rijksuniversiteit groningen

faculteit Wiskunde en Natuurwetenschappen

A dynamic model for incidents on the Dutch inland waterways

Master's thesis in Mathematics

August 2012

Student: T.W. Scholten

First supervisor: prof. dr. E.C. Wit

Second supervisor: dr.ir. R.W.C.P. Verstappen

A dynamic model for incidents on the Dutch inland waterways

Finding the extra delay, number of infeasible routes and costs due to a broken lock or vessel accident during a given length of time

Committee:
prof. dr. E.C. (Ernst) Wit
dr.ir. R.W.C.P. (Roel) Verstappen
dr. ir. J.L. (Hans) Korving
drs. A. (Arwen) Korteweg
dr. ir. A.C. (Arie) de Niet

rijksuniversiteit groningen

Witteveen+Bos

ABSTRACT

Few incidents occur on the Dutch waterways, but when they do, the consequences can be severe. A robust network therefore helps the Port of Rotterdam to maintain its reputation as a stable and reliable company. Keeping a network robust and improving it requires knowledge of these consequences but since they are so scarce, insufficient data has been gathered to analyze where the weakest links are.This leads to a desire to come up with an estimate for the delay, costs and the number of infeasible routes caused by broken locks or vessel accidents.

Similar problems have been solved in the past by models that describe the movement of vessels, but none have a focus on incidents and are able to handle interaction between vessels dynamically. A new predictive dynamical model is designed with a focus on minimizing computational time. Vessels are modeled using a discrete Markov chain while the locks in the network are updated by a deterministic process. Within this model an incident can be defined and set to last for a given length of time.

ACKNOWLEDGEMENTS

It would not have been possible to write this thesis without the help and support of the kind and patient people around me, to only some of whom it is possible to give particular mention here.

Above all I would like to thank prof. dr. **Ernst Wit** for his help, dedication, knowledge and especially the time he took to guide me through the whole process. The quality of this report would not have been the same if it had not been for the big help and great amount of input that came from his part.

I would like to thank my mentor at Witteveen+Bos dr. ir. **Hans Korving** for the guidance, advice and support. I greatly appreciate the opportunity he gave me along with the freedom to fill in the project in my own way.

At the same company dr. ir. **Arie de Niet** helped me out a great deal with the Matlab environment and I would like to thank him for the useful feedback I received from him.

From another department but also working at Witteveen+Bos I would like to thank **Peter Quist** and **Martijn Ruygers** for the advice they gave me. With their knowledge of the Dutch waterways and the time Martijn took to help me out with some of the details they have been a big help.

I would like to thank drs. **Arwen Korteweg**, the principal of the whole project, for the faith he had in the project along with the helpful input and guidance.

In order to carry out some field research I made a trip from Groningen to the Rotterdam harbor on a Barge. This could not have been possible without the hospitality and help of the Operative manager of m.s. de Carpe Diem: **Vincent Ooms**.

From Charta software in Rotterdam I would like to acknowledge **Karsten Uil** for inviting me over for an interview about the BIVAS model. Besides that, I greatly appreciate I was granted permission to use the data that was needed as the input for the model.

Among my colleagues I would like to thank **Ellen Fest** and **Joost Veurink** for their help in the practical matters within the company. Along with that they were always great company to be around.

Special thanks go out to my nephew **Ilhan Spiekman** by helping me out with a thorough review of the thesis

Finally, I would like to thank my parents, classmates and friends for their patience, input, help and sometimes prized distractions to be able to continue with a fresh look.

CONTENTS

1 MAIN OUTLINE 19
1.1 Parts thesis 19
1.2 Model introduction 19

I MOTIVATION AND PREPARATION 21

2 INVOLVED COMPANIES 23
2.1 Witteveen+Bos 23
2.2 Port of Rotterdam 23
2.3 Rijkswaterstaat 24

3 MODELS 27
3.1 SIVAK 27
3.2 SIMDAS 27
3.3 BIVAS 27

4 NETWORK DATA 29
4.1 Manipulated data 29
4.2 CEMT-types 32
4.3 Matrix 33

5 FLEET DATA 35
5.1 Current Fleet 35
5.2 Fleet expansion 35

II STOCHASTIC CONGESTION MODEL 37

6 OUTLINE MODEL 39
6.1 Setup 39
6.2 Markov model 39
6.3 Assumptions 42

7 VESSELS 45
7.1 Types 45
7.2 Harbor 46

8 PARTS 47
8.1 Links 47
8.2 Junctions 47

9 ROUTE PLANNER 49
9.1 Routes 49
9.2 Ants 50
9.3 generate matrix 53
9.4 Catching errors 55
9.5 Route Possible 57
9.6 Update 58
9.7 Overview route planner 62

10 FIT VESSELS IN A LOCK 63
10.1 Theory 63
10.2 Example 64

11 CONGESTION 69
11.1 Vessels 69
11.2 Locks 69
11.3 Parallel locks 70
11.4 number of lockings 74

12 INCIDENT AND RESULTS 75
12.1 Incident definition 75
12.2 Incident simulation 76
12.3 Stopping time 77
12.4 Overview incident 77

III RESULTS FROM CONGESTION ANALYSIS 79
13 MODEL ANALYSIS 81
13.1 (Un)finished parts 81
13.2 Verify model 82
13.3 Calculation speed 82
13.4 discussion 83
14 FUTURE EXTENSIONS 85
14.1 Costs 85
14.2 Rest places 85
14.3 locking time dependent on vessel sizes 85
14.4 Ghost links 85
14.5 Water levels 86
14.6 Bridges 86
14.7 In/out-flow 86
14.8 Behavior vessels 86
A BIVAS DATA 89

LIST OF TABLES

Table 1 legend of figure 4 30
Table 2 legend of figure 5 31
Table 4 Classification CEMT type maximum conditions in meters 32
Table 3 legend of figure 6 32
Table 5 Generated routes 52
Table 6 Vessels in lock 66
Table 7 network data BIVAS 90
Table 8 vessel data BIVAS 91

LIST OF FIGURES

Figure 1 model flow list 20
Figure 2 Broken lock Eefde 24
Figure 3 BIVAS model 28
Figure 4 full network 30
Figure 5 nodes to merge and delete 31
Figure 6 simplified 32
Figure 7 simplified with background 33
Figure 8 simplified with background zoomed 34
Figure 9 visualization 34
Figure 10 model flow list 40
Figure 11 Updating ants 50
Figure 12 Ants updating part 1 51
Figure 13 Ants updating part 2 52
Figure 14 example network 54
Figure 15 reverse present 56
Figure 16 routeplanner 61
Figure 17 route planner results 62
Figure 18 vesselfitting 67
Figure 19 running time vessel fitting 68
Figure 20 uniform distribution lock 71
Figure 21 Unequal lock decomposition 73
Figure 22 input generators 78
Figure 23 model2 78

LIST OF SYMBOLS

E	Expected value
γ	Number of nodes in the system
t	Time
Δt	Time step
S_t	State at time step t
N	Nodes
n	Single node
L	Links
l	Single link
$X(n)$	x-coordinate of a node
$Y(n)$	y-coordinate of a node
$\|l\|$	Length of a link
d	Distance
$C(l)$	CEMT type of link
$\tilde{C}_k$	k-th class of CEMT types
$D^k_{i,j}$	Distance of link (i, j) of the k-th class of CEMT types
$O_{t,j}$	State of a lock
X_{t,i_j}	State of a vessel
$N(o_i)$	Nodes of a lock
$(n_{\text{from}}, n_{\text{to}})$	Nodes of a lock
$t(o_i)$	Locking time
$\{I, II, ...\}$	CEMT types
$\tilde{C}_k$	Collection CEMT classes
$L_{\tilde{C}_k}$	Set of links containing CEMT classes
$N_{\tilde{C}_k}$	Set of nodes connected to links with CEMT classes
(a, b, c)	Route through nodes a, b and c
R	Route
$\tilde{R}$	Reversed route
V	Vessels
IL	Initial list
FL	Final list
DM	Distance matrix
RM	Route planner matrix
T	Transition matrix
D	Direction matrix
G	Go/stay matrix
$c(t)$	Current node at time t
τ	Time of first transition
$A(V)$	Arrival times of vessels
Un	Uniform distribution
f	Density function
F	Cumulative density function

PROBLEM DESCRIPTION

What are the delays, number of infeasible routes and costs due to a broken lock or vessel accident during a given length of time on the Dutch inland waterways. *If needed only a region of influence will be calculated and there will be a separation of container and bulk. Multiple analyzes are made of different incident with time stretching from the incidents till the situation returns to normal again to determine the robustness of the system.*

SUMMARY

The expectation is that the flow of goods will increase in the upcoming years. To avoid extra traffic on the road and to achieve the environment targets a giant growth is needed of inland navigation. Due to the arrival of the "tweede maasvlakte" the number of transported containers will quadruple between now and 2035. The growth of inland waterway transport can cause an increase in the number of bottlenecks and the possibility of an accident if capacity cannot accommodate the expected growth. The consequences of such an incident (broken down locks and vessels that got stuck) have been unknown until now. This is why a model is built to analyze the congestion on the Dutch waterways in the next 30 years.

DATA Available data dated from 2004 extrapolated to 2008 about the network and ship route intensity is available from the BIVAS model. Different papers based on scenarios help to provide additional information and useful data for extrapolation. On a large scale the BIVAS model and on a local scale the SIVAK model exists. The focus of the new model lies somewhere in between these models. The additional value of this new model is that it is able to analyze a small window of time and the dynamical behavior. This means ships will take into account what other ships are doing. Depending on the interest of the user and calculation speed a set of vessel types can be chosen and simulated without all other types. The scenarios that are evaluated are selected by the user and both container and bulk transportation can be analyzed. The implementation techniques make use of a Markov model where advantages and downsides will be evaluated to see how well this model solves the problem. In order to draw conclusions about the results Monte-Carlo simulations will be used and the results will be visualized using a map of the Netherlands.

SCENARIO Within one scenario the time of delay due to taking an alternative route and the amount of vessels that have nowhere to go will be specified. The duration of the simulation will be from the moment the incident takes place till the time the situation returns to normal again. This normal state will be defined using a simulation without an incident. The model assumes a static initial state which means that seasons, water levels, ice and intensities in high and low seasons are neglected.

1

MAIN OUTLINE

1.1 PARTS THESIS

This thesis is built on three blocks:

- I Motivation and preparation
- II Stochastic congestion model
- III Results from congestion analysis

MOTIVATION AND PREPARATION In the first part the source is mentioned and evaluated together with models that already exist. The added value of the new model in comparison with these existing models will be discussed. A background of the involved companies are given along with their aim for the future. Before one can use the data from the input some changes have to be made in the structure of this data. The final input will consist of the complete network of the Netherlands along with fleet data.

STOCHASTIC CONGESTION MODEL In the second part the model is explained, the complete setup followed by all the different elements in detail. In the beginning of this part the Markov assumption is made, which is the foundation of the model. Together with all the assumptions that are made a balance between simplicity and reality is created. Later on the parts of the network and the way the vessels are modeled are discussed. This is directly followed by the working of the route planner and model of the locks. In the end of this section the attention is focused on how congestion influences the flow of vessels and how an incident can be implemented.

RESULTS FROM CONGESTION ANALYSIS The last part describes the output. Here the focus lies on what kind of output can be obtained and what the results are. In order to provide this information an analysis has to be made with the newly made model. The thesis ends with a conclusion and a Recommendation report to Port of Rotterdam.

1.2 MODEL INTRODUCTION

This thesis is mainly a description of the model but occasionally some references to the actual model are made. The model is built in the Matlab environment and consists of multiple parts and every part consists of multiple files. All the files are graphically represented by grey squares in figure 1 and the different parts are grouped by the rounded squares. For now the focus lies on what the structure is like which means these blocks and all details in this figure can be disregarded for now. What is happening in these boxes will become clear along the way.

STRUCTURE The focus lies mainly on describing what happens in the code of the model. This means no explicit code is given and apart from the end of a chapter no explanation is given and no references are made to these files. There will be parts of figure 1 presented at the end of some chapters to refer to these codes and summarize what is happening. In this way all

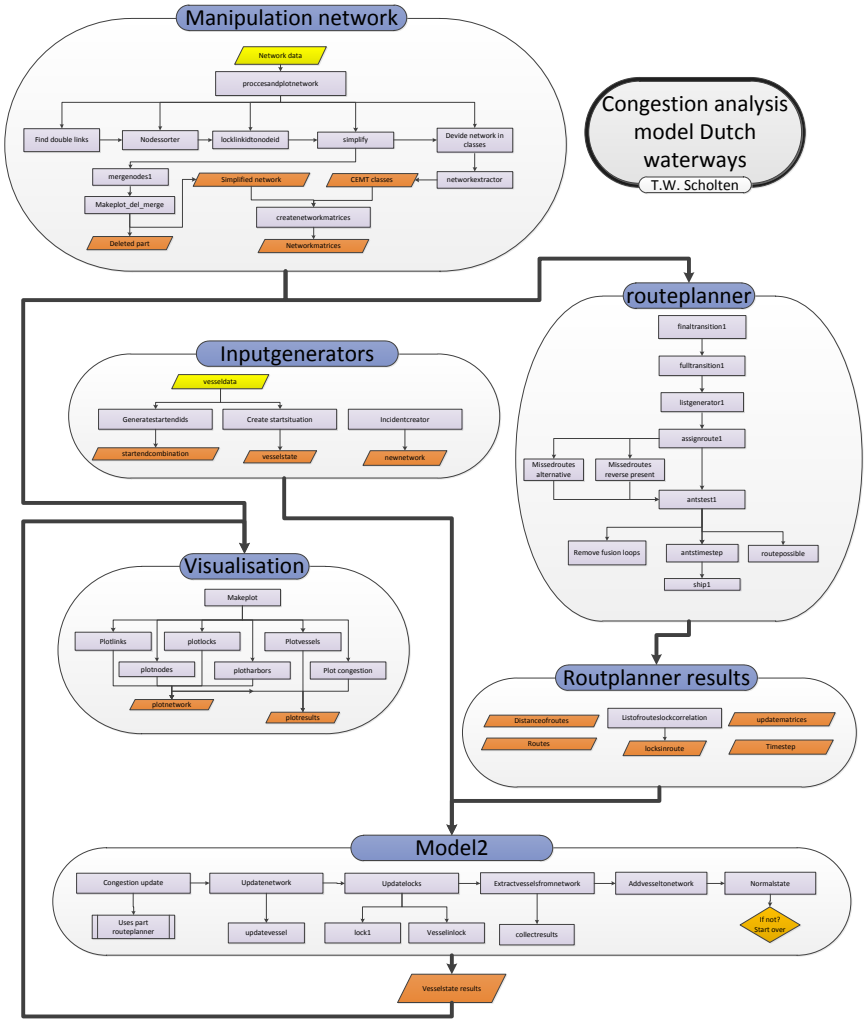


Figure 1.: model flow list

the theory, information and process are known before jumping into the raw model. This section is accompanied by a summary of the blocks contained in the displayed part of that section.

Part I

MOTIVATION AND PREPARATION

2

INVOLVED COMPANIES

There are several companies involved and/or interested in this project. The main initiator for this project was the Port of Rotterdam. They have a big interest in robust connections to the rest of the Netherlands and neighboring countries. The responsible party is Rijkswaterstaat which is a branch of the Dutch government responsible for the Dutch waterways. The third party is Witteveen+Bos, a consultancy firm with roots in all kinds of water related branches. The latter company was the supervisor and supporter in the process of writing the model and this thesis.

2.1 WITTEVEEN+BOS

Witteveen+Bos offers its clients value-added consultancy and top-quality designs for water, infrastructure, spatial development, environment and construction projects. They deliver reliable solutions built on the knowledge, experience, social insight and intellect of their employees. Their department of ports and waterways has the objective to prepare, manage and supervise the construction and development of ports and waterways qualitatively, quickly and efficiently.[10] Witteveen+Bos are specialized in:

- port development: strategic recommendations, transport economics, master planning and financial/economic feasibility
- port infrastructure: quays, jetty structures, land reclamation and other infrastructure for seaports and inland port terminals and marinas
- waterways: nautical consultancy (including analyzing bottlenecks) for port access channels, inland waterways, locks, bridges, stopping points for ships waiting to dock and other infrastructure

The main interest of Witteveen+Bos regarding this thesis is gaining knowledge and affinity with the topic of robustness and congestion management on the Dutch waterways. Since this is quite a hot topic at the moment this could be both commercially interesting as well as an opportunity to attract new parties to work with.

2.2 PORT OF ROTTERDAM

The main reason for this project is the interest from Port of Rotterdam (PoR) due to some incidents on the Dutch waterways. One of these incidents involved an acid tanker that capsized on the Rhine near Lorelei. This tanker transported 2400 tons of sulfuric acid and was traveling from Lundwigshafen in southern Germany to Antwerp, Belgium. [7]

This incident blocked the passage of vessels for a long period. This caused a damage of 50 million euros to the Dutch and German economy. These were the finding of a study of investigation and consultancy bureau NEA [8]

Another example was the recent accident of the lock at Eefde as can be seen in reffig:lockeefde. In the night of the second to the third of January 2012 the door of the lock broke down during the locking process.[11] One of the most important inland harbors in the Netherlands is situated in Hengelo behind this lock. A perambulation within the disadvantaged companies

Figure 2.: Broken lock Eefde

resulted in an estimated damage of multiple millions of euros. [12] A new parallel lock is planned to be ready in 2017 to counter a similar incident in the future. [13] This example shows that an incident first has to occur before action is taken. A model could give insight into damage, costs and could help prevent financial damage on a large scale if the right decisions are made based on the results.

CONSEQUENCES In order to maintain its current position in a highly competitive market the robustness of PoR is of great importance. The most significant part is the continual connection to other harbors and distribution centers in the Netherlands and neighboring countries. There are some possibilities to transport goods by road and train but the largest amount is transported by cargo vessel. There is some flexibility between water, road and train transportation but the maximum capacity of the road and train network is almost reached. Besides that the costs and pollution of transportation over water are lower.

It may seem that costs play the most important role, but in fact the real threat is that PoR's image as a robust port may be affected. Competitors like Antwerp have to deal with the same network, so this means that a robust image is what matters most on the international market.

If image is so important, then the following question arises: what will happen when a serious incident takes place? Since incidents are scarce by nature it is difficult to obtain enough real life experience to learn from. This is why a model can be used to give insight into the possible consequences of such incidents. Another interesting project that is currently in development is the building of 'Maasvlakte 2'. This will allow the next generation sea ships to enter the harbor. A consequence is that transshipment will increase significantly and thus there will be an increase of vessel size and number of vessels sailing through the inland rivers and canals of the Netherlands. This evaluation and prediction of growth is not covered in this thesis but alternative input can be used to evaluate congestion with or without future incidents.

