to minimise the summation of the total passage time over all vessels?

An answer on the main research question can be obtained by answering the following sub questions:

1. *What modelling method can best be applied to model a shipping lock, with 2 chambers in parallel, in which typically multiple vessels fit, for the use of lock scheduling?*
2. *At which locks is sufficient data collected to use for a lock scheduling model?*
3. *How can the collected data be used for a lock scheduling model?*
4. *With which methods can a lock scheduling model be created, that can keep the runtime within a practical limit whilst obtaining the best possible solution quality?*
5. *How can the lock scheduling model be verified and validated?*

6. *How does the lock scheduling model perform in minimising vessel passage time compared to a baseline?*

7. *Is it better to minimise economical value rather than minimising vessel passage time, as this serves the economy better?*

8. *Is it fairer for larger vessels to minimise economical value rather than passage time?*

1.3. Scope

The scope of this thesis is restricted to locks with two parallel chambers in which multiple vessels are processed. Locks in sequence are not considered. In practice this means that main areas of interest are Europe and China. In these places there are locks on the inland waterways with those properties. Shipping locks on the rivers of the Americas do mostly not process multiple ships at a time, which is also the case for most sea locks.

The locks of interest are among the largest in the Netherlands, the Krammersluizen, the Sluizen Hansweert and the Kreekraksluizen. All three locks have 2 parallel chambers of similar size. All chambers have a width of 24 meter. At the Krammersluizen and the Sluizen Hansweert, the locks have a length of 280 meter and at the Kreekraksluizen the length of the lock is 318 meter.

No passage times are collected at the locks of interest, instead SIVAK is used to translate arrival times in passage times to compare the LOSCO model with.

Recreational vessels are not included in the scope. That means that separate locks for recreational vessels are ignored and recreational vessels are filtered from the by Rijkswaterstaat obtained data sets. The scope is restricted to the professional fleet. This can be done as recreational vessels are in practise of lower priority, they are only added to the chamber if there is space left that cannot be used for professional vessels.

This study is not about lock design, the design of the locks is unaltered. Faster leveling, faster doors or other design measures could also result in lower passage times. There are only three things that can be changed in this study to achieve lower passage times, vessels can be assigned to other chambers, the time of initiation of locking can be altered and the order of vessels to sail in the chambers can be altered. To simplify the problem, lock enter times are always taken as 2 minutes per vessel, the processing is taken as 30 minutes and the exit times are takes as 2 minutes per vessel. The processing time consists of the closing of the doors, leveling and opening of the doors on the other side. Vessels cannot overlap in enter or exit time when they navigate subsequently in or out of the lock. The minimum passage time for a vessel is hence 34 minutes, which can only be achieved if it is locked on its own and locking is directly initiated after arrival and entering of the lock chamber.

A last simplification is that for safety allowance. In reality vessels are required to keep safety distances, especially vessels with dangerous goods. The safety allowances are ignored to simplify the problem.

1.4. Report Structure

The report structure follows the structure of the research questions, an overview of the structure of the chapters and corresponding sub questions can be found in Figure 1.1. In Chapter 2, different modelling approaches are presented and the literature is reviewed, in order to identify the best solution method. Then, in Chapter 3 the data is analysed. In this chapter locks are chosen to model, and the data of these locks is analysed. Passage times per vessel are not a part of the data and therefore the SIVAK model is presented in Chapter 4. SIVAK can serve as a baseline model to compare the LOSCO model with. The LOSCO model (Lock Scheduling by Constraint Optimisation) is introduced in Chapter 5. This is the model that was created in this study. The whole workings of the model is explained in this chapter. In Chapter 6 the LOSCO model is verified and validated. The results follow in Chapter 7 and Chapter 8. In Chapter 7 the results of the optimisation on minimum passage times are presented and in Chapter 8 the results for the economical optimisation. In Chapter 9 follows a discussion on the study. In Chapter 10 the main research question is answered and the study is concluded. The recommendations are also included in this chapter.

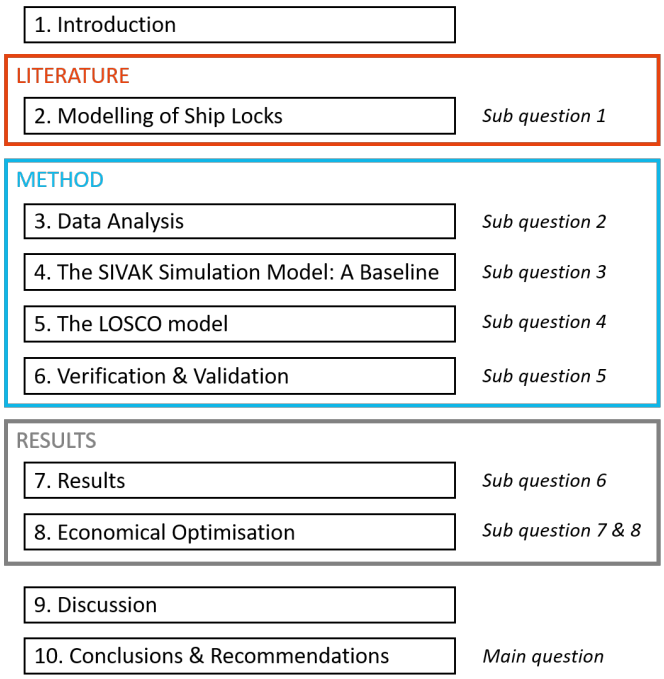


Figure 1.1: Overview of the chapters and corresponding sub questions.

2

Modelling of Ship Locks

This chapter answers the first sub question of this thesis:

What modelling method can best be applied to model a shipping lock, with 2 chambers in parallel, in which typically multiple vessels fit, for the use of lock scheduling?

In this chapter, distinct approaches for the modelling of ship locks are discussed. The lock scheduling problem is first introduced in Section 2.1. After that the different approaches for modelling this problem are discussed by taking a look at the literature available. This discussion is divided in three sections that each cover a method. The methods that are investigated are queuing theory in Section 2.2, simulation in Section 2.2 and mathematical programming / operations research in Section 2.4. The findings are concluded in the last section, Section 2.5.

2.1. Introduction to the Lock Scheduling Problem

This section introduces lock scheduling as a mathematical problem. A schematic overview of a lock complex is illustrated in Figure 2.1. This is an example of a lock with three parallel chambers. The lock scheduling problem is also valid for a lock with a single and for two chambers, but simpler, as there are less choices per vessel. The busiest locks in the Netherlands consist of multiple parallel chambers. Large chambers for inland waterway transport and sometimes smaller chambers dedicated for recreational vessels. Recreational vessels are out of scope, the smaller chambers dedicated for recreational vessels are not drawn in Figure 2.1. A lock chamber has a certain size and can lock multiple vessels at the same time, depending on the size of the vessels.

After a lockage has started, the chamber is unavailable for some time as it processes the vessels to the other side of the lock (chamber A and B in Figure 2.1). During this processing time the doors of the chamber are closed, the water is leveled with the other side of the lock and the doors are opened again. Also some extra time is needed to allow vessels to sail in the lock and to exit the lock after the vessels are processed. Vessels on the other side of the lock might be waiting already for their turn to be locked. A chamber can also process without any vessel inside (chamber A in Figure 2.1). This can be useful if there are vessels waiting at the other side of the lock. When a chamber is not processing, it is waiting on vessels to enter in one or the other side of the lock (chamber C).

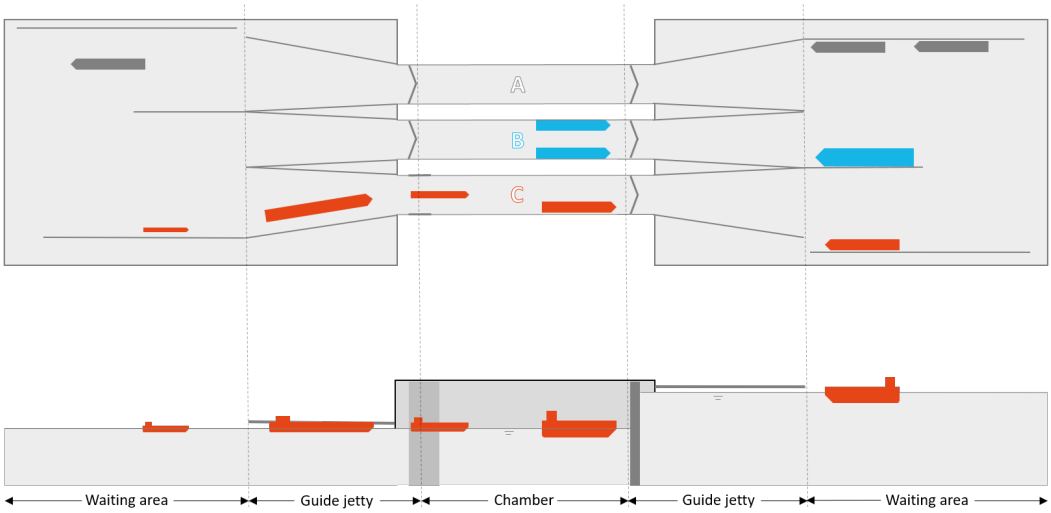


Figure 2.1: A schematic overview of a lock complex

Vessels can be in seven states. A vessel can be on its way to the lock, it can be waiting on the lock in the waiting area, it can be sailing in the lock, waiting in the lock, being processed by the lock, sailing out of the lock and continuing it's way after locking.

This thesis is not about optimising locking by changing the design of locks. For instance to speed up levelling, using faster lock doors etc. The lock scheduling problem is about optimising the lock scheduling sequence. The objective for optimisation in this thesis is to reduce the total passage time for vessels. This problem can be split in three decisions that have to be made multiple times:

- Each vessel should be assigned to a lock chamber;
- The order of sailing in for the vessels in the same lock has to be decided on;
- It should be decided at what time the lockage should be initiated.

These are hard decisions to make, as every single decision depends on many other vessels in the future and also on the other decisions.

In Chapter 5, a detailed description of the lock scheduling problem can be found. In this chapter, the literature is reviewed to come up with a modelling approach. Three modelling approaches are considered, queuing theory, simulation and mathematical programming.

2.2. Queuing Theory

Queuing theory is a modelling concept that can be applied to locks. Queuing theory is based on probability theory and is an analytical and exact model. The model is not hard for a computer to run, relative to simulation and to mathematical programming, of which the latter can be particularly computationally expensive. The mathematics of queuing theory can be complicated, which makes it harder to explain and understand relative to simulation. Queuing theory can seem a black box model relative to simulation, where each step can be followed. Queuing theory limits the use of functions to parametric distributions, which causes an approximation error. Queuing

theory provides only answers on the long term, the steady state solutions. But the most important disadvantage is that queuing theory is based on dispatching rules, that always result in approximations (Terekhov et al., 2014).

In Section 2.1 is argued that there are 3 types of choices to be made to optimise lock scheduling. These choices are about the order of sailing in a chamber, the choice of chamber per vessel and the choice of when to initiate locking in a chamber. The optimal choices are highly dependent on each other. In queuing theory the choices need to be made with rules. Examples of these rules are queuing discipline rules, that define the order in which the vessels are served. Also rules need to be defined for which chamber has priority and rules for when to initiate locking. The question is whether rules exist that are optimal. Likely, the optimal choices are different in each different situation. Since queuing theory can only deal with general rules, therefore it cannot be used to research the interplay of choices in each situation. Therefore first a model is needed that can generate the best choices at each step. From this model, general rules could be derived that can be used to model with queuing theory.

This problem is also encountered by Martinelli and Schonfeld (1995). They state that the use of queuing theory cannot be applied to a lock scheduling model. There is no first in first out queue discipline at locks and there are numerous interdependencies that queuing theory cannot deal with.

The same is concluded by Wilson (1977), as they investigated to model shipping locks with queuing theory in order to estimate the capacity of locks. For locks with a single chamber, an approximation could be made. However for multiple chambers, he states that his approximation is insufficient.

Radmilovic et al. (2007) might be the most advanced use of queuing theory to model single and parallel locks. They used batches of arriving vessels from 1 to 4 vessels, uniformly distributed. Service is also in batches. They modelled two different cases, in one case locking is initiated if only 1 vessel fits in the lock (the minimum batch service policy), in the other only if the lock is full with 4 vessels (full batch service policy). Occupancy and coefficients of variation for inter arrival and service times are varied. The conclusion is that it is better to use minimum batch service policy in all cases, however, the maximum occupation of the lock considered is 0.7. The study also states that the results are restricted due to the approximations that are made. These approximations are in the rules that are assumed.

Concluding, it can be stated that queuing theory is not a viable method to model shipping locks for optimality. That is because there are always approximations in the dispatching rules that are needed for queuing theory. The extend of the error due to these approximations cannot be known without a model that does not have the same approximations.

2.3. Simulation

A computer simulation is another method to model shipping locks. In particular a discrete event simulation. Time is not modelled discretely but events are, as most time steps would not be very interesting. As a result a discrete event-based simulation model has the potential to be faster.

There are several programs or packages that can do a discrete simulation. Options are to create a new simulation from scratch in Simio, Arena or in a Python

(or other programming language) package, such as SimPy or Salabim. SIVAK is the standard simulation model as used by Rijkswaterstaat. The newest version of SIVAK was created with Simio. SIVAK is a model that is able to simulate traffic on the inland waterways, in particular at locks and bridges. SIVAK is used as a baseline model in this thesis, more on this follows in Chapter 4.

Relative to queuing theory, simulation can cope with more complexity. Historical data can be used, in which relations of parameters and variables can be uncovered that could not be uncovered by queuing theory. Also insight on short term effects can be gained. However, still the same problem with dispatching rules as with queuing theory is present. Relatively more studies used simulation than queuing theory to model shipping locks.

Schonfeld and Ting (1998) studied the integrated control in a case with multiple locks in sequence. The essence of the problem is however different than the research question in this thesis. This is a study about American locks, where barges might have to be split up to pass locks after which they are joined again, see Figure 2.2. They created a simulation model that can minimize the costs of waiting, moving and cargo depreciation for all vessels. They use a heuristic algorithm that takes in to account the waiting time at the next lock for this. Scheduling is done at short notice, as all decisions are made based on the ships that are already available at a lock. The next lock is also considered by including the waiting time at the next lock in the algorithm.

Martinelli and Schonfeld (1995) made a simulation model in which the central question was whether or not the waiting at locks is dependent on the waiting at previous locks. He concludes with a relation for isolated waiting time divided by total waiting time for 2 and 3 lock systems in series. A major assumption that he makes is that all vessels are tows of the same size, a typical American case. This assumption cannot be made in the Netherlands.

Campbell et al. (2007) studied the same Mississippi river locks. They investigated different lock management alternatives; appointment systems, re-sequencing policies and system-wide traffic management. Appointment systems are useful, because ships can save fuel or do other productive activities instead of waiting in a queue. However the locking times were found not to be steady enough for this method. Re-sequencing means making another rule than ”first come first served”. The rule ”fastest processing time first” proved to the best re-sequencing policy considered. The cost-savings were quite low, but these savings were found to rise fast with increasing traffic. It remains a guess whether these conclusions are also useful in the Dutch case. In the Dutch case there are more vessels processed at the same time, instead of one convoy that is even split up. This likely leads to different conclusions.

Smith et al. (2011) studied more policies to locally optimize scheduling, again on the Mississippi river. He mainly discusses the order of the vessels to lock and came up with an effective heuristic.

A master thesis was done on the effect of adjusting vessel speeds to locking cycles (Hengeveld). He included a cost model to calculate the optimal sailing speeds towards the locks. He managed to achieve a reduction of 80% of the waiting times at

Figure 2.2: Left: Mississippi lock, multiple lockings are needed to process one barge convoy. Image from: [U.S. Army Corps of Engineers, 2017, commons.wikimedia.org] Right: Lock at Terneuzen, multiple vessels are processed at the same time. Image from: [Rijkswaterstaat, 2017, maritiemnieuws.nl]

the locks. It is however not an optimisation of the lock, but rather an optimisation of the travel towards the lock. Vessels are not locked faster but make less costs.

Van Haastert (2003) did her master thesis on a method for lock scheduling at the locks of IJmuiden near the port of Amsterdam. Her approach was a simulation of every logical possible order of locking vessels and selecting the optimum among those. This is hence a globally optimised case, it does however not work for the inland waterways, for similar reasons as described for the Schonfeld and Ting (1998) study. Sea locks is a different case, as the average number of vessels in the lock is much lower. This model would take too much time to run for inland waterway locks.

Overall, simulation is an applicable method for lock scheduling. A simulation is based on rules that are defined on beforehand. The main problem with simulation is that global optimisation is not possible without trying every possible alternative, this can be too time consuming for practical application, especially in busy, but not too busy, cases when scheduling likely has the biggest effect. Rule based algorithms, which can be simulated, are not sufficient for this problem, as decisions are very dependent of each other. It is unlikely that a rule based algorithm will return a good solution. Therefore, in the next section is turned to optimisation with mathematical programming.

2.4. Mathematical Optimisation

2.4.1. Planning Problems

Planning problems are part of the field of operations research. This field originated in the Second World War, when the allied developed the first methods for optimizing the use of their resources for military operations. The field is characterized by the use of mathematics and computer science to solve problems with a complexity that increases rapidly with the number of variables. Therefore many search algorithms have been developed to be able to solve problems in a reasonable amount of time.

Planning problems always deal with some limitations, the number of resources is limited (e.g. people, machines, commodities), or otherwise the time. Respecting the limitations, or constraints, the most efficient planning is searched for. The best alternative should be selected, there is an optimisation involved. In mathematics,

such a problem is called a mathematical optimisation or a mathematical programming model. A mathematical optimisation model consists of variables, constraints and an objective function. Variables are defined by a domain, a range of vessels. Constraints are functions that define the relation between the variables. The objective function is the function that should be optimised by changing the variables. In the field of computer science these kind of problems are called constraint satisfaction problems (CSP), if a feasible answer is needed. They are called a constraint optimisation problems (COP) if not only a feasible, but also an optimal answer is the goal (Russel and Norvig, 2010). In principle, these problems can be solved by evaluating all possible options after discretisation. However it is in the nature of many problems that the number of options soon explodes, so it is practically impossible to find the optimum in this way.

A well known COP and mathematical optimisation method is called linear programming, the relations (constraints) and the objective for this problem should be linear to use this method. An example that is often used for this kind of problem is the food problem (Dantzig, 1963). In this problem there are a few different kinds of food and each food has its nutritional values specified. The objective is to to have enough of each nutrient in a day while the price of the food needs to be minimised. The variables are the amount of each food to take. The variables form linear constraints to suffice each nutrient. The objective function is linear as each variable is multiplied by its price.

One of the most famous and also best algorithms to solve linear programming models is called the simplex method. This algorithm uses the fact that all relations are linear by going through all vertices of the linear relations. As the extreme points are always vertices in a linear model. The vertex in which the objective function is extreme is the optimal solution (minimizing or maximizing). In this example the domain is continuous. The domain could also be made discrete by only allowing integers. In this case it would for instance be forbidden to take half a sandwich or only a part of an apple.

When the domain is discrete, the model is called a combinatorial optimisation problem Russel and Norvig (2010). The lock scheduling problem needs discrete variables to make logic decisions. E.g. is or is the vessel not included in this cycle? This variable is called an integer variable. And if the only options are 0 and 1, it is called a binary variable. The problem is, that these variables do make a problem non-linear and the simplex algorithm is not able to solve this problem.

Often the number of alternatives is so big that it will take too much computation time to consider each and every alternative in the domain for every variable. Therefore a number of methods have been developed by mathematicians and computer scientists to solve different kinds of optimisation problems.

2.4.2. Job Shop Scheduling Problem

The lock scheduling problem shows similarities to a general formulated problem called the job shop scheduling problem (JSSP) (Google, 2020). In it's purest form ,the JSSP is as follows: There are n jobs that need to be scheduled on m machines, each job has a specific processing time and the machines are not identical. The goal of the problem is to minimize the makespan, the time from the first job to the finish-

ing of the last job. Once a machine starts a job it is obliged to finish it and it can only do one job at a time. If machines are identical, the problem is known as the flexible JSSP. In the standard version the jobs have to be carried out in a specific order and on (a) specific machine(s). Many variations on the problem exist. An example of the JSSP is the travelling salesman problem. The problem of finding the shortest route between n cities (machines). An example of the JSSP can be found in Figure 2.3. In this example there are 5 jobs. Each of the jobs is assigned a direction. The objective is to minimise the makespan using 2 machines with the constraint that the directions have to be alternated. In this small example, if there was only one machine, there are already 5! = 120 possible orders. With 2 machines, all these orders can also be divided in any way over the machines. Making $120 \cdot 5 = 600$ if the machines are exactly the same and $120 \cdot 5 \cdot 2 = 1200$ possible combinations if the machines are different. The computational complexity of this problem can be said to be $N! \cdot N \cdot 2$. This means that for N = 10 jobs, there are already more than 70 million options to choose.

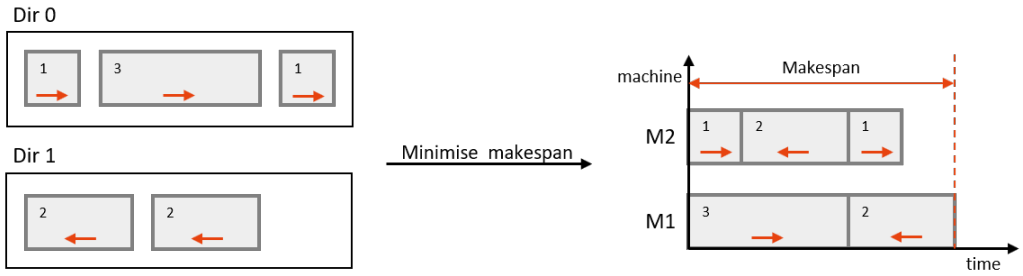


Figure 2.3: Example of a basic job shop scheduling program with 2 machines and the constraint that directions should be alternated.

The JSSP is a NP-hard problem (Hermans, 2014). NP-hard is a term from computational complexity theory. Meaning that the only algorithms applicable run in a time proportional to N! (the faculty of N). This is a problem, as the problem quickly is practically unsolvable. Therefore the conclusion is that the problem can only be limited in size to allow it to be solved, even by the best algorithms that were developed. Two algorithms that can be applied on a limited JSSP problem are Mixed Integer Programming (MIP) and Constraint Programming (CP).

2.4.3. Exact Solution Methods

Mixed Integer Programming (MIP)

One of the methods that was developed in the field of combinatorial optimisation is called mixed integer programming. MIP is a method with which a JSSP can be solved. The most generally used algorithm is called the branch-and-bound algorithm (Gurobi Optimization LLC.). In this algorithm, the problem is first converted to a linear program by allowing the integer variables to be continuous initially. The linear problem that results is called the linear-programming relaxation of the original MIP problem. The integer variables could be integer by coincidence, but it is more likely that they have a decimal value. In this case 2 new MIP problems are created. One problem forces the integer variable to be smaller than the nearest lower integer, and the other problem forces it to be larger than the nearest upper integer. E.g. for x = 3.46, x <= 3.0 for the first problem and x >= 4.0 for the second problem. These problems are

linear-programming problems and can be solved with the simplex method. In doing this as well for other integer variables, a search tree is created. Each new problem is called a node or a leaf of the tree.

If all of the integrality restrictions in a node are met, a feasible solution is found (Gurobi Optimization LLC.). No more nodes are generated from this node and the solution is compared to the last optimal solution. Some smart alterations help the model solve faster, such as cutting planes and heuristics. They function however on top of the same basic mixed integer programming model.

Constraint Programming (CP)

Constraint programming is effective for problems with many OR-logic constraints,compared to MIP. For this reason Google Google (2018) as well as Russel and Norvig (2010) indicate that for the JSSP problem, CP is the better option.

Constraint programming is solved with an algorithm called branch-and-prune. Prune is synonymous for the process of propagation. The basic idea is to create a search tree in which in each node a variable is assumed to be a certain value in its predefined domain. The value that is chosen in the nodes restricts values of neighbouring nodes through the constraints. Therefore a lot of nodes do not have to be visited, as they become infeasible. If a branch is unfeasible, the algorithms backtracks and starts researching another branch. There are many different methods created to go through the details of this process, that greatly improved the speed of constraint programming. Many of which stem from the field of artificial intelligence. The big step towards better algorithms has been the introduction of self-learning algorithms (Russel and Norvig, 2010). These algorithms can learn during the optimisation and even add learned new constraints on the go.

An example of a constraint programming algorithm is an algorithm that can solve a sudoku (Russel and Norvig, 2010). A sudoku is a puzzle that contains 81 variables. The domains of these variables are from 1 to 9. In every row and column and in 9 3x3 squares, all values should be different, these are the constraints. The CP solver does exactly the same as a human would do. It scraps values from the domain of each variable if the value is constrained. This is called checking for arc consistency (imagine the variables to be nodes and the constraints to be arcs). Some sudokus can already be solved by this inference. Harder sudokus need a stronger notion to be solved. Path consistency is an example of a stronger notion. With path consistency 3 variables are compared. If the domains of 3 variables in the same line of a Sudoku are 4,7, 4,7, 4,7,9, then it can be inferred that in the third variable the 4 and 7 can be eliminated, and the only possible value is a 9. This method can also be done for k variables, this is called k consistency. The main idea remains the same for any CP solver. Values are eliminated from domains by using local consistency, until one possible value for each variable remains.

Sudokus are designed to be solved with inference alone. For many problems, inference alone is not enough. At some time the algorithm will have to start searching. For CP problems searching was found to work best depth first. This means that repeatedly unassigned values are chosen until an inconsistency (by inference) is detected. After which the solution backtracks. This is called backtracking search (Russel and Norvig, 2010).

Inference and backtracking search are the basics of constraint programming. The algorithm can be further improved by taking the optimal order of operations, for

instance by starting search with the least constraining value. Many more heuristics exist to provide better results, such as self-learning methods. In Chapter 5, two of these methods are further elaborated on: decomposition and cut generation.

2.4.4. Approximation algorithms for solving the JSSP

Approximation algorithms are also called meta-heuristics. There are numerous examples of these algorithms. Meta-heuristics arose as algorithms that can balance the need for a good solution with the runtime of the algorithm. It is unlikely that the optimum is found with these type of algorithms. They might however be very useful in practice, because they can provide a good approximation in only a fraction of the time of an exact method. A simple set of priority rules is already a meta-heuristic. What we call simulation can hence also be interpreted as a meta-heuristic. Not one of the meta-heuristics can be said to be the best. It depends on the problem at hand (Blum and Roli, 2003).

One of the most widely used meta-heuristics are genetic algorithms. These algorithms try to mimic evolution to achieve the best solution. A first generation is created by randomly selecting a solution. The better solutions have a higher chance to be selected to sexually recombine and also mutations can occur to form a second generation. Over many generations the solution will improve.

Another approach is that of Ant Colony Optimisation. Ants follow simple rules, yet they can find the shortest paths to food if they work together. They exhibit higher intelligence working together, this is what is called swarm intelligence. This intelligence emerges as ants have a high probability to follow other ants by following their pheromones trail. The amount of pheromones and the time of evaporation can steer the other ants in the right direction.

The Bees Algorithm follows the strategy that bees have for collecting nectar. A small amount of bees are scouts, the scouts signal to other bees where they have found good nectar so that other bees can collect the nectar. In essence this is combining a global search with a local search.

Other examples include Simulated Annealing, Neural Networks, Tabu Search or the Electromagnetism like method (Blum and Roli, 2003). Also combinations between methods exist.

2.4.5. Studies to the Lock Scheduling Problem

Petersen and Taylor (1988) made a mathematical programming scheduling algorithm for a sequence of locks and narrow passages on the Welland canal in Canada. In their algorithm, only one vessel is allowed in a lock at the same time.

Nauss (2008) created a mathematical model for the clearing of queues in the Mississippi river. Such a queue forms when a lock has not been operable for some time or in case of accidents. The Mississippi river is again characterized by convoys.

Passchyn (2016) did his PhD thesis about locks scheduling on inland waterways using the mathematical approach. He focused on the complexity of different algorithms for solving lock scheduling problems. He investigated algorithms for a lock with a single chamber, for locks in series and for one lock with parallel lock chambers. However, in the case with parallel lock chambers, he only investigated the case with no waiting for vessels. Making it unusable for scheduling locks at busier times.

Hermans (2014) concluded that the multiple ship, single lock, problem is already NP-hard. Meaning it is not solvable in polynomial time. In other words, the order of the problem is N!. Therefore the multiple ship, parallel lock problem is also NP-hard, as it is an extension. The single ship, single lock problem can however be solved in polynomial time. He also proposed a method to solve this problem.

Zhang (2008) combined simulation and mathematics in order to schedule the locks at the Three Gorges Dam (Figure 2.4). These are actually 2 parallel series of 5 locks for which the total passage times can be up to 5 hours. The locks are in series because the difference in height at the dam is 113 meter. The lock in the north only transports vessels up towards the west and the southern lock only towards the east. This is a consequence of the series of locks. The core of the scheduling, or global scheduling is a simulation. Already on this simulation some algorithms are used, specifically an heuristic search method, simulated annealing and a rolling horizon scheduling method. After this also locally is optimized using a parallel genetic algorithm. In 2012 a new lock was opened at the Three Gorges Dam. This lock is a ship lift, that reduces the passage time to 40 minutes.

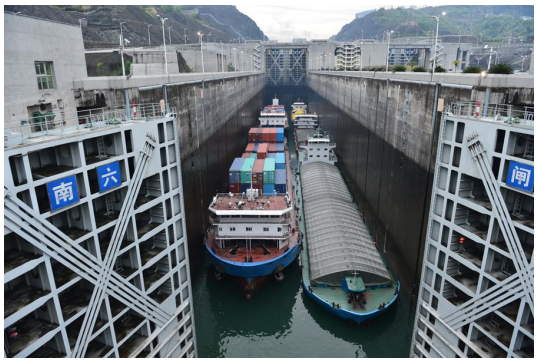


Figure 2.4: The 5 level lock system at the Three Gorges Dam. Image from: [Peoples Daily Online, 2016 ,en.people.cn]

Up to now, Verstichel (2013) has been the only study with a model for solving the lock scheduling problem for parallel lock chambers, that are not in series and for multiple vessels. He made this important contribution to the lock scheduling problem in his PhD thesis at the KU Leuven. In his model, he tried to achieve high solution quality with minimum calculation time. The result is a mixed integer programming model (MIP) that uses Bender's decomposition too gain speed. This decomposition method is also applied in this thesis an will be further elaborated on in Chapter 5. He split the problem in three interrelated sub problems: ship placement, chamber assignment and lockage operation scheduling. For each of the sub problems he found a parallel to existing problems. For the ship placement problem the parallel is to two dimensional bin packing problem, a problem that originates in the field of computer science. The other two problems have a parallel problem in operations research, specifically combinatorial optimization. Chamber assignment is related to the assignment problem and the lockage operational scheduling to the (parallel) machine scheduling problem (parallel in case of multiple locking chambers). The posing of constraints for the ship placement problem made it possible to come up with an exact answer using combinatorial optimization. However, this exact solution is too computationally extensive to use in practice. Therefore, a heuristic approach was

created, called the three-way best-fit heuristic. The heuristic consists of three steps. An ordering part, a ship placement policy and an array of gaps. The ordering part is a list of vessels that have priority, vessels are modeled as rectangles. Highest priority is placed first in the lock. Three orderings are included in the model: decreasing width, decreasing length and decreasing surface. The ship placement policy defines how individual ships should be placed, at the leftmost or rightmost side of the gap, adjacent to the tallest or shortest gap-defining rectangle or placing so that the difference in top level with its neighbour is maximal/minimal. The array of gaps is consisting of a skyline that determines the free space in the chamber. This problem and the other problem of chamber assignment was combined using a technique called Bender's decomposition. Which essentially exists of iterating solutions subsequently in order to save computational time. A decision support tool for lock masters was developed and tested successfully. The conclusion of the report is that the practical applicability of the model developed is limited by the slow convergence of the exact solution approach of the master problem.

2.5. Conclusion

With the literature that is reviewed in the chapter the first sub question can be answered:

What modelling method can best be applied to model a shipping lock, with 2 chambers in parallel, in which typically multiple vessels fit, for the use of lock scheduling?

The literature study has been summarized in Table 2.1. From the literature some conclusions can be drawn on the modelling of shipping locks.

First of all, queuing theory has been applied to model locks. Queuing theory is however dependent on rules, that make optimisation of the lock scheduling problem difficult. It is also hard to gain insight in the dynamics of the problem using queuing theory. Therefore queuing theory does not seem the best method to make a lock scheduling model.

Another approach is simulation. Simulation of the lock problem has been widely applied, however, optimisation is difficult, as this is also a rule based method. Local optimisation can be done, but a global optimisation is not possible with simulation. Simulation does however have the benefit that the most complex configurations can be modelled. Situations with multiple locks, parallel and in series and also with multiple vessels. Locks in series are not in scope of this thesis, as this would make the problem harder to solve. This might however be the next step in an a more encompassing lock scheduling model. Striking is the difference in the studies from North-America compared to Chinese and European studies. North-American IWT is characterized by barge convoys, vessels consisting of multiple barges so that only one vessel can be in the chamber in one locking. The barges can even be split up over multiple lockings, resulting in less than one barge per locking. Whereas the European and Chinese locks process multiple vessels at a time, making the job scheduling problem harder to solve. This can also be observed in Table 2.1. Many (American) studies only consider the case with only a single vessel in the lock. This is also the case, to a lesser extend, for sea locks. As Hermans (2014) states the situation with multiple vessels is a lot harder, even NP-hard, a computational complexity theory way of saying that the only algorithms available for solving the problems are of the

order N! (proportional to the faculty of N).

The lock scheduling problem can also be solved using algorithms from operations research and computer science. These methods are collected under the name mathematical programming. The generalisation of the problem is called the Job Shop Scheduling Problem and the best solution method was identified to be the constraint programming method.

Zhang (2008) created an advanced model for the scheduling of the Three Gorges Dam locks, he combined simulation with techniques from mathematical programming. Unfortunately, his model is also limited by simulation for the global optimisation problem. The nature of the problem is also different as this model consists of multiple locks in series. He can make the assumption that the two parallel lock sequences only transport vessels in one direction. What is of interest therefore is a study that can solve the lock scheduling problem for parallel lock chambers and multiple vessels per chamber but that does not necessarily schedules a series of locks.

Hitherto, only one study has been published about solving the lock scheduling problem for multiple vessels and parallel lock chambers in a pure mathematical programming way. This is the PhD thesis by Verstichel (2013). His study therefore serves as a cornerstone of this thesis. His model uses a decomposition method to split the lock scheduling problem in a master problem and a sub problem. The master problem contains the optimisation, whereas the sub problem assigns vessels to their position in the lock. The model was however found to converge too slow for practical applicability.

