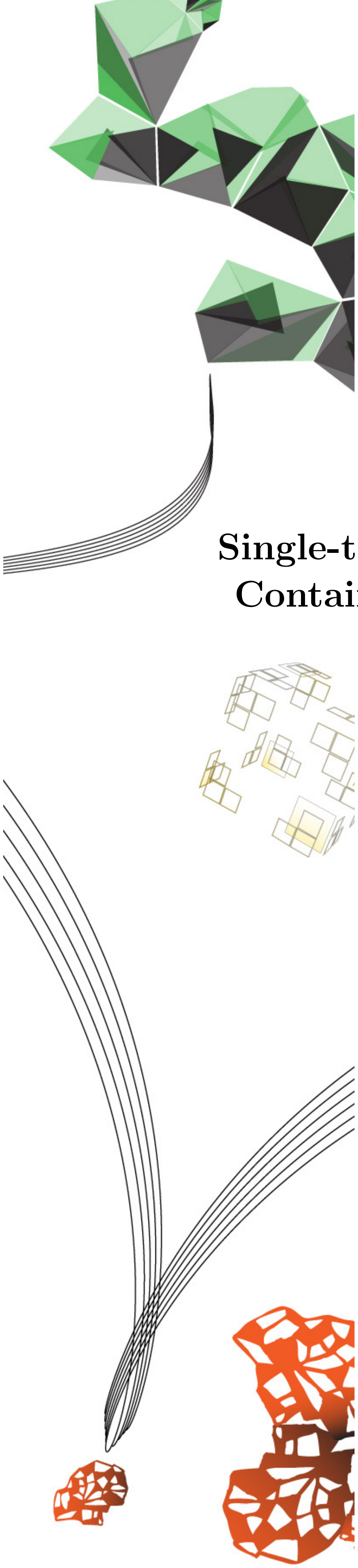


Master Thesis

Single-to-many Dynamic Traffic Assigment for Container Freight on a Multimodal Network

Ruben Fransen
s1004816

April 7, 2016

Abstract

This research presents a mathematical programming formulation for a Dynamic Traffic Assignment model and an implemented iterative shortest-path algorithm. Freight transport demand is assigned to a dynamic network of roads and waterways with the goal to minimize transportation costs. The challenge is to assign paths to a network with time-varying travel speeds and capacities, the assignment of limited capacities to different freight shipments and determining of travel and waiting times. Assigning on a multimodal and dynamic network lead to a non-classical shortest path problem.

Key words: Dynamic Traffic Assignment, Routing problem, Shortest-path problem, A* algorithm, Inland container shipping,

Preface

This master thesis is the final assignment of my study Applied Mathemathics at the University of Twente. The specialization of my masters's programme is Operations Research and this thesis was done for the group Discrete Mathematics and Mathematical Programming.

TNO

I did this research at TNO, which is an independent research organization. The mission of TNO is to connect people and knowledge to create innovations that boost the sustainable competitive strength of industry and well-being of society. For this master thesis I did an internship at the Sustainable Transport and Logistics deparment of TNO.

At this department a model has been developed that can assign freight from the Port of Rotterdam to a final destination in the hinterland. This model is used to analyse multimodal and synchromodal transport flows and the effect that route and mode choices have on emissions and the accessibility and safety of the network. The downside is that this model assigns onto a static network, meaning that all travel information is based upon daily averages and there is no specific record of at what time each shipment is where in the network.

TNO is interested if it would be possible to do a similar freight traffic assignment in a more dynamic setting. In such a dynamic setting it would be possible to take into account travel times that vary during the day and that would result in better estimations of travel times. Also the effect of temporary disruptions can be taken into accound in a dynamic setting. Having better travel time estimations and more exact interarrival times along the routes can be useful for all logistic partners. This can help shippers, terminals and infrastructure managers with improving their planning. For infrastructural design a dynamic model is not required, but it can give a better insight in the effect of infrastructural changes.

Acknowledgements

I would like to thank everyone that supported en guided me during this master thesis.

Prof. dr. Johann L. Hurink
For being my supervisor for this thesis, the valuable converstations we had on the mathematical problems at hand and the guidance for writing this thesis.

Dr. ir. Mo Zhang
For iniating this master thesis from within TNO, being co-supervisor for the first four months of my internship and increasing my knowledge on Transportation Research.

Layla Lebesque MSc.
For replacing Mo, therefore being co-supervisor for the remainder of my internship and for the guidance in finishing this thesis.

Dr. G.J. Still and Dr. J.C.W. van Ommeren
For completing my graduation committee and taking the time to assess this master thesis.

Colleagues at TNO
For a really nice atmosphere and all conversations which contributed to my internship and the quality of this research.

Family and friends
For all support that lead to finishing my Master in Applied Mathematics.

Contents

1 Introduction

Freight transport around the globe is continually increasing. This growing amount of transport requires more capacity of available infrastructure. Consequential larger networks and more interacting trade routes require more and better transport planning. The search for cost reduction and increasing efficiency on these larger and more complex systems has driven research on transport planning. This is resulting in more complex problems, which are getting harder to solve.

Transport planning takes place at three different levels: Strategic, Tactical and Operational Planning [33]. Strategic or long term planning involve highest levels of management and deal with major capital investments, mainly physical network building or resizing. Tactical planning aims at the allocation of resources and scheduling services on a medium term horizon, such that system performance is improved. Operational planning deals with all short term decisions on transport activities: scheduling of services, maintenance, crews and routing of vehicles. This research focuses on routing vehicles and therefore fits in the level of operational planning.

A decent operational planning is required to minimize costs and optimize the amount of freight demand that can be satisfied. A better planning increases the efficiency of system. The optimization of the planning lies in making efficient schedules for vehicles, ships, operations, crew and all other factors that complete the system. To be able to make an efficient planning for all these parts of the system it is required to know what freight has to be transported at what times.

This research focuses on optimizing the operational planning for the transport of shipping containers from the port of Rotterdam to destinations in the Benelux or Germany, the so called hinterland. The challenge is to plan a route for each freight shipment. This planning of routes involves the scheduling of which container will be transported by which vehicle or vehicles to get from Rotterdam to its final destination. It also involves the routing of these vehicles over the transportation network. These vehicles having limited capacity, leads to a challenge of which shipments to assign to which vehicle. The vehicles considered in the setting are trucks and barge ships, also called the transport modalities or modes. The created model can also be used for different types of goods than containers, if vehicles are considered which can transport those goods.