2.3 RIJKSWATERSTAAT

Rijkswaterstaat manages and develops the Dutch network of roads and waterways commissioned by the secretary of State of infrastructure and en-

vironment. It is the executive organization of the ministry of infrastructure and environment. Their focus is to guarantee dry feet, provide enough and clean water, ensure quick and safe road and water transportation and manage trustworthy and useful information. [9]

Since Rijkswaterstaat is responsible for the state of the waterways in the Netherlands the results of a new model may be of interest to them. Results can be used to adjust the policy of Rijkswaterstaat. Stakeholder in a lot of projects of Rijkswaterstaat is the previously mentioned Port of Rotterdam.

DVS Traffic and waterways agency (Dienst Verkeer en Scheepvaart) works within Rijkswaterstaat and focuses on knowledge and expertise in quick and safe traffic by road and water. This results in sustainable and public-oriented network management for now and in the future. DVS is the main party when it comes to data about vessel transportation. This division is responsible for the network and use of data management.

3

MODELS

There are several models on the market that simulate similar problems. To avoid making a model that already exists a brief analysis of the current models and their capability is made. The Models that are discussed have a boundary condition that they cover the Dutch waterways.

3.1 SIVAK

SIVAK stands for "simmulatiepakket voor de verkeersafwikkeling bij kunstwerken" or simulation package for the traffic flow near artifacts. This package is focused on one or more artifacts next to each other. This could for example be a number of locks and/or bridges. Since the scale of this simulation package focuses on such a detailed environment the new model will not overlap with the SIVAK model.

3.2 SIMDAS

SIMDAS is a simulation program that can be used to determine the capacity and safety of a waterway and changes in capacity and safety as a function of the behavior of waterway users, rules and regulations, waterway layouts and composition of traffic. This model only applies to the rivers in the southern half of The Netherlands and is not suited for big scenarios. [6]

3.3 BIVAS

BIVAS stands for "Binnenvaart analyse systeem" which could be translated as analysis of the inland waterway system. The main use is to answer policy questions like:

- Load of waterways and artifacts
- Calculation of management strategies
- Scenarios of maintenance
- Effects of incidents

The last question is of particular interest regarding this new model. The new model should have added value over the BIVAS model in at least one? aspect. Therefore BIVAS needs to be examined to find its strong and weak points.

DATABASE As mentioned in section 2.3 BIVAS uses a database of ship routes of the year 2004 extrapolated to 2008. This database originates from Rijkswaterstaat and is currently the most recent database of the traffic density on the Dutch waterways. This model also makes use of a fairly detailed network of the Dutch waterways. Most links include parameters as width, height, water level, bridges and CEMT-classes. These CEMT classes will be described in section 4.2. Besides the links the model contains locks, dams, seasons, ship types, ship costs, price fluctuations. This means a lot of information is both put into the model and can be extracted after calculation.

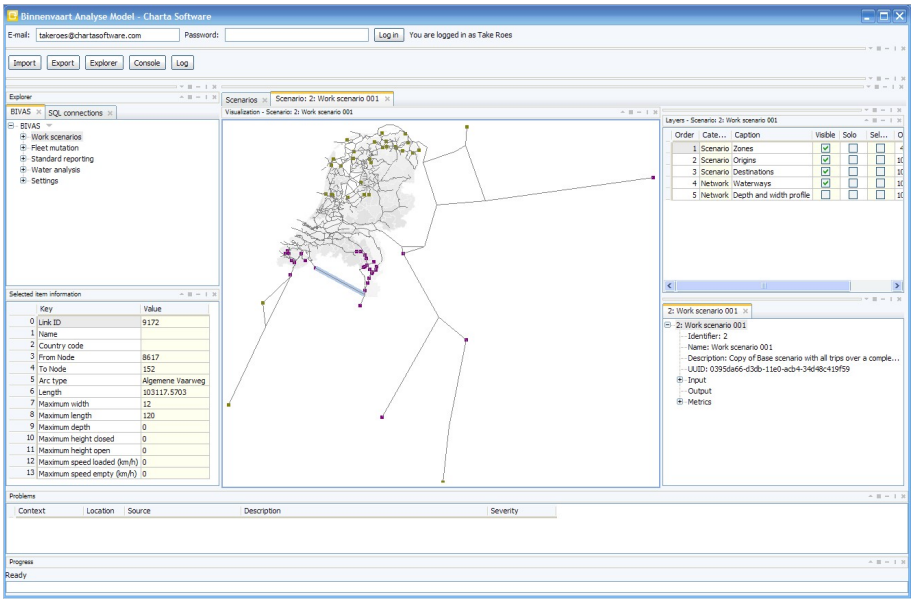


Figure 3.: BIVAS model

This amount of data also has the downside that calculations may take long to process. In order to make this model efficient enough the calculations are kept fairly simple.

7 There is an overview given in appendix A of the data about the network, which will give an idea about the information contained in the input table. also In addition there is a table that shows the parameters in the data of the vessels which can be found in table 8. To obtain the sizes of the vessel a publicly available document from the Ministry of Traffic and Water (Verkeer en Waterstaat) is used.[5]

SCENARIO A scenario is calculated by assigning the shortest or cheapest route to each vessel depending on what is chosen to optimize. A uniform distribution is used to implement some randomness to the system to avoid the case that all vessels go through the same link if two routes are almost equally beneficial. This means that vessels will not take each other into account and might run into each other on bottlenecks. Although it is possible to run the same scenario again with the waiting times of the previous run in mind this is not ideal.

If due to an incident congestion in a corridor arises vessels might take a detour with higher costs to avoid waiting.

When a scenario is calculated in BIVAS there is no choice in length of time. The default time is a whole year which takes four different seasons into account. A scenario of, for example, two weeks is therefore impossible to calculate. Also the region is fixed to the total area of The Netherlands. This is no problem in itself but it could lead to longer computing times compared to a smaller region. Depending on the speed demands, scenario and speed of the model there should be an option to select a smaller simulation region.

This means that a model which takes other ships into account, runs for a chosen length of time and is able to handle a chosen region has a lot of added value compared to the BIVAS model. Both the network and fleet databases are free to use and will be used as a the input for the new model.[3]

4

NETWORK DATA

The basis of the network data consists of nodes and links with additional information.

Definition 1. Nodes n_i consist of a x-coordinate $X(n_i) = x$ and a y-coordinate $Y(n_i) = y$.

Definition 2. Every links l_i has a corresponding combination of nodes $N(l_i) := (n_{\text{from}}, n_{\text{to}})$

To simplify notation there is an equivalent representation for each link l_i.

Definition 3. $l_i \equiv l_{(n_{\text{from}}, n_{\text{to}})}$ where n_{from} and n_{to} are the nodes the link is connected to.

This means that every i represents a unique index of a combination of two nodes.

Definition 4. Links have a distance $|l_i|$

Definition 5. Links have a CEMT type $C(l_i)$

Definition 6. A node can have a harbor on it: n_H which is a starting or endind point of a route

Definition 7. A lock has a corresponding combination of nodes $N(o_i) := (n_{\text{from}}, n_{\text{to}})$

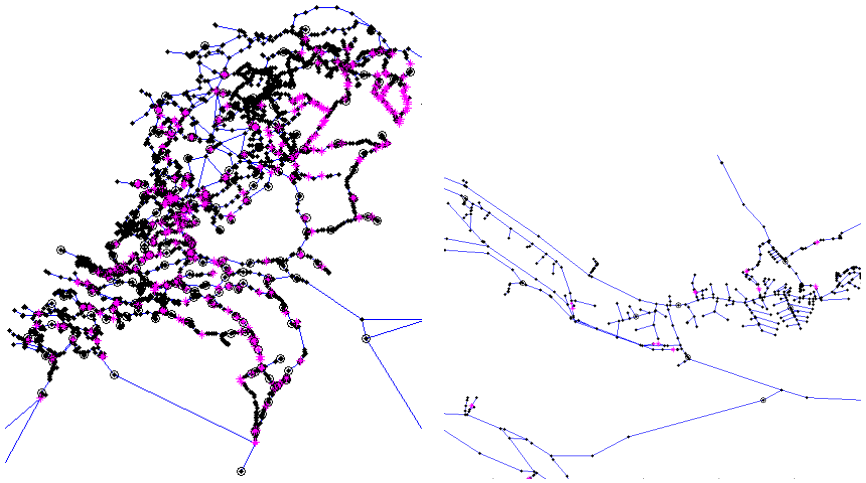
Definition 8. A lock has a locking time $t(o_i)$ which is the time for the lock to complete a full cycle

Additional information is available about the network. Examples of this information are depth of the waterways and maximal passing height. Since the model will not take these variables into account one has to be careful when interpreting the results of the model.

SOURCE Since the BIVAS model has the complete network available, this data can be used. The CEO of Charta Software Karsten Uil allowed the use of this data regarding this model.

4.1 MANIPULATED DATA

In order to be able to calculate results as quick as possible it is convenient to simplify the network as much as possible. At the same time the network has to be representative, which means that it must contain all the information. Before manipulation the network looks like 4. Here the complete network is visualized using a combined scatter plot and line plot. Together with the complete picture the area around the harbor of Rotterdam is illustrated to allow for a better overview of the details of the plot.



(a) full network (b) full network zoom

Figure 4.: full network

Lines	Links
Black dots	Nodes
Purple stars	Locks
Black circles	Harbors

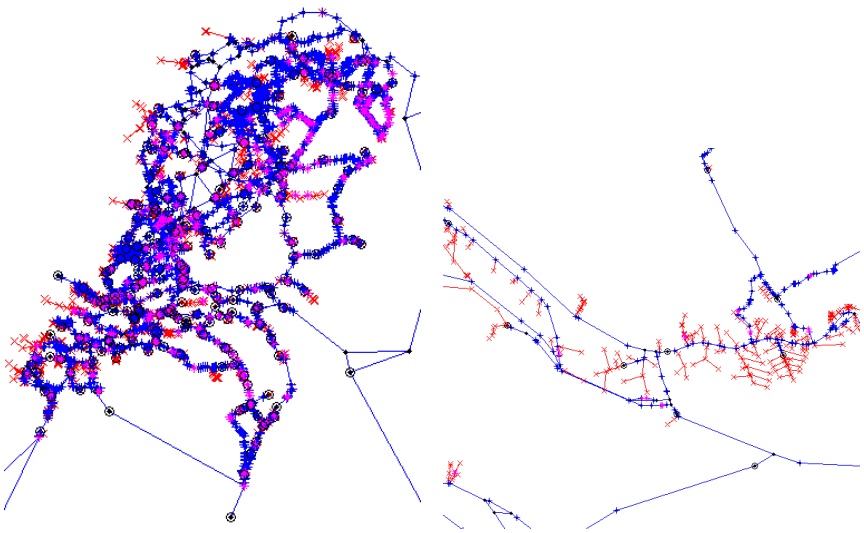
Table 1.: legend of figure 4

When looking at this normal network the first thing that strikes is that there are some blind endings. This is useful as long as there is a harbor at the end of the waterway, but in a lot of cases that is not the case. This means nodes can be deleted to improve calculation speed later on, which can be done in two ways. The first option is the deletion of the links connecting the node. The second option is to merge the connecting links to obtain a new link. Notice that this option can only be done in a node has exactly two connections.

DELETION OF NODES There are a few criteria that have to be met before deleting:

- Nodes with a connection to more than two other nodes need to be maintained.
- Nodes containing a lock ending can never be deleted.
- Nodes containing a harbor can never be deleted.
- Nodes with zero or one connection can be deleted.

Now start by iterating a process of node deletion where the above criteria are met until no node is deleted. The result is a network without any blind endings. At the same time no necessary information is lost. But this is not the only simplification that can be made.



(a) nodes to merge and delete (b) nodes to merge and delete zoom

Figure 5.: nodes to merge and delete

Blue lines	Links
Black dots	Nodes
Purple stars	Locks
Black circles	Harbors
Blue plus signs	Nodes to merge
Red lines	Links to delete
Red crosses	Nodes to delete

Table 2.: legend of figure 5

MERGING OF NODES It is also possible to merge links. For the merging process the following criteria must be met:

- Nodes with two connections can be deleted and corresponding links merged.
- Nodes containing a lock ending can never be merged.
- Nodes containing a harbor can never be merged.

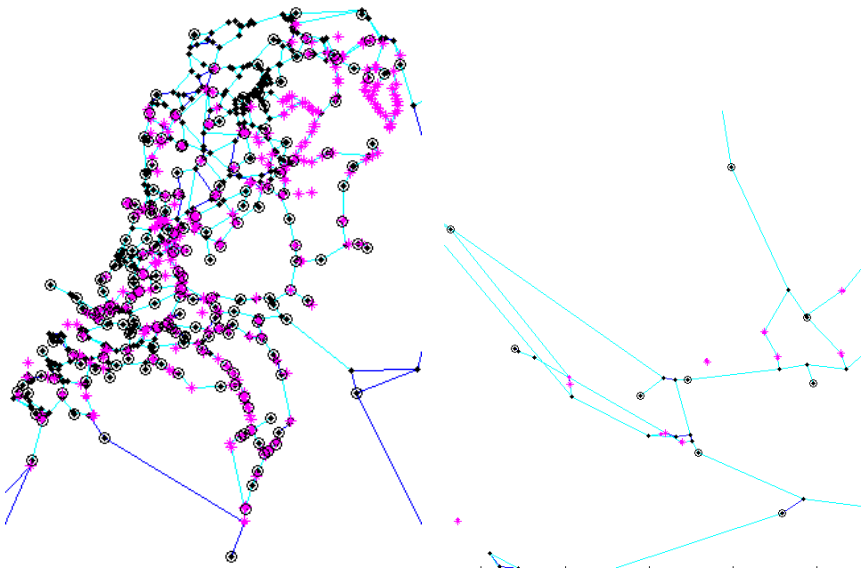
It is now possible to iterate a process of the merging of links and again no necessary information is lost. This way, the network ends up with a total number of 1310 nodes and 1584 links. Since the original network had a total of 8850 nodes and 9420 links these numbers are reduced by respectively 85,20% and 83,19%. The parts that need to be maintained and deleted are shown in figure 5 and the result in given in figure 6.

Since every link has a CEMT type two merged links may have a different CEMT type. When this happens the CEMT type of the resulting merged link should be equal to the worst one of the two. This means

CEMT types

Class	Length	Width	Depth	Height	Cargo
I	38,50	5,05	1,8-2,2	4	250-400
II	50-55	6,6	2,5	4-5	400-650
III	67-80	8,2	2,5	4-5	650-1000
IV	80-85	9,5	2,5	5,25-7	1000-1500
Va	95-110	11,4	2,5-4,5	5,25-7	1500-3000
Vb	172-185	11,4	2,5-4,5	9,1	3200
VIa	95-110	22,8	2,5-4,5	7-9,1	3200-6000
VIb	185-195	22,8	2,5-4,5	7-9,1	6400-12000
VIc	193-200	34,2	2,5-4,5	9,1	9600-18000
VIIb	195/285	34,2	2,5-4,5	9,1	14500-27000

Table 4.: Classification CEMT type maximum conditions in meters



(a) simplified (b) simplified zoom

Figure 6.: simplified

Blue lines	Original links
Black dots	Nodes
Purple stars	Locks
Black circles	Harbors
Light blue lines	Links that are merged

Table 3.: legend of figure 6

4.2 CEMT-TYPES

Now the network is simplified but not yet ready for use and there are still two steps to take. Vessels come in different sizes and not every vessel is allowed on every waterway simply because it does not always fit. This leads to a different network for each vessel. This difference in network can be classified using CEMT types, which stands for 'Confrence Europenne des Ministres de Transport'. As mentioned in the introduction of chapter 4 a CEMT class l_{ic} exists for every link l_i.

$$C(l_i) \in \{I, II, III, IV, Va, Vb, VIa, VIb, VIc, VIIb\}$$

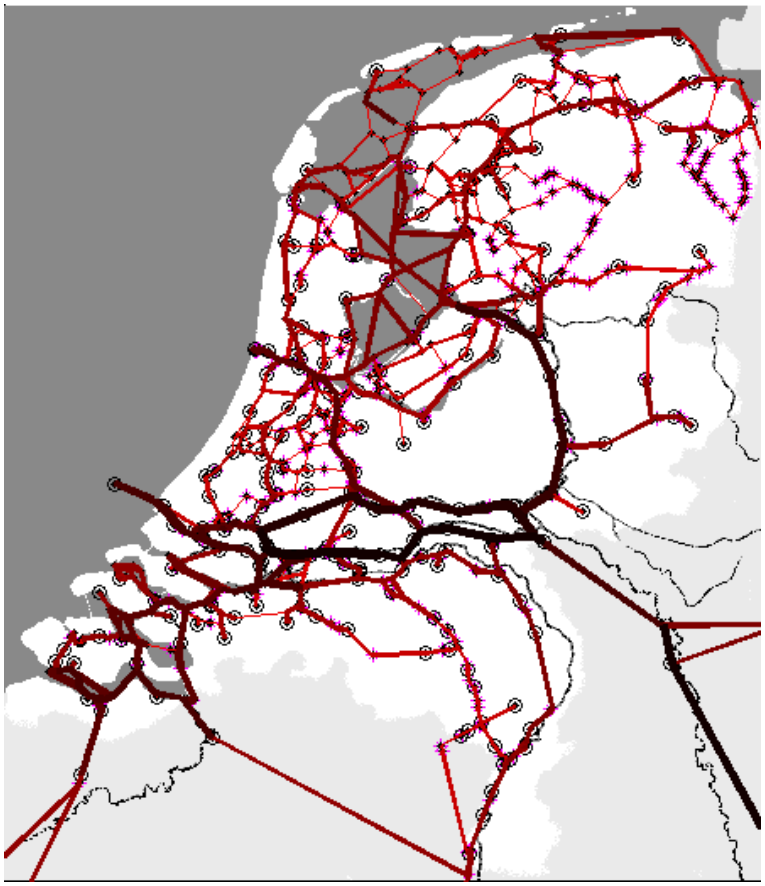


Figure 7.: simplified with background

Since small vessels are able to sail on big waterways but not the other way around the following holds:

$$\bigcup_{\{i|C(l_i)\in\{I\}\}} l_i \subset \bigcup_{\{i|C(l_i)\in\{I,II\}\}} l_i \subset \ldots \subset \bigcup_{\{i|C(l_i)\in\{\ldots,VIIa\}\}} l_i \subset \bigcup_{\{i|C(l_i)\in\{\ldots,VIIa,VIIb\}\}} l_i$$

By plotting these sets in a single figure where increase of CEMT type is implemented by an increasing size of the links figure 7 is obtained. In this figure the contour of the Netherlands is also made visible. To make writing a bit more convenient the following definition is used:

Definition 9. A CEMT class C_k is defined as the kth element from $C(l_i) \in \{I, II, III, IV, Va, Vb, VIa, VIb, VIc, VIIb\}$

Definition 10. The collection CEMT classes $\tilde{C}_k$ is defined as: $\tilde{C}_k := \bigcup_{i=1}^{k} C_k$

4.3 MATRIX

Now there is a set of links for each class $\tilde{C}_k$ these links are denoted by:

Definition 11. $L_{\tilde{C}_k} := \{l_i | C(l_i) \in \tilde{C}_k\}$.