Table 2.1: Consulted studies about the modelling of locks

Study	Topic	Method	Series?	Parallel?	Vessels?
Martinelli and Schonfeld (1995)	Waiting time dependency of locks	Queuing Theory	Series	-	Single
Wilson (1977)	Lock capacity analysis	Queuing Theory	Single	Parallel	Single
Radmilovic et al. (2007)	Lock operations analysis	Queuing Theory	Single	Parallel	Multiple
Schonfeld and Ting (1998)	Integrated lock control	Simulation	Series	Single	Single
Campbell et al. (2007)	Traffic management (decision) rules	Simulation	Series	Single	Single
Hengeveld	Adjusting vessel speeds	Simulation	Single	Parallel	Multiple
Van Haastert (2003)	Lock scheduling	Simulation	Single	Parallel	Multiple
Smith et al. (2011)	Scheduling	Simulation	Single	Single	Single
Petersen and Taylor (1988)	Optimal scheduling Welland canal	Mathematical prog.	Series	Single	Single
Nauss (2008)	Clearing of queues Mississippi	Mathematical prog.	Series	Single	Single
Passchyn (2016)	Scheduling on inland waterways	Mathematical prog.	Series	Parallel	Multiple
Hermans (2014)	Optimisation of inland shipping	Mathematical prog.	Series	Single	Single
Zhang (2008)	Scheduling Three Gorges Dam locks	Sim. & math. prog.	Series	Parallel	Multiple
Verstichel (2013)	Lock scheduling problem	Mathematical prog.	Single	Parallel	Multiple

3

Data Analysis

In this chapter the second sub question is answered:

At which locks is sufficient data collected to use for a lock scheduling model?

In the first section, Section 3.1, the properties the busiest inland locks in the Netherlands are stated and locks are selected to model. In the section that follows, Section 3.2, some of the data at the selected locks is analysed. Last, in Section 3.3, the chapter is concluded. For the code behind every figure and statement is referred to the notebook on GitHub. The link to which can be found in Appendix C.

3.1. Inland Shipping Locks in the Netherlands

The flat delta makes Dutch shipping locks often low in elevation difference, with the exception of the south of the Netherlands, where more elevation differences exist. Most of the locks have the function to allow vessels to go from a waterway with a variable water level (sea or river) to a waterway with a fixed water level (lake or canal). Shipping locks exist in many different sizes for all kinds of different ships. The busiest shipping locks are often characterized by multiple chambers for inland vessels and sometimes also separate chambers for recreational vessels, as can be observed in Table 3.1. In this table the Dutch inland shipping locks with the largest number of passages are listed. The locations of these locks are depicted in Figure 3.1.

Table 3.1: Inland locks in the Netherlands with more than 30000 passages per year, excluding recreational ships (Bureau Voorlichting Binnenvaart, 2016). The locks that were selected to model are emboldened.

Lock	Corridor	CEMT	# Chambers	# Passages
Volkeraksluizen	Schelde-Rijnverbinding	VIb	3 + small lock	110331
Kreekraksluizen	Schelde-Rijnverbinding	VIb	2	68234
Sluizen Terneuzen	Kanaal Gent-Terneuzen	VIb	3	55668
Prinses Beatrixsluizen	Lekkanaal	Vb	2 + larger new	48984
Sluizen Hansweert	Kanaal door Zuid-Beveland	VIb	2	43559
Krammersluizen	Schelde-Rijnverbinding	VIb	2 + small lock	42211
Oranjesluizen	Binnen-IJ	VIa	1 + 2 smaller	41318
Prinses Irenesluizen	Amsterdam-Rijnkanaal	VIb	2	35131
Prins Bernhardsluizen	Amsterdam-Rijnkanaal	VIb	2	32220
Houtribsluizen	IJsselmeer	Va	2	31055
Sluizen Weurt	Maas-Waalkanaal	Vb	2	30320

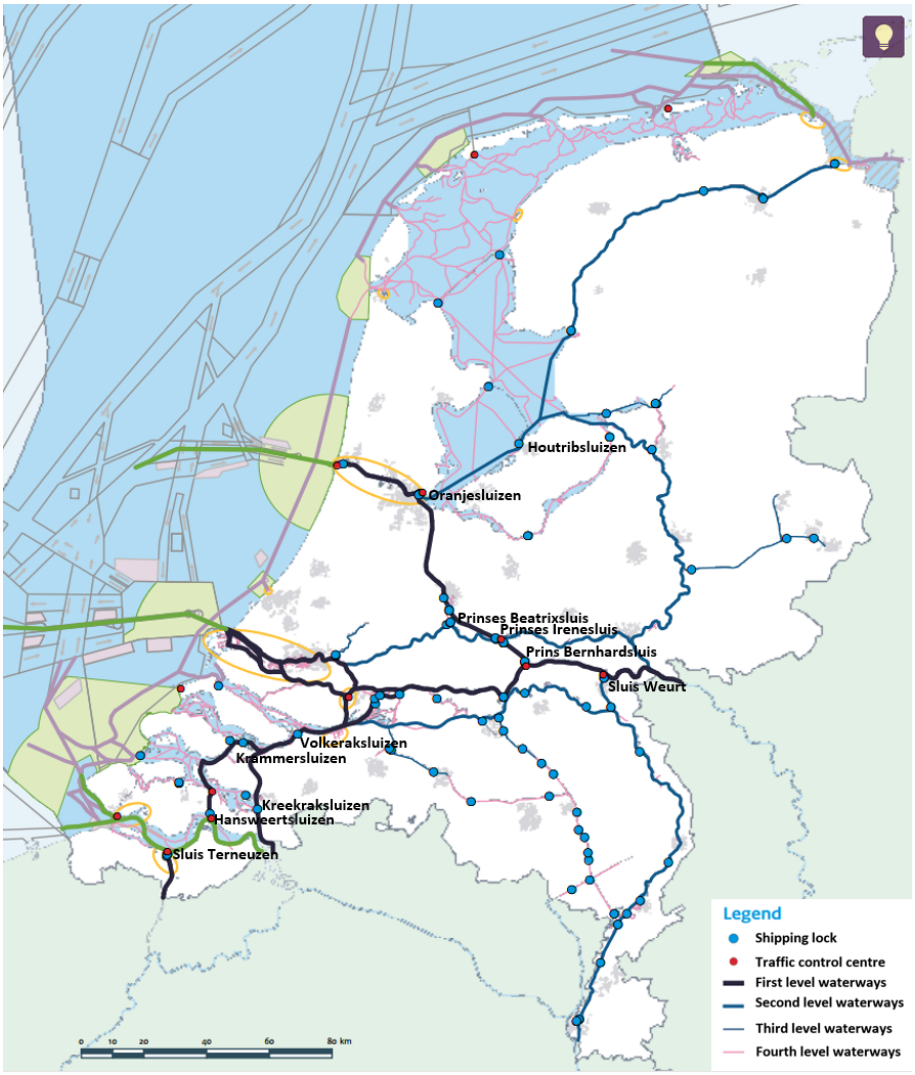


Figure 3.1: The main Dutch waterways and shipping locks. Edited map, base map by: [Rijkswaterstaat, 2015, Beheer- en ontwikkelplan voor de rijkswateren 2016 - 2021]

3.2. The Locks' Data

The data that was provided by Rijkswaterstaat consists of almost 4 years of data of lock passages at multiple locks (1st of January 2014 until 23rd of November 2017). Three locks were selected based on the number of chambers and their importance for the Dutch IWT system: The Krammersluizen, Kreekraksluizen and Hansweert-sluizen. These three locks have 2 chambers. In Table 3.1 can be observed that these locks are among the busiest 6 inland locks in the Netherlands. The locations of these locks are on the map in Figure C.1. On the map can be observed that the locks are on two different routes on the Rotterdam-Antwerp corridor. The Krammer-sluizen and sluizen Hansweert are on a westward route across the Oosterschelde and the Kreekraksluizen are on the eastern route on the canal called the Schelde-Rijnkanaal. Vessels can choose between the routes. The Volkeraksluizen are also in the data set, but it has 3 chambers and therefore it is computationally more expen-sive to solve for optimality. The Sluizen Terneuzen were also not chosen to model, because they are less representative of inland locks, as also a lot of sea vessels use this lock. The Prinses Beatrixsluizen would have been a good option, as at the time of writing a new third chamber is being build. However, this data was not provided by Rijkswaterstaat.

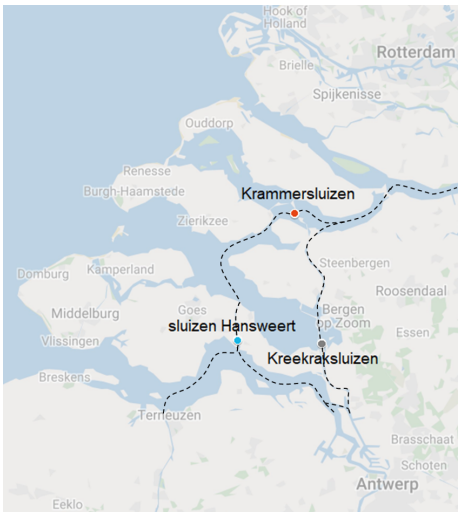


Figure 3.2: The locations of the selected shipping locks in their main corridors. Base map by [Google, 2020, google.com].

The data set consists of vessel information and the times at which they are as-signed to a chamber by the lock operator (Dutch: 'toerbeurt tijden'). Due to the lack of the actual arrival times, these times are modeled as arrival times, as the actual arrival times should not be too far off. The data does not provide passage times of vessels, nor can this be derived from the data. This poses a problem, because it means that the data cannot serve as a baseline. The data should first be processed by a model that can translate the arrival times in lock passage times. The model that is used for this is SIVAK, as is explained in the next chapter, Chapter 4

The Krammersluizen and the Sluizen Hansweert each have 2 chambers with a length of 280 meter and a width of 24.0 meter. The Kreekraksluizen has two cham-bers with a length of 318 meter and a width of 24.0 meter.

In order to get some insight in the data, the daily number of vessels is plotted in Figure 3.3 with partly transparent dots. On average 106 vessels pass at the Krammersluizen per day with a standard deviation of 26. At the Sluizen Hansweert, 113 vessels pass on average per day with a standard deviation of 29. And 188 vessels per day pass at the Kreekraksluizen with a standard deviation of 31. The standard deviations are of similar order, the higher mean for the Kreekraksluizen therefore means that the coefficient of variation is smaller: $c_{v,kreekrak} = 0.16, c_{v,krammer} = 0.25$ and $c_{v,hansweert} = 0.26$. Another way to say this is that the base load of vessels for the Kreekraksluizen is higher in absolute as well as relative sense.

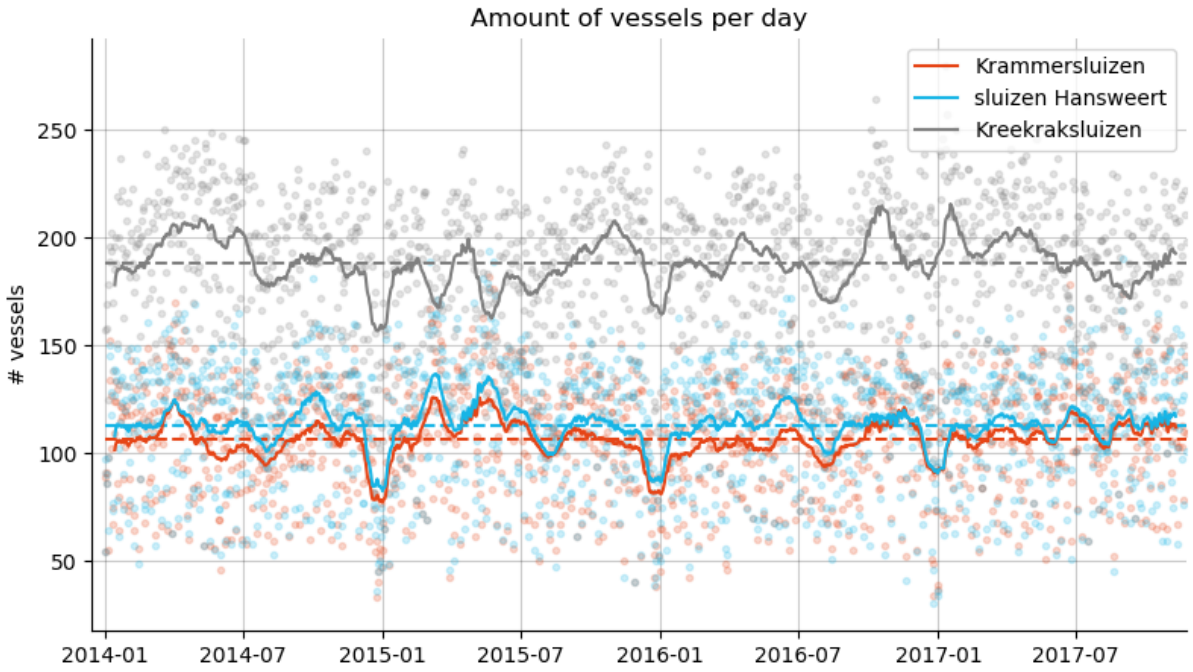


Figure 3.3: Amount of vessels passages per day in the three selected locks.

The vessels that pass the Krammersluizen and the Sluizen Hansweert are neatly divided in the two passing directions. However, the Kreekraksluizen lock 3.5 vessels southward more on average per day than northward. The difference could be due to coupling of barges or the route over sea or eastward canals and rivers.

In Figure 3.3, the means are drawn with broken lines in the respective assigned colors of the locks. The means of the Krammersluizen and Hansweert are close, in fact, a lot of vessels are the same, as they pass both of the locks on the same trip. This can be better observed with the continuous line. The continuous line represents the rolling mean for a period of 28 days, the mean of 2 weeks in front until 2 weeks later than the corresponding date. The red Krammersluizen and blue Hansweert follow similar contours, this implies that indeed at lot of vessels use the two locks on the same trip.

The Krammersluizen and Kreekraksluizen also show similar contours with the Kreekraksluizen. This can be explained by the fact that the vessels will spread along

the 2 routes, avoiding long waiting times. Also, the troughs in transport during Christmas and New Year's Day are clearly visible.

There is a weekly signal in what appears as chaos. This signal is displayed in Figure 3.4. In the weekends, the number of passages is the least. There are also differences within the work week. In the middle of the work week the number of passages is higher than during the begin or end of the work week. There are slightly more vessels travelling north on Friday and Saturday and slightly more south on Sunday and Monday. On Sundays the difference is largest.

On the daily timescale there is also a signal. The effect of day and night is clearly visible in Figure 3.5. For the Krammersluizen and the Sluizen Hansweert, more vessels go south in the morning and more vessels north in the afternoon. At the Kreekraksluizen, more vessels travel south during the night and more travel north during the day.

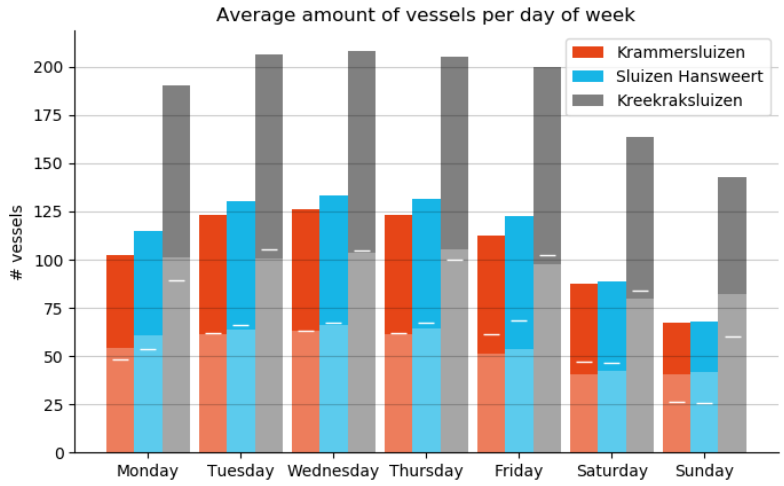


Figure 3.4: Amount of vessels passages per day of week in the three selected locks. The lower lighter part of the bars are the vessels that travel southward, the white lines indicate the part of the bars the vessels that travel northward.

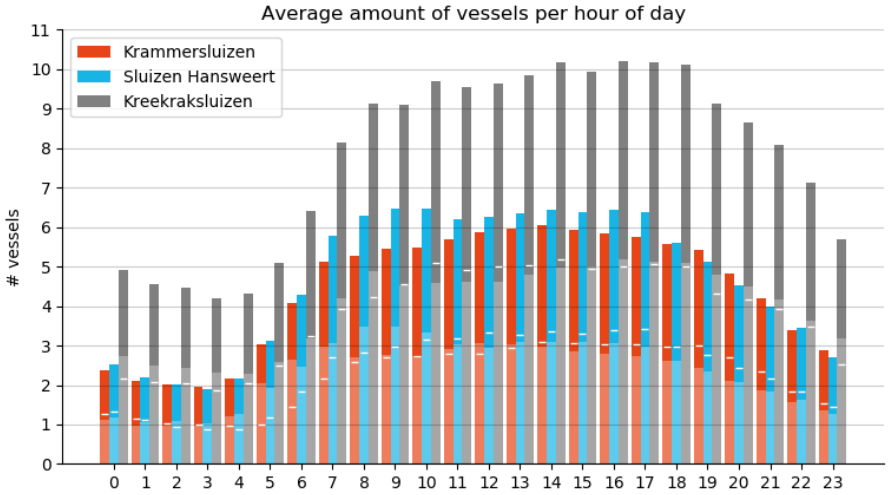


Figure 3.5: Amount of vessels passages per hour in the three selected locks. The lower lighter part of the bars are the vessels that travel southward, the white lines indicate the part of the bars the vessels that travel northward.

3.3. Conclusion

The second sub question was answered by the data analysis:

At which locks is sufficient data collected to use for a lock scheduling model?

Three locks were selected to model, the Krammersluizen, the Sluizen Hansweert and the Kreekraksluizen. These are the busiest Dutch inland locks with 2 chambers. The data that was gathered by Rijkswaterstaat is not sufficient as passage times cannot be derived from the data. Therefore the data needs to be processed in order to model passage times. The SIVAK model can be used to do this processing. Arrival times can however be estimated from the data, using the time that vessels are assigned to a chamber (Dutch: 'toerbeurt tijden').

4

The SIVAK Simulation Model: A Baseline

In this chapter the third sub question is answered:

How can the collected data be used for a lock scheduling model?

In Section 4.1 the choice for SIVAK as a baseline model is motivated. In section 4.2 the workings of the SIVAK model are explained. Lastly, in Section 4.3, the conclusions for this chapter are drawn.

4.1. Motivation for using SIVAK

In practice, lock operators make the decisions at the lock, but the data for passage time per ship is not collected, as was concluded in the previous chapter. Neither do lock operators use a method that can be modelled properly. It is possible that the lock operator makes a different decision when presented with the same problem. Therefore was resorted to the use of SIVAK as a baseline model. With SIVAK the passage times can be obtained with the approximate arrival times of vessels as input.

The passage times obtained can then be used as a baseline for the lock scheduling model. Recently, a new SIVAK version has been developed at Systems Navigator. SIVAK is used as a baseline to compare the lock scheduling model with, as data for passage times for vessels is not sufficient (Chapter 3). The reason for choosing SIVAK as a baseline model is that SIVAK is the standard for simulation of locks as used by Rijkswaterstaat.

The new SIVAK III model was build by Systems Navigator and is owned by Rijkswaterstaat. The new SIVAK model was build with the Simio simulation software. The power of Simio is in it's visualisation, which makes coding and testing easier. An example of the visualisation is shown in Figure 4.1. The model can simulate vessels on an individual level. It is a discrete event-based simulation model.

SIVAK can help with decisions in the design as well as in the use of inland waterways. Examples of uses of SIVAK are the impact on the traffic flow with respect to (according to Buro Sierenberg en De Gans, Ministerie van Verkeer en Waterstaat, Rijkswaterstaat, Dienst Verkeerskunde] (1991)):

- Dimensions of waterways;

- Congestion;
- Mixed and separate locking;
- Scaling up;
- Traffic management rules;
- Cost-benefit analyses.

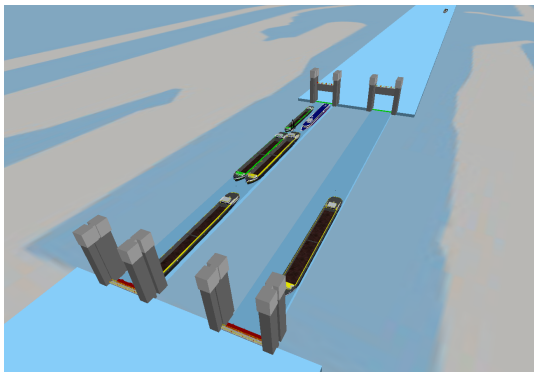


Figure 4.1: SIVAK visuals make the model behaviour better to understand.

4.2. The SIVAK simulation model

4.2.1. Overview

SIVAK is an acronym for ”simulatiepakket voor verkeersafwikkeling bij kunstwerken”, translated as ”simulation tool for traffic handling at engineering structures”. It was created for Rijkswaterstaat to study the handling of shipping traffic and road traffic at bridges and shipping locks, situated in a network of waterways.

The model can be split up in different sections that will be treated in the following subsections. The first subsection is about the network (4.2.2). The network defines the waterways that are modelled. A subsection on objects follows (4.2.3). This subsection is about bridges or locks that are in the network. The last subsection is on the fleet (4.2.4). In this subsection the actual vessels that go through the network and pass the objects are defined.

4.2.2. Network Definition

Network and Routes

The network serves as the backbone of theSIVAK simulation model. Waterways are modelled as arcs between nodes. In a node, multiple waterways or a lock or bridge can be connected. The nodes are defined by giving them 2 coordinates. The length, width and the depth of the waterways are defined as well as minimum and maximum speeds. Routes are defined in a static way. The route per vessel is defined before SIVAK is run. This means that vessels cannot choose their route dynamically, that is based on the traffic in the system. A route is a sequence of nodes that vessels pass.

Water Height

The water height can take on two different modes. It can be set fixed or tidal. In the case of a tidal water height a table can be added to describe the water height over time. All heights are relative to a specified reference level. This is also true for the bottom depth. Heights are checked for each vessel when entering a new waterway, entering a new lock or when a vessel is passing a bridge.

Traffic (Ship-ship Interaction)

Ship-ship interactions are managed before a new ship is entering a waterway. Widths, including required margins, are checked for encountering ships. If this requirement is not met, the ship has to wait at the start of the waterway, until the waterway is cleared. For overtaking, a similar width restriction was made, there is however also a length restriction in order to make sure the ship can pass within the length of the waterway. If one of these restrictions is not met, the speed of the second vessel is adjusted so that it will not overtake. In a waterway also route conflicts can be set for ships that cannot encounter each other, for instance in a harbor channel for large ships.

4.2.3. Objects (Locks)

The section objects in SIVAK defines bridges and locks. Locks being the topic of interest. A name and ID is provided for each lock and the nodes in the network that it is attached to. There are multiple settings that define the operation of the lock. With this settings the reality of the operations can be approached.

Chambers

A lock complex may exist of multiple chambers. The length and widths have to be defined for every chamber. In order to make a choice between chambers for each vessel the chamber priority setting is used. Four distinct chamber priority settings are distinguished:

- Area (A vessel will lock through the smallest chamber that it fits)
- Availability (A vessel will use the first chamber that has open doors on the side of the vessels. If chambers are equally available, the area priority is used)
- Fill (A vessel will choose the chamber that will have the highest utilization by entering)
- Custom Priority (One of the chamber is always considered first, unless it is unavailable)

There is also an option to record the water loss for each chamber.

Locking Regimes

A lock can be assigned a locking regime. The locking regimes specifies the requirements to initiate locking for a chamber. This decision highly influences the efficiency of the lock. Every regime is based on the utilization on the open and on the closed side of the lock. The utilization is defined as the summed surface area of the vessels that can be planned in the locking as a percentage of the total chamber area. The minimum utilization to initiate locking on the open and closed side can be chosen. There are three combination curves that can be chosen The first formula used is

called 'block'. This formula simply initiates locking if one of the minimal percentages is reached. The formula uses an 'OR' logic for this:

$$'block': \quad \frac{u_{act,open}}{u_{min,open}} \; OR \; \frac{u_{act,closed}}{u_{min,closed}} = 1 \tag{4.1}$$

The second formula is called 'linear'. This formula initiates locking if the summation of the actual percentages as a part of the minimal percentages reaches one:

$$'linear': \quad \frac{u_{act,open}}{u_{min,open}} + \frac{u_{act,closed}}{u_{min,closed}} = 1 \tag{4.2}$$

One of the specified minimal percentages will therefore rarely be reached. Therefore locking is initiated earlier than when using the 'block' formula.

The last formula is called 'ellipse'. It is similar to 'linear', except the two terms are squared:

$$'ellipse': \quad \left(\frac{u_{act,open}}{u_{min,open}}\right)^2 + \left(\frac{u_{act,closed}}{u_{min,closed}}\right)^2 = 1 \tag{4.3}$$

The terms are always smaller than 1 (if they reach 1, locking is initiated). This means that squaring has the effect of decreasing the terms. Therefore the decision is postponed relative to the 'linear' formula and locking is initiated later. Relative to the 'block' formula, the locking is initiated earlier, as terms will never reach 1, if only one vessel is waiting on the closed side of the lock.

There is a second mechanism that can overrule the specified combination curve. This is the maximum waiting time mechanism. Every vessel can only wait a specified amount of time. If this maximum waiting time is reached, locking in initiated.

The range represents the distance over which traffic can be detected. Only the vessels within this range are considered in the locking regime.

An example of the workings of the locking regimes can be found in Figure 4.2. Sometimes the treshold is not reached, but still locking is initiated. In this case one of the vessels reached its maximum waiting time. The average inter arrival time is low relative to the processing time. Therefore the lock does not wait in between lockings, but is continuously locking.

No Regime

There is also the option to not assign any regime. In this case the lock will initiate leveling directly if no vessel is within range on the side the chamber is open. If a new vessel comes within range before leveling is initiated, this vessel is added to the same locking.

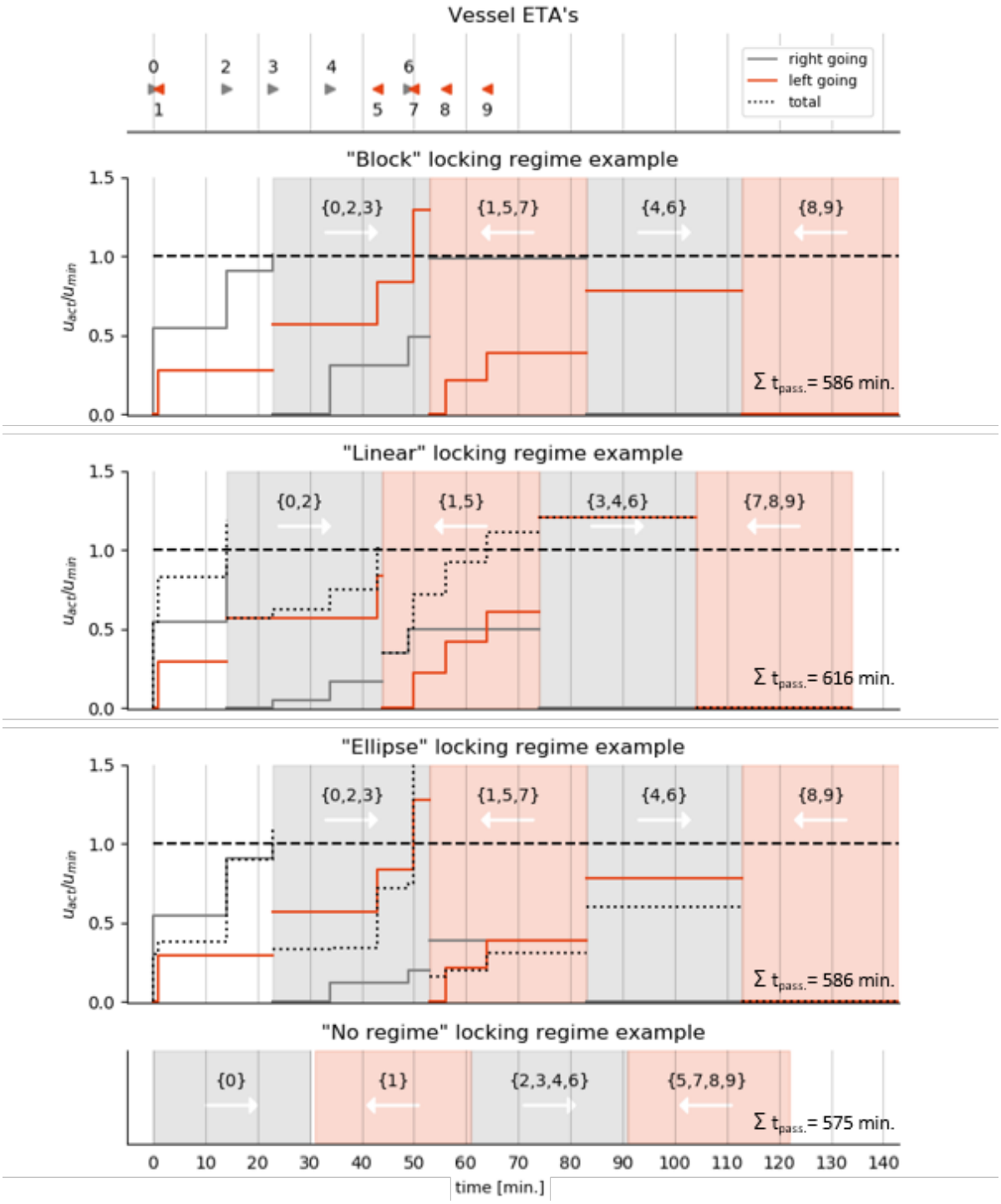


Figure 4.2: Regimes result in different decisions for locking. The range for this example is set at 5 minutes. The minimum utilization on the open side is 40% and on the closed side is 80%. The maximum waiting time is 30 minutes. The processing time of the lock in 30 minutes. The entering and exiting times are neglected in this example. In this case, the ”no regime” option is best, as the sum of the passage times for this option is lowest.

Passage Times

The passage time consists of time that is specified per vessel and time that is specified per chamber. The time per chamber consists of the times that are needed for opening and closing doors, a leveling base time and a leveling factor. The leveling factor is the time that is needed per meter of water level difference.

The time needed per vessel consists of entering and exiting times and a switch time correction. These times are specified per ship class and they are dependent on the loading of the vessel and the width of the vessel relative to the chamber entrance width. There is also a distinction between the first vessel and vessels that enter the chamber after other ships (entering or exiting subsequentially).

There is also a switch time correction. This is some extra time that is used between the last vessel that exits the lock and the first vessel to enter the lock. It is dependent on the length of the vessel that is about to enter the chamber and the length of the guide jetty.

The last addition on the passage time is due to a process called squeezing. This process models the extra time that is needed for vessels that fit in small gaps.

Lock Optimization

Lock optimization considers the position of vessels within the locking chamber. The optimisation will initiate if a vessel is within range that will not fit with the other vessels in the chamber. Vessels that were planned in the chamber earlier are always taken. The order can hence be changed solely to investigate whether the last vessel can fit in the chamber. If the first vessel starts entering the chamber, the order is definite. If another vessel comes within range the vessel can still be added to the locking if it fits, but the order is not changed anymore.

Safety Allowances

Within the chamber, vessels often maintain a certain space for safety relative to each other. This space can be based on regulations. In safety allowances these margins can be specified. A case for which safety distance are important are vessels with cones. The number of cones specifies the level of dangerousness of the goods that these vessels transport. There are regulations that specify the safety distances for these vessels.

4.2.4. Fleets

Vessels are divided in ship classes. A fleet consists of all the vessels that follow the same route on the network. Fleets can be generated in two ways, stochastically and deterministically.

Stochastic Fleet Generation

If vessels are generated in a stochastic manner, per ship class and attribute an expected value and a standard deviation is defined. Also correlations between attributes are specified. E.g. a larger vessel is likely to also have a larger dead weight tonnage (DWT). A data set for all different ship types is available in SIVAK to help with vessel attributes.

Arrival patterns have to specified in the designated table. The chances of arrival of a vessel per day per hour can be specified in this table.

Deterministic Fleet Generation

To generate vessels in a deterministic way all attributes for each vessel are entered manually in a table. Again, the standard ship classes can be used.

4.3. Conclusion

The third sub question can now be answered:

How can the collected data be used for a lock scheduling model?

SIVAK can be used as a baseline model to compare the lock scheduling model with. The SIVAK model can translate arrival times and vessel sizes into passage times at locks. This model suffices because it is already being used by Rijkswaterstaat for modelling purposes. SIVAK works based on a range that is specified for the lock. Once a chamber becomes available, SIVAK adds vessels to the lock that are within the range of the lock. The decision to start locking can be based on a regime. A regime is a function of the percentage of the area of the chamber that is filled with vessels. The regime can be overruled by the maximum waiting time that can be set. In the case no regime is used, SIVAK decides to initiate locking if there is no vessel within range on the open side of the chamber anymore.

5

The LOSCO model

This chapter provides an answer to the fourth sub question:

With which methods can a lock scheduling model be created, that can keep the runtime within a practical limit whilst obtaining the best possible solution quality?

This chapter is about the lock scheduling model that has been developed. For convenience, this model is called the LOSCO model, for "lock scheduling by constraint optimisation". First, the concept is explained in Section 5.1. An introductory example follows that makes it easier to understand the nature of the problem in Section 5.2. After that the two parts of the model are explained, the master problem, Section 5.3 and the sub problem in Section 5.4. Section 5.5 provides some information on the solver that is used and the code that was produced. Lastly, in Section 5.6 the chapter is concluded. The code of the model can be found in Appendix C.

5.1. The Model Concept

The lock scheduling model is based on the lock scheduling model by (Verstichel, 2013) that was explained in chapter 2. The concept is illustrated in figure 5.1. Just like in the Verstichel (2013) model, the model is split in two parts that are iterated until a solution is found.