One issue in this research is the routing of vehicles on a dynamic network. An example of the network dynamics is the congestion on highways during rush hours. These and other time varying effects influence the travel time of freight transport. Taking into account these network effects when calculating the travel time leads to better transport planning with more reliable travel times. This better route planning would be beneficial for all other schedules dealing with the transportation of these freight shipments. This is where the Dynamic Traffic Assignment model comes into scope, to be able to assign routes to freight traffic in a dynamic situation.

The goal of this research is to make a tool that can find a dynamic freight traffic assignment that allows all freight demand to be delivered succesfully. The tool should be able to assign freight orders to vehicles and routes and therefore be able to make a decent route schedule. Making such a decent schedule requires to know the travel times and capacities of the vehicles on the dynamic network.

First a more formal problem statement and the research question can be found in the next chapter. That will be followed by a review of literature on traffic assignment and freight transportation in Chapter 3. In the next chapter a mathematical model for solving the problem is formed in a mathematical program formulation. After this model formulation, the construction, implementation and a use case of a developed algorithm for finding a Dynamic Traffic Assignment are treated in Chapters 5,6 and 7. Subsequently an answer to the research question, the conclusions of this research and recommendations are stated in chapter 8.

2 Problem statement

The main problem considered in this research is optimally assigning freight transport onto a dynamic traffic network. The transport network is dynamic, which means that the best route from an origin to a destination is not always the same. Travel speeds on roads can fluctuate and capacities of barge ships and terminals are limited. External influences such as rush hour or network disruptions, by accidents or breakdowns, can also change the best routes. Furthermore, the already assigned freight shipments also influence the network, because they can increase rush hour congestion or use the limited capacities of terminals or barges.

To be able to determine the routes that minimize costs it is required to know the availability and transport times of vehicles. However, as mentioned before the travel speeds and capacities vary over time. So for finding the total travel time of a route it is necessary to know which vehicle(s) can be used and the travel speed of the vehicle(s). The determination of travel times based upon time varying travel speeds adds the time dimension to the routing problem. This time dimension can lead to computationally heavy route calculations. However, in practice the choice of a route often has to be done reasonably fast, to make response based upon real time information useful.

The goal of this research is to create a Dynamic Traffic Assignment (DTA) model which allows to find the cheapest routes, based upon time dependent network data. To react on real-time information it is necessary that a solution should be found in a reasonable amount of time, as this would make real-time response possible, e.g. route improvements along the way. This leads to the following question to be answered by this research:

- *How can a Dynamic Traffic Assignment model effectively assign dynamic freight demand on inland waterways and roads dealing with disruptions and congestion?*

To be able to answer the research question the following subquestions are formed.

- *How to find dynamic routes fast enough to make real-time decisions?*
- *How to assign freight demand requests to limited barge capacity?*
- *What insight in network behaviours could a dynamic traffic assignment model give when network disruptions occur?*

The first subquestion focuses on the time required to find a dynamic route. If some disturbance occurs, then some route might be blocked and routes would have to be changed. It would be useful to assign new routes within a certain amount of time. To do this new routes have to determined fast enough. Finding these routes fast enough contributes to the DTA model.

Barge ships have the capacity to transport many containers and can therefore carry different freight demand requests simultaneously. Making a decent planning for putting the shipments onto the barge ships can make sure that the barge ships are used efficiently. The DTA requires freight to be able to be assigned to barge ships and so this assignment of freight onto the ships is part of the research problem.

Next to having a model that can be used to make a traffic assignment for freight onto a dynamic network, it is good to recognise the practical insights that can be retrieved from a resulting traffic assignment. To show application possibilities of the model this last subquestion is defined. Furthermore, the DTA model should be able to give insight in freight traffic behavior and economic effects of network disruptions. It might even be used to give insight in the robustness of the traffic network.

So we have the research question: How can a Dynamic Traffic Assignment model effectively assign dynamic freight demand on inland waterways and roads dealing with disruptions and congestion? First we continue with a literature review on transportaton modelling which focuses on existing methods. Subsequently, a model is formulated and an algorithm is implemented.

3 Literature review

In this section a literature background on transportation research is given. The basis of transportation modelling and the more recent developments on (dynamic) traffic assignment problems are mentioned. Also a background on shortest-path problems is given as this has also been used for this research. In the remainder of this report, the term 'path' will be used instead of 'route', because the transportation network is considered to be described by a graph and in graph theory it is common to refer to paths instead of routes.

3.1 Transportation research

This subsection focuses on the basics of traffic assignment.

3.1.1 Operational planning

There are several levels of planning in transportation research, as mentioned in Chapter 1. This research mainly deals with the operational planning level of transportation research. Strategic planning might come into scope when more long-term elements such as available vehicles or network changes are considered. On an operational planning level studies have been done on reducing travel times and transport costs for given sets of transport requests. Most studies consider the situation where the path choices influence the costs or even the possible paths for other traffic. In these cases the goal is to find some equilibrium, meaning a solution where it is not possible to improve the current situation. The equilibria in transport modelling are known as the principles of Wardrop [37].

3.1.2 The principles of Wardrop

There are two main types of equilibria in transportation modelling. One equilibrium minimizes the average travel costs per user or equivalently the total costs of the system. This is known as Wardrop's second principle and called the system optimum (SO) or a system-optimal equilibrium. The other equilibrium is known as Wardrop's first principle and is called the user equilibrium (UE), this specifies a situation where no single user can find a cheaper path, if all the users stay at their current choice.

In a SO some individuals might have to use an individually non-optimal path to reduce the costs for other users. In that case investigation of the overall network flow is needed. Therefore, for this situation flow modelling is used instead of looking at individual optimal choices of paths. Which equilibrium is being used as a goal criterion depends on the given situation. A UE is more suited to model a network with individual users that want to reduce their personal costs, where a system optimal equilibrium is more suited when the considered traffic is less individual and is somehow controllable.

3.1.3 Modelling methods

There are several different modelling methods that are used in transportation modelling. The studies for optimizing tranportation systems on an operational level can be categorized in four broad categories. The categories are mathematical programming [23], optimal control [13], variational inequality [24] and simulation-based approach [12] [22] [34].