These sets of links have a set of nodes and are defined by:

Definition 12. $N_{\tilde{C}_k} := \{N(l_i) | l_i \in L_{\tilde{C}_k}\}$

Now that the set of nodes and the set of vertices of each CEMT class have been defined it is possible to create a distance matrix for each CEMT class. The matrix is defined as follows:

$$D^k_{i,j} = \begin{cases} 0 & \text{if link from i to j not present} \\ |l_{(i,j)}| & \text{if link from i to j present} \end{cases}$$

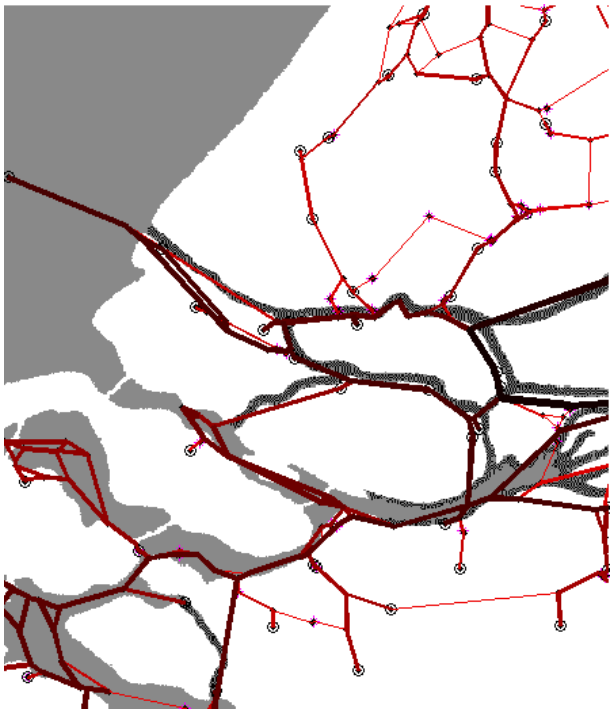


Figure 8.: simplified with background zoomed

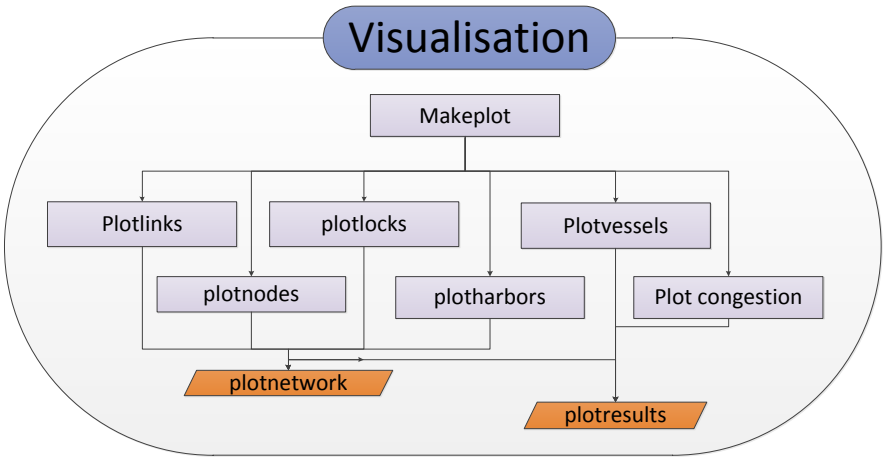


Figure 9.: visualization

Where k is the transition matrix corresponding to the set of $\tilde{C}_k$ and presence is checked in this set.

After deriving the transition matrix for each $\tilde{C}_k$, all the elements are in place for defining the Markov Model.

5

FLEET DATA

5.1 CURRENT FLEET

There is no database available of the current state of the fleet. The most relevant and available data set is one created by Rijkswaterstaat in 2004. This data was created by counting vessels during a big survey of a whole year. As part of this survey origins, destinations and vessel types were stored. In 2008 another counting was done but this time only at locks and bridges. By taking the figures from the data in 2004 and tweaking this they were able to estimate the data of all commercial vessel movements in 2008. Unfortunately there is no data available from more recent years. This is due to both a reorganization in Rijkswaterstaat and some technical problems which caused data loss in 2010. The data from 2004/2008 is used and available in the BIVAS models which can be found in 3.3.

There are some incomplete plans to provide new data but no concrete actions have been taken so far. The data that would be presented by a new survey could easily be implemented by just replacing the current data with the new one.

VESSEL TRACKING Another interesting possibility is to track vessels with the beacons they have on board. This data is publicly available on the website of 'marine traffic'.[15] Storing this data could give a good indication of vessel movements in a certain period of time and could be used to update the current database. Caution is needed here because it is not compulsory to have such a system aboard (at least, for now). This has to do with the privacy of the sailors. Consequence is that this method would not give a complete picture of the situation. What is about to happen in the future is also of great interest.

5.2 FLEET EXPANSION

Due to the creation of Maasvlakte 2, a big extension of the harbor in Rotterdam, the cargo and raw material transport will increase drastically over the next 30 years. This will have both consequences for the total transported quantity as as well as for the composition of the vessel fleet in the Netherlands. If scale increment takes place, this could lead to a completely different situation on the Dutch waterways. Increasing sizes of vessels will give them fewer possibilities of making detours making the network less robust. There are several studies estimating the different possibilities of growth. Four different scenarios are used to make estimations: Global economy, high oil price, European trend and low growth.[14]

GROWTH IMPLEMENTATION During this research the choice is made to set aside this problem to focus on the model itself. If a user of the model would like to evaluate what the results of such a growth could be in the future the data of the vessels can simply be altered according to the expected growth. This would involve generating a matrix which contains all the vessel origins, destinations and types that are expected to sail in the future.

Part II

STOCHASTIC CONGESTION MODEL

6

OUTLINE MODEL

GOAL MODEL The aim of is to build a model that is able to describe the vessel flow through the Dutch waterways while an incident takes place. The model must cover the complete network of the Netherlands and must have an extra edge compared to the existing models. The main issue that is not covered with the existing models is that none of them analyze the system in a dynamical way. Since incidents can lead to a significant increase of vessel movements on a certain corridor or waterway this might may result in congestion in such a region. Detours might may be more appealing at this time and the aim is to incorporate this into the model.

INPUT AND OUTPUT In part I the data of all the Dutch waterways and start-end-combinations of vessels during over a period of onea year was obtained. This data is used as the input of for? the model. The aim will be to obtain the extra travel and waiting time of vessels during an incident compared to the normal situation. It must be possible to visualize this data in a map. The map used for this is figure 7

6.1 SETUP

Where normal one would use a shortest path algorithm like Dijkstra's algorithm [1] would be used, but in this case a different approach is taken. The basic idea is to base it the model on a discrete time Markov chain along with some adjustments to stay close to reality.

The model will act on the level of vessels. This means each individual vessel is calculated and tracked. These vessels must be able to detect each other and adjust their route if a route gets to too busy. This congestion is assumed to form in front of the locks. A detailed description of all of these statements is worked out in this section. In section 13.4 the advantages and disadvantages will be discussed along with all the problems that need to be tackled to make this approach work.

In figure 10 the whole outline is given, which can also be found in the introduction. Each rounded rectangle represents a different part of the model, while the rectangles within them are different files containing the whole code that is used. The yellow parallelograms represent the input data and the orange parallelograms represent modified input or output that is used within the program. Since the illustrations of the individual rectangles are quite small all the individual elements are also presented at the end of their corresponding chapters.

6.2 MARKOV MODEL

This section can be seen as the engine of the model. Here the Markov chain is introduced on which the whole process will be based. This choice has a lot of consequences and some difficulties will arise later on, but it will also lead to a new way of modeling individual components in a network.

Let us start by defining time t:

Definition 13. Let t be the time with $t \in [0, T]$ with $T \in \mathbb{N}$

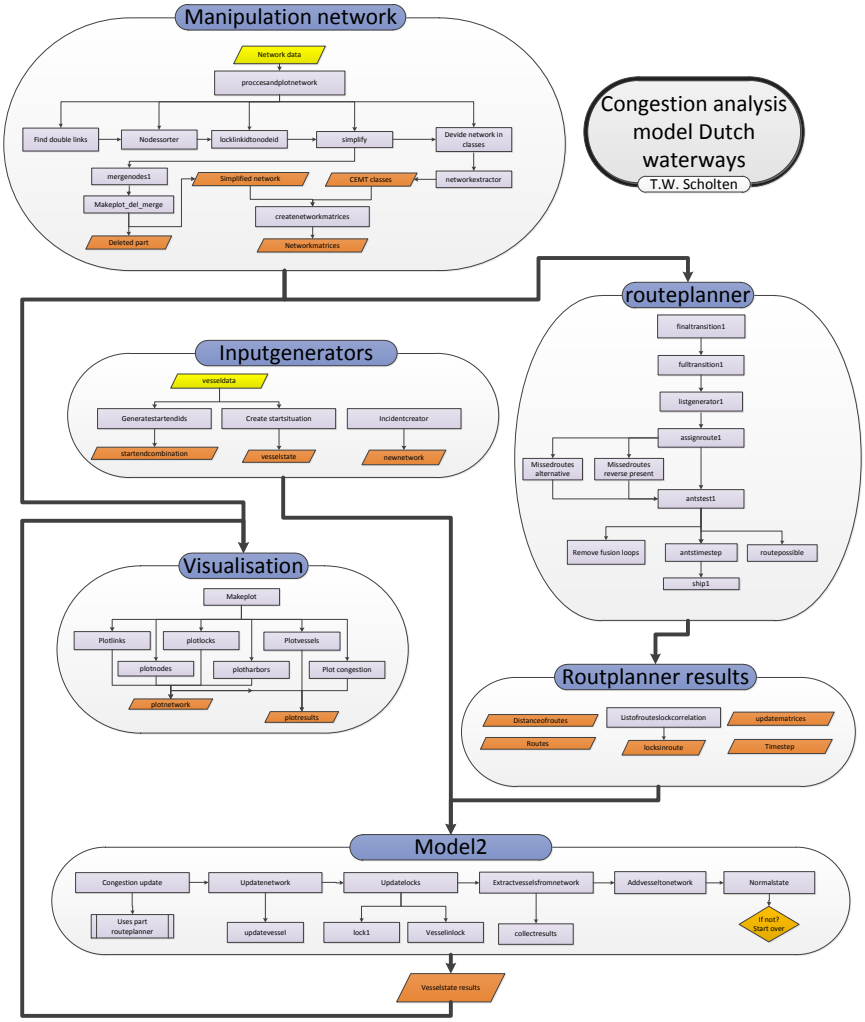


Figure 10.: model flow list

The next step is defining what a state is:

Definition 14. $S_t = (X_{t,i_1}, ..., X_{t,i_{N_t}}, O_{t,1}, ..., O_{t,O})$

Were:
- N_t are the number of vessels in the system at time t
- X_{t,i_j} vessel state of vessel i_j at time t
- $O_{t,j}$ is the state of lock j at time t

Here $X^t_{i_j}$ consists of:

$X^t_{i_j}$

Variable	Description
Type vessel	The type ID of a vessel
Location	The ID of the node where the vessel is now
Departure time	The number of time steps that passed before departure
Arrival time at current node	The number of time steps that passed before Arriving at current node
Empty/full	0 if empty 1 if full of cargo
Startid	Node ID of departure
Destinationid	Node ID of destination
Curlength	Total traveled distance up until current time
Fromtoid	Corresponding ID matching the matrix of the Start-end-combination
Leftlock	1 if it just left a lock 0 if it never entered a lock or moved at least one node

And O^t_i consists of:

O^t_i

variable	description
Locking start time	The number of time steps that passed until the start of last locking
Locking end time	The number of time steps that passed until the end of last locking
Vessels in lock	The IDs of Vessels currently in the lock
State	'up', 'down' or 'neutral'. The latter state is used after a locking time with no vessel entries

MARKOV ASSUMPTION Now the basic definitions are known it is time to define the Markov assumption:

$$P(S_t = s_t | s < t; S_s = s_s) \tag{6.2.1}$$
$$= P(S_t = s_t | S_{t-1} = s_{t-1}) \tag{6.2.2}$$
$$= \prod_{j=1}^{N_t} P(X_{t,i_j} = x_{t,i_j} | S_{t-1} = s_{t-1}) \prod_{k=1}^{L} P(O_{t,k} = o_{t,k} | S_{t-1} = s_{t-1}) \tag{6.2.3}$$

In equation 6.2.1 the probability of reaching a state given the previous states is given in continuous time. Since the model is based on discrete time this equation reduces to equation 6.2.2. Combining the last equation with definition 14, equation 6.2.3 is obtained.

CONDITIONING As can be seen the probabilities of transition are dependent on the previous time steps. Since this model deals with discrete time steps, only the previous state is relevant because only first order transitions are taken into account. When evaluating the state, the information about

the vessels and the locks is known. Here the number of locks stays fixed as well as their positions. The state of the vessels, however, is a different story, since they are allowed to vanish and spawn from the system. This results in a dynamic number of vessels and positions at every time step. The whole state space is too big to calculate, but states between time steps can be evaluated relatively easily.

TRANSITION MATRIX Since all the information at time t of $S_{t-1} = s_{t-1}$ is known $S_t = s_t$ can be calculated deterministically. Now a transition from a state at time $t-1$ S_{t-1} to a state at time t S_t needs to be described. This is done by a combination of a stochastic and deterministic process. To obtain this combination the equation 6.2.3 is split up into two parts. The first part consists of the vessels:

$$\prod_{j=1}^{N_t} P(X_{t,i_j} = x_{t,i_j} | S_{t-1} = s_{t-1})$$

Transition probabilities are assigned to these vessels and random numbers are drawn to see in which state they end up the next time step. Along with this it is possible to add and subtract some vessels in the system. How these vessels are updated will be explained in great detail in chapter 9. Locks, however, are a different story.

$$\prod_{k=1}^{L} P(O_{t,k} = o_{t,k} | S_{t-1} = s_{t-1})$$

These locks are updated by means of a deterministic process but at the same time their input is stochastic. Because a vessel arrival is a random event its waiting queue cannot be anticipated. Therefore, starting times of a locking happen stochastically while the process that follows is fully deterministic. This arrival and entering procedure is explained in 10. How a lock is updated can be found in chapter 11.

STATIC INFORMATION Besides this state some fixed information is fed into the system. This data consists of the network N, the harbors N_H, the time step Δt and a list of possible routes. How the last two mentioned are generated will be explained in detail in chapter 9. The data from the network N and harbors N_H was already mentioned in 4. Besides the state with all variables and this static information there is also a dynamic part.

DYNAMIC INFORMATION The transition probabilities of the vessels do not need to be fixed, which gives some control over the update process. Since vessels need to reach their destination preferably along a route without congestion these numbers can be chosen such that this condition is met. The dynamic property of these probabilities can be used to update the system for congestion. In chapter 11 it is explained how this congestions influences these numbers.

6.3 ASSUMPTIONS

In order to make a new model assumptions have to be made. This is one of the most important chapters of this thesis because it will define a large part of the model and can therefore be read as a summary of the model. These assumptions originate from different aspects of which given time to implement, calculation time, simplifications and necessary elements are the most important ones. All the different elements of the model have to represent reality as closely as possible while keeping in mind that they have to be easy to implement and calculate. Some of the following assumptions may not yet be explained or defined. A detailed explanation will follow in the upcoming chapters while this section is only an overview.

STRUCTURE The model has discrete time steps. This makes computations fairly easy but it is a downside when expected passing times become shorter than a time step. This means there is a maximum time step length which can make computations over a long period of time quite expensive. As mentioned before the model makes use of a Markov model to model the transitions of the vessels.

VESSEL MODELING The model is chosen to behave on the level of vessels. This gives the possibility to give information and data of each individual vessel and makes it easier to obtain detailed results from a simulation. The downside is a longer running time. Every vessel will have a start and end point and besides that a type (CEMT-category) is assigned depending on the size of the vessel. There will also be a label assigned to a vessel saying whether it transports bulk or containers.

Results must contain the differences in sail times compared to the normal situation, the number of vessels that are negatively influenced by the incident, the number of vessels that have no alternative route and the costs due to an incident.

LOCKS Locks have a variable up, down and neutral. The transition between up and down takes place in a deterministic manner and has a fixed locking time. When no vessels arrive for a period longer or equal to the locking time the lock enters a neutral state where it immediately turns into an up/down state as soon as a vessel arrives at the up/down side of the lock. In this way the lock is 'smart' enough to know from which direction the first vessel is coming.

ROUTE PLANNER Assigning routes does not happen deterministically but they are selected from a list of shortest routes taken by multiple 'stupid vessels' . How this works is explained in chapter 9). These shortest paths are not only subject to distance but also to congestion. The congestion occurs only at locks since these parts are the bottleneck of the system.

DYNAMICS AND CONGESTION Vessels have to adjust dynamically to each other. This means they have to 'see' each other and react to (upcoming) congestion and take a detour if needed. It was already mentioned that congestion occurs at locks, which means an incident free link has infinite capacity. Congestion is measured by the number and type of vessels in front of a lock at every time step. As a consequence, vessels only have to take into account the distance together with this congestion, which makes computations relatively easy.

DATA The data that is used is the data that is available from the BIVAS model based on the 2004 data from Rijkswaterstaat. This data is extrapolated to 2008 using counting from different artifacts.

WATER LEVELS The seasonal change is of no influence. Given the available time, complexity, relevance and expectation of Port of Rotterdam this elements is discarded from the model. Various other models and studies have been done [2] with respect to climate change, water levels and seasonal change. Although not meant for this use, a river or canal with critically low water levels can be evaluated by marking it as an incident.

BRIDGES This being said, there are two types of bridges, the ones that can open and the ones that have a fixed height. Both of these bridges are not included in the model but could both have a contribution to the outcome of the model. This contribution can be split into two different categories: congestion near bridges and pass restrictions. The assumption is made that this congestion can be neglected compared to the locks. Since this aspect is

not researched in this thesis the user of the model should keep this in mind while reviewing the results.