The part that contains the actual optimisation is called the master problem. The other part contains multiple functionalities and is called the sub problem. The division in a master and a sub problem is a consequence of the need for performance of the model. Without this division, the model is so slow that it is not functional. Two methods are introduced to speed up the master problem, the first method is called 'first come, first serve'. This method can either be set on or off, if it is turned on, vessels are forced to enter the lock in the same sequence as their arrival. The resolution changes the discretisation, the step size could for instance be taken as one or two minutes.
The sub problem consists of three parts, first there is a part that is called chunking. Chunking can speed up the model by dividing the vessel list into multiple parts. Thus presenting easier problems to the master problem. When the model is run, chunking is what the model starts with. Another part of the sub problem is the maximum waiting time part. When the maximum waiting time is lower, the number of possibilities

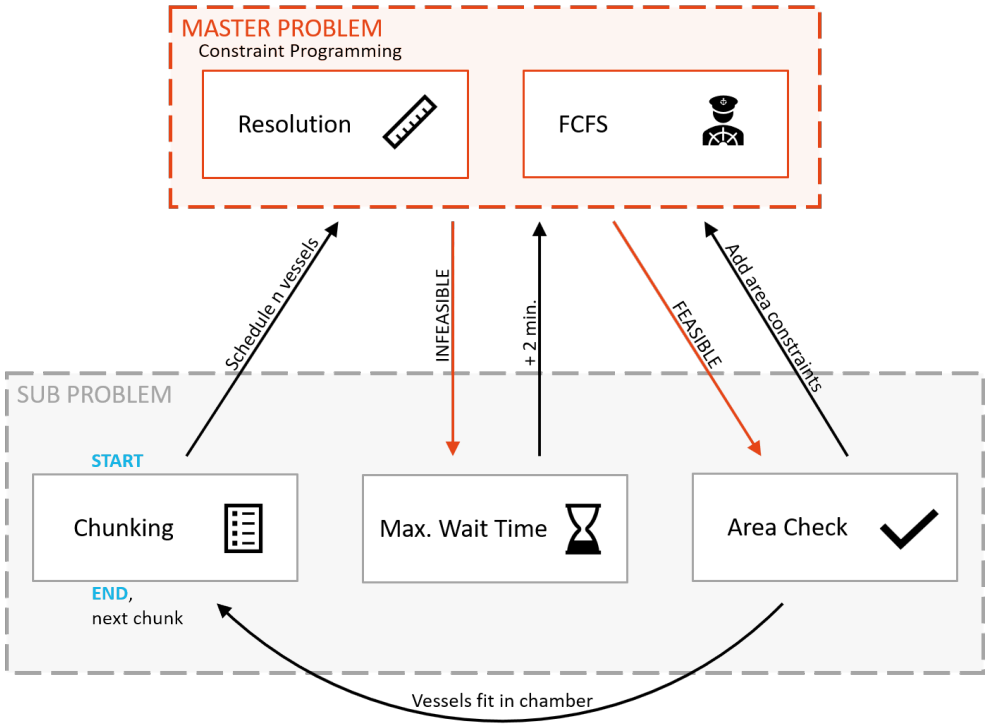


Figure 5.1: Flow diagram of the model concept.

is lower and the model is faster. If the master problem turns out to be infeasible, this part will increase the maximum waiting time for every vessel with 2 minutes. This increases the number of possible scheduling opportunities for the model and hence the chance for feasibility. If the master problem is feasible, the model precedes to the area check. The area check fits the vessels in the assigned chamber and returns constraints if sets of vessels that could not be fitted together. The iteration ends if none of the three sub problems return anything. This means that the schedule is valid. The model then procedes to the next chunk.

5.2. Introductory Example

In order to gain insight in the model, an introductory example with 4 vessels is used. The input of the example is shown in Table 5.1 and 5.2. This example is worked out manually to get a feeling for the model behaviour.

Table 5.1: The vessel input data for the example

Ship	ETA (min.)	Direction	Width (m)	Length (m)
A	10	Right	9.5	85
B	17	Left	10.3	85
C	38	Right	14.2	135
D	41	Right	7.5	65

Table 5.2: The lock data for the example

	Length (m)	Width (m)	Proc. time (min.)
Chamber 1	280	24	30

There are two questions that should be answered for each vessel in order to achieve an optimal schedule:

- At what time should leveling be initiated?
- At what order should it approach the lock relative to other vessels?

In this example, there is only one chamber. If there would be multiple chambers the choice of chambers would have been the third question to answer.

The goal is to schedule in such a way that the sum of the passage time for all is minimal. That is equal to saying the average passage time should be minimal. The smallest total passage time would be achieved if the vessels could enter the chamber at their time of arrival (ETA). Not a single vessel would have to wait. Unfortunately with a processing time of 30 minutes and entering and leaving times of 2 minutes, this is not possible for the vessels in the example. However, it would have been possible if and only if the vessels' arrival times would be spaced by more than $(30+2\cdot2=)$ 34 minutes from each other. Therefore, in our case, it is inevitable that at least one vessel will have to wait. The question then is: how can this waiting time be minimised?

The maximum number of locking cycles is eight. That is the case if there is one cycle for every vessel and the maximum amount of empty lockages. Empty locking is needed in between lockages when 2 subsequent lockages go in the same direction. With four vessels there are 4! = 24 permutations, or possible orders in which the vessels can be locked. The four vessels all fit in the same chamber concurrently and some vessels come from the same directions. With this information, already some of the permutations can be ruled out. Vessels A,C and D all go in the same direction, so for now we can assume that these vessels will be locked in the order of arrival. With this assumption there are only 4 permutations left. (B,A,C,D, A,B,C,D, A,C,B,D and A,C,D,B). However, there are more choices. Vessels can be locked at the same time, but only in the same direction. This increases the number of possibilities to 12. All these possibilities are elaborated in table 5.3. The first column contains the first possibility (B, ACD). In this option, first vessel B is locked on its own and afterwards vessels A, C and D are locked together. The minimum start time of the first lockage is 17 minutes, because that is the arrival time of vessel B. The entering and exiting time is 2 minutes, therefore, the vessel is in the lock at 19 minutes and that is also the time that the lock starts locking. The processing time is 30 minutes, so at 49 minutes the locking has ended. The vessel needs 2 minutes to sail out, so the chamber is empty at 51 minutes. Meanwhile, vessel A,C and D are waiting on this side of the lock. The three of them can sail in immediately. The vessels can sail in subsequently,the process of sailing in the lock takes 6 minutes. At 57 minutes, the locking can start. The doors will close and the water in the chamber will be levelled. It finishes at 87 minutes and vessel A exits the lock at 89 minutes, vessel C at 91 minutes and vessel D at 93 minutes. The passage time is calculated by taking the

time that the vessel left the lock and subtracting its arrival time. The passage times for each vessel are summed and the total that is obtained gives a measure of the effectiveness of the schedule.

This was calculated manually for each of the 12 possibilities in Table 5.3. The best option, the option with the least total passage time, appears to be the option in the first column. The best option only uses 2 lockages. The reason for this is that relative to the minimum processing time, the inter arrival times are small.

For such a small example, a manual calculation can be done, if there however more chambers and more vessels, a computer needs to be used. At a certain point, even the computer is not suitable anymore to evaluate each option. That is why constraint programming is introduced.

The result of the manual optimisation can also be viewed in the diagram in Figure 5.3. This type of diagram is used throughout this document. An explanation for this diagram can be found in figure 5.2.

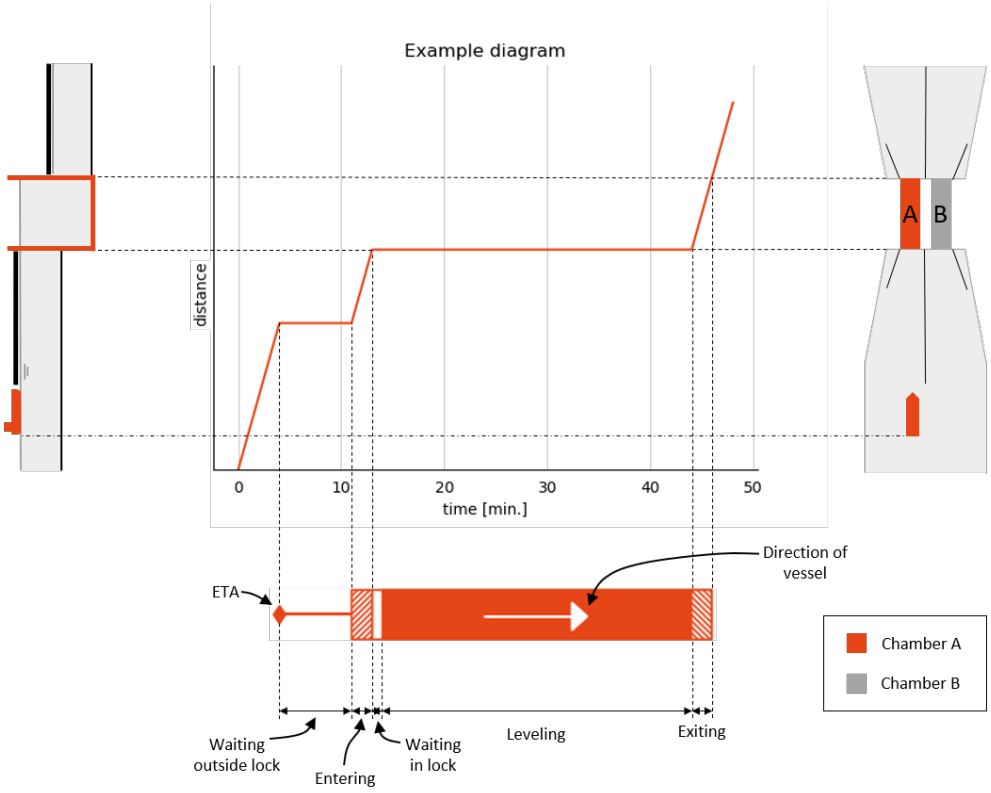


Figure 5.2: Explanation of the lock scheduling diagrams

Table 5.3: Elaboration of the example, all times are in minutes. All options are in the columns. The Roman numbers represent a lockage.

		B, ACD	**B, AC , D**	**B, A, CD**	**B, A, C, D**	**A, B, C, D**	**A, B, CD**	**A, C, B, D**	**AC, B, D**	**A, C, D, B**	**ACD, B**	**AC, D, B**	**A, CD, B**
I	min. start	17	17	17	17	10	10	10	38	10	41	38	10
	start	19	19	19	19	12	12	12	42	12	47	42	12
	end	49	49	49	49	42	42	42	72	42	77	72	42
	empty	51	51	51	51	44	44	44	76	44	83	76	44
II	min. start	51	51	51	51	44	44	74	76	74	83	106	74
	start	57	55	53	53	46	46	76	78	76	85	108	78
	end	87	85	83	83	76	76	106	108	106	115	138	108
	empty	93	89	85	85	78	78	108	110	108	117	139	112
III	min. start		117	115	115	108	108	108	110	138		139	112
	start		119	119	117	110	112	110	112	140		141	114
	end		149	149	147	140	142	140	142	170		171	144
	empty		151	153	149	142	146	142	144	172		173	146
IV	min. start				179	172		142		172			
	start				181	202		144		174			
	end				211	232		174		204			
	empty				213	234		176		206			
Passage time	A	79	77	75	75	34	34	34	64	34	69	64	34
	B	34	34	34	34	71	71	125	93	196	100	156	129
	C	53	51	113	111	114	115	70	38	70	43	38	72
	D	52	110	112	172	203	113	133	131	131	42	98	71
Total		**218**	**272**	**334**	**392**	**392**	**306**	**362**	**326**	**431**	**254**	**356**	**306**
Rank		1	3	7	10	10	4	9	12	6	2	8	4

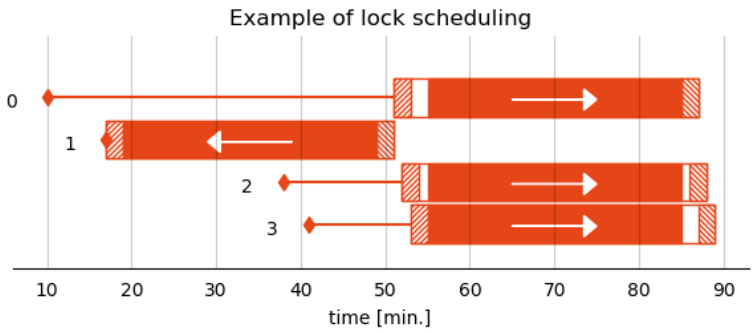


Figure 5.3: Example of lock scheduling, using one chamber. Every row represents a vessel. The diamond is the arrival time, the line means waiting outside of the lock. The hashed areas are entering and exiting. The white areas waiting inside the lock and the coloured areas represent the processing time with the direction of levelling.

5.3. The Master Problem

The master problem is a constraint programming model. This part of the algorithm makes the actual schedule. The master problem has to decide three things multiple times:

- What chamber to assign the vessel to?
- At what time is levelling in the chamber initiated?
- In what order should the vessels enter and exit the chamber?

These decisions should be made in such a way that all the constraints are satisfied and the objective function is minimised. The objective function is initially chosen to be the sum of the passage times over all vessels. Variables, parameters and constraints are defined in the master problem. A mathematical description of the master problem can be found in Appendix B. In this section a more concise and intuitive explanation is given.

The master problem receives as input a list with vessels from the chunking sub problem. The information that is in the list for every vessel is the ETA, the vessel dimensions and the direction of travel.

The three decisions that the master problem makes are illustrated with Figures 5.4 to 5.6. Note that there is no order in the decisions, all the three decision are equally important to achieve the optimal solution. In Figure 5.4, the leveling decision is displayed. The leveling for each value can be moved in time, but it cannot overlap, as is the case for vessel 0 and 1. There can be only one way that the processing can overlap, and that is when the start and end are exactly the same for two vessels and the direction is also the same. This means that they are in the same locking. In Figure 5.4, this is the case for vessels 2 and 3. The maximum waiting time is equal to the maximum amount of time that the processing can be moved from the ETA.

In Figure 5.5 the decision for the vessel entering and exiting order is displayed. If the lock dimensions were not restrictive, this would not make sense, all vessel orders would be optimal in the order of arrival. In reality, the lock dimensions can be restrictive, and sometimes it is therefore more optimal to change the order of the vessels. In this case, vessel 2 is larger, therefore it is more optimal to make vessel 0 wait until vessel 2 sailed in. If vessel 0 sailed in first, vessel 3 would not have fitted and would have to wait for a long time.

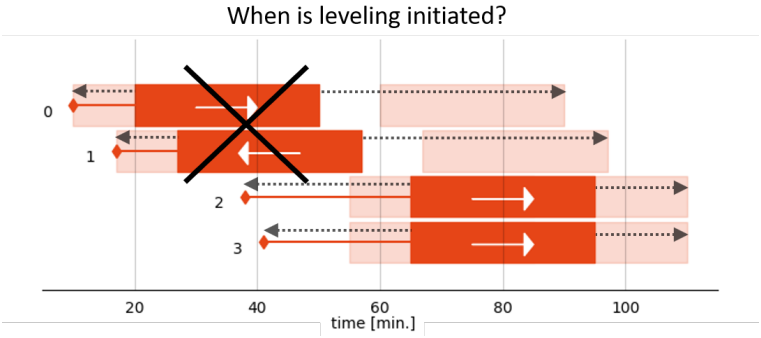


Figure 5.4: An illustration of the levelling decision. The processing per vessel can be moved in time until an optimal solution is found. The processing can only overlap if the direction of the vessels is the same and the start and end times are equal.

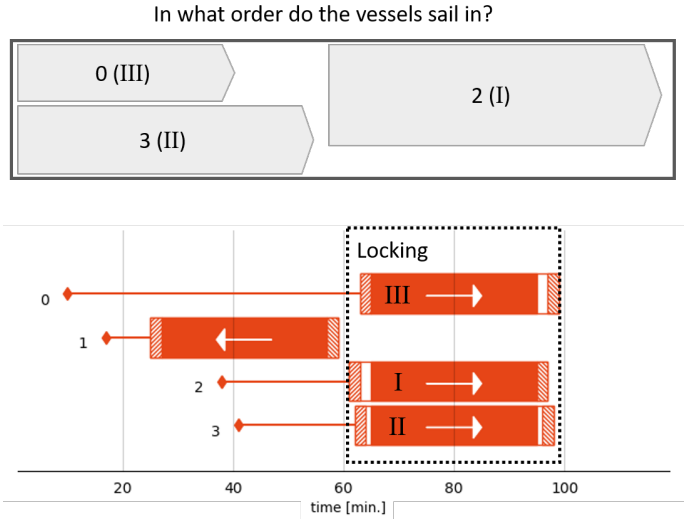


Figure 5.5: The vessel order decision. If the vessel order is not in the order of time of arrival, this is a consequence of the fit of the vessels in the chamber. The order of the vessels is counted with Roman numbers.

In Figure 5.6, the chamber assignment decision is displayed. The schedule that is shown is the optimised case for the set of vessels. In case there is more than one chamber.

5.3.1. Resolution

A method to reduce the runtime of the model is to reduce the search domain by increasing the time step. Instead of a discretisation of time per minute, the time step could be increased to multiple minutes. This reduces the number of options for the algorithm and hence makes the runtime faster. This does however come at a cost, as the average passage time will increase. Resolution is defined as the inverse of the time step. For example a time step of 2 minutes is the same as a resolution of $\frac{1}{2}$.

The smallest timescale in the problem are the exiting and entering times of the vessels. These were assumed to be 2 minutes. If a time step larger than 2 minutes is chosen, an error is made in the modelling of the entering and exiting times. To still be able to make the time step larger than 2 minutes, the model first calculated the total entering and exiting time by multiplying the number of vessels by 2 minutes if all the vessels enter the lock subsequently. Only then the closest multiplication of

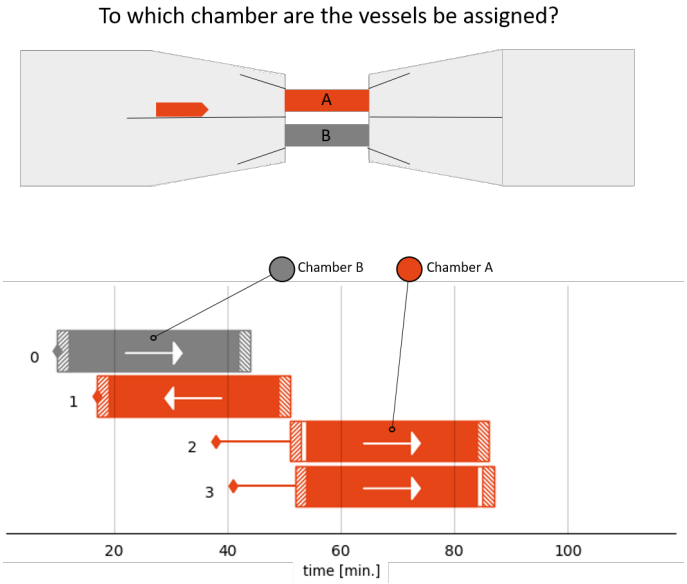


Figure 5.6: The chamber assignment and entering time decision. For every vessel, a choice between the chambers is made. The chamber is indicated by color.

the time step is taken. For instance, if the resolution is $\frac{1}{5}$, time step 5, and 3 vessels enter the lock subsequently, it would take 6 minutes. However in this discretisation the time needed is modelled as 10 minutes, which is 2 time steps.

If every time of arrival is equally likely, in the long run the losses per vessel converge to values that can be calculated. This can be explained by taking the example for a resolution of $\frac{1}{3}$. This means that all times should be stated as a multiple of 3. For a vessel arriving at time = 0, there is no time loss, however a vessel arriving at time = 1 minute is modelled at time = 3 minutes. In the model the vessel is already waiting for 2 minutes. For the vessel arriving at time = 2 minutes, 1 minute is lost, as this vessel's ETA is also modelled at time = 3 minutes. Then for a vessel arriving at time = 3 minutes, no time is lost. After which the pattern repeats. In the long run, on average $\frac{1}{3} \cdot 2 + \frac{1}{3} \cdot 1 + \frac{1}{3} \cdot 0 = 1$ minute is lost per vessel only by modelling the this error in ETA's. Besides the error in ETA, there is also an error for entering and for exiting, that was explained in the previous paragraph. The problem is essentially the same as for the ETA. The consequence is that for a time step of 3 minutes, 3 times this 1 minute is lost per vessel in the long run, making the discretisation error already 1 minute $\cdot$ 3 = 3 minutes for a resolution of $\frac{1}{3}$.

The fourth and last discretisation error is that for the processing time. The processing time was set at 30 minutes. This cannot be discretised for a resolution of $\frac{1}{4}$ for example where it is 32 minutes, introducing an error of 2 minutes for every vessel.

In Table 5.4 the discretisation errors are displayed for time steps ranging from 1 to 10. From this table it is apparent that some of the resolutions are not very useful. It is better to use the resolution $\frac{1}{10}$ than a resolution of either $\frac{1}{9}, \frac{1}{8}, \frac{1}{7}$ or $\frac{1}{6}$. And $\frac{1}{5}$ is better then $\frac{1}{4}$ and possibly also $\frac{1}{3}$ as the runtime savings are better with increasing resolution. The amount of time that can be saved with increasing resolution will result from the experiments in chapter 7.

Table 5.4: Discretisation errors per vessel

Resolution	ETA (min.)	Entering (min.)	Processing (min.)	Exiting (min.)	**Total (min.)**
1	0	0	0	0	**0**
$\frac{1}{2}$	0.5	0	0	0	**0.5**
$\frac{1}{3}$	1	1	0	1	**3**
$\frac{1}{4}$	1	1	2	1	**5**
$\frac{1}{5}$	1.12	1.12	0	1.12	**3.36**
$\frac{1}{6}$	2	2	0	2	**6**
$\frac{1}{7}$	3	3	5	3	**14**
$\frac{1}{8}$	3	3	2	3	**11**
$\frac{1}{9}$	4	4	6	4	**18**
$\frac{1}{10}$	2	2	0	2	**6**

5.3.2. First Come, First Serve

A restriction on the vessel order simplifies the problem, which could save runtime, as many possibilities are omitted. This is what is achieved with a 'first come, first serve' constraint (FCFS). The simplification could also mean that better solutions are missed. Verstichel (2013) concluded that the 'first come, first serve' should be used to decrease the calculation time. The introduction of new heuristic methods could however mean that a combination of heuristics without FCFS is better.

The 'first come, first serve' constraint for a lock with multiple chambers raises some questions. What does it mean to be served first? Are you the first to start the processing, or to enter or exit the chamber? Does it include both chambers? Does it include the arrival times on both sides of the lock?

To include both sides in the 'first come, first serve' constraint will clearly not be optimal. Situations will occur in which a vessel that could perfectly join a lockage will keep on waiting just because a vessel on the other side of the lock arrived earlier. Therefore the constraint should only count for vessels in the same direction of travel. Chambers should also be able to decide when to start a lockage independent of each other, time is lost when a chamber will wait on the other chamber only because vessels should leave in the same order as they arrive. Therefore the 'first come, first serve' constraint is taken to be in the scope of a lockage.

The usefulness of this constraint can only become apparent with experiments, in Chapter 7.

5.4. The Sub Problem

The division in master problem and sub problem was made for one reason only: it increases the performance of the model. Chunking and the maximum waiting time both trade a loss in solution quality for a gain in runtime. The area check does not come at a cost in optimality.

5.4.1. Chunking

Chunking is also known as cut separation. Chunking is the first part of the sub problem and this is also where the lock scheduling model starts. Chunking divides the list of vessels to be scheduled into smaller parts, before the optimisation starts. This makes the master problem easier and quicker to solve. Chunking can also be absolutely necessary, if the number of vessels to be scheduled is too large to solve

in a reasonable amount of time. This will later become clear with the experiments in Chapter 7.

Vessels can be independent with respect to lock scheduling. Intuitively this is easy understood in the hypothetical case when 2 vessels are spaced hours apart. The lock has ample time to anticipate on the vessels, it has the time to lock empty and can wait for the vessel to sail in. The inter arrival time is so large that neither of the vessels restricts the other vessels in any way. They can be said to be independent of each other with respect to scheduling. It is optimal to end a chunk when this inter arrival time is large enough. In the case the directions of the two vessels are opposite, the inter arrival time should be exactly larger than the processing time of the first vessel. If the directions of the two vessels are the same, the time that is needed to lock empty should be added to the processing time of the first vessel to create a boundary for independence.

In reality, the inter arrival time is rarely large enough for independence to occur. Therefore a method needs to be introduced that can cope with dependent sets of vessels. This approach aims to minimize the loss in solution quality by selecting the largest inter arrival times.

Chunking Algorithm

The chunking algorithm runs before the scheduling is started. One parameter is needed, the maximum chunk size. The algorithm consists of two stages, in the first stage, chunks are added and in the second stage, chunks are removed. The first stage starts by creating an ordered list of inter arrival times. Chunks are split from large to small inter arrival time, however, chunks are only split if the maximum chunk size is exceeded. The first stage is continued until no single chunk exceeds the maximum chunk size. This approach was found to create many small chunks and every chunk introduces an extra error if they are not independent. Therefore, in the second stage, chunks are merged to decrease the number of chunks. This time the order is from the smallest chunk size to the large ones. Often a choice can be made between two neighbours to merge the small chunk with. Preferably, the chunk with the smallest arrival time difference is deleted, but this is only possible if the maximum chunk size is not exceeded. In this way, the minimum amount of chunks is generated on effective positions. An example of the chunking algorithm can be found in Figure 5.7.

5.4.2. Maximum Waiting Time

The maximum waiting time is there to limit the possibilities for the master problem and therefore decreases the runtime of the model. This does however come at a cost: global optimality cannot be guaranteed when using this.

A standard value for maximum waiting time can be set. If the master problem cannot find a solution, the maximum waiting time is increased for each vessel by 2 minutes. By relaxing the maximum waiting time restriction more opportunities per vessel arise. The master problem is iteratively solved in this way, until a feasible solution is reached.

The maximum waiting time also makes the schedule fairer. It prevents the situation that a few vessels have very long waiting times to the benefit of other vessels. Rijkswaterstaat bases it's investment decisions for locks on a maximum waiting time of 30 minutes. If the average waiting time surpasses this maximum waiting time, action should be taken at the lock.

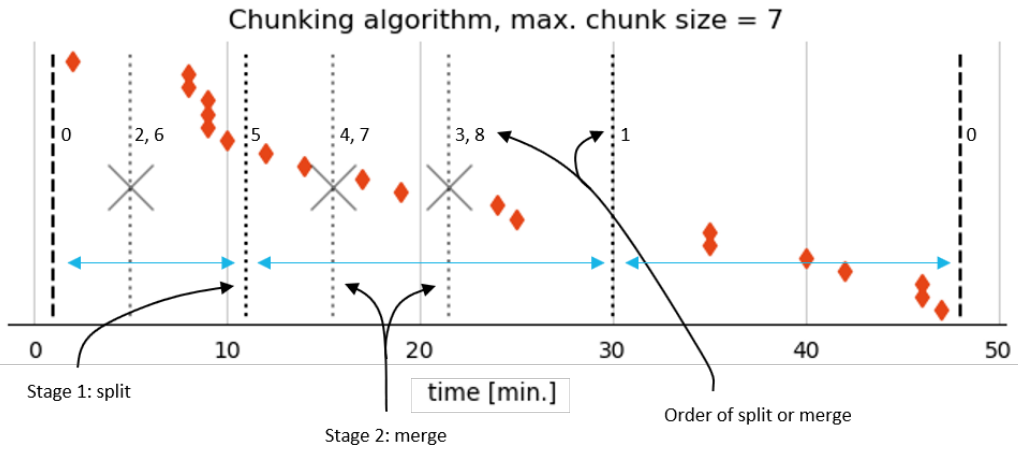


Figure 5.7: An example of the chunking algorithm with a maximum chunk size of 7 vessels. The red diamonds represent the arrival time of each vessel of 20 vessels in total. In stage 1 splits are generated (step 0 to 5), in stage 2 they are merged (step 6 to 8). The result is three chunks that are indicated by the blue arrows.

5.4.3. Area Check

The area check is the only part of the sub problem that was also used by Verstichel (2013), in what he called the lock layout problem. The area check in his work was improved. This is done by splitting the problem in different parts. Contrary to Verstichel, already in the master problem a constraint was added. Sets of vessels that have a larger cumulative area than the area of the lock chamber are already denied in the master problem. For this check all vessels are modeled as rectangles, so that a simple width times length calculation suffices. Also in the next methods, the vessels are modeled as rectangles.

In the area check sub problem, the vessels are fitted more realistically. First, the possibility for fitting the set of vessels independent of their order is checked by the heuristic Three-Way Best-Fit algorithm that was created by Verstichel (2013). Next also the order is checked with a new method. For every set that was scheduled but cannot be fitted, a constraint is send back to the master problem in the iterative process. After which the master problem tries to solve again, but it will not put the same set of vessels in the same chamber again.

Three-Way Best-Fit algorithm

The Three-Way Best-Fit algorithm is an ingenious heuristic that was created by Verstichel (2013). An example of the result of this heuristic can be seen in Figure 5.8 With this heuristic it can be checked quickly whether a set of vessels fits in a chamber at all. The heuristic is so fast because the vessels are sorted in three ways, on decreasing width, decreasing length and decreasing area. These sorted lists are combined with 6 placement methods. So that only 6 x 3 = 18 combinations are tried. If one of these combinations manages to fit all vessels in the lock, the algorithm returns $True$, otherwise it return $False$. The entire explanation of the heuristic can be found in Appendix D. If the algorithm finds a set that does not fit, all subsets are also tried to fit. This increases performance as it decreases the number of iterations on the master - sub problem level.

Area Order Check

The last area check is the area order check. The area order check is only applied if the FCFS constraint is turned off. For this check the fit of the set of vessels is checked

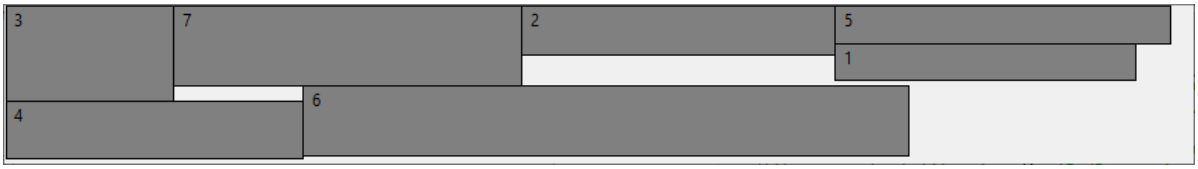


Figure 5.8: Example of the output of the three-way best-fit algorithm. A fit is achieved using the decreasing width ordering policy and the leftmost placement policy

in the order that they are assigned by the master problem. If the area order check is *False*, all permutations of the order are also checked. Again to increase performance. The working of the concepts of skyline and lowest gap in the algorithm of the area order check is illustrated in Figure 5.9 and the process of the algorithm in Figure 5.10. The area order check works similar to the Three-Way Best-Fit algorithm, but it is more precise and a lot slower. There are only two placement methods, left and right. After a vessel is placed the skyline is updated, similar to the Three-Way Best-Fit algorithm. In the skyline the lowest gap is selected. If the next vessel fits in this gap, it is placed there, left or right, dependent on the placement method. If it does not fit, the next lowest gap is tried. If all the vessels are successfully fitted in order, the algorithm is terminated and returns *True*. If after all combinations no fit can be achieved, *False* is returned.

The number of combinations that is tried is n^2. This explains why this algorithm can get slow if the number of vessels increases. The Three-Way Best-Fit algorithm always has the same number of options so it is faster.

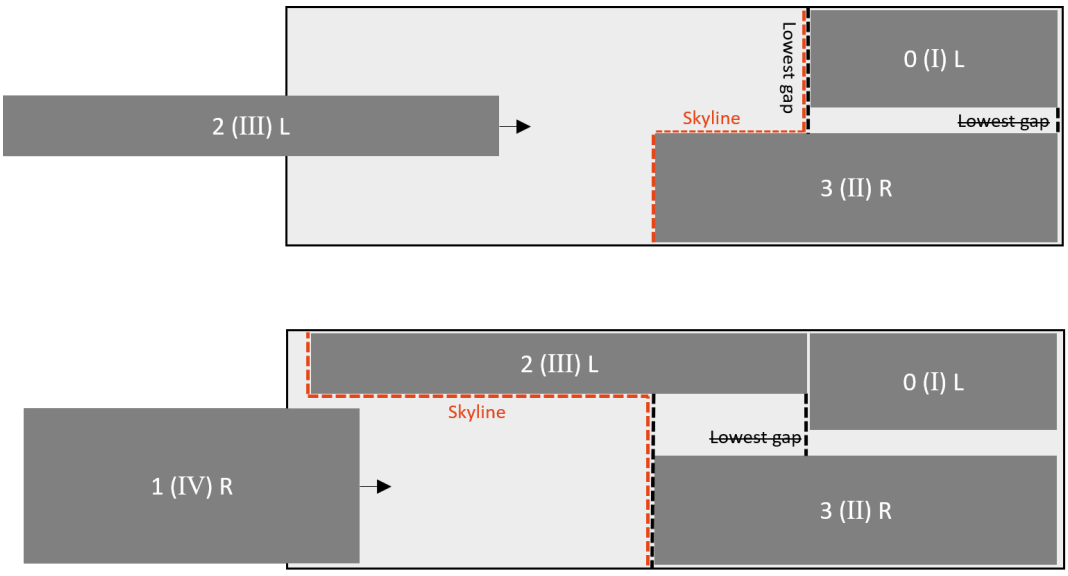


Figure 5.9: Example of the working of the concepts lowest gap and skyline. In this example 4 vessels are tried to be fitted by the area order check in the order (0,3,2,1).

It would be sufficient to only check if the vessels fit in the order that is provided by the master problem, instead of first running the Three-Way Best-Fit algorithm. This would however be very time-consuming. The number of orders for vessels is equal to $n!$. This means that for 8 vessels there are already $8! = 40320$ possible orders. All these orders would have to be returned with an area constraint. If none of the orders fit, all these constraints can be replaced by only 1 constraint with the heuristic. This

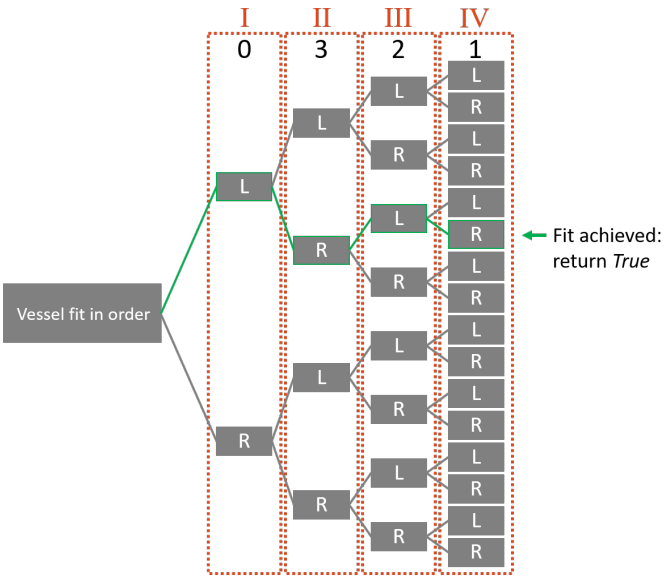


Figure 5.10: All possible options for the example are displayed. Each vessel can either be put left (L) or right (R) in the lowest gap. If a fit is achieved, the algorithm terminates and returns *True*

is the reason that first the Three-Way Best-Fit algorithm is run and after that the area order check.