Mathematical programming and optimal control approach a traffic assignment problem with a similar goal function and constraints. The main difference is that the mathematical programming approach uses a discrete-time step where the optimal control constraints are defined in a continous-time setting, resulting in a continuous-time optimal control problem [33]. Merchant and Nemhauser [23] formulated a mathematical program which has been used as a basis for other studies on dynamic traffic modelling [5] [10] [36].

The mathematical theory of variational inequality was developed to solve equilibrium problems. As such, it can also be used for a traffic flow equilibrium problem. Several algorithms and solution methods have been developed to deal with inequality problems [24]. Finding a suitable solution for a traffic assignment problem is often not very efficient, as it can take quite a lot of time to find a equilibrium. It can also be difficult to calibrate model parameters to get a realistic and practical solution. In these cases a simulation-based approach could be useful. Creating a simulation model can result in more practical results. However, a simulation does not guarantee that some optimum or equilibirum is found. A method to bring a simulation-based model closer to an equilibirum is the Method of Succesive Averages (MSA)(see section 3.1.4).

The above described methods are different in the mathematical approach, but they all may be used to represent traffic situations. For this they use similar representation to describe a transportation network or the behaviour of traffic flow. The main aspects of this modelling of traffic flow are described in section 3.1.5.

3.1.4 Method of Succesive Averages

It can be hard to find an equilibrium when using a simulation-based approach for a traffic assignment problem. Especially a SO can be hard to find. Most simulation-based approaches try to improve the traffic situation step-by-step until a equilibrium situation is found. MSA is a method to improve such a search for an equilibirum.

One way of improving a traffic situation by simulation is iteratively improving traffic flows until an SO is found. The iterative procedure can find cheaper paths for some traffic flow. However, it is difficult to determine the cost effects for other flows in advance, changing the flow to the cheaper paths might cause the total costs to increase. Therefore, it is hard to find a system wide improvement. These unpredictable reactions makes the mathematical behaviour of a simulation-based model hard to track and it is therefore difficult to guarantee convergence of such an iterative procedure [34].

MSA is an effective solution heuristic that has been implemented in simulation-based DTA models. The basic idea behind MSA is that in each iterative step all possible cheaper paths are determined, but only a part of the flow is assigned to these cheaper paths. The advantage of only rerouting part of the flow is that the cost effect for the other flows also occurs only partially. If the made changes seem to be a step in the right direction, then the method records that the change in this direction was succesful and will do more steps with the same direction.

3.1.5 Traffic flow modelling

In the subsections above possible approaches for traffic models are considered. All these approaches have in common that they need to determine the travel costs of specific flows to be able to minimize the costs of travelling over a network. In general, the travel costs per link are considered, which depend on the travel time and some other travel cost. Furthermore, the amount of time it takes to traverse a link is physically dependent on the length of the link and the speed of travelling, whereby the travel speed can vary through time in a dynamic network. An aspect that comes into scope with dynamic traffic flow modelling is the order in which traffic enters and leaves the link.

The travel speed on a link can depend on the amount of flow on the link, e.g. the decrease in travel speed on a road when rush hour congestion occurs. When more traffic is using a link, then the speed on that link decreases until some critical amount of flow is reached and congestion occurs. Therefore, the traffic flows have to be integrated into a model to take this effect into account. This can be done in several manners.

One method to relate traffic flow and travel speed is the fundamental diagram of traffic flow [7], which relates traffic speed, flow level and traffic density. The diagram can be used for different types of vehicles with different properties, where for example vehicle length influences traffic density [21]. It also possible to determine the travel time directly based upon the level of flow [32] or to do a more extensive studie on whole link flows and travel times [4], by taking a closer look at inflow, outflow and related link travel times for some constant inflow. Another method focuses on minimization of flow costs instead of transportation costs. Merchant and Nemhauser [23] relate costs to the level of flow and then cost minimization can be done to find an optimal flow situation.

3.1.6 First in, first out

In traffic flow modelling there is some order in which traffic enters and leaves a link. In static modelling the time a vehicle enters or leaves a link is not taken into account. However, in a dynamic model the time of entrance and exit of a link do get attention. For a single lane traffic link it is clear that in reality a vehicle leaves the link before the vehicles that entered the link behind him, because it is not possible to overtake on a single lane. However, for multiple lanes the situation may be different and vehicles may overtake ieach other. The non-overtaking property is also known as the First-In-First-Out constraint (FIFO). This FIFO policy, which is also known from warehousing or inventory problems, can be integrated into traffic models if the given application aks for it.

There are different methods of implementing FIFO for the different mathematical approaches. Most methods deal with some restriction or constraint for flow or travel time. If FIFO constraints are added to a variational inequality formulation, then it is no longer sure that a solution continues to exist [10]. It is also possible to add FIFO constraints to a mathematical progamming model [3], but it results in nonconvexity issues. Furthermore, if a continuous time formulation with FIFO constraints is studied, it also results in nonconvexity for the objective function [3]. Thus the FIFO policy can be integrated in the different mathematical formulations, but the consequential nonconvexity complicates searching for solutions.

In the continuous approach no new variables and fewer constraints are necessary to integrate FIFO, compared to the discrete case. It is not necessary to add direct FIFO constraints, because of the continuous time setting. Assuming the speed is the same for all flow on a link, then there never is a speed difference between seperate flows on a link. Having the same speed implies that a flow can not pass any other flow on the same link and thus the FIFO policy is followed.

A method to ensure FIFO via the link travel times is to smoothen the travel speed function, where the travel time is determined upon the link travel speed function at time of entrance [11]. The method used in the implementation of this research to ensure FIFO on road links is based upon a method by Ichoua [18]. The method puts constraints on the travel time over a link. These constraints ensure that the travelled distance over the link is at least the length of the link, see section 4.3.4 for further explanation.

3.1.7 Freight assignment

Most transport models are made to model a complete transport system, meaning that they model all flows in the network. The entire flow map of all these combined flows determines the network state, and the flow dependent travel speeds can be determined. However, in this research the assigned freight is only a part of the traffic flow on the network, meaning that there is also other traffic on the network. Assigning only a part of the flow implies that the total flow is not known and a corresponding flow speed cannot be determined directly.