The passing restrictions can at their turn be split into two different bridges: Fixed ones and bridges that can open. The fixed bridges give a restriction to height on individual links. Together with seasonal and short period change in water levels this might restrict some vessels to not pass at all or increase or decrease their load. This will result in more vessel movements at certain connections. A bridge that is able to open and close could also have some effects. During rush hours of road traffic they might stay closed for longer periods of time. Another consequence may occur when the bridge does not operate at night and is thus blocking a link for a duration of time.

INCIDENTS The model must be able to handle one or more incidents on a user specified location for a length of time that is also specified by the user. Keep in mind that the fact that bridges are not implemented in the model does not mean they cannot be the cause of an incident. An incident is defined as a deletion of a link but could be extended to other forms such as one way traffic at the same time or closed for some CEMT types. When an incident occurs not all vessels will respond immediately. This delay is, unfortunately, not implemented in the model. This is because it is quite hard to estimate the expected delayed response time.

DEPARTURE BEFORE ARRIVAL There are some assumptions that do not match with reality but are left out for simplicity. One is that it may happen that a vessel is going back and forth between two harbors. When an incident takes place in between and no detour is possible the first upcoming departure of that vessel will be a problem since the vessel was never able to arrive. In the model there is only a list of departure times and places and it does not understand which vessels are actually one and the same. The result is that multiple vessels enter the system while in fact only one is waiting with a delayed schedule.

DELAYED DEPARTURE Another problem may be that vessels will not leave or have a delayed departure because they know a certain incident has taken place ahead of them. In the model every vessel will depart if scheduled. This is not necessarily a bad thing. The information obtained from the leaving vessels, unreachable routes and extra sailed time is actually really valuable since these numbers are a representation of the extra costs.

7

VESSELS

The model is chosen to behave on the level of vessels. The other option is to model on the level of nodes. The advantage is that it is possible to give information and data to each individual vessel. This makes it easier to obtain results from a simulation. The downside is that it has running time increases faster when a network becomes bigger or the number of vessels is increased. Every vessel will have a start and an end point. Besides that a type (CEMT-category) is assigned depending on the size of the vessel and it will have a label indicating whether it is transporting bulk or containers. This variable does not play a role in the modeling but is only taken into account in the output. This is because a late container vessel costs a lot more than a similar bulk vessel with the same delay.

The Parameters of the vessels are as follows:

- type
- origin-destination
- amount of cargo/depth ship
- location
- transporting bulk or containers

7.1 TYPES

Since there are several different types of vessels they have to be updated in different manners. Each vessel has an available network a size and a speed assigned to it. This means the same type of vessels can be grouped and updated at the same time. All these different groups can be updated in parallel where no interaction takes place. There is only one exception and that is a lock. How these are updated can be found in chapter 10. For now the aim is to update a single vessel on it's own network without locks.

Each type has the following aspects

- CEMT type
- max width
- max length
- speed
- delay costs
- costs per km

The transition process is based on probabilities Π explained in chapter 9.

7.2 HARBOR

In real life vessels sail from harbor to harbor where they dock, load and unload goods. Vessels have a schedule to maintain and have departure times corresponding to this schedule. In the model a harbor is built as an entry and exit point for vessels in the system. Their departure time is used as the departure time in the model but their scheduled arrival time is not used. This is because the answer that is wanted is a delay compared to an incident free situation. Such a situation could be simulated where arrival times can be extracted. When a simulation of an incident is done a comparison can be made. The same inflow is used but by using a network with and without an incident a comparison of arrival times can now be made.

INFLOW When it is time for a vessel to leave the harbor it is added at the node where the harbor is present. To model this inflow the following information is needed:

- Departure time of the vessel
- Node id of the harbor where the vessel departs

Since the system runs in discrete time steps the departure time is rounded up to the first upcoming time step from the moment it would depart. Now a vessel is added to the system and the total state of the system is updated in accordance with the new situation.

This means that apart from the vessels present in the start situation vessels can only enter the system through a harbor. It could also be that vessels arrive not through a harbor but from the edge of the modeling space. This can still be solved by creating an artificial harbor with the same properties as a normal harbor. The delayed arrival times can also be stored here to evaluate in the output.

OUTFLOW As soon as a vessel hits the node of its destination it is extracted from the system at once. This means it enters the harbor at once and does not interact with anything in the system up to that time. As soon as a vessel arrives all the data of the arrived vessel is stored in a matrix that can be evaluated after simulating.

Note that the following issues are not included in model:

- If a harbor is too busy it is not possible to find a docking spot at once. This means that when a harbor contains too many 'recently entered' vessels it must add some extra time to the current sail time.
- In the first instance the rest places are neglected which means infinite queue capacity in front of locks. There is a possibility that they are added later on in the model.

8

PARTS

Now the model setup is defined along with the state and it is clear how the vessels are implemented. The next step is to describe the different parts of the network. These parts consist of links, junctions and locks. Since locks form a key piece of the model they are neglected for now but will be described extensively in chapter 11.2.

8.1 LINKS

As can be seen in chapter 4 the links that are present between two nodes have a length, maximal speed limit, distance $|l_i|$ and a CEMT type $C(l_i)$. In the Markov model the links are just connections between different nodes where the vessels are present. That means that a vessel is never actually on the link. But the restriction that these links imply must be met. In chapter 4 the different networks for CEMT-types are already mentioned but the length is not covered. Since a vessel is not present on the link it is waiting on a node instead and at a certain point just disappears at the last known location and emerges at the other node. The update process of this change needs to mimic reality as close as possible. How this is done is explained in section 9.6. Links can also be the location at which an incident takes place. This process is described in chapter 12. Note that depth is not mentioned while this can be important. This part is neglected for simplicity's sake but could be implemented in a future extension, which is mentioned in chapter 14.

8.2 JUNCTIONS

Since there are nodes in the network none, one, two, three or more links can be connected. Since nodes without a connection have no added value to the network they are neglected. Nodes with one connection are always connected to a harbor because of the simplification that is used described in chapter 4. The same simplification also leads to the fact that nodes with exactly $n = 2$ links are always connected to a harbor or lock. Proper junctions always consist of a node with $n > 2$ connected links where the most common one is the $n = 3$. At all of these nodes a direction has to be chosen for the vessel. Each time step an update is performed to see where the vessel is heading or if it is staying at the current node. This is done by a random number generator together with probabilities of transferring to a certain node or staying.

9

ROUTE PLANNER

To guide vessels to their destination a route has to be assigned. A problem with this route is that it has to be able to adjust to congestion. Therefore, a shortest route algorithm is not sufficient because every time step this algorithm should compute this route for all start-end IDs times of all the vessel types. A different approach is needed. The main idea will be that scaling should be possible when it comes to transition probabilities. This means that if a link is present between two nodes the transition probability can never be zero. At the same time the probability of taking the shortest route must be sufficiently large.

Before a route is planned all the vessels are grouped into classes of equivalence. These classes consist of the influence of the vessel CEMT type on the available links in the network that the vessel can use. Therefore, a different network is assigned to every class where the network consists of all possible routes that particular vessel is able to sail through. This network is a subset of the original network and inherits all other data like distances and lock locations. Now a route can be generated for every equivalence class.

9.1 ROUTES

Before describing how the route planner works a few definitions are necessary. Some definitions made in chapter 4 are used again.

DEFINITIONS First a route has to be defined.

Definition 15. A route R_x from a to b to c is defined as $R_x := (a, b, c)$

Definition 16. A route R_x starting at a and ending at b with $n \geq 0$ other nodes in between is defined as $R_x := (a, *, c)$

Definition 17. A route R_x starting at a, ending at b with at least one direct connection between i and j is defined as $R_x := (a, *, i, j, *, b)$

Routes can also be contained inside a bigger router.

Definition 18. If $\exists R_s, R_e$ s.t. $R_x = (R_s, R_y, R_e)) \Rightarrow R_y \subset R_x$

So if R_y is contained route R_x in exactly the same order and size then R_y is said to be a subset of R_x. The contrary can now also be defined.

Definition 19. If $\nexists R_s, R_e$ s.t. $R_x = (R_s, R_y, R_e)) \Rightarrow R_y \not\subset R_x$

If a route R_y is not contained route R_x (i.e. $R_x = (R_s, R_y, R_e)$) then R_y is said not to be a subset of R_x with notation $R_y \not\subset R_x$

And the last definition is about a collection of routes.

Definition 20. Routes R_x and R_y are contained in the collection of routes R_C if $R_x \in R_C$ and $R_y \in R_C$

Definition 21. $t(R_k)$ is the time it takes to sail through route k.

Definition 22. The route with the smallest passing time R_i is defined as $E(t(R_i)) <= E(t(R_j))$ for $\forall j \neq i$

9.2 ANTS

To come up with a matrix that guides the vessels along their optimal route some inspiration came from how ants operate. Individual ants do not know where they are going exactly, but as a whole they are quite effective. This kind of random walk of the individual ants will be the basis of a generator of routes.

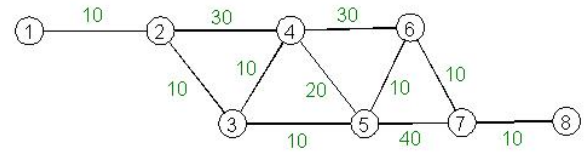
RANDOM ROUTES At this stage there is an origin, a destination, and aa available network with known lengths on each link are available. To obtain the 'virtual length' of a lock the vessel speed is divided by the expected locking time without congestion. Now it 'is time to let vessels enter the network starting from the origin. These vessels will leave with no knowledge of the network and therefore will 'sail' a random route.

To implement this there is a probability assigned to each possible transition.

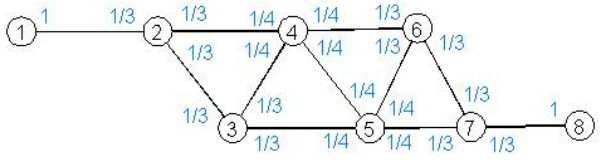
$$P(x^t = n_j | x^{t-1} = n_i) = \left(\sum_{k=1}^{\gamma} \mathbb{1} \left(|l_{(i,k)}| > 0 \right) \right)^{-1}$$

Where γ is the number of nodes.

Notice that these vessels transcend every step to a different node since the transition probabilities of all outgoing links sum up to one by construction since $\sum_{j=1}^{\gamma} \left[P(x^t = n_j | x^{t-1} = n_i) \right] = c\frac{1}{c} = 1$. Here c are the number of connections at node i. An example network with corresponding probabilities is shown in figure 11. This example network will suffice as an explanatory tool for the rest of this chapter. Now the transition rates are known, the vessels can be updated each step, but there is still one more issue that needs to be tackled.



(a) Example network



(b) Transition probabilities

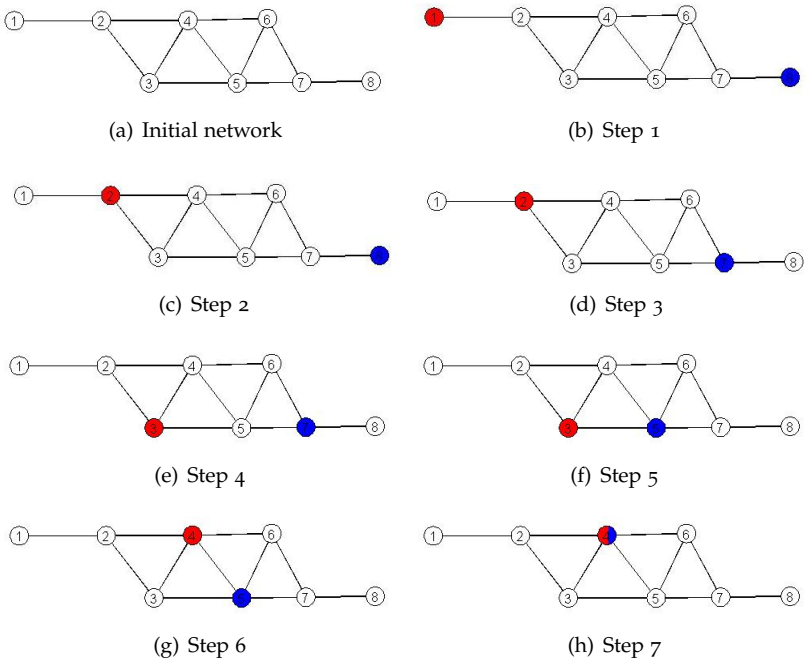
Figure 11.: Updating ants

GHOST VESSELS The trick is not only to let vessels depart from the origin but also from the destination. These vessels can 'bump into each other' by accident creating a route connecting their paths. This means that as the vessels move from the origin at random the same thing happens at the destination. Since these vessels are not actually sailing they are from now on referred to as 'ghost vessels'.

This means there are two sets of vessels, the ones that started at the origin: V_o and the ones that left at the destination V_d. In practice these sets will contain $n > 0$ vessels each but for simplicity's sake it will be assumed for now that these sets each contain only one vessel.

Now update V_o and V_d alternately. An example of how this works is shown in figure 12. Continue doing this until a vessel in V_o hits the

same node as V_d. When this happens a vessel from V_o has traveled along route $r_o = (o, *, i)$ and another vessel that departed from V_d traveled along route $r_d = (d, *, i)$. This combination gives a feasible route from origin o to destination d when this $(r_o, \tilde{r}_d) = (o, *, i, *, d)$ route is taken. $\tilde{r}_d$ is the reversed route of r_d. Note that alternate updating is necessary since simultaneous updating could lead to missing a route that otherwise would have been created.



(a) Initial network (b) Step 1

(c) Step 2 (d) Step 3

(e) Step 4 (f) Step 5

(g) Step 6 (h) Step 7

Figure 12.: Ants updating part 1

Now this route is found store it along with its total distance traveled and continue. It could happen that, for example, two vessels of both sets V_o and V_d arrive in an update step at the same node i. In this case every combination of vessels in V_o and V_d is stored.

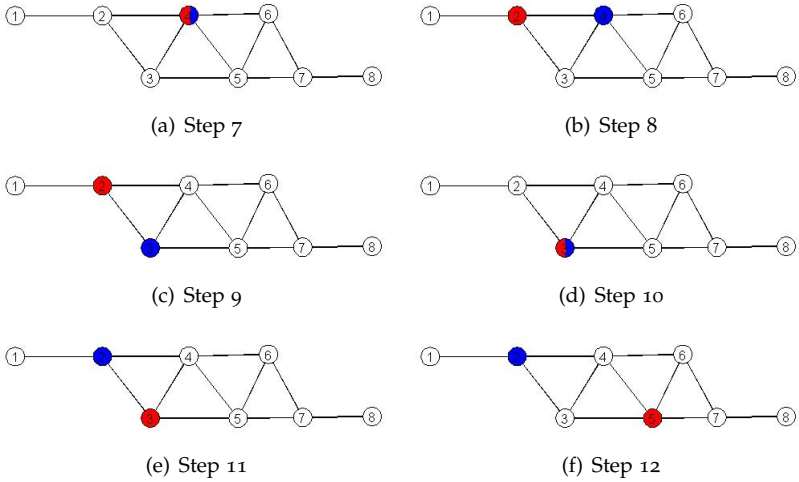
CLEANING ROUTES A downside of this method is that a lot of loops and detours are incorporated in the resulting routes. The latter is not a problem and actually something that is wanted. The reason for this will become clear in the upcoming part of this chapter. Loops, on the other hand, are preferably avoided. In order to assure this, three solutions are implemented where the first two are actually one and the same. The first one is that returning vessels are reset. That means, the route they traveled is just deleted from their memory and they continue as usual. This does not mean that all previously found routes by this vessel are deleted. The second idea to counteract loops is simply deleting them as soon as they are created.

In figure 13 the manufacturing of the routes is continued where figure 12 stopped. In sub figure the red vessel traveled through $(1, 2, 3, 4)$ with corresponding traveled distance $10 + 10 + 10 + 30 = 60$ and has to update. As can be seen in sub figure this vessels travels to node number 2. At this point it finds itself making a loop. This loop is deleted instantly and therefore route $(1, 2, 3, 4, 2)$ turns into $(1, 2)$ with traveled distance 10.

The routes generated by the example of figures 12 and 13 are shown in table 5.

Now the procedure is known about how routes are created and obtained some input is needed. This input consists of the following components:

- network
- number of vessels



(a) Step 7 (b) Step 8

(c) Step 9 (d) Step 10

(e) Step 11 (f) Step 12

Figure 13.: Ants updating part 2

Route	Distance
$(1,2,3,4,5,7,8)$	100
$(1,2,3,4,5,7,8)$	100

Table 5.: Generated routes

- time steps

This input needs to be generated automatically to let the program run autonomously. The network is the easiest one since it is already available. The number of vessels is set to be a constant. The number of vessels is set to be a constant. The number chosen here is 20. Both a number that is too high and one that is too low ends up with a longer calculation time. On the other end a number which is too low influences the quality of the outcome since significantly fewer routes are generated. The number of time steps the model makes depends on the time t_0 of first interaction. This time is measured and a linear combination is used to calculate the additional number of time steps.The initial version of the model uses $20 + 3t_0$ but different combinations or even different formulas could be used to tweak the program to obtain better or faster results.

PROCESS LIST OF ROUTES Now a list of routes has been generated that start at point a and end at point b but our goal is to obtain a transition matrix in which at every node in the network information is needed about the shortest route to its destination harbor $n_d \in N_H$. As can be seen in table 5 double routes are easily created, therefore the double routes are first deleted and afterwards ordered according to the total distance.

GENERATE LIST Now a relatively long list exists with different routes going from a to b. According to this list the update of congestion in the network will be checked. This process is described in chapter 11. For now it is important to keep in mind that this list is checked at every time step and, therefore, it should be as short as possible. At the same time this will be all the information about congestion available and it should therefore be long enough to represent the complete network. These competing aims are solved in the algorithm in figure 1. In this algorithm item: 5 may need some clarification. A route $(x_1, x_2, x_3..., x_{n-1}, x_n)$ consists of multiple links $l_{(x_1,x_2)}$, $l_{(x_2,x_3)}$,...,$l_{(x_{n-1},x_n)}$. Here the difference is made that $l_{(x_i,x_j)} \neq l_{(x_j,x_i)}$ $\forall i, j$. Now this set of links needs to have at least one unique element compared to the set of links contained in the set of already inserted routes FL

1. Sort the list of routes by increasing total distance.
2. Create an empty set of covered links: CL and an empty set of the final list of routes: FL.
3. The current route that is evaluated is the first (e.g. the shortest route).
4. If there are no more routes to evaluate: Stop.
5. If the current route has at least one directed link not in set CL add route to final list FL .
6. Add all directed links to CL.
7. Evaluate the next route and return to step 3.

Algorithm 1: Algorithm list generator

This algorithm has the nice property that if every directed link is covered by the initial set of routes the resulting set of routes is covered as well.