5.5. The Code

The code can be found in Appendix C.

No code from Verstichel (2013) is available. Therefore the entire code had to be rewritten in this thesis. The division in a master and a sub problem (Bender's Decomposition) and the Three-Way Best-Fit algorithm in the area check are taken from this study.

The model is developed in the python programming language. The solver that was used is the Google OR-tools solver. Many solver exist, examples are: Gurobi, CPLEX, PuLP, MiniZinc. However, most solvers focus on mixed integer linear programming. Therefore only the solvers that have a constraint programming capability remain. Amongst those, the solver that performs best is the Google OR-Tools solver (Monash University, 2019). This solver is also open-source, contrary to the solver that Verstichel (2013) used, Gurobi. The solver being open source makes it very suitable for research purposes.

5.6. Conclusion

This chapter attempts to answer the following sub question:

With which methods can a lock scheduling model be created, that can keep the runtime within a practical limit whilst obtaining the best possible solution quality?

The lock scheduling model that was developed was called LOSCO (Lock Scheduling by Constraint Optimisation). The model is based on a model by Verstichel (2013). This study concluded that the model was too slow for practical applicability. The challenge is therefore to get the best possible solution while keeping the calculation time low. To achieve this, three new heuristic methods are proposed. Also the no-'first come, first serve' method is dropped as a combination of the new methods could mean this provides better solutions. The new methods are a change in resolution, the introduction of a maximum waiting time and chunking.

The resolution method is a change in the discretisation of the model. The time step could be enlarged so that number of options for the master problem is decreased. Experiments can show if the balance between the decrease in calculation time and the worse solution quality is in favor of the model or not.

The maximum waiting time puts an upper limit to the search space of every vessel. This also decreases the number of options.

Chunking is a method to divide the list of vessels to be scheduled up into smaller parts or chunks. The chunks are split when the arrival times of two subsequent vessels are spaced as much as possible. The chunks provide easier problems to solve for the model. The drawback is that the model does not take into account the vessels that are not included in the chunk, resulting in a worse solution. If the chunking size can be larger, the solution will improve. This method is of practical use, as it provides a method to link solutions.

A balance needs to be found between the settings of the different methods, in order to find this balance between solution quality and calculation time, experiments will presented in Chapter 7.

6

Verification & Validation

In this chapter the fifth sub question is answered:

How can the lock scheduling model be verified and validated?

In this chapter the method of verification is explained and also the validity of the model is discussed. The difference between verification and validation is often misunderstood. Verification gives an answer to the question: Was the model build right? And validations answers: Was the right model built? Verification is hence about problems with the coding and implementation. Validation, on the other hand, is about the modelling. All code that was used in this chapter can be found on GitHub, the link to which can be found in Appendix C.

6.1. Verification

6.1.1. Master Problem Verification

To check if the model works, tests are executed. The constraints of the problem are checked for feasibility, these constraints cannot be violated:

1. The locking time is later than the ETA;
2. Empty Locking Constraint: If the direction of a vessel is unequal to the initial lock direction, then the locking time of this vessel cannot be smaller then the processing time of 30 minutes (the lock needs time to level to the other side);
3. Processing Time Constraint: All vessels that are locked in the same chamber and have the same direction have either the same locking time or have a locking time that is spaced with minimally 2 times the processing time of 30 minutes from the other vessels (the ships can be in the same locking or they have to wait until the lock is back in the same direction);
4. Processing Time Constraint (2): All vessels that are locked in the same chamber, but have a different direction are spaced with minimally the processing time (the lock will need this time to change direction);
5. Area Constraints: Vessels that are locked at the same time in the same chamber have a total area that is smaller than the chamber area;
6. Multiple chambers: If more chambers are used, the total waiting time should always stay equal or decrease.

7. Enter and exit time: Every vessel needs two minutes to enter and to exit the chamber. Subsequent vessels are therefore always spaced by 2 minutes.

Besides the constraints, also specific functions need to be tested:

8. Resolution, the discretisation is changed.
9. First come, first served (FCFS), vessels can only enter a chamber in order of arrival.
10. Economic optimisation, optimisation is on minimal monetary value instead of minimal time.

In Table 6.1 the 11 tests that take all these constraints and functions into account are displayed. In the second column is the number of the constraint in the list above that is tested. To be able to find the answer on these tests a predefined pattern is used in the input for ETA. This makes the result also predictable. Constraints are isolated in a test if that is possible. However, not all constraints can be isolated to test them. In that case, multiple different combinations with other constraints are made. Combinations have to be tested as well, as constraints might influence each other. The input on the left side of the table should match the output on the right side of the table for the test to pass.

Table 6.1: Verification tests for the master problem.

	Description	Constraints	Vessels				Lock			Output		
			ETA (min.)	Length (m)	Width (m)	Dir (l/r)	# chambers	Length (m)	Width (m)	Chamber	Passage time (min.)	Waiting time (min.)
Test 1	Closed side	1, 2	0	100	10	r	1	100	10	1	64	30
Test 2	Closed side, later arrival	1	30	100	10	r	1	100	10	1	34	0
Test 3	Multiple vessels, just in time	1, 4	0	100	10	l	1	100	10	1	34	0
			34	100	10	r				1	34	0
			68	100	10	l				1	34	0
Test 4	Multiple vessels, just in time	1, 3	0	100	10	l	1	100	10	1	34	0
			64	100	10	l				1	34	0
			128	100	10	l				1	34	0
Test 5	Arriving too early	1, 3	0	100	10	l	1	100	10	1	34	0
			34	100	10	l				1	64	30
			68	100	10	l				1	94	60
			102	100	10	l				1	124	90
Test 6	Multiple in chamber	1, 3, 4, 5, 6, 7	0	50	10	l	1	100	10	1	36	2
			2	50	10	l				1	36	2
			38	50	10	r				1	36	2
			40	50	10	r				1	36	2
Test 7	Multiple chambers	1, 3, 6	0	100	10	l	2	100	10	1	34	0
			0	100	10	l				2	34	0
			34	100	10	r				1	34	0
			34	100	10	r				2	34	0
Test 8	Mixed sizes	1, 4, 5, 6, 7	0	50	10	l	2	100	10	1	36	2
			0	50	10	l				1	38	4
			0	100	10	l				2	34	0
Test 9	Resolution = 0.5	1,4,8	0	100	10	l	1	100	10	1	34	0
			35	100	10	r				1	35	1
Test 10	First come, first serve	1, 3, 5, 6, 7, 9	0	50	10	l	1	100	10	1	36	2
			0	100	10	l				1	102	68
			0	50	10	l				1	38	4
Test 11	Economic optimisation	1, 4, 10	0 (VoT = 60)	100	10	l	1	100	10	1	98	64
			0 (VoT = 6000)	100	10	l				1	34	0

6.1.2. Sub Problem

Area Check

Recall from Chapter 5 that the area check is twofold, there is the Three-Way Best-Fit algorithm by Verstichel (2013) and the area order check.

Only the heuristic model was recreated, so cases cannot be compared with an exact method. Verstichel (2013) concluded the exact model is too slow, and is not worth modelling for validation only. In his report, there is no verification.

For verification the lock layout can be plotted in top view. A number of lock layouts can be plotted and it can visually be checked if a fit is achieved and if constraints are violated. Tests can be done with sets of ships that are known to fit or not. For instance two ships that can just fit next to each other in the lock (or not). The tests are found in Table 6.2. The model was found to pass this verification.

Chunking

The chunking function is part of the sub problem. Chunks can be identified from the eta's of the vessels lists before running. The example in Figure 5.7 on page 41 is used to verify the chunking function. The chunking function consists of two parts. In the first part chunks are added from large to small time between vessels, until all chunks are equal to or smaller than the maximum chunk size. In the second part, chunks are removed if they can be combined to achieve larger chunks. So that the minimum number of chunks is used. The test can be found in table 6.3.

Maximum waiting time

Two tests are executed for maximum waiting time verification. Test 10 and test 11. In test 10 is tested whether the model automatically increases the maximum waiting time if there is not feasible solution. In test 11 the model tests if the maximum waiting time itself works. A case is used in which the waiting time is close to the maximum of 60 minutes. If then the maximum waiting time is increased to 90 minutes, the objective function should be better. In other words, the sum of all passage times should be lower, so that a better solution is found. The tests can be found in Table 6.4.

6.2. Validation

The lock scheduling model is compared to SIVAK. Therefore the model assumptions should be the same in both models. In practice this means that some simplifications were made. In this section the impact of these assumptions is estimated, in order to validate the model.

6.2.1. Water Height

In the lock scheduling model and in SIVAK, the water level difference was assumed to be constant. In practice, tides are the main factor for the water level difference of many locks in the west of the Netherlands. The effect of the tides is that the processing times of the locks can vary, some lockages will be shorter and others longer. Tides can be simulated in SIVAK already. Tides are predictable, so they can be included in the lock scheduling model as well. The effect of adding tides will make lock processing times shorter and longer with it's imposed water level. This means that also passage times will follow this pattern. The capacity of the lock is thus dependent on the phase of the tide.

Table 6.2: Verification test for the area check. TWBF refers to the Three-Way Best-Fit algorithm.

	Description	Function	Vessels Length (m)	Vessels Width (m)	Lock Length (m)	Lock Width (m)	Output
Test 1	Vessels just fit in width	TWBF	50 50 50 50	4.5 5.5 6 4	120	10	True
Test 2	Vessels don't fit in width	TWBF	50 50 50 50	4.5 5.6 6 4	120	10	False
Test 3	Vessels just fit in length	TWBF	70 50 60 60	5 5 5 5	120	10	True
Test 4	Vessels don't fit in length	TWBF	71 50 60 60	5 5 5 5	120	10	True
Test 5	N vessels exactly fit	TWBF	50 50 60 40 40	10 10 10 5 5	100	30	True
Test 6	N vessels fit in order	Order fit	40 35 40	4 8 4	100	10	False
Test 7	Ignore order	TWBF	40 35 40	4 8 4	100	10	True
Test 8	Good order	Order fit	35 40 40	8 4 4	100	10	True

Table 6.3: Test 9 for chunking, resulting in 3 chunks. Vessel 0-6, vessel 7-12 and vessel 13-19

Vessel	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**
ETA (min.)	2	8	8	9	9	9	10	12	14	17	19	24	25	35	35	40	42	46	46	47
ETA diff. (min.)	2	6	0	1	0	0	1	2	2	3	2	5	1	10	0	5	2	4	0	1
Chunk added (order)		*2*						*5*		*4*		*3*		*1*						
Chunk deleted (order)		*6*								*7*		*8*								

Table 6.4: Test 10 and 11 for maximum waiting time verification.

	Description	Vessels ETA (min.)	Length (m)	Width (m)	Dir (l/r)	Lock # chambers	Length (m)	Width (m)	Output
Test 10	Max. wait time increase	0	100	10	l	1	100	10	Maximum waiting time > 60
		0	100	10	l				
		0	100	10	l				
		0	100	10	l				
Test 11	Run 1: Max. waiting time = 60	0	50	5	l	1	100	10	Objective run 1 < objective run 2
		0	50	5	l				
		30	100	10	r				
		50	50	5	r				
		50	50	5	r				
Test 11	Run 2: Max. waiting time = 90	0	50	5	l	1	100	10	
		0	50	5	l				
		30	100	10	r				
		50	50	5	r				
		50	50	5	r				

6.2.2. Passage Times

The SIVAK model allows to specify the entering and exiting times based on ship class, width of the vessel and the loading of the vessel. However in SIVAK as well as the lock scheduling model, the entering and exiting times were simplified. The entering and exiting times were set to 2 minutes for all vessels. The minimum spacing for vessels that subsequently enter or exit a chamber is set to 2 minutes as well.

In reality these times are variable and dependent on the situation at the lock. Also extra time should be added between the last vessel that leaves the chamber and the first one to enter the chamber again. In SIVAK this is referred to as switch time correction.

It can also be argued that the last vessel to enter the lock needs some more time if the space is small and navigating should be done carefully, this was also included in SIVAK and is called squeezing in that model.

These decisions can best be made by gathering data on real entering and exiting times. And then calibrating the model with that data. The result will be a better schedule.

6.2.3. Safety Allowance

The safety allowance does have an impact on the results. Safety allowance means that every vessel needs to keep a certain safety distance from other vessels in the lock. There is also the case of cone vessels. Cone vessels do need larger spacing and sometimes they are even required to be locked alone. To do this more efficiently, there are sometimes lock doors in the middle of the chamber.

Safety allowance was omitted altogether in the lock scheduling model as well as in the SIVAK model. This can make a difference for the results. The model that has the highest number of vessels per locking will be at a bigger disadvantage.

Cone vessels do however make the model faster to solve, as they decrease the number of scheduling options with their large spacing requirement.

6.3. Input Validation

Also the data that is input to the model is validated. Values that are known to be invalid are entered, after which the model should report an input error. The following

3 aspects are tested with respect to input validation:

1. All input values should be an integer, as this is required by the solving strategy of combinatorial optimization;
2. All parameters should be in the defined range (these ranges can be defined during the model development as this is dependent of the performance of the model);
3. If a dimension of the vessel is larger than the largest chamber, an error should be returned;

6.4. Conclusion

In this chapter the fifth sub question was answered:

How can the lock scheduling model be verified and validated?

The conclusion for verification is that the model passed the tests. The tests can be found on GitHub, the link to which can be found in Appendix C.

For validation the conclusion is more nuanced. The SIVAK model and the lock scheduling model only differ in the decisions that are made. This makes them comparable. They do however differ with respect to reality. The most important differences for the results of the model are the safety allowance and the absence of tides. The entering and exiting times also cause a difference.

7

Results

In this chapter the sixth sub question is answered:

How does the lock scheduling model perform in minimising vessel passage time compared to a baseline?

In this chapter, the LOSCO model is compared with the SIVAK model. Ideally, real passage times are used, but these cannot be derived from the data that is available (Chapter 3). Instead SIVAK is used, as Rijkswaterstaat uses this model to assess the capacity of locks and bridges and to plan possible investments, which reflects the confidence that is put in SIVAK.

In Section 7.1, the optimal simulation settings for SIVAK are explained. Then, in the Section 7.2, the optimal settings for the LOSCO model are discussed and non-efficient heuristics, non-FCFS and resolution, are discovered and eliminated. In Section 7.3, the best settings for the maximum waiting time and maximum chunk size are derived. The SIVAK and LOSCO models are compared to each other for three lock complexes for 3 weeks. The crowdedness of the three weeks are varied to be on the 3 quartile borders. In this section the behaviour of the model is described. In Section 7.4, runs are done for a year to achieve more statistically significant results. In Section 7.5 some more experiments are done to compare the models in relation to the maximum capacity of the locks. In Section 7.6, the chapter is concluded.

All code that was used in this chapter can be found on GitHub, the link to which can be found in Appendix C.

7.1. SIVAK Simulation Settings

SIVAK can be run with different settings. In order to achieve a comparison with the LOSCO model the best SIVAK settings are used. The best settings are the settings that lead to the least average lock passage times. An overview of the SIVAK model can be found in Chapter 4.

Initially, 9 SIVAK experiments were set up with different locking regimes (Table 7.1). A locking regime consists of the decision rules that are made to decide which vessels are assigned to which chamber and at what times the locking in a chamber

should be started. Three properties of the locking regimes were varied. First, the range, that is the distance that is looked ahead to new vessels that are arriving at the lock. Second, the percentage of the area of the chamber that is minimally required to be filled with vessels. Third and last, the relation between chamber area use percentages on the open and on the closed side of the lock.

A lot of other settings are the same for each experiment, to achieve the best comparability. For instance the maximum waiting time that will overrule any regime. The maximum waiting time is set at 30 minutes. If this waiting time is reached for any vessel, levelling will be initiated. Processing times of the lock are set at 30 minutes and the entering and exiting times of the vessels is set at 2 minutes for every vessel.

Table 7.1: SIVAK Settings

Name	Regime	Combination curve
A	no regime (5 min.)	-
B	no regime (10 min.)	-
C	40%, 80%	block
D	40%, 80%	ellipse
E	40%, 80%	linear
F	20%, 60%	block
G	20%, 60%	ellipse
H	20%, 60%	linear
I	no regime (15 min.)	-

7.1.1. Range

The range is specified in SIVAK as a distance. In the experiments, all vessels are assigned the same speed of 15 km/h, so that it can also be stated as a time. All vessels that are within the range of the lock, on both sides, are considered in the locking regime. Vessels that are outside of this range are unknown to the lock in the SIVAK simulation. Three different ranges are considered in the experiments. The shortest used is a range of 1.25 km, which translates to 5 min. at 15 km/h. Also 10 minutes (2.5 km) and 15 minutes (3.75 km) was used.

7.1.2. Minimal lock area utilisation

A minimal lock area utilisation can be specified for both the open and the closed side of the lock. This is the summation of the vessel areas of the vessels that are waiting on one of the sides of the lock, divided by the total chamber area. Two different settings for minimal lock area utilisation are used in the experiments. The first requires minimally 40% of the lock on the open side to be filled or otherwise 80% on the closed side. The second setting requires minimally 20% of the lock on the open side to be covered or otherwise 60% on the closed side. There is also an option to not specify the minimal lock area utilisation, this setting is referred to as 'no regime'. In this case, the levelling will be initiated once there is a vessel in the chamber, if there is another vessel within the range it will also be added to the same levelling. Locking initiates if the chamber is full or if there is no vessel in range on the current side of the lock.

7.1.3. Combination curves

All the combination curves were used: block, linear and ellipse. The combination curve defines the relation between the utilisation on the open and on the closed side of the chamber and the relation between the utilisation and the minimal utilisation. These were treated in Chapter 4. Note that in the case of 'no regime', no combination curve is used.

7.2. LOSCO Model Settings

Similarly to the SIVAK model, also the LOSCO model can be run with different settings. In Chapter 5, methods were introduced to decrease the running time whilst still maintaining a good solution. In this section, experiments are executed with different settings and the best settings are selected. The settings that are varied are the maximum chunk size, the resolution, the maximum waiting time and first come first served rule (FCFS).

A thousand vessels are generated from a exponential distribution with a parameter of 10. In Figure 7.1 the ECDF (Empirical Cumulative Distribution Function) of this sample can be compared to the real world data for three locks. Vessel sizes are randomly drawn from the data at the three locks, the Krammersluizen, the Sluizen Hansweert and the Kreekraksluizen. With these data initial tests can be run to eliminate inefficient settings. The tests are subject to a 2 hour runtime limit. This is a practical limit. As chunks are around the size of 20-30 vessels, the minimum number of chunks is around 30 and this makes the maximum runtime per chunk around 4 minutes. If the runtime is longer than this, it might not be practical in reality, as a schedule should be ready before the first vessel in a 20 to 30 vessel sequence arrives.

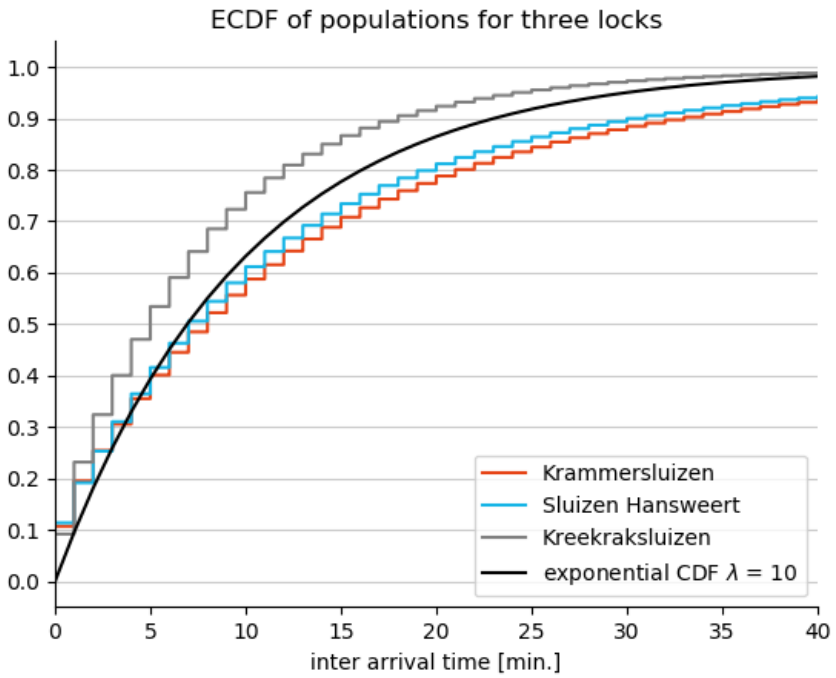


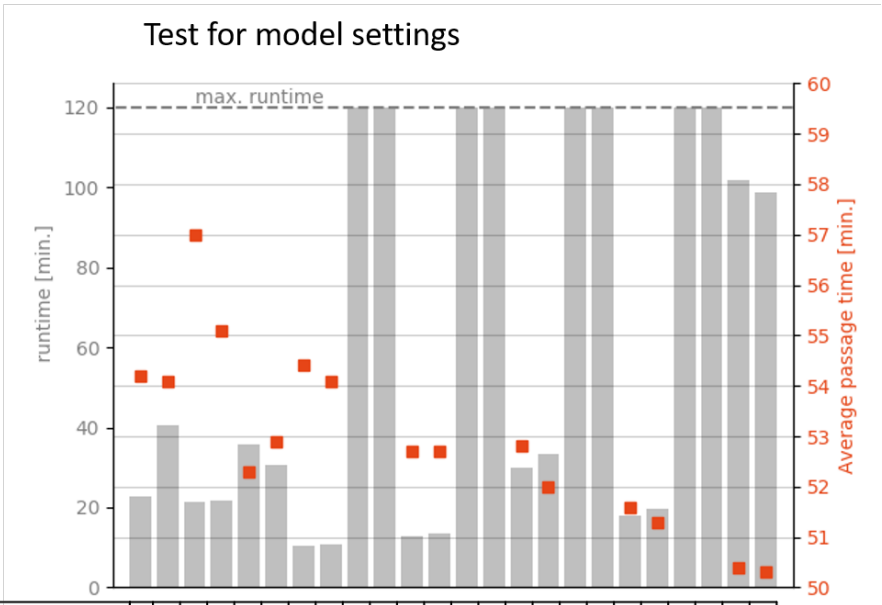
Figure 7.1: The empirical cumulative distribution function of the inter arrival times at three different locks.

Tests were done for maximum chunk sizes of 20 and 25 vessels in combination with maximum waiting times of 30, 45 and 60 minutes. These 6 tests were done with four combinations of resolution and FCFS. A resolution of 1/2 and 1 and with the FCFS turned on and off. This makes a total of 24 runs. The experiments are performed on a Toshiba Satellite C55 with an Intel Pentium CPU 2020M @ 2.40 GHz and 8 GHz of memory, running Windows 10 in 64-bits.

In Figure 7.2 the runtime and the average passage time are displayed for runs with different settings. When FCFS is off, the model fails to produce any results within 2 hours for all but max chunk size = 20 and max waiting time = 30, 45. While the quality of these results is quite good, the quality of the FCFS can be better in combination with other settings for maximum waiting time and chunk size. The FCFS option provides better results for the time it takes to calculate. Therefore runs without FCFS are eliminated.

What can also be observed is that reducing the resolution to a half barely has an effect, the runtime is barely shorter and the solution is worse. Therefore the resolution should be taken as 1.

The last setting that can be eliminated is the maximum waiting time of 30 minutes, as this does not produce any sufficient results. In this case the best solution is achieved with a maximum chunk size of 25, a maximum waiting time of 60, a resolution of 1 and FCFS turned on. Note that the runtime for this best solution is close to the max. For maximum chunk sizes of 30 vessels or more, no better solutions can be found within this maximum runtime.



FCFS	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
Resolution	½	1	½	1	½	1	½	1	½	1	½	1	½	1	½	1	½	1	½	1	½	1	½	1
Max. wait time	30				45				60				30				45				60			
Max. chunk size	20												25											

Figure 7.2: The tests from which can be concluded that FCFS = 0 and resolution = 0.5 can be eliminated

7.3. Week runs

To do the experiments with all data would take too much time to run and would therefore be inefficient. That is why samples of a week are taken of the data set that was discussed in the previous chapter. One way to generate samples is to fit a parametric distribution as was done in the previous section. It is well known that the exponential distribution is a good choice for fitting inter arrival times. By fitting a exponential distribution all dependencies would however be lost. Since the dependencies can be significant, another sampling method was chosen. For each lock 3 weeks in the data set are selected.

The data consists of times of sign up at the lock and vessel data. For three locks three weeks were scheduled. With SIVAK as well with LOSCO, so that they can be compared. One of the weeks corresponds to the 25% percentile of the busiest weeks. One week corresponds with the median (50% percentile). And one week was randomly chosen in the last quartile (>75% quantile). The distributions of the number of vessels per week for the Krammersluizen, Sluizen Hansweert and Kreekraksluizen are displayed subsequently in Figure 7.3, Figure 7.4 and Figure 7.5.

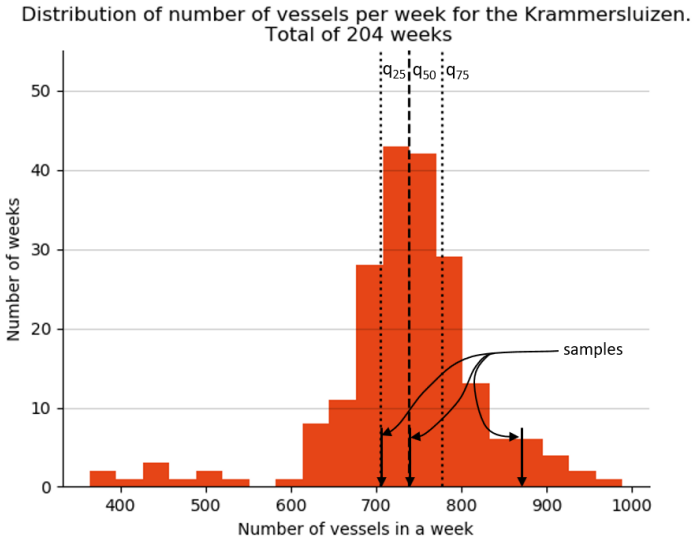


Figure 7.3: Distribution of the number of vessels per week for the Krammersluizen.

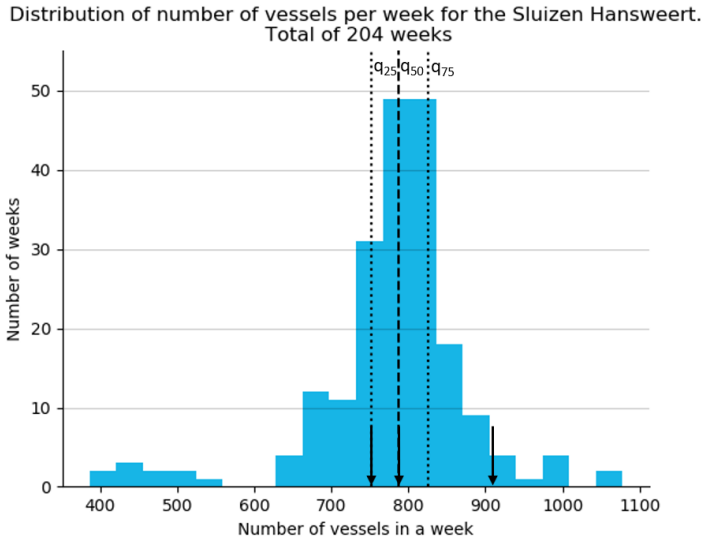


Figure 7.4: Distribution of the number of vessels per week for the Sluizen Hansweert

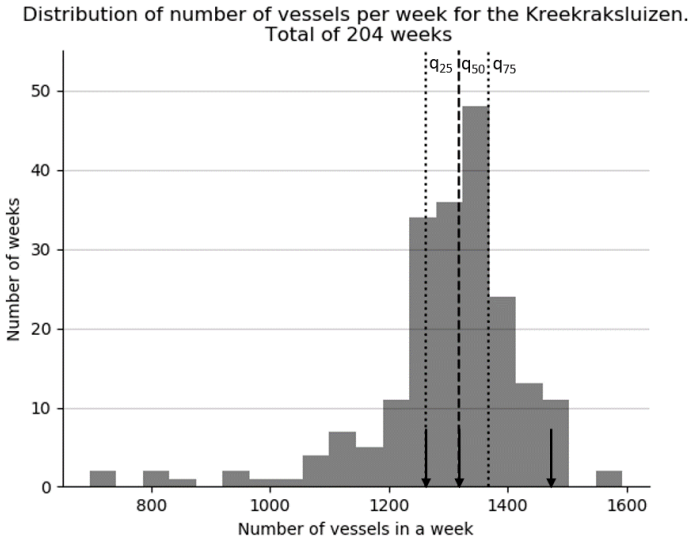


Figure 7.5: Distribution of the number of vessels per week for the Kreekraksluizen

Table 7.2: Scheduling of weeks input

Lock	Test	Num. of Vessels'	Week	Year	$\mu_{int.arr.}$ [min.]	$q_{25\%} of rollingmean$ [min.]
Krammersluizen	$q_{75\%}$	873	47	2016	11.2	7.6
Krammersluizen	$q_{50\%}$	739	10	2016	13.1	8.8
Krammersluizen	$q_{25\%}$	705	15	2016	14.0	10.8
Sluizen Hansweert	$q_{75\%}$	906	27	2015	10.7	6.9
Sluizen Hansweert	$q_{50\%}$	788	49	2016	12.4	8.4
Sluizen Hansweert	$q_{25\%}$	752	46	2015	12.7	7.4
Kreekraksluizen	$q_{75\%}$	1476	46	2015	6.8	5.0
Kreekraksluizen	$q_{50\%}$	1318	26	2015	7.5	5.6
Kreekraksluizen	$q_{25\%}$	1263	31	2015	7.9	5.8

Table 7.3: Scheduling of weeks results

Lock	Test	Best SIVAK [min.]	$\sigma_{\mu,SIVAK}$	Best LOSCO [min.]	$\sigma_{\mu,LOSCO}$	Difference	$\sigma_{\mu,diff.}$
Krammersluizen	$q_{75\%}$	55.3	0.65	50.0	0.49	5.3	0.81
Krammersluizen	$q_{50\%}$	51.9	0.58	46.0	0.39	5.9	0.70
Krammersluizen	$q_{25\%}$	50.2	0.55	43.8	0.36	6.4	0.66
Sluizen Hansweert	$q_{75\%}$	57.0	0.63	52.4	0.48	4.6	0.79
Sluizen Hansweert	$q_{50\%}$	53.9	0.65	47.4	0.43	6.5	0.78
Sluizen Hansweert	$q_{25\%}$	54.6	0.65	49.5	0.49	5.1	0.82
Kreekraksluizen	$q_{75\%}$	83.0	1.0	-	-	-	-
Kreekraksluizen	$q_{50\%}$	72.6	1.1	80.3	1.2	-7.7	1.6
Kreekraksluizen	$q_{25\%}$	68.3	0.85	67.6	0.82	0.7	1.2

In Table 7.2 the input data for the 9 weeks that were sampled can be observed.

The results of the scheduling can be observed in Table 7.3. The LOSCO model is successful in the cases of the Krammersluizen and the Sluizen Hansweert. Where the average passage times can be reduced by around 5 minutes. The LOSCO model fails to produce good results for the Kreekraksluizen. In the following subsections the differences will be explained.

7.3.1. Best SIVAK and LOSCO runs

The best SIVAK and LOSCO runs can be illustrated with the experiment for the week in the last quartile at the Krammersluizen. In Figure 7.6 all SIVAK as well as all LOSCO runs are displayed for the busiest tested week for the Krammersluizen. SIVAK_B and SIVAK_I are the best regimes of all SIVAK runs, this is also the case for all other weeks and locks that were tested. Both of these use the setting 'no regime'. The best SIVAK run therefore does not seem to depend on the crowdedness at the locks. For the LOSCO runs, most of all cases a maximum chunk size of 25 vessels and a maximum waiting time of 60 minutes produces the lowest average passage time. This is true for all but the Hansweert q_{75} case, where a maximum chunk size of 30 in combination with a maximum waiting time of 45 minutes is optimal. In Figure 7.6, all SIVAK and all LOSCO runs are compared. In this case the best LOSCO regime (avg. passage time of 50.0 min.) is 5.3 minutes better than the best SIVAK regime (avg. passage time of 55.3 min.).

The results can also be displayed as in Figure 7.7. In this figure the rolling means over 20 vessels for the average passage times as achieved by the SIVAK and the LOSCO model are displayed. Also the development of the mean and the 95% confidence interval of the mean is displayed. On the right axis, the rolling mean for the inter arrival times is plotted on top of the other graphs. With this graph the crowd-

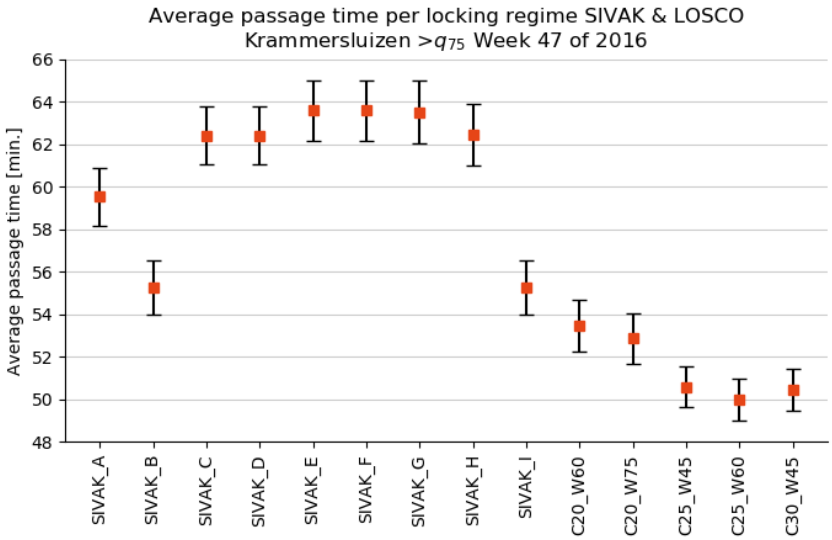


Figure 7.6: Means and 95% confidence intervals for all Krammersluizen q_{75} week runs

edness can be estimated, a clear day and night pattern can be observed. The SIVAK model is worse on average, however, when it is very busy the model achieves better results at times than the LOSCO model (eg. on Thursday evening can be observed that SIVAK can manage to achieve lower passage times than the LOSCO model). The graphs for all other experiments can be found in Appendix E.