The mentioned property, that we assign only a part of the traffic flow, makes this research different from most DTA problem, because only a part of the network flow is known. The unknown part of network information may get specified by input distributions, similar distributions for the network state have also been used by [9] and [28].

3.2 Iterative path choice

As mentioned in the previous section the network state is not directly known when the assigned freight flow of a traffic system is only a part of the total trafffic. In this situation we could have the network state as some given input. This would allow different ways of finding a path for each freight order instead of modelling traffic flows. One possibility is using a shortest-path algorithm to find a shortest path from origin to destination for each shipment. If the travel time of links is used as input for a shortest-path algorithm instead of the length or weight of the links, then it is possible to use such an algorithm to find a fastest path in a dynamic network with FIFO consistency [9].

Individually finding a cheapest and possible path for each freight order could be a decent solution for the traffic assignment. However, if the assigned paths have some influence on the network state and therefore some influence on the path choice of other freight orders, then this approach only approximates the real situation. If this influence could be constrained in such a way that costs of already assigned paths are not influenced, then it could be possible to iteratively assign cheapest paths to find a UE. That is because iteratively picking a cheapest path and not influencing costs of earlier picks, implies that no freight shipment could have picked a cheaper path.

Iteratively assigning cheapest paths has been applied in literature and also in this research to find a solution for a DTA problem. The following sections describe the development of shortest path algorithms and their use in dynamic networks.

3.2.1 Static shortest-paths

The most famous algorithm for finding a shortest-path between 2 vertices of a graph is the Dijkstra algorithm [8]. The algorithm builds a shortest-path tree from the start node until the goal node is found. A downside is that this method is computationally heavy for large networks.

A reduction in computational effort for the Dijkstra algorithm has been achieved with the A* algorithm [16]. This algorithm uses a guiding heuristic function to direct the search towards the destination. Where Dijkstra's method builds a shortest-path tree from the origin in all directions, the A* method only computes the shortest-path tree from the origin towards the destination. This requires that some positioning of nodes in the network is known, e.g. locations in an infrastructure network are positioned on a map, on this map the Euclidean distance between nodes can be determined. This Euclidean distance is a possible heuristic funtion that can give the A* algorithm a search direction. With the heuristic the algorithm constructs a smaller shortest-path tree before it reaches the goal node, compared to Dijkstra's algorithm. Note that the performance of A* depends on the quality of the heuristic function and there are several conditions that the function must satisfy [20].

A newer and computationally faster method for determining shortest-paths on road networks is the reach-based method. The paper of Gutman [14] describes this algorithm and gives a comparison between the performance of Dijkstra, A*, Reach, Reach combined with A* and Exact Reach. The reach-based method is computationally faster than the A* algorithm if the algorithm has already preprocessed the network. However, this preprocessing takes quite some time. As a consequence only if the number of the to-be-assigned paths is large enough then the total computation time required by the Reach algorithm is less then the total computation time required by A*.

3.2.2 Dynamic optimal path choice

The shortest-path algorithms are initially made for static graphs, so they had to be altered to able to find a fastest paths in a dynamic network. One of the first papers on dynamic path choice was [6]. An iterative scheme is devised to compute a shortest path in a network with time dependent transit times. It originates from Bellman's iteration scheme [1], which was based upon his principle of optimality.

One of the first papers on a minimum-time-path algorithm for a network with time-varying link travel times has been written by Dreyfus [9]. He states that Dijkstra's algorithm can be used to find a shortest travel time path in a time-dependent network. Orda and Rom [28] made a general analysis for a shortest-path algorithm for a network with time-dependent link delays, which is similar to link travel times being time dependent. They conclude that efficient solutions exist if waiting at a node is only allowed at the source node, general waiting contraints for all nodes make the problem very complex. Hall [15] states that Dijkstra's algorithm does not find minimal expected travel time paths if stochastics are introduced into time-dependent travel speeds and travel times.

Horn [17] formulates a method using shortest-path algorithms to find a quickest path in a dynamic network. He uses A* and the Reach algorithm to determine shortest-travel-time paths taking into account time-varying link speeds and subsequent link travel times. This method assumes that time dependent travel times estimates are known.

3.2.3 A* algorithm with dynamic link weights

The A* algorithm will be used in this research, because the dynamic link weights complicate the problem and therefore a initially faster shortest-path algorithm has not been used. Reversed search from destination to origin has not been used because the starting times at links are required for determining the travel costs. A preprocessing algorithm is not used because of the dynamic network. Preprocessing would have to be done for a lot of(or all) time steps and when capacities or travel speeds change then the network also changes and preprocessing has to be revised.

The A* algorithm has been used before with dynamic link weights. Horn[17] used the A* algorithm with varying travel times as link weights, the travel times were required to be FIFO-consistent. In the problem of this research we will have costs as the link weights, they would also require similar consistency to have A* function properly.

3.2.4 Time-Expanded network

Another method of dealing with a dynamic network is by using a Time-Expanded network [5] [27], or also called Space Time Network [29]. A Time-Expanded network can be created by partitioning a time horizon in a number of discrete periods and duplicate the nodes, such that each node is represented in all time periods. The link traversal times are rounded to a corresponding number of time periods and the given links (i, j) are duplicated and now connect nodes i and j in different time periods. More precisely, links $(i_t, j_{t'})$ are created where i_t denotes node i at time period t, node $j_{t'}$ denotes node j at time t', where $t' - t$ equals the travel time from i to j at time t. Now the Time-Expanded network can be seen as a static graph and a static shortest-path algorithm can be used to find a path in this graph.

A disadvantage of this network propagation is that an already large network gets even larger, as the number of nodes and links get multiplied by the amount of time expansions. This decreases the tractability of the cost minimization problems. Ziliaskopoulos and Wardell [39] confirm that time expansion is computationally expensive for large networks and therefore usually impractical.

3.2.5 Multimodality

Within the domain on freight modelling we may have to deal with different transport modalities and the transfer of freight between these different modalities. For example, in this paper freight has to be assigned on trucks or barge ships.

This is another aspect which was at first challenging for shortest-path algorithms. To solve this challenge, Jourquin [19] defined a 'supernetwork' where virtual links are placed to represent transfers between modes, taking into account loading times or other specific transfer properties. These transfer links were already introduced by Sheffi [35]. A shortest-path algorithm can find a path via several modes in such a supernetwork.