Theorem 1. *If* $l_{(i,j)} \in IL \Rightarrow l_{(i,j)} \in FL$

Where IL is the initial list and FL the final list.

Proof. If $l_{(i,j)} \in IL$ it means there must be a set of routes R_0 where $l_{(i,j)} \in R_0$. Of all these routes at least the route with the smallest total distance will be added to the final list. This is because before that step $l_{(i,j)} \notin CL$. This gives that $l_{(i,j)} \in FL$ □

The set FL now consists of a set with increasing length where every additional route covers at least one link that has not been covered before. Now the only thing that is needed is that the initial set covers all directed links. This is not always fulfilled especially since the returned routes were reset. This means $l_{(*,a)}$ will never be covered. This does not mean a mistake was made by deleting loops but it does need a fix. The reason for loop deletion is still valid since optimal routes needed to be obtained. Therefore additional routes need to be added.

9.3 GENERATE MATRIX

This means it is possible to find routes for every combination of begin and endpoint. The goal, however, is to find a matrix that updates the complete network. This matrix contains information about the network, vessel speed, vessel type and about the congestion on the waterways. The focus now is only to find a matrix that contains the information about the shortest routes. That means congestion and speed of vessels are neglected for now. In section 9.6 the separation between shortest route and speed of the vessels is made and in chapter 11 a detailed explanation covers the congestion.

COMBINE START NODES The vessel types, start IDs and end IDs were all separated and each combination would have its own matrix. But the start IDs are not relevant at all. This is because the update matrix works as an update process where vessels are attracted to their destination along the shortest path given their position at each time step. Regardless of their departure node, if two vessels were having the same destination and are present at the same node the route to be taken is the same. This can be exploited to reduce the total number of matrices.

In section 9.1 routes were generated which can be used right now. When combining the generated routes of one end node n_e, all the start nodes N_s available for a vessel type C and one vessel type a collection of routes $R_{C,e}$ is combined. This collection is now used to create a distance matrix.

```
network =

     0    10     0     0     0     0     0     0
    10     0    10    30     0     0     0     0
     0    10     0    10    10     0     0     0
     0    30    10     0    20    30     0     0
     0     0    10    20     0    10    40     0
     0     0     0    30    10     0    10     0
     0     0     0     0    40    10     0    10
     0     0     0     0     0     0    10     0

>> assignroutes{1}

ans =

         0    1.0000         0         0         0         0         0         0
         0         0    0.9830    0.0170         0         0         0         0
         0    0.0506         0    0.0506    0.8988         0         0         0
         0    0.0057    0.3314         0    0.3314    0.3314         0         0
         0         0    0.0503    0.0054         0    0.8939    0.0503         0
         0         0         0    0.0009    0.0533         0    0.9458         0
         0         0         0         0    0.0002    0.0533         0    0.9465
         0         0         0         0         0         0         0         0
```

Figure 14.: example network

DISTANCE MATRIX Now a distance matrix DM can be generated. To do this look at link from node i to j and find in all routes that pass through this link: $\{R_{C,e}^{(i,j)} | (i,j) \in R_{C,e}\}$. Of each route take $(i,j,*,e)$ such that $(i,j,*,e) \notin (i,j,*,i,j,*,e)$ and call this set $\bar{R}_{C,e}^{(i,j)}$. Now take of all these routes the one with the smallest distance $DM(i,j) = min(|\bar{R}_{C,e}^{(i,j)}|)$. In this way a matrix is generated that has elements on (i,j) which tell how big the remaining distance is to the end point if present at node i and heading to node j. But a problem arises since it could happen that $\bar{R}_{C,e}^{(i,j)} = \emptyset$ because simply no route passes trough the link. In section 9.4 this error is countered and solved such that no link present in the network is free of routes.

TUNING FACTOR Now the matrix consists of distances to the desired destination but transition probabilities are required. Since the link with the minimal distance is desired to have the highest probability the following procedure is used:

$$RM(i,j) = \frac{\left(\frac{1}{DM(i,j)}\right)^{\tau}}{\sum_{\forall k}\left(\frac{1}{DM(i,k)}\right)^{\tau}} \tag{9.3.1}$$

In equation 9.3.1 RM stands for the route planner matrix. The tune factor τ can be chosen according to how aggressive the model should scale down the probability of a detour. The denominator makes sure the rows in TM sum up to one since a direction must always be chosen exactly once.

EXAMPLE As an example the same network is taken as in figure 9.11(a) so the matrix from figure 14 is obtained where a route needs to be found from node 1 to 8. In this same figure the found transition matrix is shown. The route with the shortest path of distance 60 is $(1,2,3,5,6,7,8)$. The matrix with transition probabilities gives exactly the same route if the highest probabilities are taken for each step. As mentioned before some probabilities are missing. For example, there exists a link from 2 to 1. It is extremely unlikely that traveling through this link leads to an optimal route but congestion could be the cause that a vessel is obliged to take such a route. Therefore some extra routes need to be added to solve this problem. This problem will be addressed in section 9.4.

9.4 CATCHING ERRORS

It might be possible that some links are not covered by a route at all or just in one direction. This could lead to problems both in normal but especially in a situation where one or more links are deleted.

PROBLEM The first thing to ask ourself is why a missing link could be a problem. This is because a link that is not covered by a route has no information at all about the distance to travel towards the end point. No information leads to an update matrix where the mentioned link has a probability of zero which means no vessel will pass trough this link ever.

For example the update matrix in figure 14 would never allow vessels to sail back from node 2 to node 1. But imagine a extra link directly from node 1 to node 8 but with such a long length that all other routes are shorter. Now a vessel left the harbor, is halfway the system but the link between node 7 and 8 is the subject of an incident. Now another route needs to be found but since routes from node 2 to 1 were not present the vessel will think it got stuck. This is exactly the reason to have all the links covered by at least one route.

In order to obtain a transition matrix that represents the complete network it is necessary have at least one route covering each link in both directions. To do this two different tricks are implemented. A directed route can be missing but the reverse could be present or no route is available at all at a link. Lets first look at the case were a reverse link is present.

REVERSE ROUTE PRESENT Now let us assume a link in one direction is missing but the other way is present in the list of generated routes. The idea used here is to start at the end of the link from where no route is available. From here pass trough this undirected link and follow the routes that are present in the reversed order back towards the origin. Now the routes will at some point intersect some other route that passes around this link. As soon as these routes are found they are followed towards the destination. Now let us try to define this properly.

It is possible to split the set of routes R in two subsets R_R, R_D where $R_R \cup R_D = R$ and $R_R \cap R_D = \emptyset$. These two sets will be referred to as the removed routes R_R and detour routes R_D. These names will be explained in the upcoming section.

The set of removed routes all pass trough (i, j) and therefore so will all detour routes $(i, j) \not\subset R_D$. In figure 15 this is represented as a route straight from p to r or from q to s. The rounded squares here represent a collection of nodes that have at least one feasible path through them. The circles represent simply ordinary nodes. But this means that if $R_R \neq \emptyset$ there exists a route $R_k \in R_R$ with the shortest distance. This path consists of a start part ${R_k}^s := (a, *, i)$, middle part ${R_k}^m := (i, j)$ and end part ${R_k}^e := (j, *, b)$. Here the index k is equal to the index of the route with the shortest distance through (i, j) e.g.

$$k = \{k | \min_{\forall i} ({R_i}^s + {R_i}^m + {R_i}^e) = {R_k}^s + {R_k}^m + {R_k}^e\}$$

.

A simple solution would be to add $(j, i, j$ to the route to obtain: $(a, *, i, j, i, j, *, b)$ now $R_{k\tilde{m}}$ is guaranteed to be present in the route and is feasible but not ideal since a vessel would not pass up and down a link in real life. To avoid this problem a little trick is introduced.

It can still be assumed that the start part ${R_k}^s := (a, *, i)$ and middle part ${R_k}^m := (i, j)$ are available. With these parts the first part of a new route can be formed. After this part the reversed link (j, i) is taken but now it is necessary not to let the route pass back through i, j again. To assure this the link in the network along with all links that pass through it are removed. This set is exactly equal to R_R. This leaves us with the detour routes which

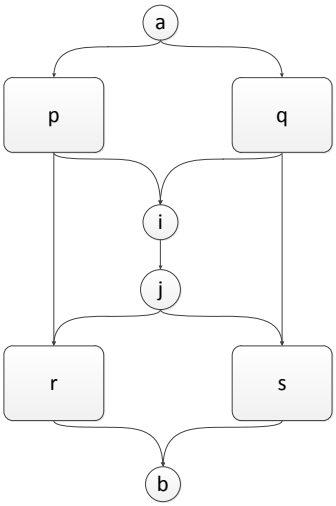


Figure 15.: reverse present

go from a through p, q, r and/or s. But the set that was just deleted can still be of use. If the start parts $\tilde{R}_{R^s}$ are reversed routes going from i to a are obtained. Together with the detours it is possible to create a new end part.

Before this can be done there are three conditions that need to be fulfilled:

- $R_R \neq \emptyset$
- $R_D \neq \emptyset$
- After deletion of link (i, j) (and (j, i)) a route $(a, *, b)$ must still be possible.

The latter can be checked with the test: 'routepossible' described in section 9.5. If one of these conditions is not fulfilled the program returns that is failed using this method and 'Reverse not present' which can be found in the next paragraph is used to solve this problem.

In order to create this new end part the reversed parts of $\tilde{R}_{R_s}$ are evaluated at every node starting at i. If a route intersects with one of the R_D this combination of the reversed route $\tilde{R}_{k^s}$ and R_D^l) is stored. As soon as a reversed start part $\tilde{R}_{k^s}$ found an intersection the rest of the route is not evaluated anymore. If all possible routes are evaluated the route with the minimal total distance is stored as the final route. That means this route consist of the following parts:

$$(R_k{}^s, i, j, i, R_{E_1}, R_{E_2})$$

Where R_{E_1} is the first part of $\tilde{R}_{k_s}$ and R_{E_2} is the last part of R_D

Along with the routes that pass through (i, j) there are paths that do not pass through. $\forall xs.t.(i, j) \not\subset R_D$

At the same time the reversed part $R_{k\tilde{m}} := (j, i)$ is not present in the list of routes R that was generated.

REVERSE ROUTE NOT PRESENT If the reverse is not present an alternative is needed. Here a solution would be to just generate new routes between a and j and between i and b but generating these routes is very costly in computing time and therefore not the most effective solution. The idea that is used here is to spread outwards to other routes from the point where the connection is missing. This means a similar approach with the reverse present except now a route is not followed back but every connection is checked.

ALGORITHM The pseudo code is presented in algorithm 2. A stop counter is used to set the number of steps that are taken after the first route is found. More steps lead to more possible options for a route that is created and that leads to routes with smaller distances. In the model this number of steps is chosen to be 3 but could be adjusted by a user. Note that this is not a very elegant solution but a last measure to at least have all links in the system.

Input : Network, existing routes, startid, stopnr = n_s
Output: Smallest route

Set new found route distance to $|R_{new}| = \infty$
Generate a vector $V = \overrightarrow{0}$ with the length equal to all the nodes present.
Define a list C_t of 'nodes to check' in iteration step t.
Set $t = 1$, done=0, stop=0 and set $C_1 = n_s$
while *stop* $\neq$ *stopnr* **do**
  for $\forall i \in C_t$ **do**
    add all unique neighbors of n_i to C_{t+1} with the exception of nodes in C_l with $l < t + 1$
    if n_i *intersects one or more routes* **then**
      Generate new routes $R_{generated}$ (explained in the text)
      if $|min(R_{generated})| < |R_{new}|$ **then**
        $|R_{new}| = |min(R_{generated})|$
        smallest route $= min(R_{generated})$
        done=1
      end
    end
  end
  $t = t + 1$
  if *done==1* **then**
    stop=stop+1
  end
end
Return smallest route

Algorithm 2: reverse not present

Missing from the code is how these routes are generated when an intersection is found.

MAKE A LOOP The only thing missing in algorithm 2 is what $R_{generated}$ means. Now a route $(a, *, c, *, b)$ is given and the algorithm also found a connection between i and c. Tthis route can be combined to a route $(a, *, c, *, i, j, i, *, c, *, b)$. In this way a loop is created which might not be ideal but in some cases the only solution. Since this is a last measure solution to find the last missing pieces the consequences of these loops being generated are minor.

9.5 ROUTE POSSIBLE

To find if the network is connected to all its parts a small test has to be done. This is necessary in order to find out if a route is possible at all. This problem was solved in two different ways. The first involves the calculation of eigenvalues and the second method spreads out over the network until it s cannot go any further. An explanation and analysis is given in the next two paragraphs.

EIGENVALUES The first solution is the calculation of the eigenvalues and eigenvectors of the network. If the eigenvectors contain a zero if that node can be reached and none zero entry otherwise.

- $V_{ij} = 0 \rightarrow$ node j is not reachable from node i
- $V_{ij} \neq 0 \rightarrow$ node j is reachable from node i

From this information the number of communicating classes can be obtained and within a class which nodes can be reached. Thus by looking at the node where the route starts and checking if the end node is contained in the class it becomes clear whether a route is feasible or not.

A big downside is that calculating eigenvalues is very costly in computing time. This means that calculating the feasibility of a route is takes more time than finding the routes themselves. This was the reason to find an alternative.

SPREAD TROUGH NETWORK The goal now is to find if parts of a network are connected and being able to calculate this within a reasonable time. The method that is used is similar to dijkstra's algorithm [1] without testing for the shortest distance but just storing the nodes that are connected. That means starting at the origin and check if all are connected.

input : Network, startid, endid
output: routepossible

Generate a vector V with the length equal to all the nodes present consisting of zeros. This vector represents all the visited nodes.
Define a list C of 'all nodes to check' consisting of only the initial node.
keepgoing=1
while *keepgoing* **do**
  Set the current node to be the first node from the list of nodes to check and delete it from the list
  For the current node, consider all of its unvisited neighbors N and set the neighboring nodes in the vector V_N to 1.
  Add all these neighbors to the list of all nodes to check with the exception of the already visited nodes.
  if *the destination node d in the visited nodes list is* $V_d = 1$
  then
    routepossible='route possible';
    keepgoing=0
  else
    if *the list of all nodes to check is empty* $C = \emptyset$ **then**
      routepossible='route not possible';
      keepgoing=0
    end
  end
end
return:routepossible

Algorithm 3: Spread trough network

Some readers may already have noticed that the idea behind this algorithm is quite similar to algorithm 2. This makes perfect sense since their problems reduce to the same issue of finding which nodes are connected.

9.6 UPDATE

Apart from the congestion it is now possible to generate a matrix that chooses a direction for each vessel. So it can safely be assumed from now on that a vessel is present in the network and it knows where to go. But

knowing where to go and actually moving along the network is a different story. This is because every vessel has a certain speed. To incorporate this into the updating probabilities the time it will take to travel along a link is needed.

PARTITION UPDATE PROBABILITY Lets start by defining the transition matrix.

Definition 23. Let T_{ij} be the transition from node i to node j. Now $P(T_{ij})$ is the probability of transition from node i to j

The aim will be to obtain this matrix so the total transition probability between every two nodes is needed. The probability in a direction is known but the speed still needs to be added. To do this the transition probability is split into these two parts.

Definition 24.

$$D_{ij} = \text{Take } i \to j \text{ direction.}$$
$$G_{ij} = \text{Go in } i \to j \text{ direction.}$$

Going in a direction needs some clarification. In reality a vessel moves with an approximately constant speed through a link. The model, however, requires that a vessel either stays in a node or transfers to another node. The goal now is to make sure that the expected number of time steps it would take a vessel to move from one node to another is equal to the time steps in reality. This probability $P(G_{ij})$ therefore has to represent this number. But how does this work?

The total transition probability is split up into two parts.

$$P(T_{ij}) = P(D_{ij} \cap G_{ij}) = P(D_{ij})P(G_{ij}|D_{ij})$$

for $\forall i \neq j$

This means $P(D_{ij})$ and $P(G_{ij}|D_{ij})$ have to be found for each i and j. Besides that It must hold that $\sum_{j=i}^{\gamma} P(T_{ij}) = 1$ because of all the options including staying at a node it must choose exactly one.

DIRECTION Since the route planner is working the direction for a vessel is known at this point which means that $P(D_{ij})$ is known. It is also known that $\sum_{j=o}^{\gamma} P(D_{ij}) = 1$. This was because of normalization applied in equation 9.3.1

SPEED But now the connection between the speed of a vessel, the length of a link and the transition probability have to be dealt with. First a definition of the location of a vessel at time t is defined.

Definition 25. $c(t)$ is the current node at time t

This vessel can move between nodes. The interesting thing now is when it makes its first transition.

Definition 26. $\tau = min_t\{(c(t) = n_j|c(0) = n_i)\}$

τ now defines the first time a vessel moves from i to j. Due to the conditional probability that a direction is already known it's certain that the current destination of transition is j. Now it's possible to state what the goal is.

$$E(\tau) = \frac{|l_{ij}|}{\Delta t \times V_k}$$

Since Δt is the time step of the system, $|l_{ij}|$ the length of the current link and V_k the speed of a vessel in category k the fraction gives the total time

steps you would expect from a vessel that needs to sail between node i and j. A bit of caution is needed for V_k since this is the minimum of the maximum vessels speed of CEMT-type k and the maximum speed on the current link by a vessel of this type. Since these numbers can be fixed prior to the calculations no problems appear here.

Now the expected value for the time steps it would take to transfer from one node to another must be the same as the time a vessel would take to cross in reality. This is best described by a geometric distribution with success probability p. Here the expected value of the first occurrence of success require k number of independent trials is given by $\frac{1}{p}$. This means if the probability of transition is taken as one over the expected time steps our goal is satisfied.

$$P(G_{ij}|D_{ij}) = \frac{1}{E(\tau)}$$

Which leads to a transition rate:

$$P(G_{ij}|D_{ij}) = \frac{\Delta t \times V_k}{|l_{ij}|}$$

V_k is the speed in kmh
$|l_{ij}|$ is the distance in km of link l_{ij}
Δt is the time step in h

$$\sum_{j=1, j\neq i}^{\gamma} P(G_{ij}) \neq 1$$

NO TRANSITION The identity $P_{ij} = P(D_{ij})P(G_{ij})$ for $\forall i \neq j$ is known, but what about $i = j$?

$$\begin{aligned}
P(T_{ii}) &= \sum_{j\neq i} P(D_{ij} \cap G_{ij}^C) \\
&= \sum_{j\neq i} P(D_{ij}) P(G_{ij}^C|D_{ij}) \\
&= \sum_{j\neq i} P(D_{ij})(1 - P(G_{ij}|A_{ij})) \\
&= \sum_{j\neq i} P(D_{ij}) - \sum_{j\neq i} P(D_{ij})P(G_{ij}|D_{ij}) \\
&= 1 - \sum_{j\neq i} P(T_{ij})
\end{aligned}$$

Since the route planner is available, maximal speeds on links or with vessels and length of links are known it is now possible to generate the transition probabilities for all networks and vessels.