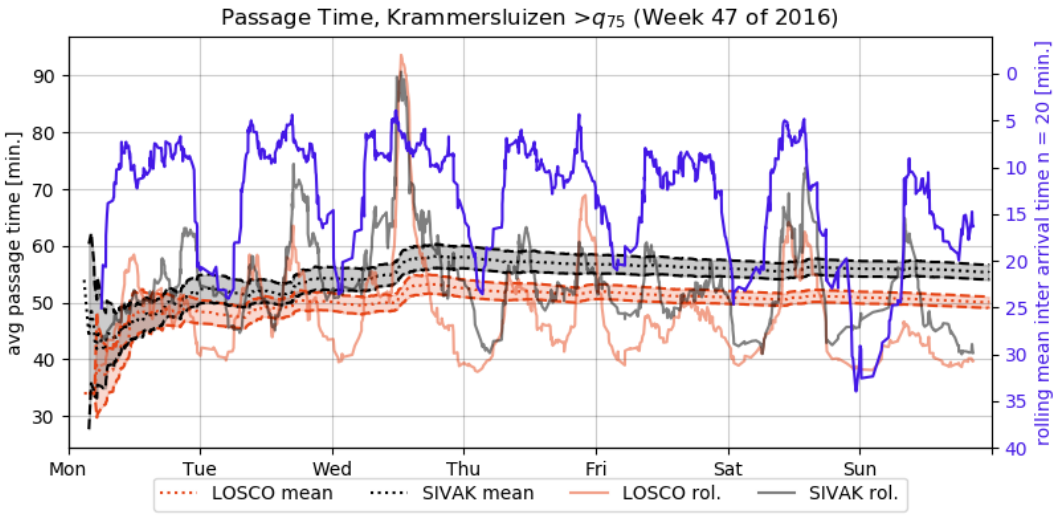


Figure 7.7: The best SIVAK run and the best LOSCO run for the Krammersluizen q_{75} week

7.3.2. Interpretation as Waiting Time

It might be more meaningful to interpret the passage time as waiting time. The minimum passage time is 34 minutes, 30 minutes of locking and 2 minutes of sailing in and out. If the minimum passage time is substracted from the passage time, the waiting time is obtained (also the waiting inside the lock when other vessels sail in

or out). This is when the value of the LOSCO model really becomes clear. For the Sluizen Hansweert and Krammersluizen tests, a 20% to 40 % reduction in waiting time can be observed (Figure 7.8). This does not apply to the Kreekraksluizen tests, as these are too busy.

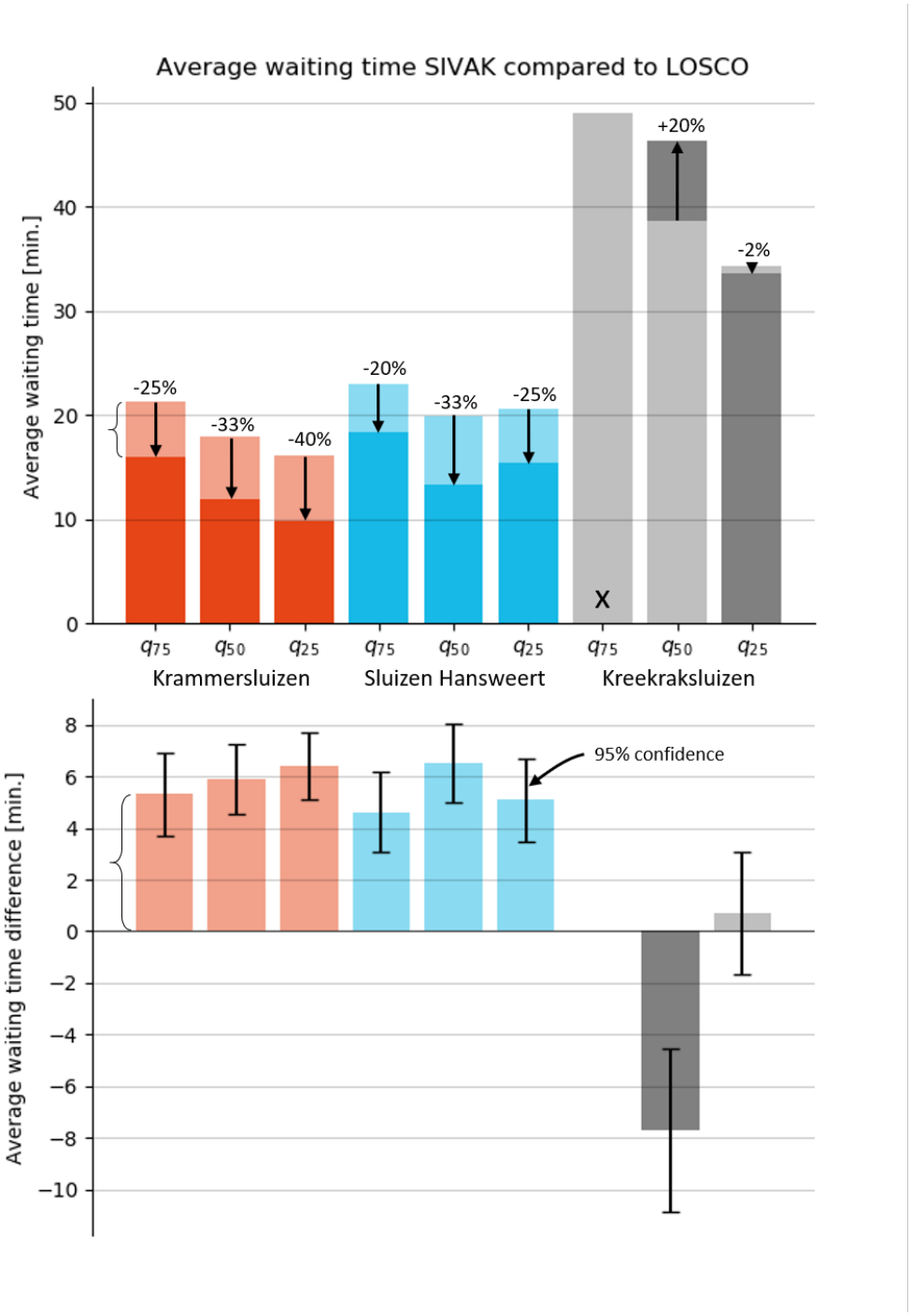


Figure 7.8: The results as average waiting time.

The runtimes for the test are displayed in the graph of Figure 7.9. The runtime has a weak relation with the average passage time that is achieved. Randomness plays a large role in the runtime. It is a matter of chance if the model can find a set

of vessels that exactly fits in the lock (area check). If this set does not fit, another iteration is tried. Another reason is the nature of the algorithm, as it scales with a factorial. The differences in runtimes for chunks have been observed to a order of 10^3.

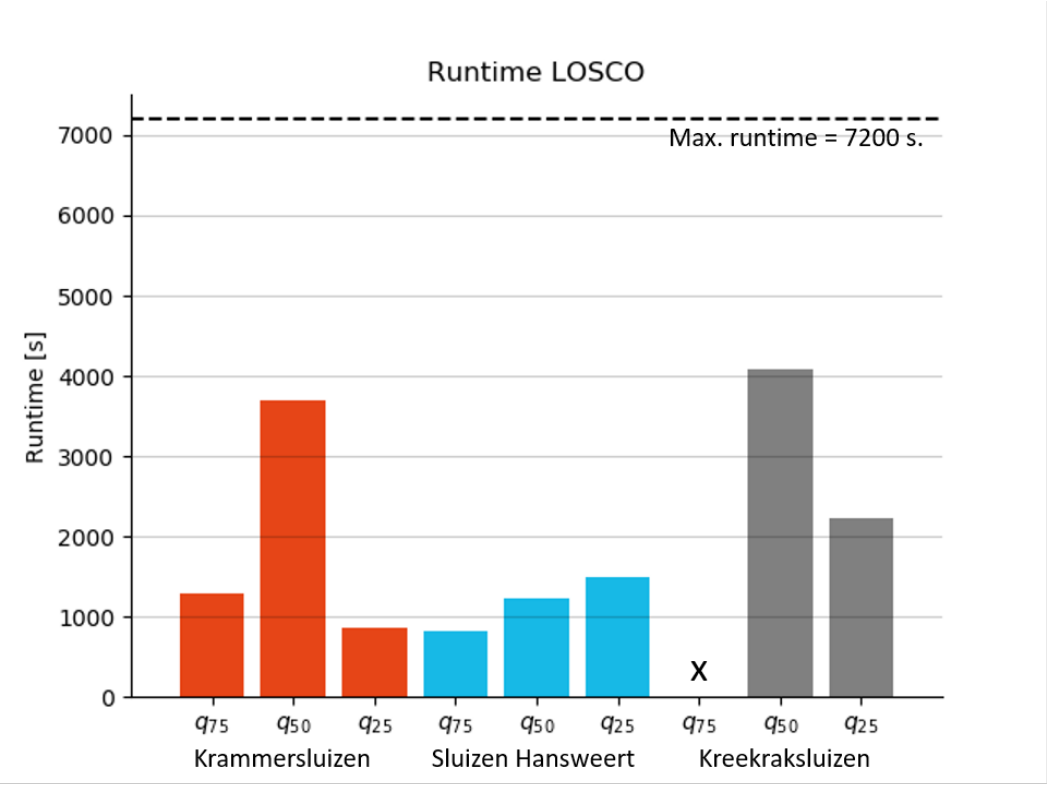


Figure 7.9: The runtime per experiment

7.3.3. Notable Hansweert Results

The results for Sluizen Hansweert q_{25} and Sluizen Hansweert q_{50} can be considered surprising as the average interarrival time is lower for the q_{25} case (12.4 minutes against 12.7 minutes), but the SIVAK as well as the LOSCO results are in favour of the q_{50} case. In Table 7.2 can be observed that a quarter of the vessels is spaced within 7.4 minutes of each other in the q_{25} case, while in the q_{50} case a quarter of the vessels is spaced in less than 8.4 minutes, meaning that the peaks in crowdedness are busier in the q_{25} week.

7.3.4. Kreekraksluizen Results

The Kreekraksluizen are too busy for the LOSCO model. In Figure 7.10 the development of the rolling means can be observed. Again, it is the peaks in crowdedness that can explain the differences. On Monday, Tuesday, Wednesday and Thursday afternoon and evening, the LOSCO model cannot deal with the crowdedness as well as the SIVAK model. For the least busy week, the LOSCO model, can compensate during less busy times, but for the median and most busy week, the SIVAK model performs better in total.

In Figure 7.11, the main problem of the LOSCO model is shown. In Chapter 5 an explanation This is a case of the beginning of Monday evening for the least busy

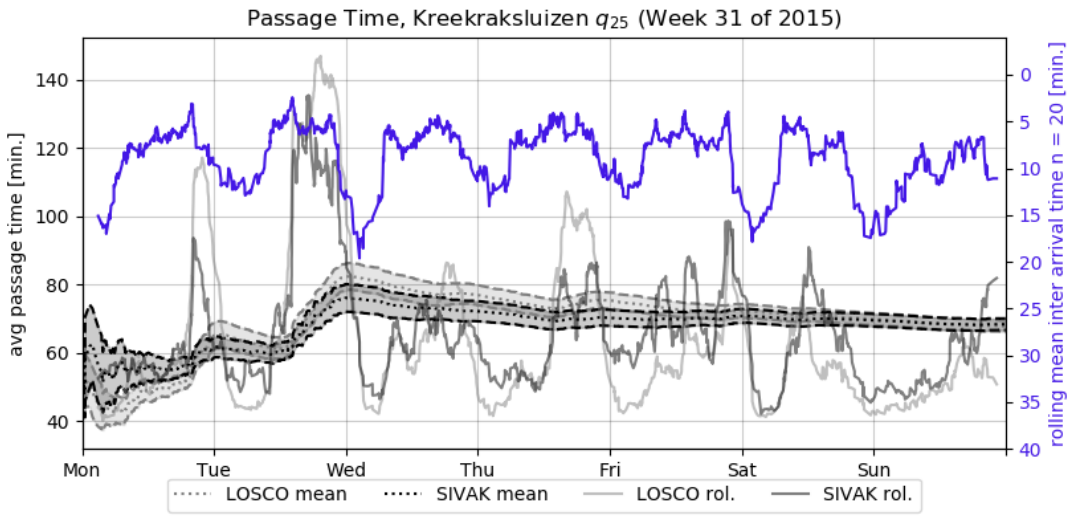


Figure 7.10: The best SIVAK run and the best LOSCO run for the Kreekraksluizen q_{25} week

Kreekrak week. The LOSCO model decided to create a new chunk from vessel 132, as the gap between the inter arrival times of vessel 131 and 132 is relatively big. Therefore, the vessels that arrived earlier than vessel 132 are independently scheduled from the vessel that arrive after vessel 132. A lot of vessels therefore have to wait for vessel 128 in the new chunk. The waiting times grow consequently and in Figure 7.10 can be observed that on this evening, the passage times at a certain moment surpass 100 minutes. Almost until midnight, after the peak in crowdedness has long gone.

In contrast with this observation is Figure 7.12. The spacing between the ETAs of vessel 95 and 96 is larger, which makes it more suitable for a chunk. As a consequence, the passage times after the chunk increase by a lesser amount.

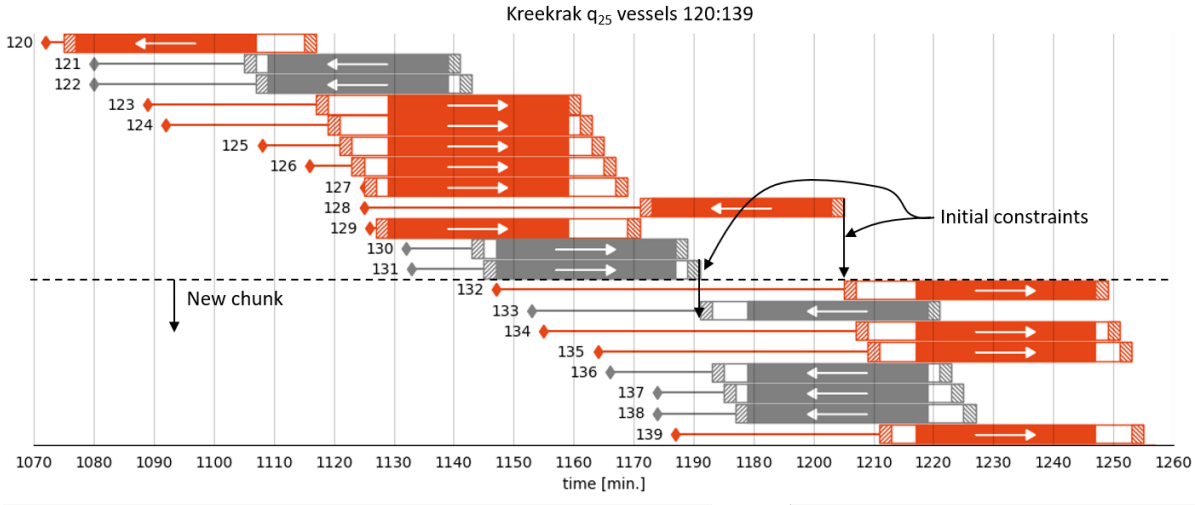


Figure 7.11: The effect of chunking.

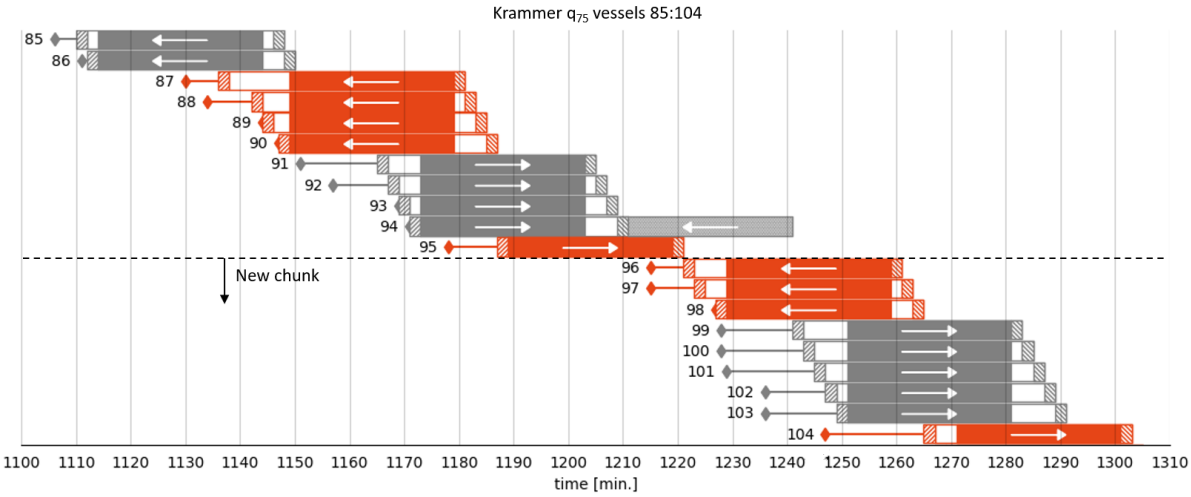


Figure 7.12: The effect of chunking.

7.4. Year runs

To achieve a more statistically significant result, a year was run for the Sluizen Hansweert and the Krammersluizen. For both locks, the data of 2016 was taken. Over the year 2016 on average every 14.0 minutes a vessel arrives at the Krammersluizen. For the Sluizen Hansweert, this is 13.0 minutes on average. The Kreekraksluizen are busier at a rate of 7.8 minutes on average between vessel arrivals. No solution for the Kreekraksluizen could be found. The results are displayed in Figure 7.13. In the left figure, the results with respect to the average passage time is shown, the same results are shown in the right figure, but in terms of average waiting time. For the Krammersluizen run, the LOSCO model (48.4 min./vessel) is 3.9 $\pm$ 0.12 SE minutes per vessel better than the SIVAK model (52.2 min./vessel). For the Sluizen Hansweert, the LOSCO model (52.9 min./vessel) is 2.0 $\pm$ 0.14 SE minutes per vessel better than the SIVAK model (54.9 min./vessel).

7.5. Result in Relation to Capacity

The capacity of a lock is not an obvious concept. As this thesis shows, the capacity might change depending on the operational management of the lock. The main problem is that vessel sizes differ. If all vessel sizes would be the same, it is easy to say that in every locking so many vessels fit. The number of vessels can than be divided by the processing time to achieve a capacity estimate. Nevertheless estimates for the capacity can be made.

All vessels that pass in the year 2016 are assumed to be waiting for the lock from the first moment in a long row, so that the lock never has to wait for vessels. They are added to the chambers in order of arrival (FCFS). They are checked for a fit in the chamber using the LOSCO Area Check, so that the chambers are optimally filled with vessels. The processing time for this amount of vessels is calculated as 2 minutes for entering and exiting per vessel and 30 minutes for the locking itself. This is repeated until all vessels have passed the lock. The capacity can now be estimated as the total amount of vessels divided by the total amount of time it took to process the vessels. The results for these capacity estimates can be observed in Table 7.4 .

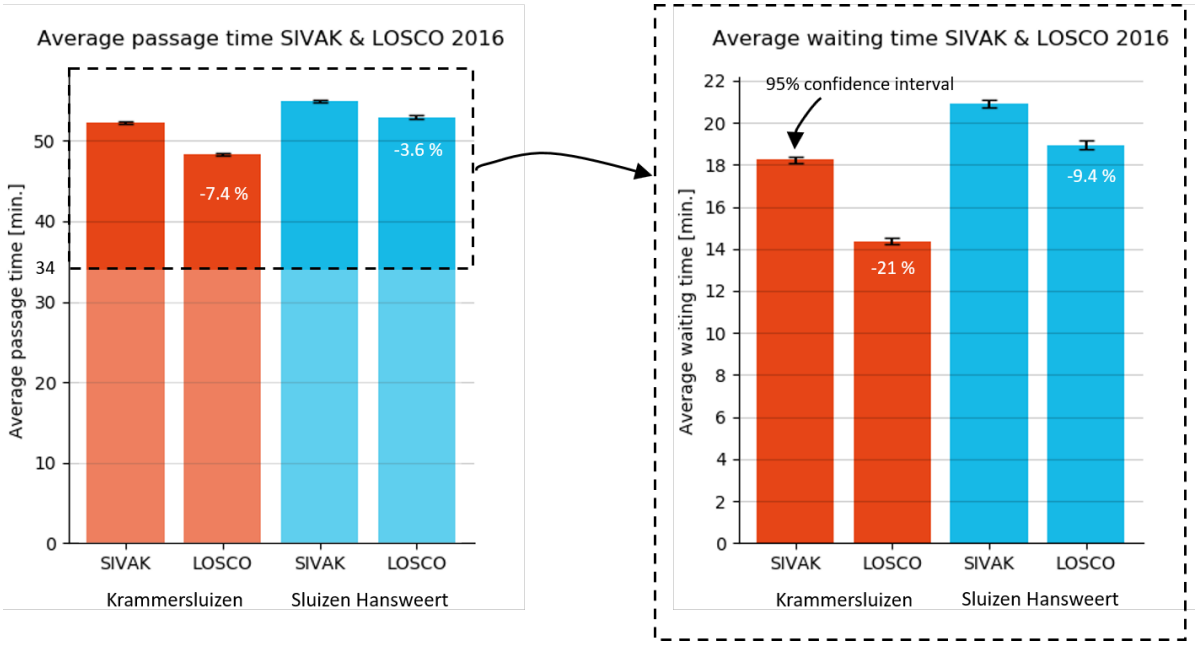


Figure 7.13: Runs for all vessels in 2016. On the left displayed as passage time, on the right as waiting time (the minimum passage time of 34 minutes is subtracted).

Table 7.4: Lock capacities. Note that all these three locks have 2 chambers.

	Krammersluizen	Sluizen Hansweert	Kreekraksluizen
capacity (ves./h.)	10.94	11.32	11.24
capacity (min./ves.)	5.5	5.3	5.3
lock length (m)	280	280	318
lock width (m)	24	24	24
avg. vessel length (m)	97.0	93.3	103.5
avg. vessel width (m)	10.5	10.2	11.0
avg. vessels/locking	4.31	4.55	4.49

The difference between the capacity of the Krammersluizen and the Sluizen Hansweert is especially interesting, as the chambers of these two locks are exactly the same size. The processing time is assumed the same as well, therefore the difference in the capacities of the locks is entirely up to the differences in vessel sizes at the two locks. As can be observed in Table 7.4, the capacity of the Sluizen Hansweert is higher, reflecting the smaller average vessel size at this lock.

The performance of the SIVAK and the LOSCO model can be displayed in relation to this capacity. In order to do this a fourth experiment was initiated. In this experiment, 1000 vessels are generated with exponential distributions with average inter arrival times ranging from 20 to 5 minutes. The vessel sizes are randomly drawn from the data at each lock. Every 1000 vessels can be processed by both models to achieve the graphs displayed in Figure 7.14. In this graph it can be observed that the SIVAK model starts to outperform the LOSCO model when the intensity is getting below 9 minutes of inter arrival time. Below this intensity the average passage time starts to climb rapidly, and as a result, below 8 minutes of inter arrival time, the LOSCO model, cannot even find a solution anymore as the calculation takes too

long. 95% confidence intervals cannot be generated for this experiment as there are 2 variables that influence this graph, the vessel size and the inter arrival time.

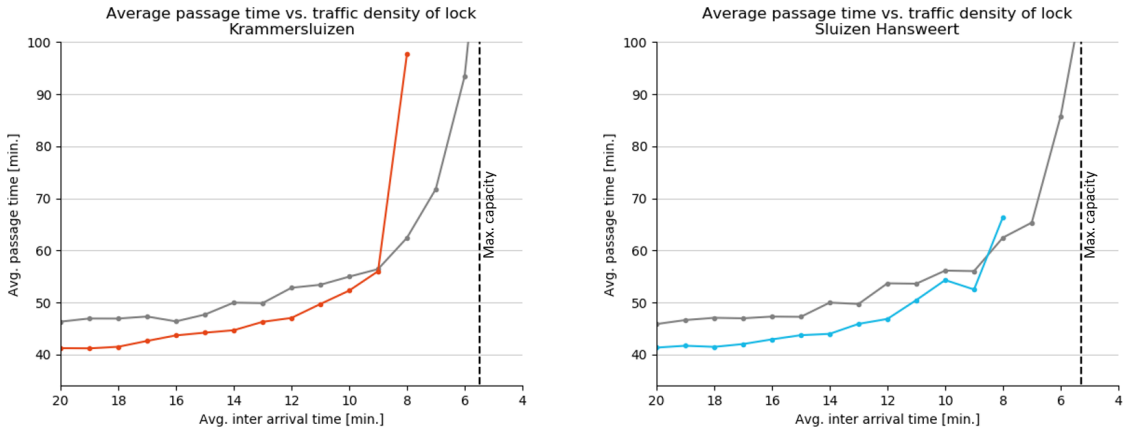


Figure 7.14: Experiments for which the inter arrival time is varied. The LOSCO model is better at longer inter arrival times, SIVAK is better when it is busy.

7.6. Conclusion

In this chapter, the experiments are presented to answer the following sub question:

How does the lock scheduling model perform in minimising vessel passage time compared to a baseline?

In order to achieve the best possible comparability between the SIVAK and LOSCO model, the best settings for both were selected. The best settings for the SIVAK model at the selected locks were found to be without a regime and with a range of 10 to 15 minutes.

In a first experiment for the LOSCO model, the no-FCFS setting and the resolution were found to not be effective.

In a second experiment, three weeks were scheduled for each lock, three weeks that differ in the amount of vessels that passed the lock. In this experiment was found that a maximum chunk size of 25 vessels in combination with a maximum waiting time of 60 minutes is most effective. For these settings, the calculation time stays within the set limit of 2 hours for 1000 vessels, and produces the best solution quality. For the Krammersluizen and the Sluizen Hansweert, the lock scheduling model manages to reduce the average passage time per vessel with around 5 minutes. However for the, the Kreekraksluizen, which are busier, the LOSCO model is outperformed by SIVAK. The LOSCO model was also found to converge slowly, so that after a week of scheduling the Hansweert median case could still achieve a lower passage time than the Hansweert case with the least vessels in a week. The SIVAK model constantly outperforms the LOSCO model if the lock gets too busy. This happens because the last vessels in a chunk are scheduled in the lock when the first vessels of the new chunk already arrive. This causes an overlap. When it is quiet at the lock, this overlap does not form, and then the LOSCO model outperforms the SIVAK model, because it makes better decisions. Decisions that consider each of the

vessels in the chunk.

The model was run to schedule a year at the Krammersluizen and the Sluizen Hansweert for the data of 2016. Not for the Kreekraksluizen, as this would take too long to run. At the Krammersluizen, the LOSCO model was 3.9 ± 0.12 SE minutes per vessel better. For the Sluizen Hansweert, the model was 2.0 ± 0.14 SE minutes per vessel better.

To identify at which intensity of vessels per hour the SIVAK model starts to outperform the LOSCO model a fourth experiment was done. In this experiment was found that below an intensity of 9 minutes of inter arrival time, the SIVAK model works better. And rapidly at 7 minutes of inter arrival time, the LOSCO model cannot even find a result anymore.

Overall, the LOSCO model is better than the SIVAK model if it is not to crowded. The boundary seems to lie at an average inter arrival time of about 9 minutes for the locks that were investigated. However, the average inter arrival time might not be the most important statistic, as the differences arise in extremes, the busiest time of the day, that is typically midweek in the afternoon.

8

Economic Optimisation

This purpose of this chapter is to answer the following two sub questions:

Is it better to minimise economical value rather than minimising vessel passage time, as this serves the economy better?
Is it fairer for larger vessels to minimise economical value rather than passage time?

In Section 8.1 the method of carrying out an economical optimisation and the motivation behind it is explained. The results follow in Section 8.2. In Section 8.3, this chapter is concluded.

All code that was used in this chapter can be found on GitHub, the link to which can be found in Appendix C.

8.1. Method

An economic optimisation is carried out for two reasons. The first reason is that that an economic optimisation might be more beneficial to the economy instead of optimisation on passage time. The second reason is that an economic optimisation might be fairer than optimisation on passage time.

If lock scheduling is optimised on vessel passage time, the schedule might not be fair. Larger vessels will in theory be disadvantaged as their vessels fill a larger share of the lock chamber. When optimisation is done on time only, it is more efficient for the model to assign multiple smaller vessels to the same space that a large vessel can be assigned to. In that case the total passage time will decrease faster as every vessel is weighed the same. In this chapter, the lock scheduling model is optimised on the economic benefit. It is argued that optimising on a monetary value can increase the fairness of scheduling. Moreover, not only the decrease of lock passage time is of value for vessels, also the predictability of travel time has a value. Therefore also the deviation in passage time is considered in the optimisation calculation.

The Kennisinstutuut voor Mobiliteitsbeleid (2018) did a survey in different transport sectors to the value of time (VoT) and the value of reliability (VoR). The purpose of this study is to be able to compare infrastructural changes more objectively, by

Table 8.1: The results of Kennisinstutuut voor Mobiliteitsbeleid (2018) for IWT for waiting times at locks.

	Trade-Off Ratio (TR)	Reliability Rate (RR)
Container	1.16	0.08
Non-Container	1.06	0.08

assigning a monetary value to the advantages and disadvantages. The study is based on costs that vessels make and a estimate of profit was also made by doing interviews. The results that are used in this study are displayed in Table **??**. The value of time is equal to the factor costs multiplied by the trade-off ratio. The value of reliability is equal to the standard deviation of the factor costs multiplied with the reliability rate. The costs of transport can be found in the 'kostenbarometer' (Rijkswaterstaat, 2018), which was created by Panteia. The table for factor costs per hour is displayed in Appendix F. Vessels are divided in different ship classes. And also a rough division is made between different goods transported. The VoT and the VoR can be used to get an indication of the economical value if they are multiplied by respectively the passage time and the standard deviation of the passage time.

The economical optimisation is only based on the VoT value, not on the VoR value. The VoR value is calculated only after optimisation. To take into account the VoR value in the optimisation would require changes in the model that take too much time in comparison with the extra benefit that is gained, as the reliability rate is way smaller than the trade-off ratio. As larger vessels also make more costs it is expected that the fairness will be restored again using this method.

8.2. Results

8.2.1. Contibution to Economy

The results for the optimisation on value of time are displayed in Figure 8.1. As can be observed, the optimisation on value of time is not very effective compared to the optimisation on time. The time optimisation arrives at similar costs as the optimisation on value of time. The results are even within each other's 95% confidence interval.

The results can also be presented as a value per year that is saved due to the LOSCO model. For the Krammersluizen, the value that could be saved in 2016 is € 390,000.- (Figure 8.2). For the Sluizen Hansweert, the value that could be saved in 2016 is € 110,000.- (Figure 8.3). The reliability of the scheduling, as measured by the standard deviation of the passage times, does not change significantly. In the figures can be observed that the reliability is slightly lower for Krammersluizen and slightly higher for the Sluizen Hansweert.

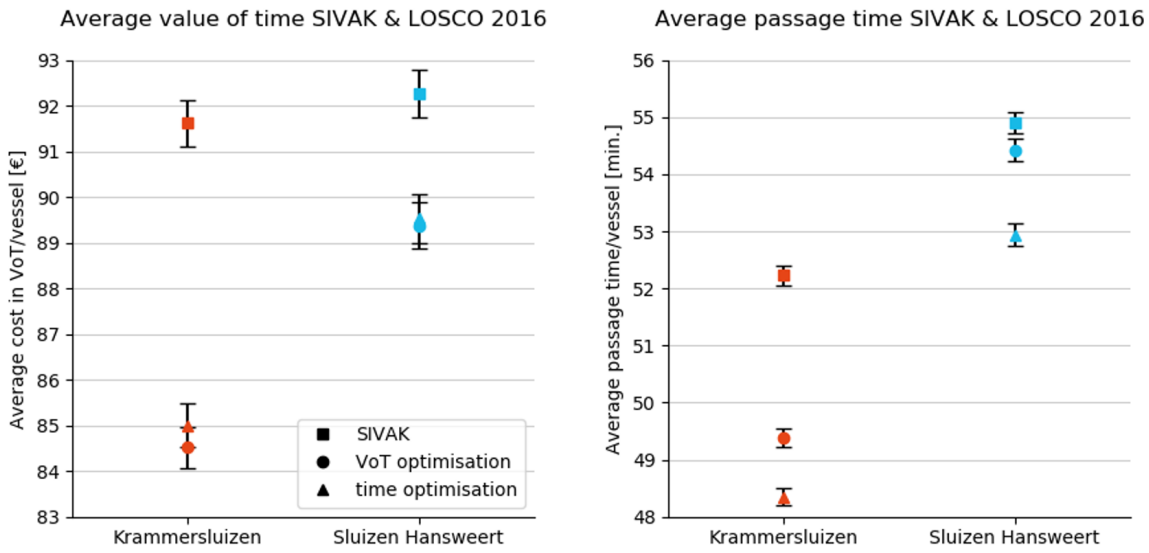


Figure 8.1: Results of the SIVAK model and the LOSCO model for VoT and time optimisation displayed in value of time.

Krammersluizen (2016)

	Average		Total	
SIVAK VoT	€ 87.59	(96%)	€	3,290,437
SIVAK VoR	€ 4.03	(4%)	€	151,297 -
SIVAK tot.	€ 91.62		**€**	**3,441,734**
time opt. VoT	€ 81.24	(96%)	€	3,051,605
time opt. VoR	€ 3.76	(4%)	€	141,407 -
time opt. Tot	€ 85.00		**€**	**3,051,605**
			€	**3,441,734**
			€	**3,051,605** -
			€	**390,129**

Figure 8.2: Savings per year for applying the LOSCO model to the Krammersluizen.

Sluizen Hansweert (2016)

	Average		Total	
SIVAK VoT	€ 87.98	(95%)	€	3,558,812
SIVAK VoR	€ 4.28	(5%)	€	173,078 -
SIVAK tot.	€ 92.26		**€**	**3,731,890**
time opt. VoT	€ 85.16	(95%)	€	3,444,780
time opt. VoR	€ 4.37	(5%)	€	176,796 -
time opt. Tot	€ 89.53		**€**	**3,621,576**
			€	**3,731,890**
			€	**3,621,576** -
			€	**110,314**

Figure 8.3: Savings per year for applying the LOSCO model to the Sluizen Hansweert

The real value of improved locking operations might not be for the shippers but for Rijkswaterstaat itself. If improved locking operations can keep the average waiting time at the lock below a certain threshold, the investment of building a new chamber or enlarging an existing chamber can be postponed. The money that is saved with the postponing of this investment can be calculated by taking the net present value of the money that is spend in a few years from now.