4 Mathematical Model

In this section a mathematical program is formulated. The objective of this program is to find the best solution for the DTA problem. First some required assumptions are described and afterwards the input, to-be-made decisions, constraints and objective function are put into a mathematical framework. This framework is only used to formalize the problem and it is not a model that can be used directly by a solver.

4.1 The problem

The problem considered in this thesis is to find cheapest paths for freight orders from a single origin to several destinations in a given network. For this mathematical formulation paths are denoted as a series of connected links in the transport network, because each link adds some link specific costs to the total path.

Freight can be shipped by two different transport modes, road and waterway. To model the behaviour of these transport modalities some assumptions have been made.

The first assumption (***4.1***) states that trucks from assigned freight orders do not influence the time dependent travel speeds of road links. In general, the travel speed on the road is dependent on all traffic on the road and not only the assigned freight trucks. The main reason for making this assumption is that in the real situation on most roads the fraction of trucks assigned by the considered traffic assignment problem is small. This implies that the rest of the flow (all other traffic) mainly determines the speed of travel.

Not making this assumption would imply that the travel speeds are influenced by the assigned freight. To determine this influence the total flow of the network should be taken into account. Then it would be possible to determine the effect of the assigned freight on the total flow and the consequential travel speeds. To do this would require distributions of network flow through time. However, these are not available as input and modelling the entire network flow would really increase the level of complexity of the model.

The travel speed distributions are considered to be input. These distributions are considered to be based upon some expected amount of traffic on the network. In this expected amount of traffic there is also some expected amount of freight traffic. The better the freight assignment meets these expectation the more realistic the travel speed distributions are, if all other flow behaves accordingly.

Assumption 4.1 *Road links are uncapacitated and travel speeds on roads are not dependent on the assigned freight.*

The next assumption (*4.2*) is made to be able to model the assignment of freight orders onto barge ships. It is assumed that barges sail a predetermined path defined by a predetermined schedule and each barge has a specified capacity. It would be more realistic if the model could also schedule the paths sailed by barges. However, it turns out that this assumption is necessary to be able to develop an efficient solution method (see Chapter 5). Determining the paths sailed by barge ships and the sailing times is considered to be a seperate optimization problem, which will not be treated in this thesis.

Assumption 4.2 *Waterway links are sailed by barges on a predetermined schedule, each barge has a specific route, sailing times and capacity.*

4.2 Input

In this section all input for the considered dynamic freight traffic assignment problem is specified.

4.2.1 Time horizon

The time dimension has to be taken into account, because we are dealing with a dynamic problem. We define T to denote a point in time, t to denote an amount of time and $\mathbb{T}$ a time interval. All T considered in the model will be in the time interval $\mathbb{T}^{horizon}$. This interval will be referred to as the time horizon and is defined by a starting point T^0 and an end point T^{end}: $\mathbb{T}^{horizon} := [T^0, T^{end})$.

For later purpose we define all T to be integer. Subsequently, t defined as the difference between two points in time is also integer.

4.2.2 Network

The freight demand needs to be assigned to paths. These paths should be found in a traffic network. Therefore a traffic network $G(N, A)$ is given, where N is the set of nodes and A is the set of directed links of the network. Each node represents a geographical location and can be either a centroid, a terminal or a junction of links.

- A centroid represents an origin or destination of freight.
- A terminal is a location where freight can change from transport modality, e.g. at a terminal containers can be loaded from a barge onto trucks.
- A junction is a node where several links of the same modality are connected.

Each link $a \in A$ connects two nodes, a link is given by $a = (i, j)$ whereby $i, j \in N$ are the nodes connected by a and the link is directed from i to j. The set of links A is the union of the following three subsets: roads R, inland waterways W and virtual links V. For each link a we have link length L_a, freight flow capacity F_a and costs c_a^F. Virtual links are used to include multimodality in the model, see 3.2.5.

- A virtual link connects the node of a transport modality to a terminal node. These links are used to take into account the time it takes to (un)load a truck or barge and also the capacity and working hours of the terminal.
- Road and waterway links represent the real world roads and waterways.

For later use, for each node we define sets of links that contain all links entering and leaving the specific node. For each node $i \in N$, A_i^{in} contains all links directed towards i and A_i^{out} the links directed out of node i. Formally, $A_i^{in} := \{a \mid a \in A, j \in N, a := (j,i)\}$ and $A_i^{out} := \{a \mid a \in A, j \in N, a := (i,j)\}$.

4.2.3 Truck travel speeds

Assumption *4.1* implies that travel speeds on roads are not influenced by assigned freight. Therefore the travel speeds for trucks on the given network are considered to be given as input. These input travel speeds are time dependent. As $T \in \mathbb{T}^{horizion}$ are integer values these travel speeds are specified by vectors. For every road link $a \in R$ the vector is $v_a = \big(v_a(T^0), v_a(T^0+1), ..., v_a(T^{end}-1)\big)$. Where $v_a(T)$ defines the travel on speed on link a on the interval $[T, T+1)$.

4.2.4 Barge schedule

This section describes how the model can assign freight demand to travel by barge ships. A barge schedule is considered to be a given input by assumption *4.2*. Hereby, the paths sailed by barges, the interarrival times at nodes along the path and the capacity of each barge are given.

A barge can only be loaded and unloaded at a terminal. This means that if a freight shipment is loaded onto a barge at some terminal that this shipment at least stays on this barge until the next terminal that is visited by this barge. For freight shipment only the travel time from terminal to terminal and the capacity of the barge are important.

Therefore the paths sailed by barges are denoted by a vector of connected trips, where each trip defines a path from terminal to terminal with arrival and departure times. These trips should be able to be part of paths assigned to freight demand. As such a path is a series of connected links these trips will be treated as new links. Therefore the given barge paths are considered to consist of links from terminal to terminal with interarrival times. We define input set B with links $a \in B$, where each a defines a link from terminal to terminal with departure time $T_a^{departure}$ and arrival time $T_a^{arrival}$ for a single barge.

This set B will now be considered as part of the traffic network $G(N, A)$ and therefore part of the set of links A.

4.2.5 Link capacities

For virtual links $a \in V$ a remaining capacity $F_a(T)$ is defined for all $T \in \mathbb{T}^{horizon}$ and for each barge link $a \in B$ a capacity of F_a is defined. This barge capacity has no time variance because the capacity is used for the entire link with an already specified time window: $T_a^{departure}$ until $T_a^{arrival}$. Road links have no capacity constraints.