TIME STEP Now its clear to see how vessels are updated a big consequence comes to surface. A probability to update a vessel can never be bigger than one. To maintain this precondition a condition arises for the maximal time step the system can handle. This condition depends on the shortest links combined with the maximal vessel speed on those links. Depending on the total time steps, size and complexity of the network and number of vessels it could lead to long calculation times. A brief solution that could be implemented is addressed in section 14.4.

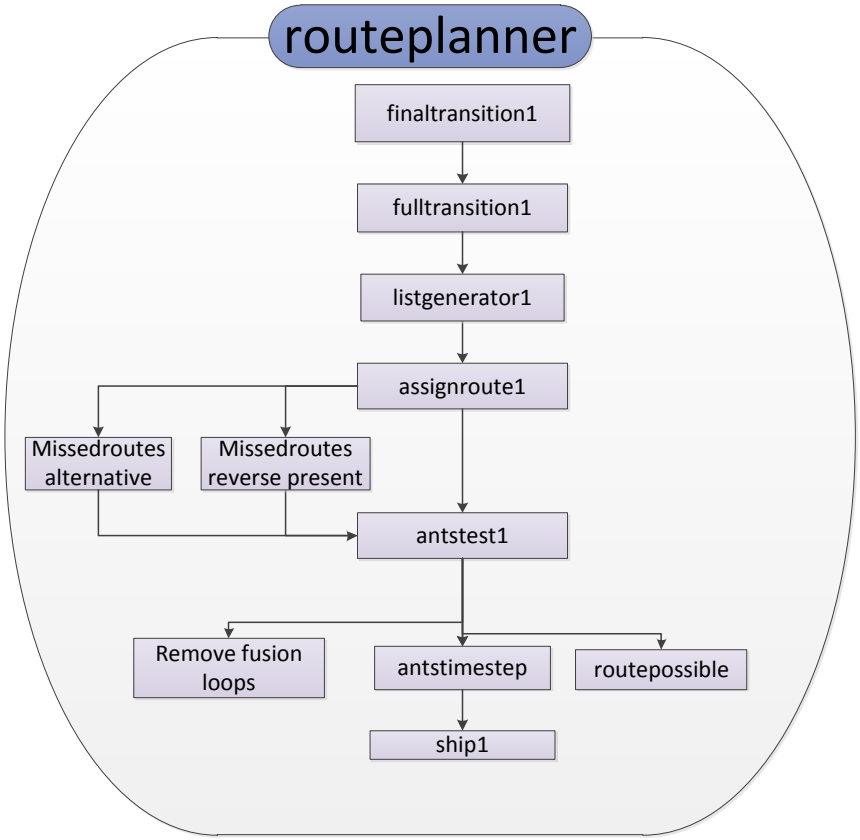


Figure 16.: routeplanner

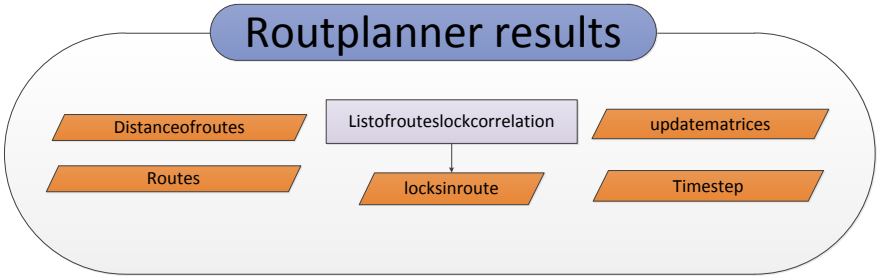


Figure 17.: route planner results

9.7 OVERVIEW ROUTE PLANNER

ROUTE PLANNER Now the route planner is clear it is time to take a step back and look at what the model does. In figure 16 this process is shown. Starting at the bottom the program 'ship1' lets vessels take a step and 'antstimestep' does that for each vessel in a time step. In the next step a test is needed to see if a route is possible and the fusion loops need to be cleared. These two things are used by 'antstest1'. This program uses multiple time steps to generate routes from point *a* to *b*. 'assignroute1' generates all the routes that need to be used and the two 'missedroutes' are used to cover the whole network. The rest of the programs make sure the update matrix is generated in the way described in this chapter.

RESULTS In figure 17 the results that this chapter has generated are shown. This means that from now on it is possible to assume that a list of routes, their distances, an update matrix, and a time step are available. Along with this there is a program that tells which locks are present in the routes and in which direction they are passed. These results will turn out to be very useful in the upcoming chapters.

10

FIT VESSELS IN A LOCK

The question that will be answered in this chapter is how many vessels can enter a lock and which ones can when given certain conditions. Since both vessels and locks come in a variety of different sizes, calculating if a vessel can enter the lock is not always trivial. After the theory an example will be given to see what is happening.

10.1 THEORY

There is a rule which claims that the fist vessel to arrive at a lock is processed first. [4] From now on this will be referred to as the first come first serve (fcfs) rule. Since these vessels arrive at different time steps a calculation has to be made to establish which vessels are able to enter a lock together. If more than one vessel arrives at the same time step their arriving order is randomized. There could be a situation where a smaller vessel behind a larger one is able to fit but the larger one does not. In this case the smaller vessel must be placed in the lock. This means the input is given as a vector of arrival times with corresponding vessel types together with a lock width and length. The output wanted is a vector consisting of ones and zeros where 'one' means 'fits in lock' and 'zero' means 'does not fit in lock'.

INPUT To solve this problem the input] first has to be evaluated. At a certain time step there is a lock that is ready to start its next locking cycle. Therefore, vessels that have been waiting need to enter. Since every vessel keeps track of its arrival time at its last entered node all the vessels combined at the entrance of the lock can be evaluated and ordered by arrival time. The vessels themselves can all have different sizes as well as the lock itself.

IDEA The vessels are fitted into the lock by algorithm 4. The main idea behind this algorithm is to use corners that are available to add a vessel. The other thing used are the edges of locks and vessels. These are stored into two lists, a vertical and horizontal line list. When a vessel is placed the edges are stored into those same lists. When the first vessel arrives only the top left corner is available and if it fits in the lock it is placed there. Now two different corners are created. One on the right next to it and one on the bottom of the vessel. The corner that has just been used is noted as a used corner and cannot be used again. The new corners are sorted from top to bottom. When two corners are on the same height they are sorted from left to right. Now find the first corner from the list and try to fit the vessel. If it fits place it at the current corner and delete all corners inside or on the left or top edge of the currently placed vessel.

When new vessels arrive, however, the problem becomes more complex. More possibilities of corners are generated and the right side and bottom of the lock are not the only conditions for placing the vessels. This problem is equivalent to the function 'place' in the algorithm.

PLACE Now it is known how vessels are placed or omitted from entering but the function place is not explained properly. The new corners must be placed in a lock where several conditions have to be met. How these vessels are placed and how corresponding corners are updated is described

input : Vessel types: $C(V)$, arrival times $A(V)$ and lock size: (O_w, O_l)
output: vector E of entered vessels

initiate variables
deletedcorners=$\emptyset$
CEMTtypesorted=CEMT types sorted by arrival time
E= $\overrightarrow{0}$

linesx = top and bottom line lock
linesy = left and right line lock
corners = top left corner

```
for j = 1 to number of vessels do
    for i = 1 to #corners do
        if vessel j length fits in corner i then
            if vessel j width fits in corner i then
                corners, deletedcorners, linesx, linesy ←
                place(); E(j)=1
                Break
            end
        end
        i=i+1
    end
end
```

Algorithm 4: vessel fitting

in algorithm 5. These corners consist of a (x, y) coordinate and a x and y 'till' value. These combined values give the maximal rectangle that is completely free of vessels and therefore able to contain a vessel within those sizes.

If they fail to fit at all possible corners a zero is returned for the vessel that tries to enter. When it does fit a one is returned and in both cases the next vessel tries to fit into the lock until all vessels have tried and the list of the vessels that entered is returned.

10.2 EXAMPLE

Now the algorithm is given it is time for an example to get a feeling what is happening. The input for the vessels is randomly chosen while the lock is the largest in the data set. Since large locks can contain a lot of vessels these ones are the most interesting to evaluate.

INPUT As input the following is used:

```
vesselsinlock=lock1([1 2 3 4 5 5 6 7 7 8 8 9],
[5 6 1 8 2 4 1 3 2 10 1 3], CEMTdata_max, newlocks, 191, 1)
```

Where most important are the arrival times:

```
[1 2 3 4 5 5 6 7 7 8 8 9]
```

These can be ordered like above or completely randomly. As long as this vector has equal length to the vessel types no problems will arise. This vessel type vector is:

```
[5 6 1 8 2 4 1 3 2 10 1 3]
```

The `CEMTdata_max` is needed to obtain vessel sizes and `newlocks` gives the lock sizes. Finally there is an id for the lock and an index if the results need a plot. This plot is not relevant for the final model but gives a nice insight into what is happening.

```
input  : vessel length, vessel width, corners, deleted corners,
         linesx, linesy, corner id
output : corners, deletedcorners, linesx, linesy

add horizontal lines of vessel to linesx
add vertical lines of vessel to linesy

for i = 1 to #corners do
    if corner i inside current placed vessel then
        add corner i to deleted corners
    end
end

//go from bottom to top
for j =#linesx to 1 do
    if right side vessel < x_2 and bottom side vessel > y_2 then
        if current node ∉ deleted node then
            find right till
            find bottom till
            place corner in sorted list
            if line j is cut then
                Break
            end
        end
    end
end

//go from right to left
for j =#linesy to 1 do
    if bottom side vessel < y_2 and right side vessel > x_1 then
        if current node ∉ deleted node then
            find right till
            find bottom till
            place corner in sorted list
            if line j is cut then
                Break
            end
        end
    end
end

//update all the till values with the new added lines for
k = 1 to #corners do
    update corner k x-till value w.r.t. new linesx
    update corner k y-till value w.r.t. new linesy
end
```

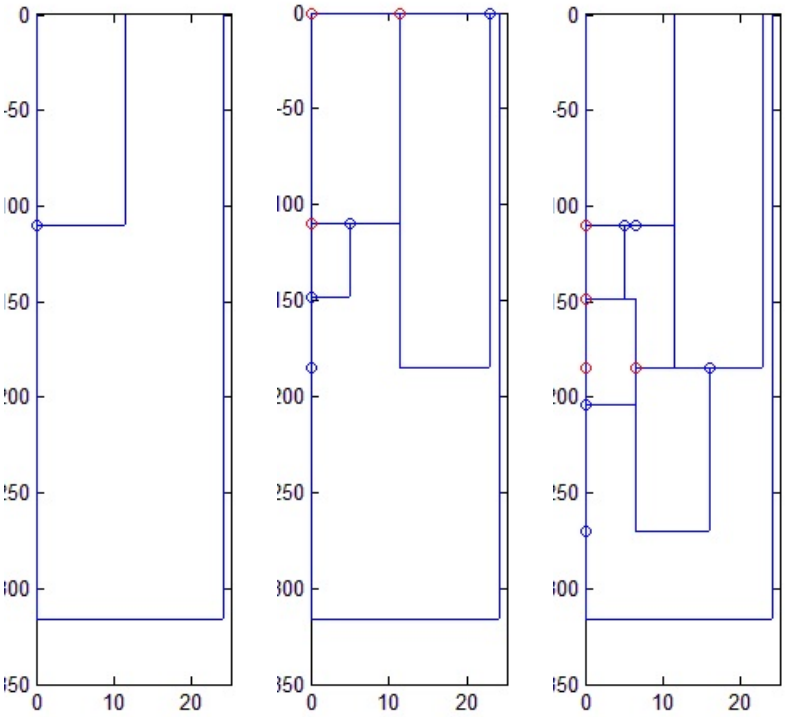
Algorithm 5: Place

Enter lock	Vessel width	Vessel length
1	110	11,40
1	185	11,40
1	38,50	5,05
0	140	15
1	55	6,60
1	85	9,50
1	38,50	5,05
0	80	8,20
1	55	6,60
0	280	22,80
1	38,50	5,05
0	80	8,20

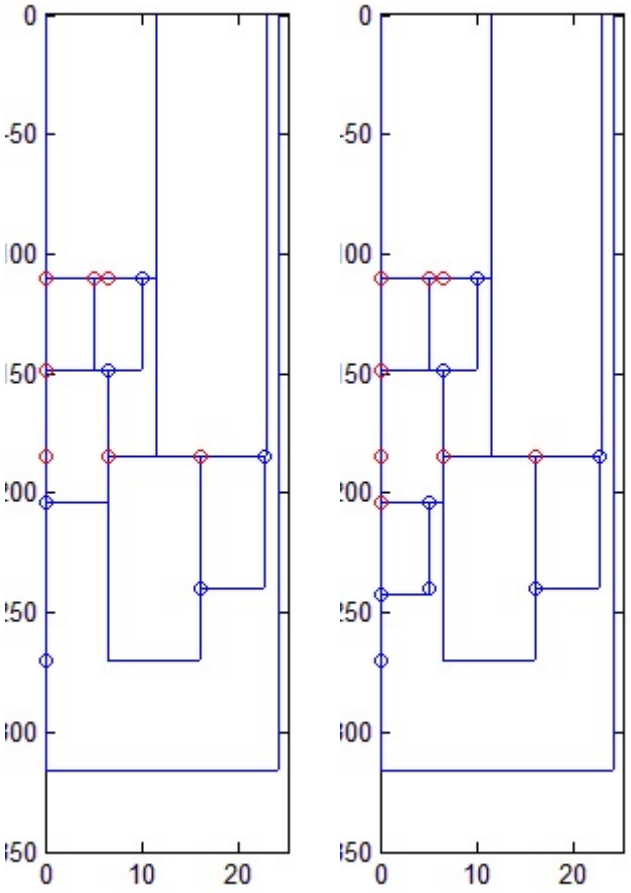
Table 6.: Vessels in lock

OUTPUT Every new vessel that tries to enter the lock will generate a separate figure. A selection of these figures is shown in figure 18. As can be seen here the new corners in blue are added on lines where they are supposed to be and the red circles represent the deleted corners where no vessels can fit anymore. Table now gives the output for this instance. As can be seen not only the first few vessel will enter but some smaller vessels that arrived later are able to fit while bigger earlier arrived vessels have to wait. Note here that the entering of the lock is not a chronological time event but just a model that checks which vessels fit. This means that after this 'plan' to fit the vessels is made the vessels can sail to their right spot. This allows for small vessels to squeeze behind larger vessels.

SIMULATION SPEED To obtain the time it takes for the algorithm to fill a lock a monte carlo simulation is done. This means multiple similar simulations are done for random lock sizes and random vessels that try to enter the lock. The number of these vessels is fixed which means every lock has to try and fit the same number of vessels. Now 10000 simulations are done to extract the average computation time from this number. In figure 19 the number of vessels that arrive at every lock is printed on the horizontal axis while the vertical axis represents the average running time of one lock processing the given number of vessels.



(a) Subfigure (b) 4 vessels (c) 6 vessels



(d) 9 vessels (e) 14 vessels

Figure 18.: vesselfitting

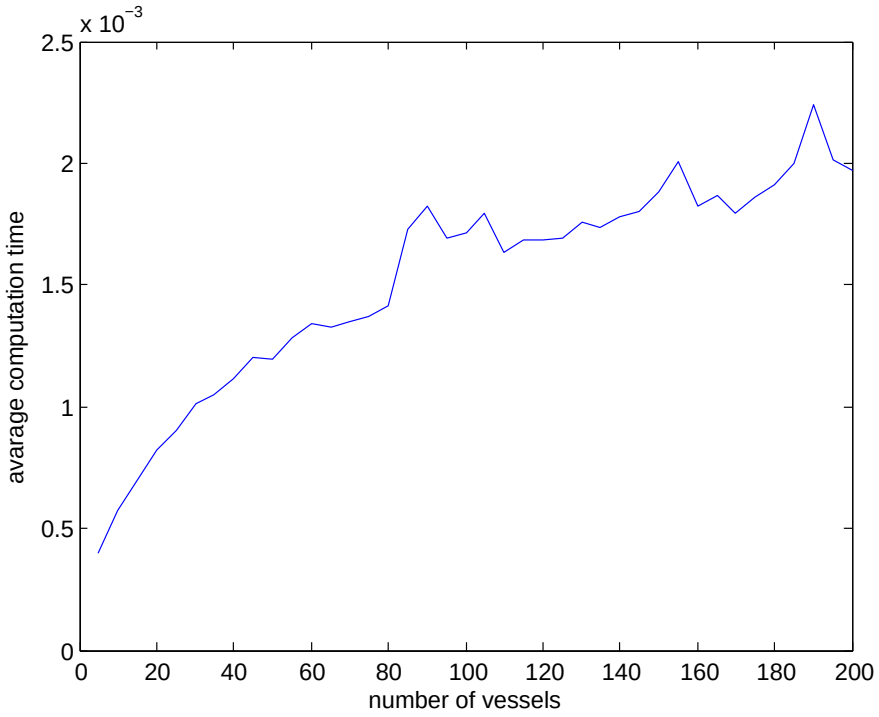


Figure 19.: running time vessel fitting

11

CONGESTION

When an incident blocks a pathway it is imminent that accumulations will build up. If the system is left running without giving the vessels some kind of information the chance of taking a useful detour is absent in the model but present in reality. A solution could be changing the transition probabilities in our system according to the congestion in a certain route. But how should these routes measure the congestion?

11.1 VESSELS

Since the same number of vessels in two different nodes might lead to long waiting times in one but have no consequences in the other, one has to be careful about how to define congestion. The vessels sailing across the Dutch canals have the most influence on each other near artifacts as bridges and especially locks. The idea is to measure the amount of vessels waiting for a lock to estimate the rate of congestion on a route.

ROUTES For each route that was generated in section 9.2 it is possible to check for locks inside this route. If, along with which lock(s), the direction is stored it is possible to add some extra distance to a route. This virtual distance is the congestion on this route generated by waiting times in front of locks. This extra virtual distance is calculated as $d = \frac{v_d}{E(t_w)}$ where v_d is the speed of vessel type d and $E(t_w)$ the extra expected waiting time. This means this number depends on the speed of the vessel and the total waiting time. Since the routes are already created the only update that has to occur every time step is the congestion at locks. Because there is a fixed number of locks this can be done fairly quickly. Finding which locks are present inside a route also has to be done only once, which means that only the found numbers have to be added to the routes.