The investment of building a new lock could be € 200 million. If this money is not spend now but in a few years, the money can be invested in something else. Rijkswaterstaat uses a nominal discount rate of 4.5% for infrastructure cost-benefit analyses (Rijkswaterstaat, 2020). This needs to be adjusted for inflation to arrive at the real discount rate. The European Central Bank's target is to have an inflation rate of 2%. That means that we can use a 2.5% real discount rate for a back-of-the-envelope calculation. If the investment can be postponed by 2 years, the net present value of the investment is:

$$NPV = \frac{R_t}{(1+i)^t} = frac200(1+0.02)^2 = €190\,million \tag{8.1}$$

This is a ten million euro saving. For five years the NPV is € 177 million. And if the investment can be postponed by 10 years the NPV is € 156 million. Which is almost a € 46 million saving in today money.

8.2.2. Fairness

The results with respect to fairness can be found in Figure 8.4 and 8.5. For simplicity the data was split into 13 RWS categories, based on the length of the vessel. These figures tell a different story than hypothesised. The hypothesis was that the time optimisation would be unfair, but this does not seem to be true, as there is no general trend of larger passage times for larger vessels. Also in the SIVAK baseline case, no trend can be observed. For the VoT optimisation, a trend can be observed. The trend is that larger vessels have a smaller average passage time than smaller vessels.

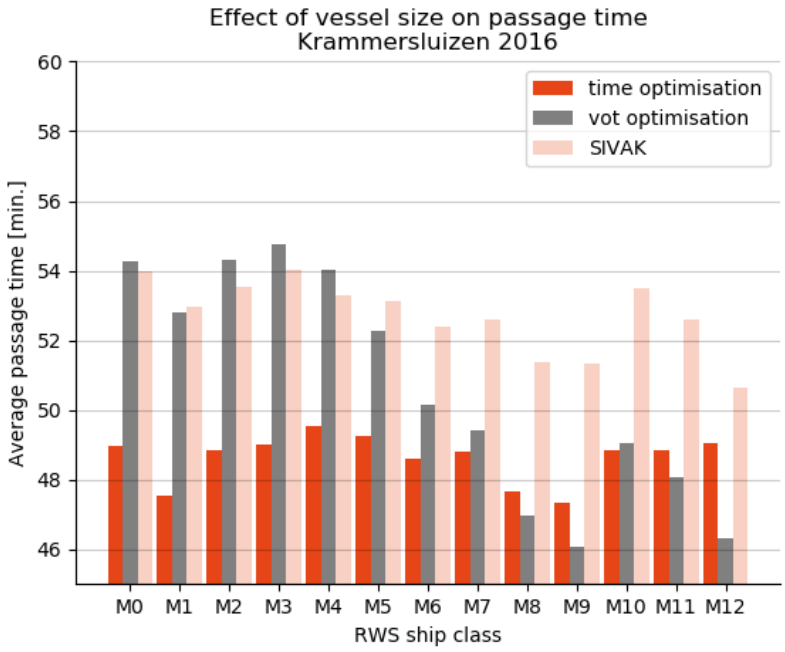


Figure 8.4

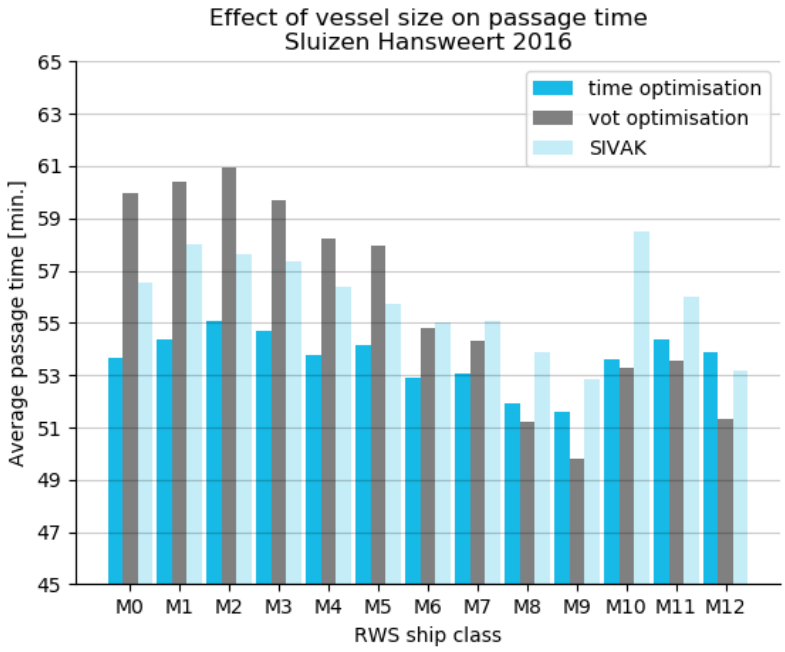


Figure 8.5

8.3. Conclusion

The first of the two sub questions in this chapter was:

Is it better to minimise economical value rather than minimising vessel passage time, as this serves the economy better?

The value of time for vessel was modelled according to method provided by Kennisinstutuut voor Mobiliteitsbeleid (2018). This means that factor costs for vessels are multiplied by a specified factor to arrive at a value of time (VoT) and value of reliability (VoR) per vessel. The LOSCO model was optimised based on the VoT. The optimisation on VoT was not found to be very effective, the costs per vessels are almost equal to the optimisation on time, while the time optimisation does decrease the time by a larger share.

The reliability of the scheduling, as measured by the standard deviation of the passage times, does not change significantly.

The second sub question in this chapter is:

Is it fairer for larger vessels to minimise economical value rather than passage time?

The fairness can be an issue for the passage time, as larger vessels are treated equal as smaller vessels, whereas larger vessels cover more space inside the locking chamber. In practice this effect was not found. In fact the opposite was found to be true, scheduling on VoT rather than on time caused larger vessel an advantage and their average passage time was decreased in relation to smaller vessels.

9 Discussion

In order to decide if the right conclusions can be drawn about the model, something has to be said about the methodology that was used to answer the questions that were posed, and in addition, the issue of generalization of the model and the optimal settings will be addressed.

One issue is the use of SIVAK instead of passage time data that was collected at the considered locks. Data for passage times per vessel is not collected at locks. Instead in this thesis SIVAK was used to compare the created LOSCO model with. It is however unknown how well SIVAK approaches reality. In this thesis the SIVAK settings were chosen to minimise the passage time in order to achieve the most conservative comparison with LOSCO. The choice for the best SIVAK settings makes it likely to be better than reality. Moreover, the method of SIVAK is based on strict rules, these do not exist in reality, therefore human errors are expected that lead to illogical choices and consequently worse performance than SIVAK.

A second important issue is the simplification of safety allowance. In the LOSCO as well as the SIVAK model, safety allowance is disregarded in this study. In reality, vessels need some space relative to each other to maneuver in the lock. For vessels that transport dangerous goods, there are rules for minimum distances that should be kept to other vessels in the lock. Vessels that carry dangerous goods are obliged to carry one to three cones. The number of cones increases with the increased danger of the goods. Vessels that carry three cones are not even allowed to be locked with any other vessel. Introducing safety allowance will make LOSCO more realistic and decreases the capacity of the locks relative to the current model. The consequence of introducing safety allowances for the model is dependent on two effects. If the capacity decreases and the same number of vessels passes, there is a higher chance for a queue at the lock. The average passage time will therefore increase and so will the maximum waiting time. A higher maximum waiting time is a tougher calculation for the LOSCO model and problems with overlapping chunks start to increase. On the other hand, less vessels can fit in the lock, so there are less combinations of vessels that can fit, which makes an easier problem for the LOSCO model. If the LOSCO model performs better or worse can therefore not be predicted and is dependent on the balance between the two effects. Experiments should be able to uncover the sensitivity of both effects.

Another issue is that in reality vessel ETAs are not a certainty. In the LOSCO model, there is no possibility for delay. In reality vessels might be delayed and miss this ETA. Therefore it might be a good idea to add some robustness to the model. Not every vessel delay is critical for the schedule, some vessels have a planned waiting time in their schedule. The focus should however be on the vessels that have no planned waiting time (they need to enter the chamber as soon as they arrive at the lock).

Furthermore, the objective function is an issue. In this thesis the objective of the lock is to lock vessels as fast as possible, to minimise their passage time. In Chapter 8 the objective was changed to the economic function, the costs that are made by the vessels is minimised. There are however more functions that the lock serves, that can also be represented in a objective function. The lock could have a function to prevent loss of water or to prevent salt intrusion. In both these cases the number of lockings should be minimised, this could be the objective function. A combination of multiple objectives with weigh factors could also be considered. If the objective function is expanded to other objective, the passage times will rise, consequently the problems of overlapping chunks will increase. Therefore it is not recommended to work with other objectives than minimum passage time before the model is improved. This behaviour could also be observed for the economical objective in Chapter 8. In this chapter, also a second conclusion was drawn. The hypothesis was that economical optimisation would result in a fairer schedule. In practice, the opposite seemed to be true. This is because a first come first serve constraint was used, so that larger vessels cannot be excluded. Removal of this constraint could mean that the opposite conclusion is drawn.

Processing times and the enter and exit times of the vessels are also simplified. The processing time was taken standard as 30 minutes. In reality, there are tides at the three locks that were scheduled, therefore, the processing time will change during the day. The astronomical tide is known in advance, therefore it could be added to the model, it does however add another layer of complexity to the model, that decreases performance. Entering and exiting times were set at 2 minutes for every vessel, in reality these can differ per vessel. If the total processing time is less in reality than was assumed, the LOSCO model can achieve better results. If the total processing time is larger, the opposite is true. A larger spread in processing times (such as by introducing tides) is bad for the LOSCO model, as the behaviour is non-linear. If processing times are larger during busy times, the LOSCO model will perform proportionately worse than it performs better when processing times are short at busy times.

Can the results of this study be generalised? The lock sizes in this study are limited to some of the biggest and most economically important inland locks of the Netherlands. The locks that were tested are all about the same size. They are also all 2 locks in parallel. However, a lock that is smaller might be beneficial for the LOSCO model in relation to SIVAK. In particular the average number of vessels that can fit the chamber at maximum capacity. If this number is lower than at the locks that were tested, the LOSCO model should work better, because there is a lower number of options. An easier problem might provide a better solution, as solution quality is balanced with runtime in the LOSCO model. In general it can be said that the results

can be generalised to locks with 2 parallel chambers in the range of the same average number of vessels per locking at maximum capacity.

The range of validity of the obtained result might be limited due to varied reasons. For instance, the heuristics no-FCFS and resolution were not found to be effective. This conclusion can only be drawn in the current set of heuristics. They could be effective in combination with other heuristics that were not used in this study. Moreover, the optimal settings were found to be chunks with a maximum size of 25 vessels and a maximum waiting time of 60 minutes. This is the result of a balance between run time and solution quality. It is likely that the optimum settings change when a faster processor is used. The effect is however expected to be limited, as the problem grows fast. Also for these settings counts that the optimal settings could change radically if new heuristics are introduced. As the optimal settings are dependent on the interplay of the heuristics. One of the main problems that needs to be solved to improve the model is the overlap that the chunks get at busy times, this is the main reason that the SIVAK model is faster at busy times than the LOSCO model.

10

Conclusions & Recommendations

10.1. Conclusions

Following the sub questions in every chapter, makes it possible to answer the research question:

How can a model be made that schedules vessels in locks with parallel chambers with the objective to minimise the summation of the total passage time over all vessels?

In order to decrease the average passage time of vessels at locks with parallel chambers, an optimisation model was developed to schedule vessels in locks with parallel chambers. This model was called LOSCO for lock scheduling by constraint optimisation. The model is an instance of the general optimisation problem called the Job Shop Scheduling Problem (JSSP). The JSSP is known to be very computationally expensive. The lock scheduling problem cannot be solved exactly with the algorithms that exist for the JSSP. Therefore the problem is simplified with heuristics. The heuristics allow the problem to be solved by finding a balance between the runtime of the model and the quality of the solution that is obtained.

The LOSCO model uses the same general approach as Verstichel (2013), hitherto the only study about the scheduling of parallel lock chambers. In his model, he generates solutions that are fitted in the lock afterwards, using the Three-Way Best-Fit heuristic that he developed. However, some heuristics were added that make the model perform better and more practical to use.

One of the heuristics is the 'first come, first serve' (FCFS) heuristic, this heuristic is also in the Verstichel (2013) model. The FCFS heuristic was dropped in the LOSCO model to investigate whether the model performance would get better without this heuristic. To this end the Three-Way Best-Fit heuristic was extended, so that it creates constraints for orders of vessels that do not fit. However, during testing it was found that the solution quality is not improved enough, compared to the extra runtime it costs. The conclusion is that the FCFS assumption is a good assumption. Another heuristic that was tested is the resolution. In this heuristic the discretisation is made coarser to reduce the number of options for the solver. This heuristic is also not effective and the best resolution is 1 minute.

An effective heuristic is the use of chunking. This also makes the model more practical, as less ETAs of vessels will have to be known in advance. The maximum waiting time heuristic is also very effective. It reduces the number of options that the model has to go through while also making the schedule more equal.

The model is compared to SIVAK as there is no data collected of passage times. There is however data of times of arrival at the lock. This data was used to test the model for three locks, the Krammersluizen, the Sluizen Hansweert and the Kreekraksluizen. These are three lock complexes with 2 parallel chambers and similar chamber sizes. The LOSCO model is effective for reducing the average passage time per vessel with about 3.9 ± 0.12 SE minutes for the Krammersluizen and 2.0 ± 0.14 SE minutes for the Sluizen Hansweert compared to the SIVAK model. Unfortunately, the LOSCO model performs worse if the lock complex is busier as was the case at the Kreekraksluizen. The LOSCO model was even unable to find a solution in this case. The interarrival time is up to three times lower during the day, which results in better performance for the SIVAK model at the peak hours in the afternoon and better performance of the LOSCO model at less busy times. The inter arrival time is around 8 to 10 minutes when the 2 models perform equal for the tested locks. During a year on average at the Krammersluizen every 14.0 minutes a vessel arrives, for the Sluizen Hansweert this is every 13.0 minutes. For the Kreekraksluizen a vessel arrives every 7.8 minutes on average.

The LOSCO model was also run on an optimisation with value of time as objective. Value of time can be obtained by multiplying the costs of a vessel by a factor as defined in Kennisinstutuut voor Mobiliteitsbeleid (2018). The optimisation on value of time was not found to be successful. The optimisation on time was able to almost reach the same costs for vessels, whilst the average passage time could be reduced by a larger share. The reliability, measured as the standard deviation of the passage time/passage value of time does not change significantly between SIVAK or LOSCO. Optimisation on time was found to be fair for large and small vessels. Larger vessels are not at an disadvantage, despite their larger size. Contrary, optimisation on value of time was found to be unfair for smaller vessels, as larger vessels got significantly shorter passage times than smaller vessels.

Savings for vessels at the Krammersluizen over 2016 are € 390,000.- if the LOSCO model would be used compared to SIVAK. For the Sluizen Hansweert the savings over 2016 are € 110,000.-. These values are low compared to the money that could be saved for postponing the building of a new lock chamber. However, the LOSCO model is not able to increase the capacity at the lock.

Overall, the LOSCO model is a step ahead towards a practical lock scheduling model. As this discussion points out, there are still some hurdles to overcome to achieve a fully functioning practical lock scheduling model. For some locks, the current LOSCO model as presented can already decrease the average passage time. It might however not be unrealistic to achieve a model that can constantly perform better with respect to SIVAK. In the recommendations, the steps that could be taken towards this model are explained.

10.2. Recommendations

In this section the recommendations for future research are given. The main recommendation is to first focus on improvement of the model before adding more features. Improvement of the model here means to achieve better solutions in the same runtime. First recommendations for the model improvement are given, then the recommendations for more features of the model follow.

Model improvement

The simplest method for improving the model might be to use a faster computer. Processor speed is limited. With a faster computer the maximum chunk size could be increased. The maximum waiting time would not be effective to increase as this also provides some fairness. The amount of maximum chunk size that can be increased is however expected to be limited, even with a computer that is a lot faster. That is because the problem gets harder proportional to the faculty with increasing chunk size.

A relatively simple way to improve the LOSCO model is to take safety distances in the lock chambers into account. As was discussed in the discussion. This seems a complication to the model, but the opposite might be true. This is dependent on the balance between the effect of larger maximum waiting times and the effect of smaller amounts of vessels per locking.

A third way to improve the model is to add more heuristics. One idea is to change the objective function of the model based on the average inter arrival time of a chunk. The main problem with busy chunks is that an overlap forms with the next chunk. Instead of minimizing the passage time for all vessels, the time it takes for all vessels to be locked could be minimised. This time is called the makespan in job shop scheduling problems (JSSP). If the makespan is minimised, the overlap in time with the next chunk will be smaller and this could lead to better solutions for busy chunks. Although the average passage time for the chunk itself will become higher, the connection to the next chunk will be better, leading to lower passage times overall. More heuristics that can be tried can be found in literature on JSSP problems.

It is also recommended to apply the model to more locks with two parallel chambers to gain more insights. It is expected that the LOSCO model works better at locks with lower maximum capacities as measured by the maximum number of vessels per locking.

A more radical change could be to change the algorithm to a meta-heuristic (approximation algorithm). The use of meta-heuristics might be better than using an exact approach such as mixed integer linear programming or constraint programming. Meta heuristics, such as genetic algorithms or simulated annealing, were shortly treated in chapter 2. In these type of models, the basis of the search itself becomes a heuristic method. They are less likely to find the global optimum, but they might be a more run time efficient method to find better solutions.

The problem would be greatly reduced if dispatching rules could be derived which could be used in a simulation model. It is however not very likely that exact dispatching rules exist, as the problem is a JSSP problem. Approximate dispatching

rules could however also be very helpful for the creation of new heuristics.

Another recommendation is to collect passage time data at locks. This does not necessarily improve the model, but it puts the model in perspective. Once data is collected, the LOSCO and the SIVAK model can be compared to reality. Only then, the real impact of lock scheduling can be observed.

Finally, the hardest but most impactful way to improve the lock scheduling model is to solve the P vs. NP problem (Polynomial time versus non-polynomial time). This is one of the seven Millenium Prize Problems as presented by the Clay Mathematics Institute with an award of 1 million dollar (Clay Mathematics Institute, 2020). An NP problem is a problem for which the solution can be checked easily but to come up with a solution seems practically impossible. The problems get so big that it is impossible to solve them by brute force. If this problem can be solved, that would mean that a lock scheduling model can be solved exactly, without the use of heuristics. Moreover any JSSP problem could be solved exactly, in fact, almost any optimisation problem could be solved.

Features

The following recommendations are only recommended once a better lock scheduling model is achieved.

A necessary feature for a practical lock scheduling model is that it should be able to deal with vessel delays. An idea to deal with this could for instance be to generate alternative solutions. The alternative solutions can be used when some critical vessels miss their ETA. Otherwise a schedule could be created where every vessel's ETA standard accounts for some delay, but this will mean a lower solution quality. It is recommended to look into literature on the aviation industry, where this is also a problem, when scheduling airplanes to landing strips.

Some features were discussed in the discussion to achieve a more realistic model, such as safety allowances and the inclusion of tides to processing times. The inclusion of tides is certainly better to include later in the model development. Safety allowances is dependent on the balance of the two effects explained in the discussion.

The model could be applied to a lock with a single chamber. It is expected that this is easier to compute, compared to two parallel chambers as the decision for each vessel to choose a chamber does not have to be made. Therefore the LOSCO model might achieve better solutions with larger chunk sizes for a single chamber problem. It would be interesting to see how much better it performs for a single chamber, as this can also say something about how much better the solution for a two chamber problem could be.

The model could also be tested at the Volkeraksluizen, which is a locking complex with three parallel chambers. This is the busiest lock complex in Europe, therefore it is expected that this is too hard for the current LOSCO model to handle. Instead of changing the number of chambers parallel, a model can be developed that can optimise the flow of vessels through locks in series. The model could also be coupled to other scheduling models, for instance the scheduling of terminals, or traffic management models.

A

Developments in Inland Waterway Transport

This chapter is about inland waterway transport. It serves to give a general background of inland waterway transport (IWT) and to identify opportunities and threads. To place inland waterway transport into perspective, the first section is about trends in transportation in general. In the second section, the inland water way system is discussed. This section is divided into subsections. In the first subsection, the functions of the waterways are treated and the situation in the Netherlands is sketched. In the second subsection, the impact of climate change on the rivers is discussed. Third, the types of vessels are explained in the subsection fleet, with a focus on the trend in vessel size. The fourth subsection treats the basics of shipping locks and in the fifth subsection the importance of the recent developments of River Information Services is discussed. Finally, in the conclusion the most important findings are summarised.

A.1. IWT in perspective: Trends in Transportation

Governmental institutions such as the EU, the Dutch government, and the Port Authority of Rotterdam try to realise a modal shift towards inland waterway transport (IWT) (European Commision, 2011) (Ministry of Infrastructure and the Environment, 2015) (Rotterdam Port Authority, 2014). This modal shift is in their interest, because IWT is the cheapest, the most environmentally friendly and the safest way of inland transport. The modal shift is also a measure to counteract the congestion of roads, which is an ever growing concern.

A division in 3 main types of cargo can be made; dry bulk, liquid bulk and containers. Inland waterway transport (IWT) is especially well developed in dry bulk and liquid bulk (Central Commission for the Navigation of the Rhine, 2017). However, a decline in the transport of liquid and dry bulk can be observed, and it is expected that this trend will continue (van Dorsser et al., 2018). The decline of bulk transport and rise of container transport is partially explained by Caris et al. (2014). They argue that more and more bulk processing industries (eg. steel plants and refineries) are being situated closer to their bulk sources. This causes a rise in the import of end products, that are often transported in containers (figure A.2). Since 2000 the number of containers in Rotterdam doubled to 3 million TEU and in Antwerp it even quadrupled to 2.5 million TEU in 2016 (Central Commission for the Navigation of the

Rhine, 2017). The modal split for container transport is around 35% for Rotterdam and Antwerp in 2016 (Central Commission for the Navigation of the Rhine, 2017), the 2 biggest container ports in Europe. The modal split for the port of Rotterdam in 2016 is in figure A.1. In this figure can be observed that the modal split for IWT is just over one third.

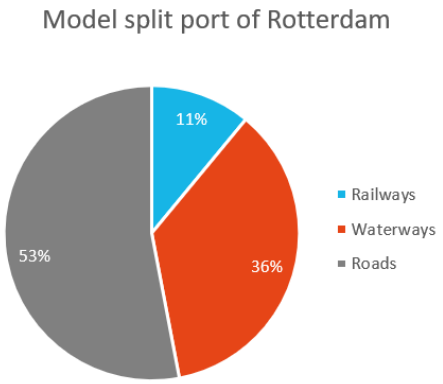


Figure A.1: The modal split of hinterland transport in the port in Rotterdam in 2016. Data from Central Commission for the Navigation of the Rhine (2017)

A.1.1. Modal Split

A division in 3 main types of cargo can be made; dry bulk, liquid bulk and containers. Inland waterway transport (IWT) is especially well developed in dry bulk and liquid bulk (Central Commission for the Navigation of the Rhine, 2017). However, a decline in the transport of liquid and dry bulk can be observed, and it is expected that this trend will continue (van Dorsser et al., 2018). The decline of bulk transport and rise of container transport is partially explained by Caris et al. (2014). They argue that more and more bulk processing industries (eg. steel plants and refineries) are being situated closer to their bulk sources. This causes a rise in the import of end products, that are often transported in containers (figure A.2). Since 2000 the number of containers in Rotterdam doubled to 3 million TEU and in Antwerp it even quadrupled to 2.5 million TEU in 2016 (Central Commission for the Navigation of the Rhine, 2017). The modal split for container transport is around 35% for Rotterdam and Antwerp in 2016 (Central Commission for the Navigation of the Rhine, 2017), the 2 biggest container ports in Europe. The Rotterdam Port Authority set a goal to increase this percentage to 45% for 2035 (Rotterdam Port Authority, 2014). Not only ports, also the European Union (European Commision, 2011) and the Dutch Government (Ministry of Infrastructure and the Environment, 2015) try to establish a model shift towards IWT. They are interested because IWT is the most environmentally friendly means of inland transport and road congestion is a concern.

Despite governmental policies, the modal split for container barging is not growing enough. This has several reasons. One reason is that many stakeholders do not have the direct incentive to organise the whole system. The IWT sector is characterized by small family companies that often operate only one ship, therefore they have a weak negotiation position, of which other parties take advantage. The current contractual relations and lack of information sharing do result in coordination problems, result-

Figure A.2: An inland container vessel of the Rhinemax class (+/- 500 TEU). The growth of container transport is an opportunity for IWT. Photo by: [S.J Reiling, 2017, emprove.nl]

ing in bad efficiency for the IWT sector (van der Horst et al., 2019). Road transport is easier to organise, a truck often only transports the goods of one company, because trucks are smaller, utilisation is not a problem. With IWT, this needs to be coordinated, the vessel is only cheaper if it can transport enough containers and there should also be an agreement on which ports to visit. A truck can get everywhere, whereas for IWT still a truck is needed to transport the container from the port to the end destination. On the other hand, road transport is facing congestion, higher fuel prices and environmental costs. This causes opportunities for other modes like rail and inland waterways. Transport mode selection is mostly dependent on cost, transportation time and shipment size. In order to facilitate a model shift, at least one of the three should be improved relative to other transport modes.

A.1.2. Future Trends

Some trends might however create a different situation. An example is the disruptive development that the introduction of containers caused. At the moment, there are other trends and possible disruptions that might change the supply and demand of IWT. One of these trends is 3D printing. A lot of consumer goods that are transported by container now, might not need any transportation anymore if they can be printed by the consumer themselves.

Foldable containers are also an interesting trend. This development reduces the cost of transporting empty containers. Transport of empty containers can also be minimised by good coordination.

A counter-intuitive trend is that of reverse modal shift. Inland waterway transport is more sustainable than road transport, but there is more innovation in road transport. This is because the modality of road transport is larger, it is also more visible to the general public and investors. The IWT sector is not able to adapt as fast as road transport, because vessels have a far longer lifetime than trucks. IWT transport is more energy efficient in theory, but if all research is focused on environmentally friendly trucks, IWT might still lose. Therefore governments plays a central role in the future of IWT.

A.2. Inland Waterway System

A.2.1. Waterways in the Netherlands

The inland waterway system consists of canals, rivers and lakes. These waterways have different functions that complicate the management of the waterways (Rijkswaterstaat, 2015). An important function of some waterways is flood risk management, especially in the Netherlands. The waterways with this function are designed to cope with a determined maximum of water discharge. The waterways also serve as a source of fresh water, which requires the water to be clean and protected against salinisation from sea. Also social and economic user functions are in need of good water quality as well as the sustainability function. Rijkswaterstaat is the authority responsible for the functions of the waterways in the Netherlands. Rijkswaterstaat is continuously busy finding the balance between the functions of the waterways.

The main interest of this study on waterways is on the function of navigability. Inland navigation precedes history as a cheap method of transport. Transport over water always has been the most economical option for long distances, as the energy consumption is low and the scalability is high. It is no coincidence that many of the largest ports in the world are situated near a river. Nowadays there is 4800 km of waterway suitable for transport in the Netherlands (Bureau Voorlichting Binnenvaart, 2016). Inland waterway transport is mostly in favour of heavy goods and goods with high volumes or goods with large sizes. It is an environmentally friendly mode of transport, compared to other modes (Rijkswaterstaat, 2000).

The Netherlands is situated in the delta of the rivers Rhine and Meuse. The river Rhine is one of the busiest navigated rivers worldwide (Pauli, 2010). Despite the relative crowdedness, the capacity is ample to deal with this (Rijkswaterstaat, 2015). The Dutch IWT system connects the major seaports of Rotterdam, Antwerp and Amsterdam to the large inland industrial areas: the heavy industry of the Ruhrgebiet in Germany and the chemical industry around Basel.

Rijkswaterstaat divided the waterways of (inter)national importance in three levels. The river Rhine as a route to Germany is of the first level, see figure A.3. The corridors that are also in this level include the corridor Rotterdam-Antwerpen, including the port of Rotterdam all the way to sea, the Amsterdam-Rijnkanaal and from Amsterdam to sea. Of second level importance are the river Meuse and the corridor over the IJsselmeer to the north of the Netherlands and the rivers IJssel, Nederrijn and Lek. Besides the main corridors, the Netherlands also has a network of smaller waterways, making many places accessible from water.

Figure A.3: All navigable waterways in the Netherlands, CEMT class II or bigger. Map by: [Rijkswaterstaat, 2008, Vaarwegenkaart naar bevaarbaarheidsklasse CEMT 2008, commons.wikimedia.org]

A.2.2. Climate Change

Not every river is navigable. This can be illustrated with the examples of the Rhine and Meuse. The Rhine is the bigger river of the two, with an average discharge of 2000 m^3/s. The Meuse only has a discharge of 230 m^3/s. A bigger river is often deeper, making it better navigable. The Rhine river discharge is also more constant. One reason is the larger catchment area of the Rhine. Another reason is the contribution of glaciers to the river Rhine, this is a more constant source compared to rainfall. Because of the differences in discharge, the Meuse had to be canalized in order to be better navigable. In figure A.4 is an example of a weir in combination with locks on the river Meuse. Resulting in a large number of locks for navigation to overcome the water level differences. In the Rhine there are also locks, but the first lock in the main Rhine channel is far upstream in Iffezheim south of Karlsruhe.

Figure A.4: The locks and weir at Lith, the Netherlands. Photo by: [J. van Houdt & Rijkswaterstaat, 2002, beeldbank.rws.nl]

Inland waterways have always been subject to the climate. However anthropogenic causes of climate change might challenge IWT even more. Climate effects are very location dependent, but there is an overall trend. The global increase of temperature will facilitate more extreme rainfall and longer periods of drought. The melting of glaciers will cause the river Rhine to move to a more rainfall dominated river system, which is characterized by more droughts in the summer(Hurkmans et al., 2010). The same study concludes that the effects are different for the first and the second part of the century. The first part is dominated by increased precipitation that may increase the number of floods, and only from 2050 a decrease in summer discharge is found, with consequent droughts.

Insufficient water levels because of droughts are expected to have the most disruptive impact to inland navigation (Christodoulou and Demirel, 2018). Vessels will have to reduce the amount cargo to reduce their draught. Vessels may even have to wait for the water level to rise, resulting in a longer travel time and higher costs. Hekkenberg et al. (2017) states that building longer and wider vessels can decrease the draught. Also smaller vessels might be used.

Not only a shortage, but also a surplus of rainfall is a problem. Water levels that

are to high make inland ports inaccessible and can damage cargo stored at a port. There are also navigation restrictions, as for instance the air draught for passing bridges is smaller. Now, IWT is considered a very reliable mode of transport, climate change might change this, resulting in a decrease of the modal shift towards more climate change resilient modes. The modes that are ironically more environmentally harmful. Climate change could result in locks in the downstream section of the river Rhine.

A.2.3. The Fleet: Vessel Sizes

The fleet can be classified by different characteristics of vessels. It can, for instance, be classified by the goods that the vessel transport: liquid bulk, dry bulk, containers and other. The classification that is most used is the classification on the size of vessels. Historically, the dimensions of canals and shipping locks led to an almost discrete division of vessel sizes. The system that is used to classify ships stems from the 1992 Conférence Européenne des Ministres de Transport (CEMT). In 2011, Rijkswaterstaat upgraded the old system and started using the RWS-class. The classification can be observed in table A.1. The waterways are classified according to these classes, as depicted in figure A.3.

Table A.1: CEMT and RWS classification

		Characteristics normative vessel				Classification	
CEMT Class	**RWS Class**	**Types of vessels**	**Width (m)**	**L.O.A. (m)**	**Draught (m)**	**Width and length (m)**	**Capacity (tons)**
	M0	other				W <= 5.00 or L <= 38.00	1-250
I	**M1**	Peniche (Spits)	5.05	38.5	1.8-2.2	W = 5.01-5.10 and L >= 38.01	251-400
II	**M2**	Campine (Kempenaar)	6.6	50-55	2.5	W = 5.11-6.70 and L >= 38.01	401-650
III	**M3**	Hagenaar	7.2	55-70	2.6	W = 6.71-7.30 and L >= 38.01	651-800
	M4	Dortmund-Ems	8.2	67-73	2.7	W = 7.31-8.30 and L = 38.01-74.00	801-1050
	M5	Elongated Dortmund-Ems	8.2	80-85	2.7	W = 7.31-8.30 and L >= 74.01	1051-1250
IV	**M6**	Rhine-Herne	9.5	80-85	2.9	W = 8.31-9.60 and L = 38.01-86.00	1251-1750
	M7	Elongated Rhine-Herne	9.5	105	3.0	W = 8.31-9.60 and L >= 86.01	1751-2050
Va	**M8**	Large Rhine vessel	11.4	110	3.5	W = 9.61-11.50 and L=38.01-111.00	2051-3300
	M9	Elong. Large Rhine vessel	11.4	135	3.5	W = 9.61-11.50 and L >= 111.01	3301-4000
Vb	**BII-2l**	2 push unit long	11.4	170-190			
	BII-2b	2 push unit wide	22.8	95-145			
VIa	**M10**		13.5	110	4.0	W = 11.51-14.30 and L = 38.01-111.00	4001-4300
	M11		14.20	135	4.0	W = 11.51-14.30 and L => 111.01	4301-5600
	M12	Rijnmax	17.0	135	4.0	W >= 14.31 and L >=38.01	>= 5601
VIb	**BII-4**	4 push unit	22.8	185-195	2.5-4.5		6400-12000
VIc	**BII-6l**	6 push unit long	22.8	270-280	2.5-4.5		9600-18000
VII	**BII-6b**	6 push unit wide	34.2	195-200	2.5-4.5		9600-18000

Data of the fleet are given in the yearly report of the Central Commission for the Navigation of the Rhine (2017). According to them, there are 13500 inland transport vessels registered in the Rhine countries. More than half of these are Dutch. The number of vessels has declined by 12% from 2005-2015. While the loading capacity has risen in the same period with 20%. This means that the size of the vessels has increased. Of this 13500 vessels, 1200 classify as push and tug vessels. Also combined arrangements are common. This is a motor vessel in combination with one or more barges. In Germany, Belgium and the Netherlands, half of the fleet is more than 50 years old. 15% of this fleet is even more than 75 years old. In the years before the 2008 crisis, many new vessels were built in the Rhine countries, resulting in overcapacity since then, but this has slowly been recovering (Central Commission for the Navigation of the Rhine, 2017).