4.2.6 Travel costs

The goal of the assignment is to minimize the transportation costs. The transportation costs are determined by a direct cost factor per link c_a^f, a travel time cost factor per link c_a^t and a waiting time cost factor $c_a^{t,wait}$. All these costs are per unit of freight.

4.2.7 Freight demand

Now only one input for the model is left to describe, which is the freight demand that has to be assigned. The freight demand is given and specified by set[1] OD. Each freight demand $d \in OD$ defines a transport request of F_d units from an origin node Ori_d to a destination node Des_d with release time $T_d^{release}$.

4.3 Mathematical program

In the previous section we have defined the given input. For this input we want to find the cheapest paths for all freight demand $d \in OD$. In the following a mathematical program is formulated, which can be used to solve this cheapest path problem. For this mathematical model, decision variables, constraints and an objective function are defined in the following subsections.

4.3.1 Decision variables

The general goal is to find a cheapest path for each freight order. So for each $d \in OD$ we want to find a series of connected links which form a path P_d from Ori_d to Des_d. We have to determine which links are used for such a path. Another decision to be made is the time schedule for using the path. This means that we have to determine when the freight transport starts and when and how much waiting time to take at nodes along the path.

[1]In (static) traffic modelling it is more common to use a Origin-Destination matrix, however because we are dealing with individual freight requests with specific properties a different notation is needed.

Note, in this model waiting is only possible at terminal nodes and the first node Ori_d is a terminal. In reality it might happen that a barge with oil is ordered to wait along the route, because fluctuating oil prices can make this profitable.

Now we define decision variables which can be used to determine the cheapest path including the required departure times. To assign a specific path consisting of links $a \in A$ to a freight shipment $d \in OD$ we define an indicator variable $\delta_{d,a}$. This variable indicates if a link is in P_d or not. This choice for a decision variable on link level is made to be able to define total costs of paths and define flow constraints. For waiting at a terminal an integer variable waiting time $t_{d,a}^{wait}$ is introduced and for travelling over a link an integer variable for travel time $t_{d,a}$ is introduced for all links $a \in A$ and freight demand $d \in OD$.

$$t_{d,a}, t_{d,a}^{wait} \in \mathbb{Z}^+.$$
$$\delta_{d,a} \in \{0,1\},\ \delta_{d,a} = 1 \quad \Leftrightarrow \quad a \in P_d.$$

4.3.2 Constraints

The constraints are split up in 3 different groups; routing contraints, capacity contraints and time constraints.

4.3.3 Path Constraints

The path constraints have to make sure that for each $d \in D$ all $a \in A$ which have $\delta_{d,a} = 1$ form a path P_d between Ori_d and Des_d. First we have to make sure that the path originates at Ori_d and ends at Des_d. Furthermore, to ensure that the path is connected, a constraint is added to make sure that each node n where a link enters also a link leaves node for $n \neq Ori_d$ and $n \neq Des_d$. This results in the following contraints:

$$\sum_{a \in A_i^{in}} \delta_{d,a} - \sum_{a \in A_i^{out}} \delta_{d,a} = \begin{cases} -1 & i = Ori_d. \\ 0 & \forall i \in N \ /\{Ori_d, Des_d\}. \\ 1 & i = Des_d. \end{cases} \tag{1}$$

It is possible that a path through the network contains a cycle. In a cycle at least one node would have more then one entering link. However all travel times and direct costs are positive. This implies that a cycle will never be in a minimal cost path.

4.3.4 Travel time constraints

As the cost of a freight transport depends on the travel times, we need to know the travel time on links and waiting times before using links. The waiting time has been introduced already as a variable.

As travel times on links are time dependent, we need to know at what time each link is entered. This time of entrance depends on all foregoing travel and waiting times, and the initial release time. To model this, let $t_{d,a}$ denote the travel time for link $a \in A$ and a shipment $d \in D$ and let $T^{in}_{d,a}$ denote the entrance time of link $a \in A$ for shipment $d \in D$. All these variables are integer within the time horizon $\mathbb{T}^{horizon}$ and they are defined by the following equality constraints. Where the entrance time of the next link a is defined as the sum of the entrance time of the previous link, travel time on the previous link and the variable waiting time before entering the link a.

$$T^{in}_{d,a} = \begin{cases} t^{wait}_{d,a} + \delta_{d,a} T^{release}_{d} & \forall a \in A^{out}_{Ori_d}. \\ t^{wait}_{d,a} + \sum_{a' \in A^{in}_{i}} \left(\delta_{d,a'}(T^{in}_{d,a} + t_{d,a})\right) & \forall i \in N/\{Ori_d, Des_d\}, a \in A^{out}_{i}. \\ 0 & \forall a \in A^{out}_{Des_d} \end{cases} \tag{2}$$

These constraints are quadratic, because of the multiplication of variables. It is however possible to rewrite these quadratic equality constraints to linear inequality constraints. To do this we first define a large number M, with $-M < T^0$ and $M > T^{end}$. With this large number M we define linear inequality constraints that ensure all $T^{in}_{d,a}$ satisfy the contraints above. The variable $T^{in}_{d,a}$ should equal zero if the link a is not used by freigth d, equivalent to $\delta_{d,a} = 0$.

This first linear contraint ensures $T^{in}_{d,a}$equals zero if the indicator $\delta_{d,a} = 0$. If $\delta_{d,a} = 1$ then this constraint would allow $T^{in}_{d,a}$ to take any value between 0 and M.

The second and the third inequality contraint ensure that, if $\delta_{d,a} = 1$ then the entrance time $T_{d,a}$ of link a is larger or equal the previous entrance $T^{in}_{d,a'}$ plus travel time $t_{d,a'}$ and waiting time $t^{wait}_{d,a}$. Note that there can only be one $a' \in A^{in}_{i}$ for which $T^{in}_{d,a'}$ is larger than zero, because there can only be one link entering the node i between a' and a. If the indicator variable of link a equals zero then these constraints require $T^{in}_{d,a}$ to be at least larger then $-M$, which is always true because $T^{in}_{d,a} \in \mathbb{T}^{horizion}$.