UPDATE Now all these routes have a new virtual distance the same trick as described in section 9.3 can be used to generate the matrix, only now with the distances. What has to be done now is define what $E(t_w)$ should be.

11.2 LOCKS

When vessels update they may at some point end up in front of a lock and be forced by the route planner to pass through it. In contrast to the updating of the vessels on the waterways the updating of a lock happens deterministically. This means that as soon as a lock has work to do it starts and it will update conform to its real locking time.

Definition 27. A locking time $t(O_j)$ of lock j is equal to the time it takes a lock to complete one cycle and end up in the same situation.

IN- AND OUTFLOW In chapter 10 it is already explained which and how many vessels can enter a lock simultaneously. This procedure is used here to calculate which vessels can enter. This means that as soon as the lock is ready to start a cycle vessels are able to enter. To maintain the nature of how locks operate a state of a lock has to be introduced.

STATE OF A LOCK Because a lock can only serve vessels in one direction at a time and never twice in the same direction a restriction has to be added to the process of a lock. An up/down-variable is introduced to represent the current state of a lock which means the lock is either in the up- or downstate. While transferring to the other state its label is still the last known state with a restriction that no vessels are allowed in.

DISCRETE TIME PROBLEM A problem with this structure is the discrete time steps and the continuous nature of a lock. The transition between up and down in discrete time is not the problem but the start and ending time is. As an example take a time step $\Delta t = 5$ and a locking time $t(O) = 7$. If three time steps would have passed by, the lock could have handled two complete cycles but if it is only allowed to start at the beginning of a time step the lock is not done after time step number three. To solve this the time the lock would have started locking is used to calculate its residual locking time. While this rounded time is stored the time the lock does start is at the beginning of a time step. This is to stay in the structure where discrete time steps are used and the connection with the vessels that arrive in discrete time is maintained.

NEUTRAL STATE Along with these two states a neutral state is added to make sure a lock is not waiting in the 'wrong' state while a vessel is arriving. This state is only reached if the lock is idle for half its locking time. This means it could have reached both states and can now instantly switch to the state it needs to be in to serve vessels best. It could also happen that a lock just reached an idle situation and within half its locking time a vessel arrives at the other state. In this situation the lock would in reality already have been starting to transfer to its other state. This is solved by using the difference between the last time a locking was completed and the current time. This difference is then subtracted from the upcoming transition given that no vessels arrived at the same time in the state the lock was waiting in.

11.3 PARALLEL LOCKS

Apart from single locks, it is also possible to use parallel locks, which prevent a waterway from being blocked if one lock is broken and they allow for an increase in capacity and speed of the locking process. These parallel locks not only cut waiting times if there is a waiting queue but they also increase the probability of arriving when the waiting time before entering is small. This, as can been seen in section 11.2, is because the lock can be in an 'up'-state or 'down'-state. With multiple locks a vessel can always choose the best option. To incorporate this effect in the model the expected waiting times are calculated for two, three and four identical locks. In the case a lock will be built in the future that is placed parallel to a lock with different locking times a section is added that explains what will happen to the expected waiting time in that scenario.

EXPECTATION SINGLE LOCK When calculating the expected passing time trough a lock a distribution has to be found for the arriving vessels. The distribution that fits the process the best will be the uniform distribution. In figure 20 the state of a busy lock is sketched on the bottom and it has to be assumed a vessel is on the 'down' side of the lock. When it arrives it first has to wait until the lock is in the down state for it to enter the lock. After entering it still has to wait $\frac{1}{2}p$ where p is the time of the lock going up and down and therefore the complete locking time. This means the best case scenario of an arrival is if it arrives where the vertical black arrows are drawn. Then it enters immediately. Arriving earlier has the consequence of waiting exactly the extra time it takes to move the lock to the 'down' state.

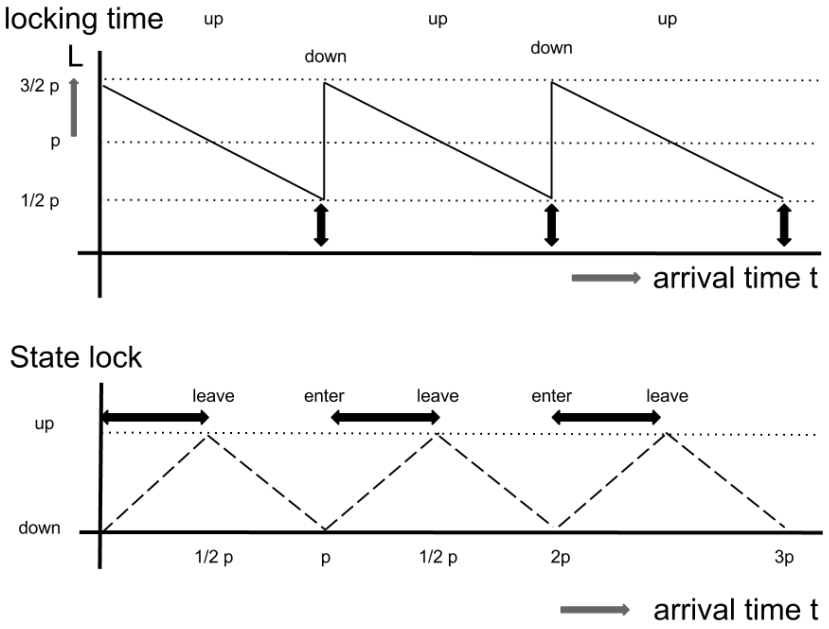


Figure 20.: uniform distribution lock

Since there is no knowledge about the exact arrival time it can be assumed the arrival time is completely random. This combined gives that:

$$L \sim Un(\frac{1}{2}p, \frac{3}{2}p)$$

EXPECTATION MULTIPLE LOCKS PARALLEL When parallel locks are available an arriving vessel will take the lock with the least amount of waiting time. Thus the problem looks as follows:

$$X_1 \sim Un(a_1, b_1)$$
$$X_2 \sim Un(a_2, b_2)$$

The expected waiting time depends on the lock with the smallest amount of waiting time, so therefore the minimum is taken over both distributions.

$$Y = min(X_1, X_2)$$

First the Cumulative distribution looks as follows

$$F_Y(y) = P(Y \leq y) = 1 - P(Y > y)$$
$$1 - P(X_1 > y, X_2 > y)$$
$$1 - (1 - F_{X_1}(y))(1 - F_{X_2}(y))$$

To obtain the density function the derivative is taken.

$$f_Y(y) = \frac{\partial}{\partial y} F_Y(y)$$

This gives the following result:

$$(1 - F_{X_1}(y))f_{X_2}(y) + (1 - F_{X_2}(y))f_{X_1}(y)$$
$$E(Y) = \int_{-\infty}^{\infty} y[(1 - F_{X_1}(y))f_{X_2}(y) + (1 - F_{X_2}(y))f_{X_1}(y)]$$

$$E(X_2) + E(X_1) - \int_{range(X_2)} y(F_{X_1}(y))f_{X_2}(y)dy - \int_{range(X_1)} y(F_{X_2}(y))f_{X_1}(y) \tag{11.3.1}$$

IDENTICAL DISTRIBUTION Let us first assume the two parallel locks have the same locking time and therefore the same expected value. This also means $X = X_1 = X_2$ which makes (11.3.1) a lot easier. Also if the same parameters are taken as in the case of a single lock:

$$X \sim Un(\frac{1}{2}P, \frac{3}{2}P)$$

$$2 * E(X) - 2 * \int_{range(X)} y(F_X(y))f_X(y)dy$$

$$2 * E(X) - 2 * \int_{\frac{1}{2}P}^{\frac{3}{2}P} yp^{-1}(y - 1/2 * p)\frac{1}{p}(y)dy = \qquad 2 * p - 2 * 1/12$$

And this leaves the answer:

$$E(Y) = \frac{5}{6}p \tag{11.3.2}$$

THREE IDENTICAL LOCKS

$$1 - (1 - F_X(y))^3$$

$$f_Y(y) = \frac{\partial}{\partial y}F_Y(y)$$

This gives the following result:

$$\begin{aligned} E(Y) &= \int_{-\infty}^{\infty} y[3(1 - F_X(y))^2 f_X(y)]dy \\ &= 3\int_{-\infty}^{\infty} y[f_X(y) - 2F_X(y)f_X(y) + (F_X(y))^2 f_X(y)]dy \\ &= 3 * E(X) - 6 * \int_{\frac{1}{2}P}^{\frac{3}{2}P} yp^{-1}(y - 1/2 * p)\frac{1}{p}dy + 3\int_{\frac{1}{2}P}^{\frac{3}{2}P} y(\frac{y}{p} - 1/2)^2\frac{1}{p}dy \\ &= \frac{9}{12}p \end{aligned}$$

FOUR IDENTICAL LOCKS

$$1 - (1 - F_X(y))^4$$

$$f_Y(y) = \frac{\partial}{\partial y}F_Y(y)$$

This gives the following result:

$$\begin{aligned} E(Y) &= \int_{-\infty}^{\infty} y[4(1 - F_X(y))^3 f_X(y)]dy \\ &= 4\int_{-\infty}^{\infty} y[f_X(y) - 3F_X(y)f_X(y) + 3(F_X(y))^2 f_X(y) - (F_X(y))^3 f_X(y)]dy \\ &= 4 * E(X) - 12 * \int_{\frac{1}{2}P}^{\frac{3}{2}P} yp^{-1}(y - 1/2 * p)\frac{1}{p}dy + \\ &\quad 12\int_{\frac{1}{2}P}^{\frac{3}{2}P} y(\frac{y}{p} - 1/2)^2\frac{1}{p}dy - \int_{\frac{1}{2}P}^{\frac{3}{2}P} y(\frac{y}{p} - 1/2)^3\frac{1}{p}dy \\ &= \frac{7}{10}p \end{aligned}$$

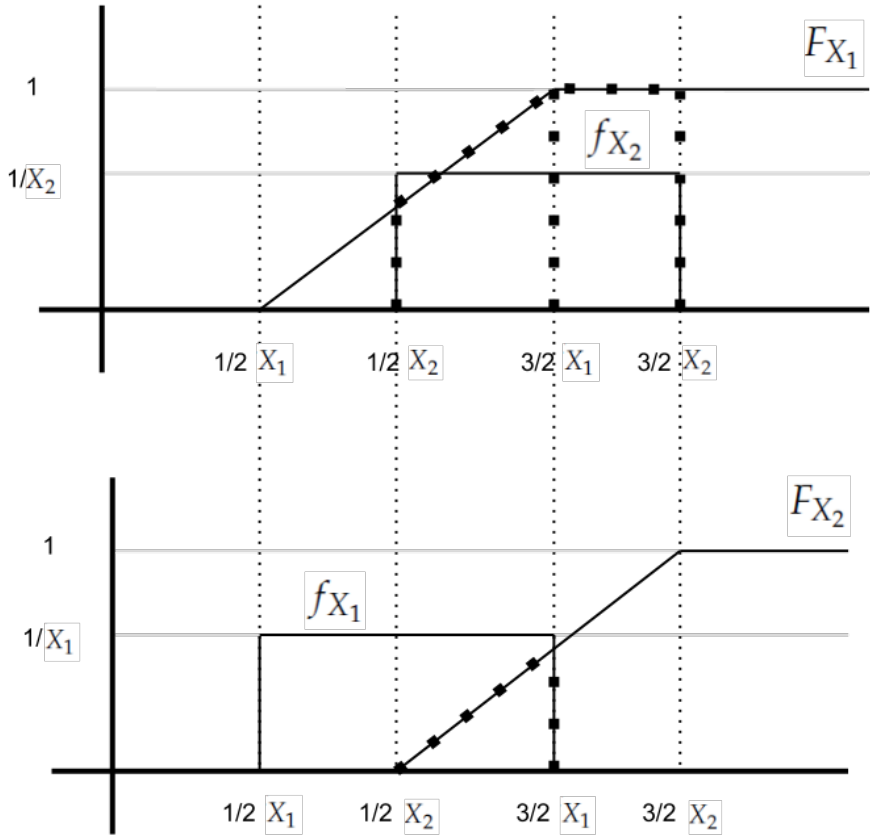


Figure 21.: Unequal lock decomposition

TWO NON-IDENTICAL LOCKS It's also possible that these locks have a different locking time. To calculate the expected value equation (11.3.1) is taken and split the cumulative and probability density function into different parts. Assume without loss of generality that $p_1 \leq p_2$

$$X_1 \sim Un(\frac{1}{2}P_1, \frac{3}{2}P_1)$$
$$X_2 \sim Un(\frac{1}{2}P_2, \frac{3}{2}P_2)$$

$$E(X_2) + E(X_1) - \int_{range(X_2)} y(F_{X_1}(y))f_{X_2}(y)dy - \int_{range(X_1)} y(F_{X_2}(y))f_{X_1}(y)$$

Using figure 21 the first integral gives us:

$$\int_{1/2\times p_2}^{3/2\times p_1} (y/p_2 * (1/p_1(y - 1/2 \times p_1))dy + \int_{3/2\times p_1}^{3/2\times p_2} (y/p_2)dy$$

and the second one:

$$\int_{1/2\times p_2}^{3/2\times p_1} (y/p_1 \times (1/p_2(y - 1/2 \times p_2))dy$$

Since $E(X_1) = p_1$ and $E(X_2) = p_2$ the results yields:

$$-\frac{9p_1^2}{16p_2} + \frac{p_2^2}{48p_1} + \frac{25p_1}{16} - \frac{3p_2}{16}$$

And if the locking times are identical $p_1 = p_2$ the result from equation 11.3.2 is found again.

11.4 NUMBER OF LOCKINGS

So far there was no mention about how to calculate the total expected number of lockings. This depends on the total number of vessels that are waiting, the sizes of those vessels and of course the type of lock. Since both locks and vessels come in different sizes there is not always a perfect way to fill the lock. This means some space in the lock is unused every locking. The question is how big is this number.

CALCULATE EFFICIENCY LOCK First a way to measure the efficiency needs to be introduced. This is done by calculating the ratio of used surface by vessels versus the total surface of the lock. These numbers can easily be calculated since its only a summation of surfaces of rectangles. An algorithm to come up with the total number of vessels that fits in a lock is already introduced in chapter 10. This algorithm can here be used here again to let a random list of vessels approach a lock and calculate the proposed ratio. For now a fixed ratio is used for all locks. Since this ratio can differ if lock sizes change a differentiation between lock sizes would be a nice extension in the future.

By doing a monte carlo simulation over all possible locks where locks without any vessels inside are neglected ratio obtained is equal to 48%. But now take a look at the example of a lock with a size of 280 by 24 meter. The efficiency of such a lock increases to stunning 85%. This means the warning to use this simplified approach shouldn't be taken lightly. The other side of the story is that these numbers represent an upper bound. Because an almost infinite queue containing small vessels will almost never occur in reality small vessels boost up the ratio by filling in all gaps. With this being said everything is now set to define and implement an incident.

12

INCIDENT AND RESULTS

Now everything is set up to simulate an incident. The route planner, the congestion update, locks and vessels are all modeled and working. To do this first an incident has to be defined and implemented in a proper way. Doing this also creates some problems that need to be solved.

12.1 INCIDENT DEFINITION

An incident on the Dutch waterways may have many different causes but the only relevant aspect here is the consequence. The consequence can be one of the following:

1. Blocked link
2. Partly blocked link
3. Blocked junction
4. Blocked lock
5. Lock speed reduction
6. Lock capacity reduction

These incidents all have a slightly different implementation and impact. First the different incident are set apart but the whole process is set aside. After that the main idea of how these incidents are modeled is described in section 12.2.

PARTLY BLOCKED LINKS Partly blocked links can have two different consequences. Vessels might be forced to sail in convoy in one direction at a time due to narrowing of the waterway. It could also be that certain vessels are not able to pass through anymore. An example would be a bridge that is not able to open anymore causing a height restriction. If a convoy is needed a link can be modeled in the same way a lock is modeled. Vessels can only enter from one direction at a time, half the locking time is set equal to the expected passing time but the parts where vessels are fitted into locks are neglected. If a restriction to the vessels is the consequence the link type is changed to the one fitting this restriction best.

LOCK SPEED AND CAPACITY REDUCTION If a lock speed or capacity would change these numbers are simply changed in the model. An example of this is the incident with the lock in Eefde explained in 2.2. Here a crane would open and close the lock instead of the original doors which increased locking time.

BLOCKED LINKS AND LOCKS Number 1 and 4 from list 6 can be handled in the same way. These incidents are also the most interesting since they were the main focus of the whole project and have the biggest consequences. These are modeled by simply deleting the links or locks from the network and run the model.

BLOCKED JUNCTIONS The last incident is a blocked junction. Since modifying the network before analyzing is not restricted to just one modification more than one and even a mixture of them can be set to happen. If a junction is blocked it's also possible to just delete all connecting links. This way one incident is split up in multiple incidents which describe the same situation but all separate are just link and/or lock deletions.

12.2 INCIDENT SIMULATION

The different incidents are described but now a simulation has to be made. In some cases links are deleted or have new conditions for vessels. It's possible to try and find all the new routes trough the network but regenerating all these routes takes some time. A smart trick could be done here to avoid this time spend on extra calculations. Since the biggest part of the network is still the same as before the original found routes could be used. But a deletion of a link or lock could lead to situations where parts end up not connected and infeasible routes.

NO CONNECTION When a incident is defined by the user the algorithm from section 9.5 can be used to find if there is still a feasible connection. When vessels can find no such route they are immediately stored in a list of vessels that are stuck. This list forms one of the outputs of the model.

COVERAGE Deletion of network parts also means that routes that were possible in the past might be blocked now and therefore impossible. It could even be that some vessel have no place at all to go anymore. Infeasible routes are now broken up into two parts, a part before and a part after de deleted link. The part after the deleted link can be maintained without any problems but the part before you leads vessels to a blocked area. This means deletion of these parts of the routes is the only option right now. But deleting these parts could lead to links that are not covered by the residual routes. Since one of the requirements was that every link had to be covered this leaves a problem. But this problem is already been solved in section 9.4. Using the same principal the gaps in the network can be covered by these new generated routes. But let's not throw to much caution to the wind. These algorithms were designed to find the leftovers that were not covered, not completely empty pieces. This could lead to strange routes within the list of resulting routes. Since no additional extension are made right now a future improvement here could lead to more accurate results.