The maximum dimensions for a non-divisible vessel on the Rhine are a length

of 186.5 m and a beam of 22.9 m (Hekkenberg, 2013). These dimensions allow the ship to access the most important ports in the Rhine region. The same study states that coupled vessels and push tow vessels might have dimensions of up to 280 m in length and a beam of 34.2 m. To get an idea of this size compared to the M1 class they are displayed next to each other to scale in figure A.5. The cause for the trend towards bigger vessels can be found in economies of scale. In 1990, overcapacity was a serious problem in the barge sector and a European policy was created to give a bonus to skippers that would scrap their vessels(Konings, 2007). Many small vessels were scrapped, resulting in an even larger share of bigger vessels. A larger vessel might however not always be the optimal one. As Hekkenberg (2013) explains, smaller ships have their own advantages. They have larger geographic flexibility, because they can sail on more (smaller) waterways. Also goods that are of high value and of low annual demand favor smaller ships. Smaller ships with smaller draughts can be favourable in the future when droughts can be a bigger problem. Smaller ships are also more flexible, because handling times are shorter. The advantage of the relative old age of small vessels is that they have low capital costs, making them more competitive (International Buck Consultants Rotterdam, 2008). In the same study it is concluded that the share of crew costs, maintenance costs and financing costs are a lot larger for smaller ships. These costs can not be lowered, unless less crew on a smaller ship is allowed, in this way legislation poses a problem for small ships. Air draft is mainly a challenge for inland container vessels. Inland container vessels are limited in (vertical) size by bridges. A study by Konings (2007) states that the maintenance of many small waterways has been neglected. He argues that the main competition of small vessels is road transport. Also the technical requirements on smaller ships are relatively expensive. There are numerous cases of projects that have tried to develop container transport on small waterways with innovative concepts. Konings (2007) and International Buck Consultants Rotterdam (2008) mention several examples: Neokemp is a small and fast vessel, special low-cost inland terminals, self-unloading vessels, push-barges. Also lighter composite ships are an example (e.g. CompocaNord), or the trunk-feeder service concept, for which barge convoys are split up for smaller waterways.

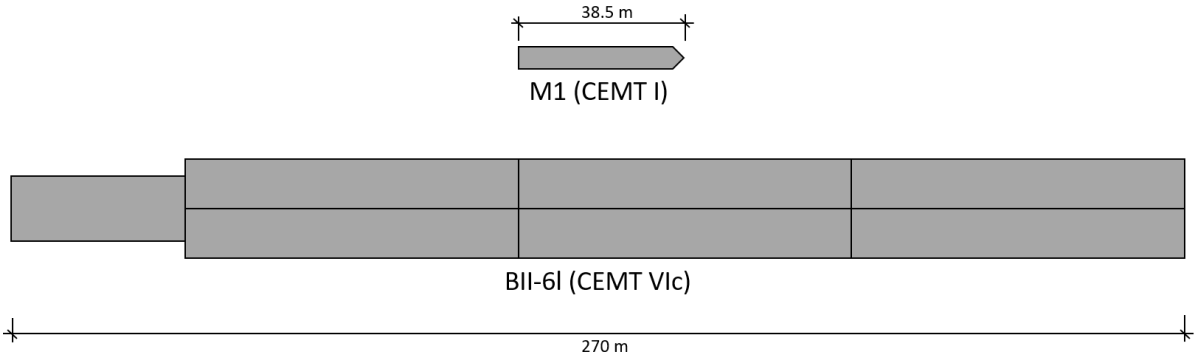


Figure A.5: A Peniche (Spits) of class M1, the smallest transport vessel, next to a class BII-6l push convoy, the biggest vessel that navigates the Rhine.

There are also less tangible reasons for the lagging development of smaller vessels. There is for instance a lack of service and cooperation with some of the smaller vessel exploiters (International Buck Consultants Rotterdam, 2008). There is also a lack of qualified manpower. The life of living on a vessel is not attractive for families. The

lack of availability of small vessels is in itself also a problem, this does decrease the reliability (Rijkswaterstaat, 2014a).

Rijkswaterstaat does take recreational boats and ships very seriously, motivating it as an important economic industry. The intensity of recreational boats is characterized by a large seasonality (Rijkswaterstaat, 2015). The summer season, holiday periods, weekends and the weather are good predictors of the amount of recreational boats.

Autonomous Vessels

Autonomous ships have the power to transform the sector completely. The process of automation follows certain levels. First steering and speed is optimized given a route plan. Then a route plan can be uploaded from shore. Next, the vessel itself will calculate a route plan. After that, a supervisor only has to monitor and intervene, the approval is asked. The next step is monitored autonomy. And the last step is full autonomy (SmartPort, 2018). Lectures by van Dorsser (2019) identified the challenges and opportunities in a transition towards autonomous shipping. A disadvantage is that the market for ships is way smaller than for autonomous cars. Therefore there is more incentive to create autonomous cars than ships. The business case is also bad because the smallest vessels relatively have the most crew costs. Larger ships are not so interesting to automate, because they need a crew for maintenance anyways. Therefore only smaller vessels are interesting to automate.

There are however also opportunities for autonomous shipping. Smaller inland vessels might for instance become more competitive. There is also a lack of interest for a career in inland shipping, which is an opportunity for automation. Automation of vessels of any type is hence not expected in the near future. However, if the technology for cars becomes widely available, smaller vessels are the most interesting opportunity for automation.

A.2.4. Shipping Locks

Shipping locks are used to help ships overcome water level differences. A lock also has the function of flood defence and is often part of road infrastructure. Another function of shipping locks is water management. For instance to prevent salt intrusion or to minimise the loss of water in a canal (**?**). The CEMT class of the waterway is often the result of the chamber size of a lock in that waterway. The CEMT class defines the sizes of largest vessel that can navigate in the waterway. Ships are often designed with this class in mind, leading to standardized sizes for vessels.

Shipping locks are expensive projects. At the moment of writing, a third locking chamber is being constructed for the Beatrixsluis and a new lock is constructed in Terneuzen. Projects that are high in value, respectively more than € 200 million and almost a billion euros. These investments are so large, that advanced tools can be developed to make better investment decisions. If projects of this value can be postponed by introducing better locking operations, a lot of money can be saved.

Locking operations always follow a certain sequence, see figure A.6. First vessels must line up in the waiting area, the doors of the lock need to be opened, then vessels sail in. The vessels are moored in the lock while the doors can start to close. The water is levelled in the lock, after which the gates are opened again. Often there are guiding jetty's that the vessels need to pass before the vessels on the other side of the lock can sail in. This is represented by the switch time (t_{switch}). The average time of

locking in inland waterways is around 20-40 minutes (Molenaar, 2011). This makes them often the bottleneck in the capacity of the waterway.
Remote control of locks (and bridges) is getting more common on the Dutch waterways, mainly for smaller locks (Rijkswaterstaat, 2015). This is a first step towards automation of the locks. Paving the way for lock scheduling algorithms.

Vessels with dangerous goods often need special treatment in locks. Blue cones are used to signal that a vessel carries dangerous goods. A vessel can have 1 to 3 cones, where the most dangerous goods are indicated by 3 cones. They should keep distance from other vessels in the lock, depending on the number of cones. Vessels with 3 cones should even be locked entirely separate.

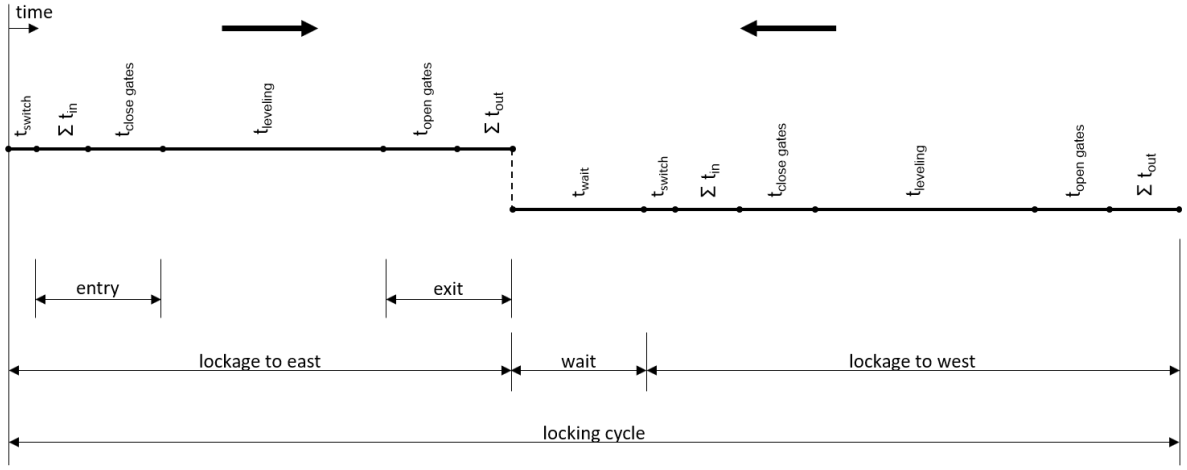


Figure A.6: The operations of which the locking cycle consists.

A.2.5. River Information Services

Information sharing is a trend that is visible in the logistics sector. There are a lot of benefits to this cooperation. The last development towards this trend is synchromodal transport. Synchromodal transport means that continuously the optimal transportation plan over different modes is being updated, so that last minute changes can be made. The benefit of this is that the utilization rate can be raised and the planning will be made more flexible, which helps with uncertainties and disturbances (Negenborn and Dekker, 2013). Negenborn and Dekker (2013) also mention the extended gate terminal concept. For instance ECT provides this service by offering inland transport by rail and by waterway. Of course this concept does not have to stay within a company. This is also stated by Crainic and Laporte (1997). They say that a limit has been reached for optimizing transportation costs for carriers and transporters on their own. What is next is a higher quality of service that is needed, defined by three parts: on-time delivery, delivery speed and reliability. This is still actual today. Therefore synchromodality is a logical consequence. On the Rhine, line service is the most used type of service. However, only a few terminals generate sufficiently large volumes to enable a ”one stop” in the hinterland. E.g. Duisburg and hubs on Mosel and Main. Especially this type of service can benefit from the sharing of information by bundling as much containers with the same destination as possible.

Caris et al. (2014) identify that a system wide model is missing concerning the detailed operations in inland waterway transport. According to them there are three reasons this could be helpful. The first reason is that it can demonstrate the benefits of synchronization. Which means that lock schedules and priority rules may optimize the system. Secondly, it serves decision makers to make better decisions for system interventions. Where normally crude forecasts are used. The last reason is that the effect of alternative priority rules can be assessed across the network. The same study by Caris et al. (2014) states that current literature ignores that lock operations are a bi-objective planning problem. Individual shippers want to minimize their own time, and waterway administrators want to optimize the system. These objectives can contradict in some cases.

Rijkswaterstaat (2011) states that there is a problem on the inland waterways with increasing waiting times for locks, movable bridges and terminals. There is a lack of a central information service so that individual ships cannot make a good decision for route choice. River Information Services (RIS) are an initiative of the European Union in order to enhance the safety and improve efficiency of the waterways. This is done by enhancing the exchange of information. Examples of research projects by the EU in this field are MARNIS (Maritime Navigation and Information Services) and NAIADES I & II. An example of a system that is under the umbrella of RIS is AIS (Automatic Identification System). AIS is a system of transponders on each vessel that send information about the vessel and it’s location. The system makes seas and waterways safer, puts the infrastructure to better use, and can help to plan ahead for vessels, terminals or waterway operators (Rijkswaterstaat, 2014b). The Netherlands can be considered ahead in these developments, BICS (Binnenvaart Informatie- en Communicatie Systeem, Inland Waterway Transport Communication System) was developed and now is even compulsory for every inland vessel. BICS is an electronic system that serves to collect every vessel’s travel and cargo data. BICS uses AIS amongst other methods. Rijkswaterstaat is more ambitious than the EU

on this matter and says its focus is on 'corridorgerichte begeleiding', literally translated as 'corridor-oriented guidance'. The current local traffic management is to shift to corridor-oriented traffic management. Information services provided by Rijkswaterstaat are water levels, traffic information, water depths, bridge heights and tidal windows in sea ports. They collect statistics regarding traffic, cargoes and calamities. As they are responsible for maintenance of the waterways they also provide information on maintenance (Rijkswaterstaat, 2015).

To facilitate a better modal split, Rijkswaterstaat initiated an extensive innovation program in 2010 when the Tweede Maasvlakte was build. This program was called IDVV (Impuls Dynamisch Verkeersmanagement Vaarwegen, literally translated as Impulse Dynamic Traffic Management Waterways). The core of the project was to share information and to cooperate in the logistic chain, as well as an attractive inland waterway transport (IWT) and better predictable arrival times(Rijkswaterstaat, 2014a). One of the subjects in the IDVV program was the lack of cooperation in the sector (Rijkswaterstaat, 2014a). Parties (mainly skippers) are seeing themselves mainly as competitors. While cooperation can make the chain safer, more effective, more efficient, more reliable and more sustainable. For instance, sometimes empty containers are transported in opposite directions on the same waterway. Inland container vessels are sometimes calling at each and every terminal in the port of Rotterdam, with small call sizes. Containers should be bundled. Planning should be done together, that is beneficial for the whole sector, as a lower price and better reliability and sustainability will attract more cargo. Another subject in the IDVV program is the better use of infrastructure. This involves smarter journey planning by vessels, facilitated by a better infrastructure information service. Information about berths for overnight stay, real time water levels and the resulting bridge heights and maximum draughts and ship locking planning amongst others (Rijkswaterstaat, 2014a).

Rijkswaterstaat is also investigating the possibilities of lock scheduling in a project called the Trajectplanner, a global description is given in Rijkswaterstaat (2014b). The main goal for the Trajectplanner is to increase the reliability of ETAs for the next terminal a ship is going to visit and to improve the flow of vessels on the waterways and it's objects. A distinction is made for shipping lock planning, bridge passage planning and berth registration. The lock planning consists of three model parts. Corridor/Network planning, lock object planning and chamber layout. The corridor/network planning provides ETAs for vessels, the lock object planning uses this to make a planning, then the chamber layout uses the output of the lock object planning. This layout is input to the lock object planning again, as well as the ETDs and ATDs in the lock object planning is output to the corridor/network planning. ETAs are regularly updated to increase the reliability of the planning. The lock object planning's function is to support the ETDs for the next terminal to load or unload, so that terminal planning is minimally disturbed. This means that the current first in first out policy will no longer be used. At the time of writing, the status of the Trajectplanner can be best described by being a feasibility study. No concrete plans are yet made for development of the model.

A.3. Conclusion

Transportation is under large influence of governments, as they mostly build and maintain the infrastructure that is needed. Governments can hence influence the

modal split. Governments want the model split for IWT to grow, because roads are facing congestion and IWT is an environmentally friendly way of transport. The inland waterway network of the Netherlands is the busiest in the world. Still, the capacity of waterways is ample to grow, but locks form the main bottlenecks in the capacity of waterways. The largest opportunity for inland waterways to grow is in the growth of container transport. Whereas climate change is an opportunity for inland waterways, it is also a thread. Climate change could result in more frequent floods and droughts, resulting in inaccessibility for vessels. The increase in droughts could mean more shipping locks are needed. The average vessel size has been increasing on the river Rhine, whether this trend will continue cannot be stated with any certainty, as there are many possible futures. Locks are big investments, a lot of money can be saved if these investments can be postponed by introducing better locking operations. There is a lack of cooperation in the IWT sector, the introduction of RIS is a good opportunity to increase this cooperation. The data that is the result of RIS can be applied to lock scheduling.

B

Master problem: The Lock Scheduling Algorithm

The master problem of the algorithm consists of the constraint programming problem. The model consists of three essential components: variables, constraints and an objective function. First, variables are added and their bounds are defined. The variables are denied certain value combinations by adding constraints. For instance the constraint that every vessel is only scheduled once. Then, the best option available is defined by the objective function, that is the minimum total passage time for all vessels. The model considers all possible values until the objective function is minimal.

Continuous variables are introduced for times that are of interest: start of locking, end of locking, waiting time, passage time. Decisions are assigned Boolean (binary) variables, for instance to attach every vessel to only one cycle or variables that check if 2 vessels are in the same cycle.

The model needs input parameters to run. It needs the number of vessels, arrival times of vessels, dimensions of vessels and directions of vessels. Processing times per chamber need to be set and the entering and leaving times per vessel. The chamber dimensions are set. A maximum waiting time is declared. This is needed to define the search space for the model. Standard the maximum waiting time is set at 30 minutes. If there is no feasible solution with this maximum waiting time, the waiting time is increased by 5 minutes by the model, until a feasible solution is found. This way of setting the maximum waiting time increases the speed of the model as the number of options is initially lower. There are also parameters that are output of the sub problem. These parameters are the area constraints, the area order constraints and the initial constraints. The master problem and the sub problem are solved subsequently, this means that the first in the first iteration these parameters are zero.

Sets:

N: set of all ships $i : \{0, 1, 2, \ldots, n\}$

T: set of all chambers $c : \{1, 2, 3\}$

K: set of all lockages $k : \{0, 1, 2, \ldots, n\}$

S: set of all vessel combinations $(i, j) : \{(0, 1), (0, 2), (1, 2), \ldots, (n-1, n)\}$

Variables:

v_{ick}: Binary variable, 1 if vessel i, $i \in N$ is assigned to chamber c, $c \in T$ and lockage k, $k \in K$.

va_i: Integer variable, time of arrival vessel i in chamber, $i \in N$.

vs_i: Integer variable, time of start locking of vessel i, $i \in N$.

vo_i: Integer variable, time that vessel i has sailed out of the chamber, $i \in N$.

p_i: Integer variable, passage time of vessel i, $i \in N$.

w_i: Integer variable, waiting time of vessel i, $i \in N$.

Co_{ck}: Integer variable, time that vessels can start to enter chamber c, $c \in T$ and lockage k, $k \in K$.

Cs_{ck}: Integer variable, time locking is initiated in chamber c, $c \in T$ and lockage k, $k \in K$.

Cc_{ck}: Integer variable, time locking is completed in chamber c, $c \in T$ and lockage k, $k \in K$.

Ce_{ck}: Integer variable, time that all vessels have sailed out in chamber c, $c \in T$ and lockage k, $k \in K$.

D_{ck}: Binary variable, direction of locking in chamber c, $c \in T$ and lockage k, $k \in K$.

W_{ck}: Integer variable, time to wait for all vessels to sail in and out in chamber c, $c \in T$ and lockage k, $k \in K$.

U_{ck}: Binary variable, usage in chamber c, $c \in T$ and lockage k, $k \in K$.

E_{ck}: Binary variable, deletes empty cycles in chamber c, $c \in T$ and lockage k, $k \in K$.

o_{ij}: Binary variable, vessel order for vessel combination $(i,j) \in S$.

W_{ijck}: Binary variable, vessels in the same cycle.

V_{ij}: Binary variable, vessels in the same cycle.

oa_{ij} : Binary variable,

Aor : Binary variable,

Parameters:

$maxwait$:	maximum waiting time.
eta_i:	arrival time for each vessel i, $i \in N$.
$sail$:	sailing time required for a vessel to enter or exit the lock.
$subseq$:	sailing time required for a vessel to enter or exit the lock if it enters subsequently after another vessel.
$proc_c$:	standard processing time of chamber c, $c \in T$.
$area_i$:	vessel area for each vessel i, $i \in N$.
$chamarea_c$:	chamber area for each chamber c, $c \in T$.
dir_i:	direction of each vessel i, $i \in N$.
$acnstrs_{ac_x}$:	set of area constraints.
$aordcnstrs_{aordc_x}$:	set of area order constraints.
ac_x:	a single area constraint
$aordc_x$:	a single area order constraint
$inittime_c$:	time
$initdir_c$:	direction
$fcfs$:	first come first serve, boolean

The variables and parameters are subject to the following constraints:

Equation B.1 states that every vessel can only be scheduled once.

$$\sum_{c \in T} \sum_{k \in K} v_{ick} = 1, \quad \forall i \in N \tag{B.1}$$

Equation B.2 states that time between the arrival of every vessel and the moment the vessel is in the chamber is larger or equal to the time needed for sailing in. In other words, the vessel first has to sail in before it can be in the chamber.

$$va_i \geq eta_i + sail, \quad \forall i \in N \tag{B.2}$$

The vessel waiting time is defined in equation B.3 as the time that the vessel sailed out of the chamber minus the arrival time, the time needed for sailing in the chamber and the processing time. In B.4 the passage time is defined as the time the vessel has sailed out of the chamber minus the arrival time of the vessel.

$$w_i = vo_i - eta_i - 2 \cdot sail - proc_c, \quad \forall i \in N \tag{B.3}$$

$$p_i = vo_i - eta_i, \quad \forall i \in N \tag{B.4}$$

Equation B.5 states that the start time of a locking is always spaced with the processing time to the completion time. And equations B.6 and B.7 define the waiting time of the locking, that is the time needed for the vessels to sail in and sail out.

$$Cc_{ck} = Cs_{ck} + proc_c, \quad \forall c \in T, \forall k \in K \tag{B.5}$$

$$Co_{ck} \leq Cs_{ck} - W_{ck}, \quad \forall c \in T, \forall k \in K \tag{B.6}$$

$$Ce_{ck} = Cc_{ck} + W_{ck}, \quad \forall c \in T, \forall k \in K \tag{B.7}$$

Equations B.8 and B.9 label empty lockages.

$$U_{ck} = 1 \quad \Rightarrow \quad \sum_{i \in N} v_{ick} > 0 \quad \forall c \in T, \forall k \in K \tag{B.8}$$

$$U_{ck} = 0 \quad \Rightarrow \quad \sum_{i \in N} v_{ick} = 0 \quad \forall c \in T, \forall k \in K \tag{B.9}$$

Equations B.10 and B.11 calculate the time for every locking that is needed for the vessels to sail in and out.

$$U_{ck} = 1 \quad \Rightarrow \quad W_{ck} = \sum_{i \in N} v_{ick} \cdot subseq - subseq + sail \quad \forall c \in T, \forall k \in K \tag{B.10}$$

$$U_{ck} = 0 \quad \Rightarrow \quad W_{ck} = 0 \quad \forall c \in T, \forall k \in K \tag{B.11}$$

Equation B.12 is a first quick check for the area of the chamber. The sum of the areas of the vessels assigned to the lockage cannot be larger than the chamber area.

$$\sum_{i \in N} v_{ick} \cdot area_i \leq chamarea_c \quad \forall c \in T, \forall k \in K \tag{B.12}$$

Equations B.13, B.14 and B.15 are trivial but necessary constraints. Equation B.13 requires that locking the start time of a vessel is equal to the start time of it's lockage and equation B.14 that the direction of both is equal. Equation B.15 states that all vessels must be in the chamber before locking can be started.

$$v_{ick} = 1 \quad \Rightarrow \quad vs_i = Cs_{ck} \quad \forall i \in N, \forall c \in T, \forall k \in K \tag{B.13}$$

$$v_{ick} = 1 \quad \Rightarrow \quad D_{ck} = dir_i \quad \forall i \in N, \forall c \in T, \forall k \in K \tag{B.14}$$

$$v_{ick} = 1 \quad \Rightarrow \quad va_i \leq Cs_{ck} \quad \forall i \in N, \forall c \in T, \forall k \in K \tag{B.15}$$

Equations B.16 and B.17 make sure that vessels can only sail in and out when the doors of the chamber are open. Equation B.18 declares the chamber empty after all vessels sailed out.

$$v_{ick} = 1 \quad \Rightarrow \quad va_i \geq Co_{ck} + sail \quad \forall i \in N, \forall c \in T, \forall k \in K \tag{B.16}$$

$$v_{ick} = 1 \quad \Rightarrow \quad vo_i \geq Cc_{ck} + sail \quad \forall i \in N, \forall c \in T, \forall k \in K \tag{B.17}$$

$$v_{ick} = 1 \quad \Rightarrow \quad vo_i \leq Ce_{ck} \quad \forall i \in N, \forall c \in T, \forall k \in K \tag{B.18}$$

Equation B.19 states that every subsequent lockage in a chamber should go in the opposite direction. Equation B.20 makes sure that the subsequent lockage start of sailing in always waits for the vessels to sail out.

$$D_{ck} \neq D_{ck+1} \quad \forall c \in T, \forall \{0, \ldots, k-1\} \in K \tag{B.19}$$

$$Co_{ck+1} \geq Ce_{ck} \quad \forall c \in T, \forall \{0, \ldots, k-1\} \in K \tag{B.20}$$

Equation (B.21) and (B.22) label all cycles that are unused at the end of the planning so that they can be removed.

$$E_{ck} = 1 \quad \Rightarrow \quad \sum_{k_0}^{k_n} U_{ck} = 0 \quad \forall c \in T, \forall k \in K \tag{B.21}$$

$$E_{ck} = 0 \quad \Rightarrow \quad \sum_{k_0}^{k_n} U_{ck} > 0 \quad \forall c \in T, \forall k \in K \tag{B.22}$$

Equation B.23 makes sure that a chamber never has 2 subsequent empty lockings, in that case it could better have waited.

$$E_{ck} = 0 \quad \Rightarrow \quad U_{ck} + U_{ck+1} \neq 0 \quad \forall c \in T, \forall \{0, \dots, k-1\} \in K \tag{B.23}$$

Equations B.24 until B.27 make sure that vessels sail in and out of the lockage in the same order.

$$o_{ij} = 1 \quad \Rightarrow \quad va_i \geq va_j \quad \forall i, j \in S \tag{B.24}$$

$$o_{ij} = 0 \quad \Rightarrow \quad va_i < va_j \quad \forall i, j \in S \tag{B.25}$$

$$o_{ij} = 1 \quad \Rightarrow \quad vo_i \geq vo_j \quad \forall i, j \in S \tag{B.26}$$

$$o_{ij} = 0 \quad \Rightarrow \quad vo_i < vo_j \quad \forall i, j \in S \tag{B.27}$$

Equations B.28 and B.29 label each vessel combination in the same lock and equations B.30 and B.31 use this to state that they cannot enter or exit a lock at the same time.

$$W_{ijck} = 1 \quad \Rightarrow \quad v_{ick} + v_{jck} = 2 \quad \forall i, j \in S \tag{B.28}$$

$$W_{ijck} = 0 \quad \Rightarrow \quad v_{ick} + v_{jck} < 2 \quad \forall i, j \in S \tag{B.29}$$

$$W_{ijck} = 1 \quad \Rightarrow \quad va_i \neq va_j \quad \forall i, j \in S \tag{B.30}$$

$$W_{ijck} = 1 \quad \Rightarrow \quad vo_i \neq vo_j \quad \forall i, j \in S \tag{B.31}$$

Equations B.32 and B.33 create helper variables to deal with the area constraints.

$$V_{ij} = 1 \quad \Rightarrow \quad \sum_{c \in T, k \in K} W_{ijck} = 1 \quad \forall i, j \in S \tag{B.32}$$

$$V_{ij} = 0 \quad \Rightarrow \quad \sum_{c \in T, k \in K} W_{ijck} = 0 \quad \forall i, j \in S \tag{B.33}$$

Equation B.34 is the area constraint, it restricts vessels to be in the same lockage if an area constraint holds. Same cycle vessels is equal to 1 if the two vessels that it refers to are in the same lockage. Therefore at least one of these values should be 0 or the sum of all these vessels cannot be equal to the amount of values. Note that the vertical stripes mean cardinality (amount of values in the set) and not absolute value in this particular case.

$$\sum_{i,j \in ac_x} V_{ij}! = |V_{ij}| \quad \forall ac_x \in acnstrs_{ac_x} \tag{B.34}$$

The following equations (B.35 until B.39) represent the area order constraint. This constraint checks if the vessel do fit in the assigned chamber in the order that they are assigned.

The variables oa_{ij} are introduced in equation B.35 and B.36. These variables store the order of the vessels that are constrained.

$$oa_{ij} = 0 \quad \Rightarrow \quad index(x_i) \leq index(x_j) \quad \forall i,j \in aordc_x \forall aordc_x \in aordcnstrs_{aordc_x} \quad \text{(B.35)}$$

$$oa_{ij} = 1 \quad \Rightarrow \quad index(x_i) \geq index(x_j) \quad \forall i,j \in aordc_x \forall aordc_x \in aordcnstrs_{aordc_x} \quad \text{(B.36)}$$

Equation B.37, B.38 and B.39 represent the core of the area order constraint. The vessels can be in the same cycle (B.37), but then the order of the vessels has to be changed (B.38). The vessels can also simply be scheduled in another locking cycle (B.39). The variables Aor are introduced to make a decision, either equations B.37 and B.38 hold or B.39.

$$Aor_{aordc_x} = 0 \quad \Rightarrow \quad \sum_{i,j \in Ac} V_{ij} == |V_{ij}| \quad \forall i,j \in aordc_x \forall aordc_x \in aordcnstrs_{aordc_x} \quad \text{(B.37)}$$

$$Aor_{aordc_x} = 0 \quad \Rightarrow \quad \{o_{ij} \in aordc_x\} \neq \{oa_{ij} \in aordc_x\} \quad \text{(B.38)}$$

$$Aor_{aordc_x} = 1 \quad \Rightarrow \quad \sum_{i,j \in Ac} V_{ij}! = |V_{ij}| \quad \forall i,j \in aordc_x \forall aordc_x \in aordcnstrs_{aordc_x} \quad \text{(B.39)}$$

Equations B.40 and B.41 initiate the start time and the start directions for each chamber.

$$D_{c0} = initdir_c \quad \forall c \in T \quad \text{(B.40)}$$

$$Co_{c0} = inittime_c \quad \forall c \in T \quad \text{(B.41)}$$

In equation B.42 the maximum waiting time is defined.

$$w_i \leq maxwait \quad \forall i \in N \quad \text{(B.42)}$$

The following constraint is the first come first serve constraint, it is only used if FCFS = 1.

$$fcfs = 1 \quad \wedge \quad V_{ij} = 1 \quad \Rightarrow \quad o_{ij} = 0 \quad \forall i,j \in S \quad \text{(B.43)}$$

C

Code

All relevant code, files and other digital material can be found on:

https://github.com/TUDelft-CITG/Lock-Simulation

Figure C.1: Scanning this QR code opens the link to the code.

D

The Three-Way Best-Fit Algorithm

The pseudo code of this algorithm can be found in algorithm 1 on page 108. References to the pseudo code are made in this paragraph in italic. The three-way best-fit heuristic is based on eighteen combinations of three ordering methods and six placement methods (*OrderingPolicy* and *PlacementPolicy*). An ordering method is the sequence in which the ships are fitted in the lock chamber. A placement method is the way the vessels are positioned in the lock. The algorithm subsequently tries all of the combinations until a possible fit of the set of ships is found. If no fit can be found, the set of ships can be considered unfeasible for the lock chamber. Initially, a basic version of the algorithm is implemented.

In the algorithm first the *Skyline* is defined, it will keep track of the contours of the vessels that are in the lock as is illustrated in figure **??**. At the start the lock is empty, so every value of the skyline is zero except for the first and last value, these values are infinite to represent the chamber walls. The first ordering method that is tried is *DecreasingWidth*, in combination with the placement policy of *LeftMost*. The ships are consequently sorted in decreasing width (*line 6*). The first vessel in the list, which is the widest vessel, is first fitted in the lock. This will fit, as there is no vessel yet in the lock. It is placed in the far left corner of the chamber, because this is implied by the first placement policy. The vessel is deleted from the *VesselList*. The skyline is updated to accommodate the vessel. Now a *LowestGap* is defined, this is the space that is left next to the first fitted vessel (the values in the skyline with the lowest x-values). This lowest gap will be tried to fill with another vessel in our vessel list. The vessel list is therefore consulted for the first vessel that fits in the gap. If this gap is wide enough a vessel can be found to fit this gap. If the vessel is fitted, it is also deleted from the vessel list and the skyline is updated again. After which the next vessel in order can be tried to fit the next lowest gap. If the lowest gap is too small to fit any vessel, the gap is filled up to the length of the shortest vessel along this gap in the skyline and the next lowest gap is defined. If the skyline surpasses the length of the chamber, the algorithm returns false, because the combination of ordering and placement could not fit the vessels in the chamber. Another attempt is made with the next combination of ordering and placement policy. If the algorithm doesn't return false after trying all the combinations, the vessels fit in the chamber, the algorithm is ended and true is returned.

In total there are three ordering policies. Vessels are sorted based on their width *DecreasingWidth*, on their length *DecreasingLength* and on their area *DecreasingArea*. They are all sorted from large to small. This strategy makes use of the fact

that a larger vessel is harder to fit than a smaller one, therefore the largest is tried first. The three ordering policies are combined with six placement policies. Vessels are always placed in the lowest gap, but on which side of the gap is determined by the placement policies. The *LeftMost* and *RightMost* placement policies simply place a vessel on the left or the right side of the lowest gap. *MaximumDifference* and *MinimumDifference* place the vessel on the side with the maximum or minimum difference in height with is neighbour. *MaximumDifference* will tend to position vessels to the chambers walls as they are defined as infinite height. *MinimumDifference* will tend to show opposite behaviour, it will place a vessel adjacent another vessel rather then to the chamber walls. The *ShortestNeighbour* and *TallestNeighbour* ordering policies simply place the vessel next to the side that is lowest in case of the *ShortestNeighbour* and the highest in the *TallestNeighbour* case. Note that this is different than the difference with the neighbouring vessel.