The fourth constraint below is required to ensure the equality of the contraints above. $T^{in}_{d,a}$ should be equal to the $T^{in}_{d,a'}$ added with $t^{wait}_{d,a'}$ and $t_{d,a'}$ of predecessing link a'. This constraint is added to ensure the equality for links a and a', leaving and entering some node i.

$$\begin{aligned} T^{in}_{d,a} &\leq \delta_{d,a} M \\ T^{in}_{d,a} &\geq t^{wait}_{d,a} + T^{release}_{d} - (1 - \delta_{d,a})M & \forall a \in A^{out}_{Ori_d}. \\ T^{in}_{d,a} &\geq t^{wait}_{d,a} + T^{in}_{d,a'} + t_{d,a'} - (1 - \delta_{d,a})M & \forall i \in N, a \in A^{out}_{i}, a' \in A^{in}_{i}. \\ t^{wait}_{d,a} &\geq T^{in}_{d,a} - T^{in}_{d,a'} + t_{d,a'} - (2 - \delta_{d,a} - \delta_{d,a'})M & \forall i \in N, a \in A^{out}_{i}, a' \in A^{in}_{i}. \end{aligned} \tag{3}$$

Following constraints define the travel times $t_{d,a}$ for the different modalities. The decision variables $\delta_{d,a}$ return in these constraints, because the travel time for unused links should be equal to zero. As links that are not used should not influence the objective function.

Travel time on virtual links

Let α_a^t denote the time to transfer one unit of freight via link a and let β_a be the basic handling time for a freight shipment. Then the travel on the virtual link is defined as.

$$t_{d,a} = \delta_{d,a}\big(\alpha_a^t F_d + \beta_a\big) \qquad \forall a \in V. \tag{4}$$

Furthermore, terminals have specific operating hours, during these hours of the day containers can be processed. These operating hours are modelled in the capacity contraints, which will be specified later.

Travel time on roads

The time dependent travel speeds v_a on all road links $a \in R$ are given. Ichoua [18] defines a method to deal with these varying travel speeds in a discrete time setting. This method puts a constraint on the travel time $t_{d,a}$ for $a \in R$. The constraint makes sure that enough time is spend on road link a to travel the total distance L_a with travel speed from vector v_a.

Initially with Ichoua's method the travelled distance on road link a is defined by summing the travelled distance in discrete time periods τ_i. These periods are defined by a an amount of time $\Delta\tau$: $\tau_i = [T_i^0, T_i^0 + \Delta\tau)$, $\tau_{i+1} = [T_{i+1}^0, T_{i+1}^0 + \Delta\tau) = [T_i^0 + \Delta\tau, T_i^0 + 2\Delta\tau)$. The amount of time $\Delta\tau$ and the travel speed during period i being $v(T_i^0)$ imply the travelled distance in interval τ_i equals $\Delta\tau v(T_i^0)$. The next equation defines the constraint where the summation over periods τ_i should ensure that the total link length L_a is travelled.

$$\sum_{\tau_i} \big(\Delta\tau v_a(T_i^0)\big) \geq L_a \qquad \forall a \in R. \tag{5}$$

Now we make a constraint from this inequality. The travel times in the assignment problem are considered to be integer, therefore the width of the time periods $\Delta\tau$ is considered to equal 1. The first time period of a freight shipment $d \in OD$ on link a therefore starts at $T_{d,a}^{in}$ and the travel time $t_{d,a}$ determines the upperbound of the summation. This upperbound will be $T_{d,a}^{in} + t_{d,a}$. The constraints should only set a travel time for links in the path P_d. Therefore the distance to be travelled will equal the link length L_a multiplied with $\delta_{d,a}$. The resulting constraint for the mathematical program is.

$$\sum_{t=T_{d,a}^{in}}^{T_{d,a}^{in}+t_{d,a}} \big(v_a(t)\big) \geq \delta_{d,a} L_a \qquad \forall a \in R. \tag{6}$$

As the given travel speeds are all positive, there always exists a minimal $t^*_{d,a}$ for each d and $a \in P_d$ that satisfy this contraint for given value of $T^{in}_{d,a}$. All travel speeds being positive implies that the constraint is also satisfied for all $t_{d,a} \geq t^*_{d,a}$. However, minimizing the travel time costs is part of the objective, which implies that the model favors the minimal $t^*_{d,a}$.

Barge travel times

The set of barge link B has been added to the network. Each barge link $a \in B$ has a fixed arrival $T_a^{arrival}$ and departure time $T_a^{departure}$. The resulting constraint for the travel time on barge links $a \in B$ is:

$$t_{d,a} = \delta_{d,a}(T_a^{arrival} - T_a^{departure}) \qquad \forall a \in B. \tag{7}$$

4.3.5 Capacity constraints

The capacity contraints have to ensure that the amount of flow on a link does not exceed the input boundaries F_a for $a \in B$ and $F_a(T)$ for $a \in V$. Assigning a path to a single freight shipment $d \in OD$ requires that at least F_d capacity is available on the links. This means that the following constraints have to be satisfied.

$$\begin{aligned} &\delta_{d,a}F_d \leq F_a(T) && \forall a \in V,\ T \in [T_a^{in}, T_a^{in} + t_{d,a}). \\ &\delta^{d,a}F_d \leq F_a && \forall a \in B. \end{aligned} \tag{8}$$

4.3.6 The objective

Now that we have defined all variables and constraints, what is left to do is to define an objective. First an objective function for a single freight demand problem is formulated.

For single freight demand $d \in OD$ the goal is to find a cheapest path P_d. This means that the costs of the resulting path should be minimal, where the total cost for a path is the sum of direct costs and time dependent costs for using links. We have the cost factors $c_a^f, c_a^t, c_a^{t,wait}$. Time variables $t_{d,a}$ and $t_{d,a}^{wait}$ denote the time travelled and waiting time on link a for freight d. Freight demand d has shipment size F_d and the path will be defined by decisions variables $\delta_{d,a}$ and $t_{d,a}^{wait}$.