COMPARISON Now two different simulations can be made: one prior to the incident and one during and posterior. These two simulations can now be compared to each other. Since the interest lies in what the consequences are and not what the total outcome is of a single simulation, the results are given in differences between the two runs. The data that is returned consists of the following:

- overall average + variance delayed arrival time
- specified average + variance of delayed arrival time
- overall number of infeasible routes
- specified number of infeasible routes
- overall traveled distance
- specified traveled distance

Each of these specifications can be set to meet different criteria consisting of a combination of the following things:

- CEMT types
- departure nodes
- arrival nodes
- departure times
- arrival times
- total traveled time bigger or smaller than threshold

All this information can simply be extracted from the list of vessels that arrives, the list of vessels that are still in the system and the list of vessels with an infeasible route.

12.3 STOPPING TIME

A simulation is set to run for the length of the incident but when the incident is over there could still be a queue of vessels in some part of the network. This means a bit of additional time has to be added to the simulation. This can be done by observing when the system returns to its 'normal' state. There are several ways of implement such a stopping criteria. three of them are explained here and could even be used together at the same time. Analysis will have to be done to find which one will be suited the best.

CONGESTION Since a normal state would imply that there is at most as much congestion as the normal situation. Since the total congestion can be measured this number can be used for a stopping criterium. Because congestion is measured in front of locks the total number movements a lock has to make in the whole system to process the current waiting vessels seems a logic number. This number is set to be equal to the average number obtained from the initial run. The second simulation is stopped when both all the incidents are cleared and this number is below $(100 + c)\%$ of the initial number. The number $c > 0$ can be chosen to make sure the system comes to a stop but does not stop too early.

ARRIVAL TIMES Another option is to look at the arrival times of the vessels. Since their departure time is stored along with the total distance they've traveled the total delay of individual vessels can be measured. Again a threshold $(100 + c)\%$ can be used to compare these numbers with the normal situation.

NUMBER OF ARRIVED VESSELS The last option is the base the stopping criterium on the number of arrived vessels. Its possible to measure many vessels arrive within an interval. This number can be obtained in both the normal as in the incident situation an for the last time a threshold $(100 + c)\%$ can be used to compare these numbers.

In addition to these stopping criterion a t_{max} is fed in to the system to make sure it always stops. When t_{max} is the cause of the system to stop a warning is shown to the user.

12.4 OVERVIEW INCIDENT

Now that all theory is clear let us go back to the model itself. In figure 22 the input of a simulation is generated. The start and end IDs, an initial simulation to fill up the system and an incident are all ran to obtain the data that is necessary to do a simulation. With this input it is time to set the model in motion. This is done in figure 23. The congestion is checked, network is updated, locks are updated, arrived vessels are extracted, new vessels are added and a check is done if the model should stop. If this is

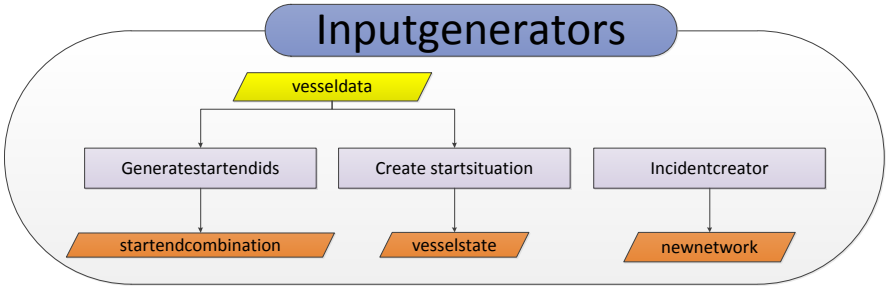


Figure 22.: input generators

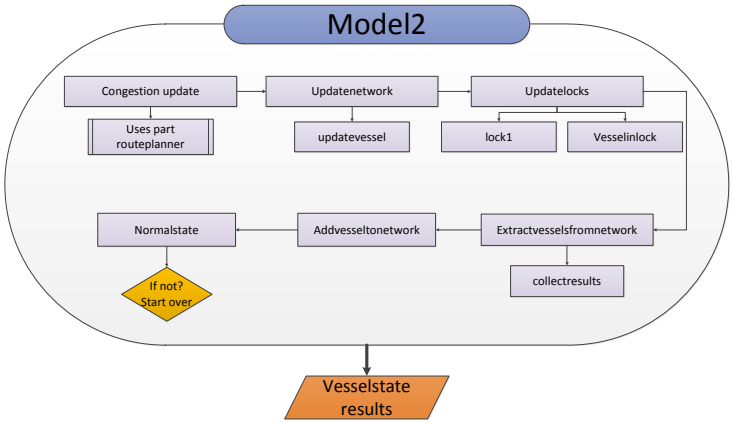


Figure 23.: model2

not the case, the cycle is started over again until the program 'normalstate' stops the cycle. The results obtained now only need to be processed and will be displayed in a similar picture as 7. Now everything is done it is time for the final part iii where the model and its simulations will be reviewed.

Part III

RESULTS FROM CONGESTION ANALYSIS

13

MODEL ANALYSIS

Now the model is fully described the model itself can be analyzed, scenarios simulated and results derived. All the different parts mentioned in part i and part ii are implemented in the model shown in figure 1. The only problem now is that due to time restrictions the part connecting all the different aspects is not working properly yet. Consequence is that a complete analysis can not be done. The aim of this last part therefore is to give an overview of everything thats working. A evaluation about those parts is made. Of all the other parts a description is given about what needs to happen to make them work. The reader will see that most unfinished parts are fairly easily added and implemented to the system. Finally a recommendation for improvement is given that can be used in the future.

13.1 (UN)FINISHED PARTS

NETWORK AND FLEET DATA The network that was introduced is fully functional. This means the data is imported into the Matlab environment and processed in such a way the data can be plotted. Besides that the same data is converted to matrices to be able to make calculations. The fleet data is also imported and available to use. Potential new data can be imported and replace the old data or extend it.

ROUTE PLANNER AND CONGESTION The route planner is almost completely functional. This means finding routes is implemented, the simplification of the list and links that are not covered can be found. The different origins that head to the same destination are not yet combined. This should give the output a higher quality. Generating the route planner matrix also works fine. Now the only thing missing to cope with the congestion is the locks that have to add the extra waiting time to the routes. This is partly done but an automated programm which links locks to the routes need still to be completed.

MODEL This is the section where all the parts need to be linked together. First the model needs to start up doing the following.

- Createnetworks
- Createstartsituation
- Incidentcreator
- Listofrouteslockcorrelation

After that it needs to complete the following cycle.

- Congestionupdate
- Updatenetwork
- Updatelocks
- Addvesselstonetwork

- Extractvesselsfromnetwork
- Normalstate

This is the main piece that links everything together but is not finished yet. Implementing this would lead to a fully working programm.

VISUALIZATION The last part is the visualization. This is already described and the following programs need to run:

- Collectresults
- Processresults
- Visualizeresults

Since no results were obtained yet a visualization isn't made yet

13.2 VERIFY MODEL

A choice is made to model the vessels by a markov chain and update locks deterministically but no mention were made about the results so far. Since it is possible for vessels to go faster than they could go in reality but also wait for a long time with no congestion strange results might come out. At the same time the average numbers that come out of the results should represent what is happening in reality. There are several ways to verify these results. A few of the options for verification are given here:

- Compare to real life normal flow examples
- Compare to real life incidents
- Compare to other models
- Consulting experts
- Use the marine traffic website to track vessels

It may be necessary to tweak the model a bit to modify the outcomes. The mentioned verification is a good way to know how well the model is preforming and witch parts should be adjusted.

13.3 CALCULATION SPEED

There are a some choices made that slow down calculation speed but improve the quality of the output while at the same time some other choices did the opposed. Obviously all these choices together influence the total calculation speed. Without a model that can be tested these parts are briefly clarified.

LEVEL OF VESSELS The choice to give every vessel its own information is necessary if a user of the model wants to obtain specific information about for example certain types of vessels, different corridors or intensities on a route. Although this choice needs both more

STOCHASTIC PROCESS The process of finding routes is based upon random movement of vessels. On forehand it is hard to tell if this leads to extra or less calculation time. Since these runs do not have to be done all over again every time it is not of big importance. This is because routes can be found in the network before a user starts using the model. This is a huge advantage where every new simulation only need a few small adjustments depending on the incidents that are modeled.

CONGESTION Since the assumption was that congestion only in locks this part reduced significantly. The congestion is measured by a total surface of vessels which is just a summation along with some linear operations. Adding extra artificial length to the route is also a linear process which only leaves the creation of the transition matrix. Since also this last operation also only consists of linear operations the whole update process should be quite fast.

VESSELS IN LOCK The updating of the locks is a process of comparting sizes of vessels with available space of the different corners. The other part here is placing these vessels and creating new corners. This is basically checking intersections of vertical and horizontal lines. For example if 1000 random locks are evaluated all with 300 random vessels waiting in front of them the average calculation time is 3.15 seconds. This means that although large locks with a lot of waiting vessels could lead to an increase of calculation time the expectation is that this should not be a problem.

TIME STEP Biggest bottleneck will probably be the constraint on the time step. This constraint leads to a maximum length of this time step which implies updating vessels and the congestion updates occur relatively often. The latter could be compensated by saying the congestion is only measured and updated every $n > 1$ time steps. Since the updating of the locks occur deterministically this time step does not involve the number of times the vessels need to be fitted inside a lock. Other solutions are proposed in chapter 14.

13.4 DISCUSSION

Before using the results a few words of caution are in its place. The choices made in in section 6.3 have some impact on the results and not all parts are fully optimized. A list of recommendation for the future are done in chapter 14. These can be seen as both improvements and extensions of model. A disquisition is now made grouping the advantages of the system that is chosen followed by the disadvantages.

ADVANTAGES The biggest advantage in this model is that preparations can be done before using the model. This saves a lot of calculation time and should result in a program that is able to generate results really fast. Another nice thing is the dynamical adjustment of the vessels to each other. There is no other model on the market that takes this into account. The last advantage is the stochastic behavior of the vessels which captures the irregularities of the vessel movement in real life.

DISADVANTAGES A big downside is that the length of the links, speed of the vessels and locking times imply a upper bound for Δt. This could imply a long calculation time when dealing with incidents that have a large lifespan. Another problem is that a lot of simplifications have been made before reaching this point and all these choices influence the outcome.

14

FUTURE EXTENSIONS

During the creation of the model several ideas came to mind which did not made it to be implemented in the final model. There are several reasons why this happened but the main reason was simplicity together with the lack of time. The ideas are listed in this chapter and may not be worked out completely but are listed to give an overview of possible future improvements and extensions.

14.1 COSTS

A result of the model is the individual and total delay of certain vessels but another interesting aspect is the corresponding costs. The easiest way to implement these cost is to relate these delays linear to costs. Although the easiest way is the linear approach all kind of functions can be chosen here. One has to keep in mind that not only delay but also traveled distance contributes to the total extra cost. The assumption was made to only take into account the direct costs. This means another extension could be to incorporate the indirect costs. Since these costs differ greatly from area to area depending on alternative infrastructure, transported goods, type of vessel etc. this might be a difficult thing to implement.

14.2 REST PLACES

In the current version the rest places are neglected. This implies an infinite queues and waiting in the middle of canals is possible. This may be impossible in reality or cause different problems in reality such as extra blocked links. The reason here for neglecting this aspect is the expectation of low impact on the results. Since this is not verified a small study to the consequences or even implementing this property might be a good idea.

14.3 LOCKING TIME DEPENDENT ON VESSEL SIZES

The larger a vessel is the longer it takes to complete a locking cycle. This means the assumption that a lock change state deterministically is not true assumed the vessels inside are not known. If the sizes of the vessels are known up front there might exist a function that approximates the total locking time of a lock. Also here is a question of how important the consequences are.

14.4 GHOST LINKS

Since the biggest problem in the model is the existence of short links. To account for this problem several tricks can be used. One of them is to add ghost links. These links can be viewed as second order transitions. The ghost links are added at the places where the probabilities of transition are greater than one when Δt is increased while the links itself are deleted. This can only be done if no locks are present and it has to be proven that this manipulation of the network yields the same results

14.5 WATER LEVELS

The assumption is made that water levels are posing no problems to the flow of the vessels. In reality waterways have a certain depth and vessels carry a load. With low water levels and/or high payloads vessels may be unable to pass trough the network. Consequence can be that to transport the same cargo an increase of vessel movements may occur. Another result may be that certain vessel types are not able to pass trough links at all anymore. This last consequence can actually already be implemented by viewing this as an incident.

14.6 BRIDGES

Another problem might be not low water levels but high ones. Because different this combination together with stacked containers of a certain height determine if a ship can pass beneath a bridge. This means vessels in the model have to keep track about the cargo and corresponding height to determine if a vessel can pass. Another result is that vessels should be able tot choose how much cargo they take since this influences the available routes they can take.

There are also bridges that open and close if vessels pass trough. Some of these bridges have opening times which mean that passing during the night is not an option. Another thing that might happen is that a bridge stays closed because of traffic across it. Both these examples may have an impact on the results while not incorporated in the model.

14.7 IN/OUT-FLOW

For some simulations the user may only be interested in a small area or corridor. Especially when the duration of an incident is very long the calculation time might be long. If this is the case it is probably a good idea to not model the whole network but just a part of it. This means the borders of the small section the user is interested in become in and out flow points.

The idea is to just handle all vessels with departure and destination harbor within the borders as usual. The other vessel all have either a destination or departure harbor outside the simulated area. For every such departing vessel the harbors are replaced with one of the in/outflow points. If it is an inflow point the expected entering time can be calculated on which the vessels enters. These in and/or outflow points are calculated by what would normally be the shortest route towards their destination.

14.8 BEHAVIOR VESSELS

The last recommendation for future improvement is about the behavior of the vessels. When an incident occurs vessels may depart later or not at all due to their knowledge of the incident. Since these decisions depend on factors as available detours, infrastructure of road and rail, time dependency of the cargo transportation it is hard to fit them into a model. The best solution is probably a cost function which decides if a vessels should leave depending on previously mentioned factors along with the delayed time.

Vessels may also be not informed of an incident that just took place which would require a reaction time in which vessels just continue as before. Here the implementation is fairly simple but a question remains about how long this reaction time should be.

BIBLIOGRAPHY

[1] Dijkstra, E. W. (1959). *"A note on two problems in connexion with graphs"*. Numerische Mathematik 1: 269271. DOI:10.1007/BF01386390

[2] Bosschieter, C.G. (2005). *"Klimaatverandering en binnenvaart. Effecten op de binnenvaart van meer extreem lage (en hoge) waterstanden op de Rijn"*. Port Research Centre Rotterdam-Delft ISBN:9056381423

[3] BIVAS `bivas.chartasoftware.com`

[4] Rules lock `http://wetten.overheid.nl/BWBR0003628/DeelI/Hoofdstuk6/AfdelingV/Artikel628/geldigheidsdatum_15-08-2012`

[5] Vesselsizes `http://www.scribd.com/doc/59715985/34/Tabel-7-1-Karakteristieke-duwbakken-met-kenmerken`

[6] Marin `marin.nl`

[7] Lorelei incident `http://www.spiegel.de/international/germany/accident-near-lorelei-sulfuric-acid-tanker-capsizes-on-the-rhine-a-739272.html`

[8] Lorelei consequences `http://www.evo.nl/site/00A91F76123B7FDFC12579220041FB0D`

[9] Rijkswaterstaat `www.rijkswaterstaat.nl`

[10] Witteveen+Bos `http://www.witteveenbos.com/en/mission-and-vision`

[11] TNO lock Eefde `http://www.tno.nl/content.cfm?context=overtno&content=nieuwsbericht&laag1=37&laag2=2&item_id=2012-03-27%2017:19:11.0`

[12] Metro news lock Eefde `http://www.metronieuws.nl/nieuws/schade-kapotte-sluis-eefde-loopt-in-miljoenen/IWIlaf!nSYce26VCTBB@D@@dkG1Hg/`

[13] Schuttevaer new lock Eefde `http://www.schuttevaer.nl/nieuws/actueel/nid17080-tweede-sluiskolk-eefde-in-2017-klaar.html`

[14] Growth expectations Port of Rotterdam `http://www.infrasite.nl/documents/bedrijven/2032/Portvisie%20Rotterdam%202030.pdf`

[15] Marine traffic `http://www.marinetraffic.com/ais/nl/default.aspx`

A

BIVAS DATA

arcs	**locks**	**cemt_types**
ID	fromid	ID
FromNodeID	toid	Description
ToNodeID	LockageTime	Width
Name	WaitTime	CrossSection
CountryCode	OverlayTime	Depth
Length	CrossSectionAreaUpperGate	CurrentSpeed
Width	CrossSectionAreaLowerGate	Category
ArcTypeID	SwitchDistanceUpperGate	BranchSetID
CemtTypeID	SwitchDistanceLowerGate	
MaxDangerousGoodsLevelID	NumberOfLocks	
MaxLength	LockLength	
MaxWidth	LockWidth	
MaxSpeedLoaded	OperatingTIme	
MaxSpeedEmpty	RecreationalShare	
Delay	BranchSetID	
CurrentDirection		
BranchSetID		

arc_vin_trajectory_con	**bridges**	**nodes**	**arc_types**
ArcID	ArcID	ID	ID
TrajectCode	ServiceTime	XCoordinate	Label
StartKilometer	PassTime	YCoordinate	Description
EndKilometer	BranchSetID	BranchSetID	
OriginalBVMS			

dams	**dest_trip_end_points**	**or_trip_end_points**
ArcID	NodeID	NodeID
MinimumWaterLevel	Name	Name
BranchSetID	BranchSetID	BranchSetID

Table 7.: network data BIVAS

travel_costs	**trips**
AppearanceTypeID	ID
ShipTypeID	Date
LaborCostsPerHour_Euro	OriginTripEndPointNodeID
LaborCostsPerHourWaitingOnLoad_Euro	DestinationTripEndPointNodeID
MaintenanceCostsPerHour_Euro	ShipTypeID
MaintenanceCostsPerKilometer_Euro	DangerousGoodsLevelID
InsuranceCostsPerHour_Euro	LoadTypeID
InterestCostsPerPercentPerHour_Euro	NumberOfShipments
DepreciationCostsPerHour_Euro	TotalWeight
HarborFeesPerHour_Euro	NstrTypeCode
OtherCostsPerHour_Euro	AppearanceTypeID
LoadCostsPerTon_Euro	Depth
UnloadCostsPerTon_Euro	LoadCapacity
Deprecated_CostsEmpty	LoadCapacityClass
Deprecated_CostsLoaded	NumberOfTrips
Deprecated_CostsWaiting	

ship_speeds	**ship_costs**	**ship_types**
ShipTypeID	ID	ID
CemtTypeID	Description	Label
LoadTypeID	BaseScenarioID	Description
Speed	BaseScenarioDescription	CEMTTypeID
EstimatedAverageSpeed		
BranchSetID		

Table 8.: vessel data BIVAS