Algorithm 1 ThreeWayBestFit

```
Input: VesselList(VesselWidth, VesselLength)
Input: ChamberWidth, ChamberLength
 1: OrderingPolicy ←(DecreasingWidth, DecreasingLength, DecreasingArea)
 2: PlacementPolicy ←(LeftMost, ShortestNeighbour, MaximumDifference,
    RightMost, TallestNeighbour, ShortestDifference)
 3: for all OrderingPolicy do
 4:   for all PlacementPolicy do
 5:     Skyline ←list(Inf., 0, 1, 2, ..., ChamberWidth - 1, ChamberWidth, Inf.)
 6:     SortedVessels ←sort VesselList according to OrderingPolicy
 7:     VesselInChamber ←empty list
 8:     while SortedVessels not empty do
 9:       LowestGap ←minimum(skyline)
10:       LowestGapWidth ←width of LowestGap
11:       if LowestGapWidth ≥ VesselWidth and  then
12:         if LowestGap + VesselLength > ChamberLength then
13:           go to line 4 next PlacementPolicy
14:         end if
15:         VesselInChamber ←Vessel according to PlacementPolicy
16:         SortedVessels ←remove Vessel
17:         Skyline ←add VesselLength according to PlacementPolicy
18:       else if LowestGapWidth < VesselWidth and Vessel is last in VesselList then
19:         Skyline ←raise to second minimum(Skyline)
20:       else if SortedVessels is empty then
21:         return  True
22:       end if
23:     end while
24:   end for
25: end for
26: return  False
```

E

All experiments

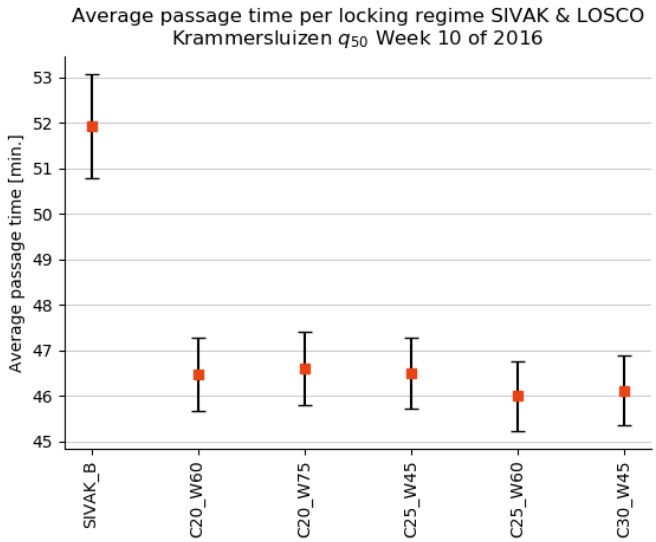


Figure E.1

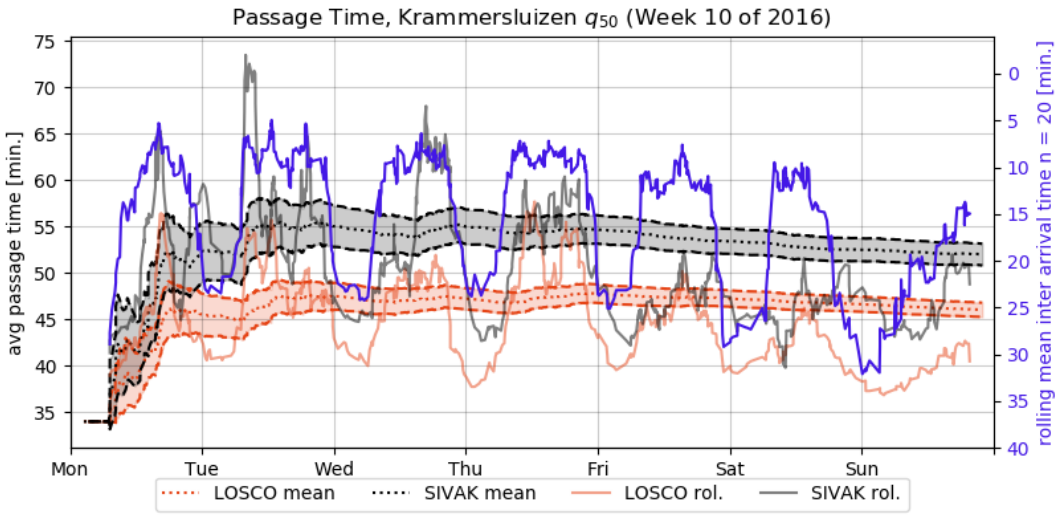


Figure E.2

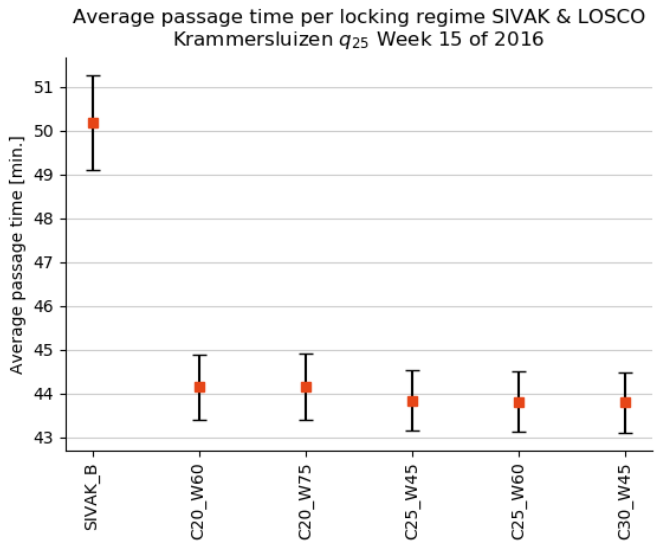


Figure E.3

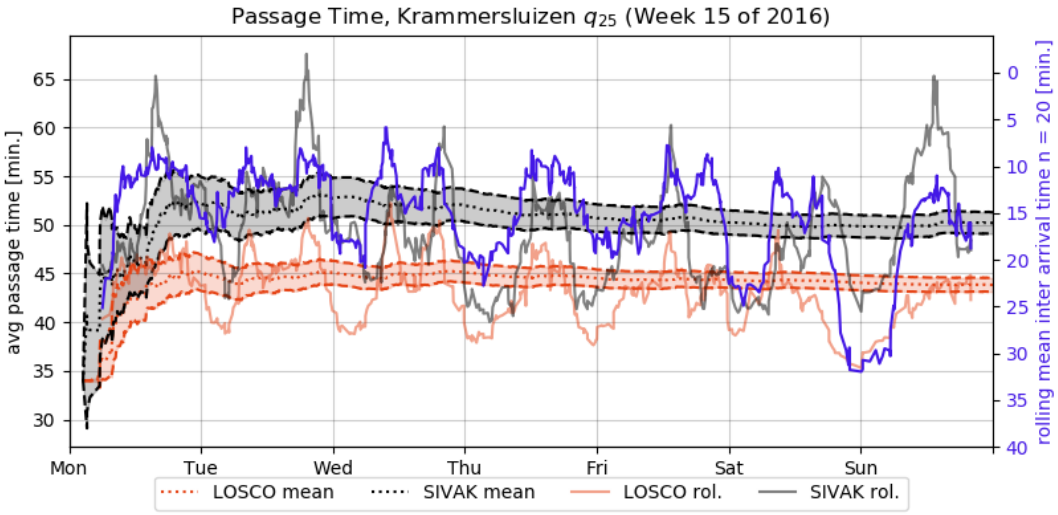


Figure E.4

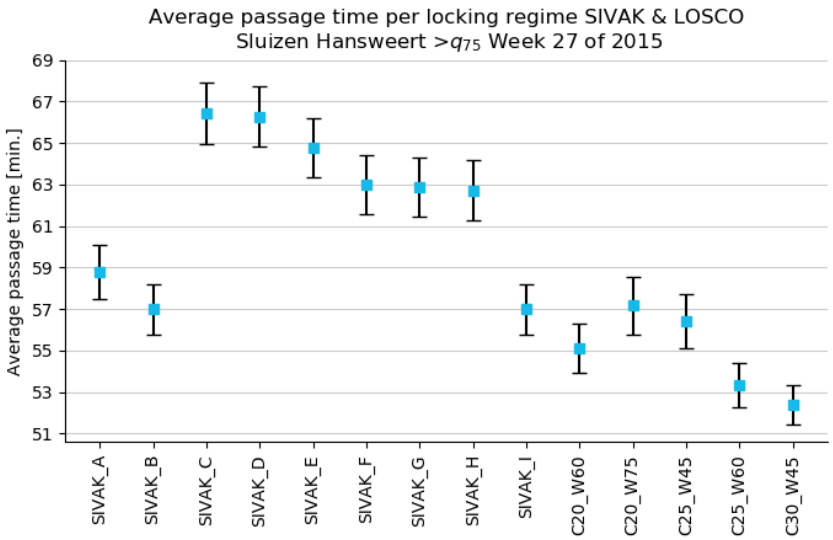


Figure E.5

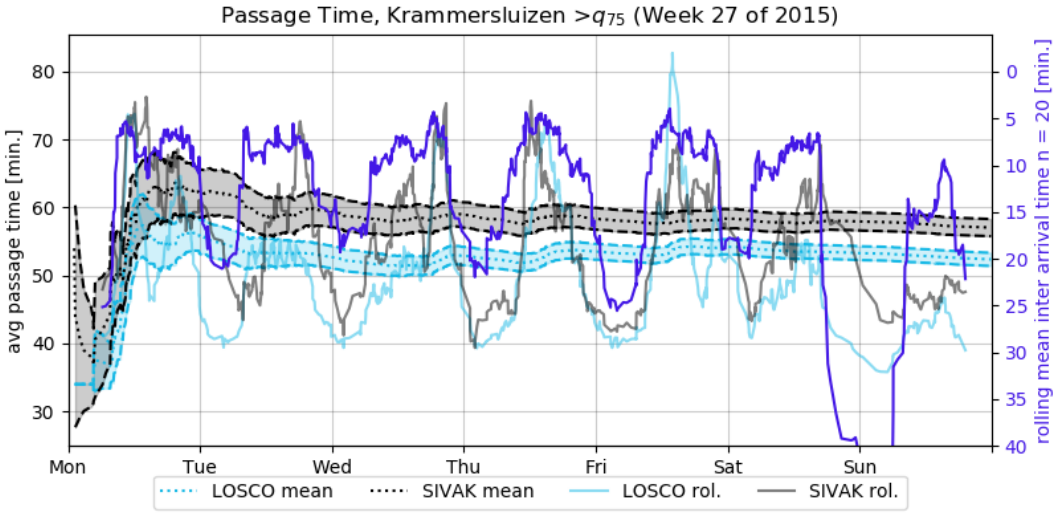


Figure E.6

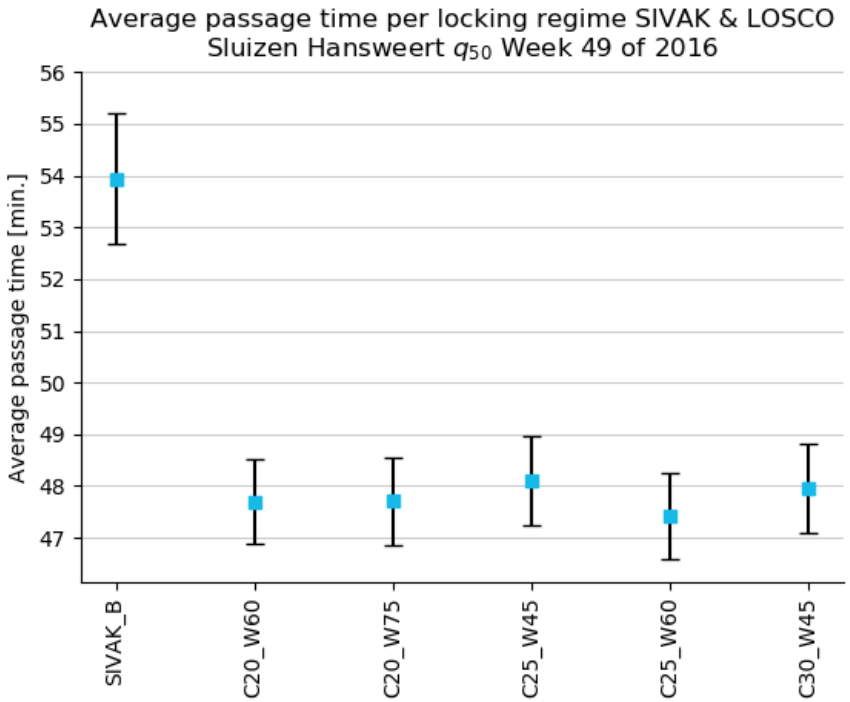


Figure E.7

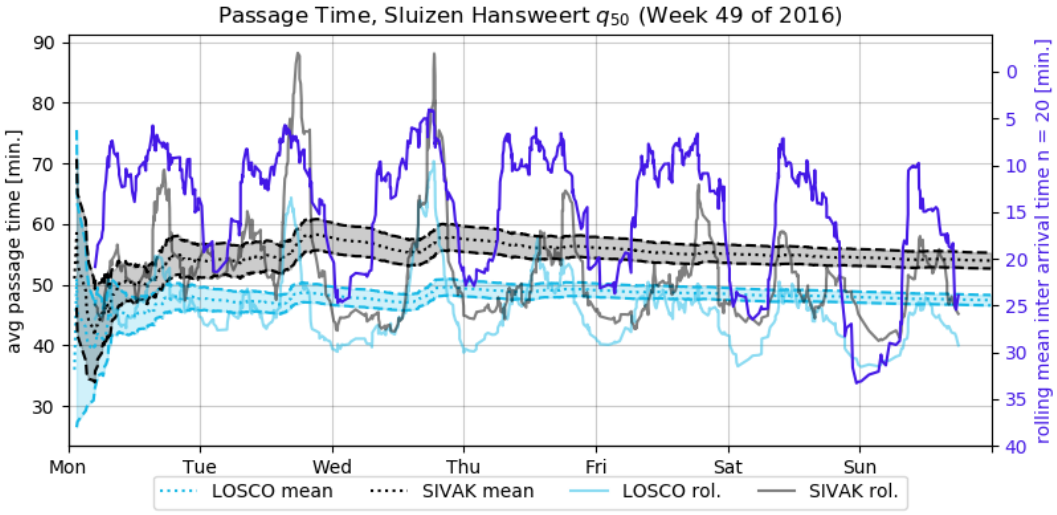


Figure E.8

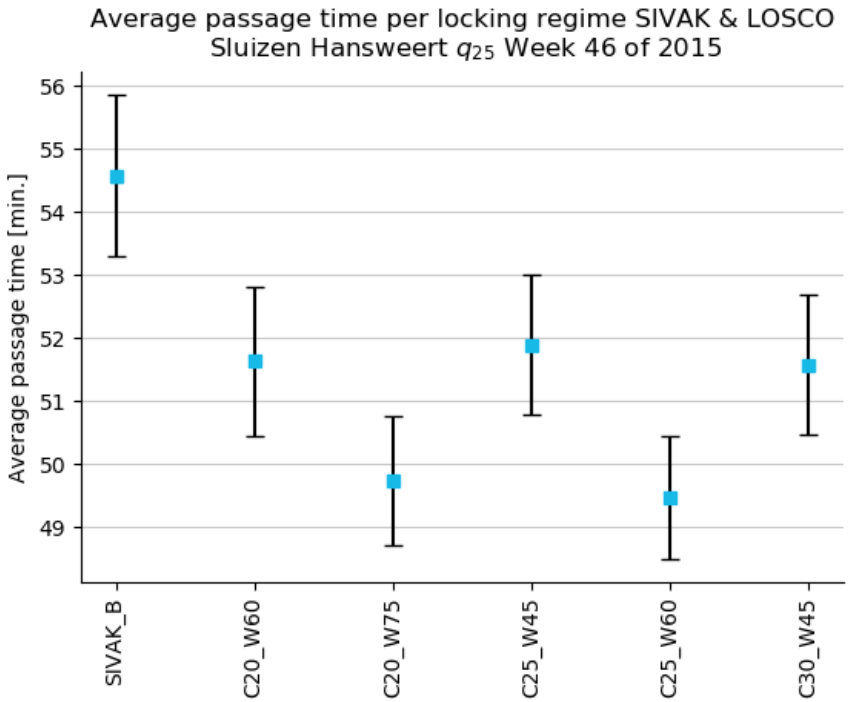


Figure E.9

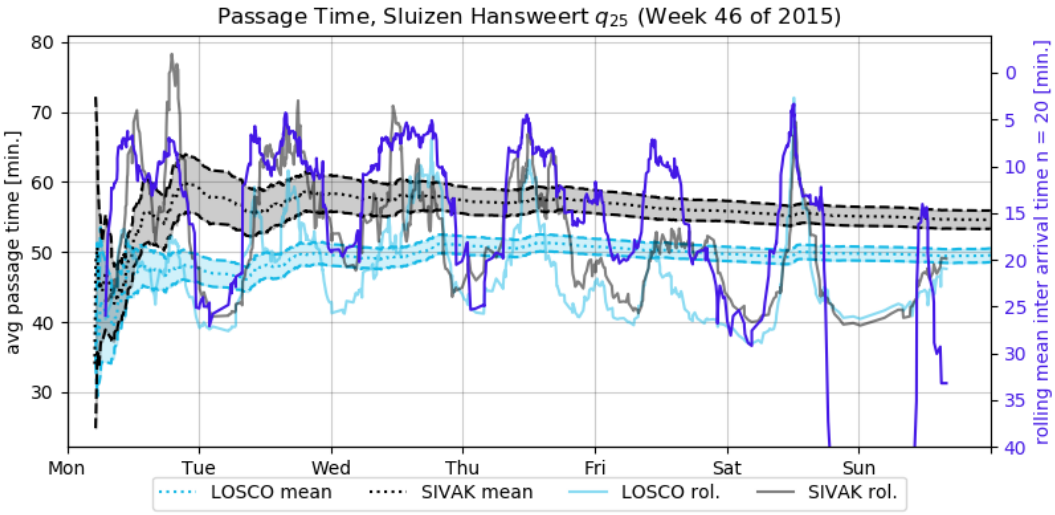


Figure E.10

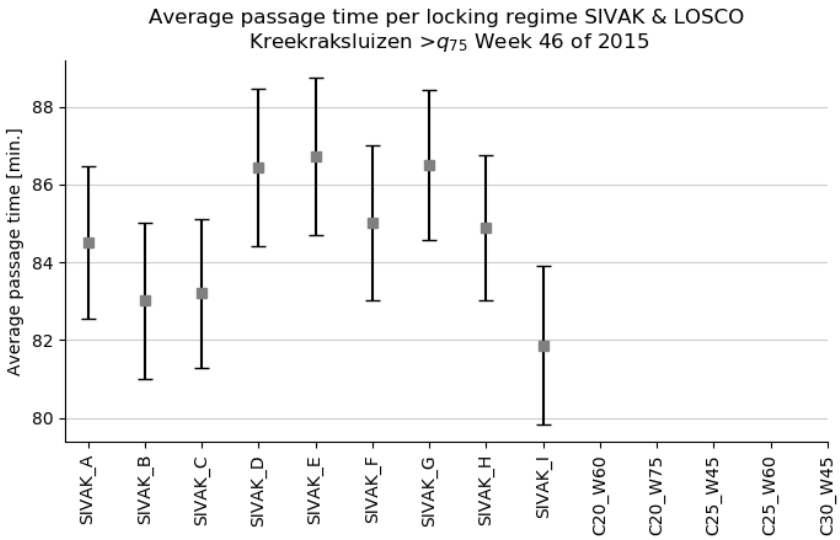


Figure E.11

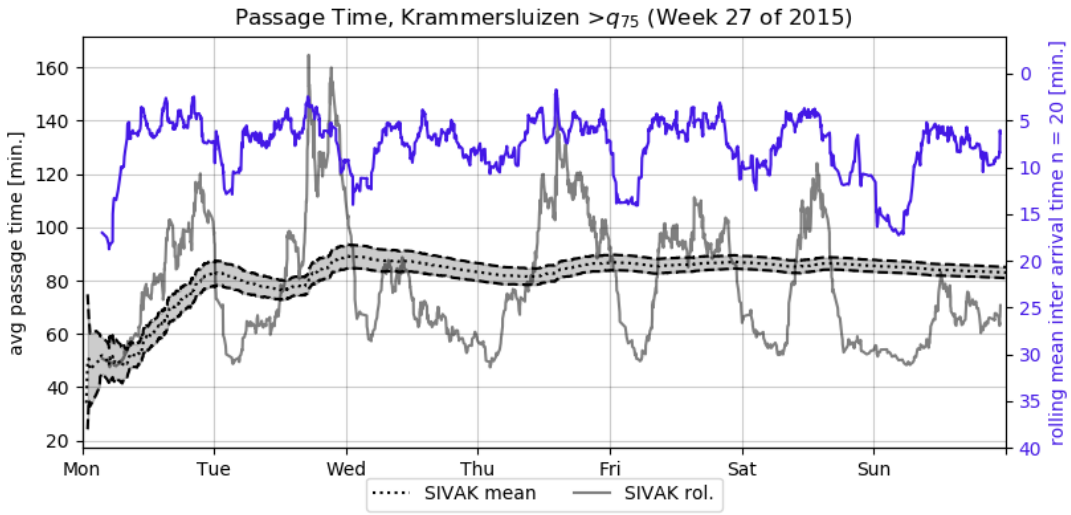


Figure E.12

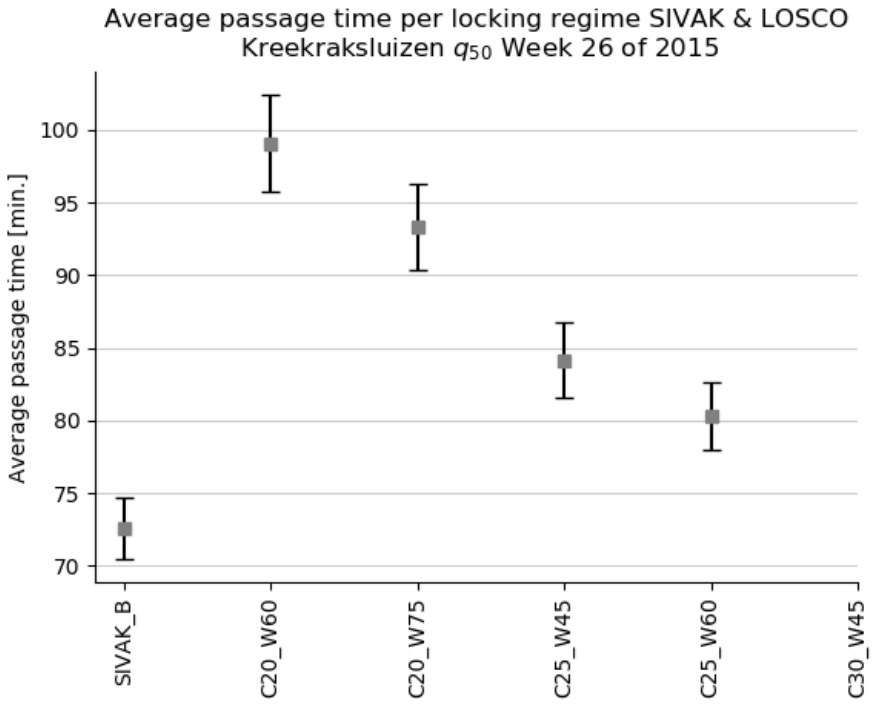


Figure E.13

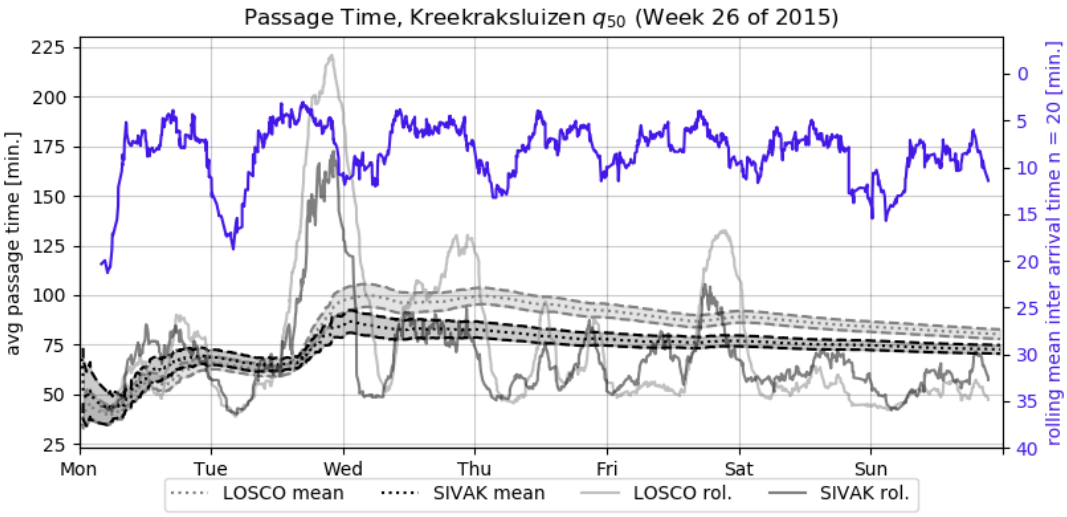

Passage Time, Kreekraksluizen q_{50} (Week 26 of 2015)
avg passage time [min.]
rolling mean inter arrival time n = 20 [min.]
225
200
175
150
125
100
75
50
25
0
5
10
15
20
25
30
35
40
Mon
Tue
Wed
Thu
Fri
Sat
Sun
LOSCO mean
SIVAK mean
LOSCO rol.
SIVAK rol.

Figure E.14

F

Factor costs table

Table F.1

Vracht-binnenvaart RWS-klasse	Overig	Containers	Nat	lengte (m)
M0	30	30	30	
M1	31.53			38.5
M2	36.08	39.23		55
M3	39.1	42.45		65
M4	42.76			73
M5	50.83			80
M6	56.33	65.37	94.46	85
M7	82.62	86.67	123.91	105
M8	96.58	101.25	137.17	110
M9	108.93	118.08	148.98	135
M10	110	129.51	156.41	
M11	126	136.9	172.01	
M12	140	151.05	192.26	
B01	43.15			
B02	47.6			
B03	50.21			
B04	52.4			
BI	87.57			
BII-1	91.77			
BIIa-1	96.26			
BIIL-1	105.74			
BII-2L	146.1			
BII-2b	142.97			
BII-4	244.83			
BII-6l	302.34			
BII-6b	301.83			
C1l	44.75			
C1b	44.95			
C2l	110.37			
C2b	109.89			
C3l	132.45	125.88		
C3b	129.96	125.77		
C4	165.76	163.19		

List of Abbreviations

AIS Automatic Identification System. 94

ATD Actual Time of Departure. 95

BICS Binnenvaart Informatie- en Communicatie Systeem (Inland Waterway Transport Communication System). 94

CEMT Conférence Européenne des Ministres de Transport (vessel classification). 87, 90, 92

COP Constraint Optimisation Problem. 10

CP Constraint Programming. 11, 12

CSP Constraint Satisfaction Problem. 10

ECDF Empirical Cumulative Distribution Function. 55

ETA Estimated Time of Arrival. 95

ETD Estimated Time of Departure. 95

FCFS First Come, First Served. 41, 46, 55

IDVV Impuls Dynamisch Verkeersmanagement Vaarwegen (Impulse Dynamic Traffic Management Waterways). 95

IWT Inland Waterways Transport. viii, 1, 2, 83–86, 88, 89, 95

JSSP Job Shop Scheduling Problem. 10–12, 81

MARNIS Maritime Navigation and Information Service (EU research project). 94

MIP Mixed Integer Programming. 11, 12, 14

RIS River Information Services. 2, 94, 96

RWS Rijkswaterstaat. 90

SIVAK Simulatiepakket voor Verkeersafwikkeling bij Kunstwerken, (translated as simulation tool for traffic handling at engineering structures). vii, 8, 23–25, 28, 29

TEU Twenty Food Equivalent Unit (container size). 85

References

Blum, C. and Roli, A. (2003). Metaheuristics in Combinatorial Optimization : Overview and Conceptual Comparison. 35(3):268–308.

Bureau Voorlichting Binnenvaart (2016). Waardevol Transport. Technical report, Bureau Voorlichting Binnenvaart.

Buro Sierenberg en De Gans, Ministerie van Verkeer en Waterstaat, Rijkswaterstaat, Dienst Verkeerskunde], P. H. (1991). *SIVAK handleiding : SImulatie pakket voor Verkeers Afwikkeling bij Kunstwerken van Rijkswaterstaat Dienst Verkeerskunde te Rotterdam : versie 1.01*. Rotterdam : Ministerie van Verkeer en Waterstaat, Rijkswaterstaat, Dienst Verkeerskunde (RWS, DVK), Rotterdam.

Campbell, J. F., Smith, L. D., Sweeney, D. C., Mundy, R., and Nauss, R. M. (2007). Decision tools for reducing congestion at locks on the upper Mississippi river. *Proceedings of the Annual Hawaii International Conference on System Sciences*, pages 1–10.

Caris, A., Limbourg, S., Macharis, C., van Lier, T., and Cools, M. (2014). Integration of inland waterway transport in the intermodal supply chain: A taxonomy of research challenges. *Journal of Transport Geography*, 41:126–136.

Central Commission for the Navigation of the Rhine (2017). Inland Navigation in Europe Market Observation (No. Annual Report 2017). Technical report, CCR, Strasbourg.

Christodoulou, A. and Demirel, H. (2018). Impacts of climate change on transport - A focus on airports, seaports and inland waterways. Technical report, European Commission, Luxembourg.

Clay Mathematics Institute (2020). P vs NP Problem. https://www.claymath.org/millennium-problems/p-vs-np-problem visited at 10-5-2020.

Crainic, T. G. and Laporte, G. (1997). Planning models for freight transportation. *European Journal of Operational Research*, 97(1):409–438.

Dantzig, G. B. (1963). *Linear Programming and Extensions*. Princeton University Press, Princeton.

European Commision (2011). White paper 2011 - 'Roadmap to a Single European Transport Area'. Technical report, European Union, Brussels.

Google (2018). CP Approach to Integer Optimization | OR-Tools | Google Developers. https://developers.google.com/optimization/mip/integer_opt_cp visited at 15-8-2019.

Google (2020). The Job Shop Problem. https://developers.google.com/optimization/scheduling/job_s visited at 2-4-2020.

Gurobi Optimization LLC. Mixed-Integer Programming (MIP) – A Primer on the Basics. https://www.gurobi.com/resource/mip-basics/ visited at 20-1-2020.

Hekkenberg, R. G. (2013). *Inland Ships for Efficient Transport Chains*. PhD thesis, TU Delft, Delft.

Hekkenberg, R. G., Van Dorsser, C., and Schweighofer, J. (2017). Modelling sailing time and cost for inland waterway transport. *European Journal of Transport and Infrastructure Research*, 17(4):508–529.

Hengeveld, J. *Optimization to reduce waiting times at locks*. MSc, TU Delft.

Hermans, J. (2014). Optimization of inland shipping A polynomial time algorithm for the single-ship single-lock optimization problem. *Journal of Scheduling*, 17(4):305–319.

Hurkmans, R., Terink, W., Uijlenhoet, R., Torfs, P., Jacob, D., and Troch, P. A. (2010). Changes in streamflow dynamics in the Rhine basin under three high-resolution regional climate scenarios. *Journal of Climate*, 23(3):679–699.

International Buck Consultants Rotterdam (2008). Een goede toekomst voor het kleine schip - Visie en actieplan. Technical Report November, Rotterdam.

Kennisinstutuut voor Mobiliteitsbeleid (2018). Kerncijfers Mobiliteit 2018. Technical report, Ministry of Infrastructure and Water Management, Den Haag.

Konings, R. (2007). Development of Container Barge Transport on Small Waterways: From Increasing Scale to Increasing Scope. *Transportation Research Record: Journal of the Transportation Research Board*, 1871(1):24–32.

Martinelli, D. and Schonfeld, P. (1995). Approximating Delays at Interdependent Locks. *Journal of Waterway, Port, Coastal and Ocean Engineering*, 121(November/Kennisinstutuut voor Mobiliteitsbeleid (2018)):300–307.

Ministry of Infrastructure and the Environment (2015). The Dutch Maritime Strategy. Technical report, The Hague.

Molenaar, W. (2011). *Hydraulic Structures: Locks*. TU Delft, Delft, March 2011 edition.

Monash University (2019). MiniZinc Challenge 2019 Results. https://www.minizinc.org/challenge2019/results2019.html visited at 20-3-2020.

Nauss, R. M. (2008). Optimal sequencing in the presence of setup times for tow / barge traffic through a river lock. *European Journal of Operational Research*, 187:1268–1281.

Negenborn, R. R. and Dekker, R. (2013). Synchromodal Container Transport: An Overview of Current Topics and Research Opportunities. Technical Report March 2016, Delft University of Technology, Erasmus University Rotterdam.

Passchyn, W. (2016). *Scheduling Locks on Inland Waterways*. PhD thesis, KU Leuven.

Pauli, G. (2010). Sustainable transport : A case study of Rhine navigation. *Natural Resources Forum*, 34:236–254.

Petersen, E. R. and Taylor, A. J. (1988). An Optimal Scheduling System for the Welland Canal. *Transportation Science*, 22(3):161–230.

Radmilovic, Z., Maras, V., and Jovanovic, S. (2007). Ship Lock as General Queuing system with Batch Arrivals and Batch Service. *Promet - Traffic & Transportation*, 19(6):343–352.

Rijkswaterstaat (2000). *Ontwerp van schutsluizen*. Rijkswaterstaat, Utrecht.

Rijkswaterstaat (2011). Kennisopbouw met betrekking tot groeiend vervoer van goederen over het water. Technical report.

Rijkswaterstaat (2014a). Beter benutten van de vaarwegen - IDVV. Technical report.

Rijkswaterstaat (2014b). Pilot Verkeerscentrale van Morgen - Technische Realisatie - Globale beschrijving Trajectplanner. Technical report.

Rijkswaterstaat (2015). Beheer- en ontwikkelplan voor de rijkswateren 2016 - 2021. Technical report, Rijkswaterstaat.

Rijkswaterstaat (2018). Kostenbarometer en binnenvaarttool.

Rijkswaterstaat (2020). Steunpunt Economische Expertise - Discontovoet. https://www.rwseconomie.nl/discontovoet%0A visited at 20-4-2020.

Rotterdam Port Authority (2014). Progress Report 2014 Port Vision 2030. Technical report, Rotterdam.

Russel, S. and Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, Upper Saddle River, New Jersey, 3rd edition.

Schonfeld, P. and Ting, C. J. (1998). Integrated Control for Series of Waterway Locks. *Journal of Waterway, Port, Coastal and Ocean Engineering*, 124(July/August):199–206.

SmartPort (2018). Smart ships and the changing maritime ecosystem. Technical report, SmartPort.

Smith, L. D., Nauss, R. M., Christian, D., Li, J., Ehmke, J. F., and Reindl, M. (2011). Scheduling operations at system choke points with sequence-dependent delays and processing times q. *Transportation Research Part E*, 47(5):669–680.

Terekhov, D., Down, D. G., and Beck, J. C. (2014). Queueing-Theoretic Approaches for Dynamic Scheduling : A Survey. *Surveys in Operations Research and Management Science*, 19(2):105–129.

van der Horst, M., Kort, M., Kuipers, B., and Geerlings, H. (2019). Coordination problems in container barging in the port of Rotterdam: an institutional analysis. *Transportation Planning and Technology*, 42(2):187–199.

van Dorsser, C. (2019). TU Delft CIE5306 Ports and Waterways II - Autonomous Shipping (Lecture) 12-6-2019.

van Dorsser, C., Taneja, P., and Vellinga, T. (2018). PORT METATRENDS Impact of long term trends on business activities, spatial use and maritime infrastructure requirements in the Port of Rotterdam. Technical report, TU Delft, Delft.

Van Haastert, M. (2003). *Planningsmodel scheepvaartafhandeling bij de Noordersluis in IJmuiden*. MSc thesis, TU Delft.

Verstichel, J. (2013). *The lock scheduling problem*. PhD thesis, KU Leuven.

Wilson, H. G. (1977). On the Applicability of Queuing Theory to Lock Capacity Analysis. 12(3):175–180.

Zhang, X. (2008). Simulation Based Parallel Genetic Algorithm to the Lockage Coscheduling of the Three Gorges Project. *2008 International Conference on Computer Science and Software Engineering*, 3:324–327.