The objective function is below this paragraph, the following steps lead to this function. All cost factors are defined per unit of freight, so therefore the sum representing the path costs for 1 unit of freight is multiplied with F_d. The sum consists of direct link costs and time dependent link costs. The direct link costs c_a^f have to be paid if the link is part of the path, equivalent to $\delta_{d,a} = 1$. The time dependent link costs c_a^t are multiplied with the time the freight shipment is on link a, $t_{d,a}$, and the waiting time costs $c_a^{t,wait}$ with the waiting time $t_{d,a}^{wait}$. Now the goal is to find $\delta_{d,a}$ and $t_{d,a}^{wait}$ that minimize the total path costs for $d \in OD$.

$$\min_{\delta_{d,a},t_{d,a}^{wait}} \qquad F_d\big(\sum_{a\in A} \delta_{d,a}c_a^f + t_{d,a}c_a^t + t_{d,a}^{wait}c_a^{t,wait}\big) \qquad d \in OD. \tag{9}$$

With this objective function and all given constrains a complete mathematical program is formulated. The program with this objective function for a single freight demand can be found in appendix A. Note, that current formulation of the program is not linear because the constraints with summations depending on the decision variables. The solution of this program would be the cheapest path for a single freight demand $d \in OD$, therefore the problem that can solved with this program will be referred to as the Single Cheapest-path Problem (SCP).

4.3.7 Objective for a System Optimum

Solving the program with objective function (9) would result in a cheapest path for the specific demand $d \in OD$. However, there are more freight demand requests in the set OD and we are interested in a joint best solution. This leads to a different mathematical program, where the objective function minimizes the total costs for all freight demand in the set OD and the constraints have to be formulated such that all paths for all demand remain feasible.

This situation where the total cost of the system is minimized is called the System Optimum (SO) (see also section 3.1.2). The objective would be to minimize the total costs of all paths simultaneously. The objective function would become the following summation of the path costs of all freight demand $d \in OD$.

$$\min_{\delta_{d,a},t_{d,a}^{wait}} \qquad \sum_{d\in OD}\Big(F_d\big(\sum_{a\in A} \delta_{d,a}c_a^f + t_{d,a}c_a^t + t_{d,a}^{wait}c_a^{t,wait}\big)\Big). \tag{10}$$

A mathematical program for this SO problem would be more be difficult to solve. The capacity constraints would have to be reformulated such that the sum of assigned freight to a link does not exceed the available capacity. Having to solve a problem with many more decision variables and the choice of which freight demand should be assigned to which available capacity significantly increases the complexity of the problem. Therefore reformulating to a program to find a SO does not seem to be the proper choice in our setting.

4.3.8 Objective for the Dynamic Traffic Assignment

The total cost minimization of a SO does not imply that an individual freight demand $d \in OD$ gets its cheapest path. That is because the limited capacities for terminals and barges can cause some freight demand to get assigned a more expensive path, to allow other freight demand to use that capacity and leading to the resulting total system costs being minimal.

In reality it is likely that each freight demand wants to get its cheapest path. It might be that several freight shipments belong to a single company which would like to minimize its total costs, however we consider all freight demand $d \in OD$ to be individual. We can solve the SCP for each freight demand $d \in OD$, however it is possible that in this case more freight will be assigned to a barge or terminal than there is available capacity.

This problem can be overcome by updating the remaining capacities after assigning a path to a freight shipment. After updating the capacities a next freight shipment can be assigned and subsequently remaining capacities can be updated. This would make sure that each assigned path has enough available capacity. However, for this approach the demands are assigned in some order. This order has an impact on the total costs and the costs for the individual demands. This problem of iteratively finding cheapest paths will be referred to as the Iterative Cheapest-path Problem(ICP).

The assigning of freight shipments one by one can be seen as a process of iteratively solving the individual mathematical program. The idea of iteratively finding a solution for the DTA problem is discussed further in Chapter 5, also the order of iteration will be discussed in that chapter. The used method does not directly find a solution to the mathematical program with some solver, but a different approach is chosen. As mentioned before, the mathematical program formulation in this chapter has been made to formalize the DTA problem.

5 Approach

The DTA problem has been formalized by a mathematical program formulation in the previous chapter. However, it has been decided to not solve the mathematical program to find a solution to the Iterative Cheapest-path Problem (ICP). The reason for this is that within the implementation no solver for a mathematical program should be used and also that such a solver might use to much computational time. This chapter describes how a different approach is used to find a solution for the DTA problem and also the downsides of this approach are discussed.

5.1 Shortest-path problem

As mentioned before the ICP consists of a series of SCP. Therefore we first describe how we approach a SCP. As described in section 3.2 it is possible to use the shortest-path algorithm A* to find fastest paths on a network with dynamic travel times, if the travel times over links are FIFO-consistent. This means we can solve a SCP with A* if the costs per link are defined as the link weights and if these costs have a consistency similar to FIFO. This similar consistency will be called Cheapest-In Cheapest-Out(CICO).

A classical shortest-path problem can be solved by using the A* algorithm. We show that the SCP in our case is not a classiscal shortest-path problem and this troubles the solutions found with A*. SCP differs from a classical shortest-path problem because the costs we want to minimize depend on time and we also want include the possibility of waiting in the paths. The waiting time problem and a solution will discussed further on in this section. We now first formalize the definition of FIFO, than take a look at how to determine the costs per link and subsequently check if these costs satisfy CICO.

5.1.1 FIFO

The FIFO-consistency for links in the network implies that when a vehicle, that enters a link a at time T, cannot leave the link earlier than any vehicle that entered the link at a time prior to T. To formalize this we introduce the arrival time function $AT_a(T)$ depending on time of entrance T, where $AT_a(T)$ defines the time of arriving at the end node of link a (equivalent to the time of leaving link a) when the link is entered at time T.

The formal definition of FIFO is that the function $AT_a(T)$ is a monotonically increasing function. Monotonically increasing means that AT_a can only increase if the time of entrance T increases. In the following we check if a similar monotonic function exists for the costs. Therefore we will first define the costs per link and show that some conditions are required to have a monotonic total cost function.

5.1.2 Costs per link

In Chapter 4 the following cost minimizing objective function (9) was defined for a SCP, where the total costs of a path are defined by the sum of the costs per link.

$$\min_{\delta_{d,a}, t_{d,a}^{wait}} \qquad F_d\big(\sum_{a \in A} \delta_{d,a} c_a^f + t_{d,a} c_a^t + t_{d,a}^{wait} c_a^{t,wait}\big) \qquad d \in OD.$$

This implies that the costs C_a per link a which can be used as link weights are defined below. Note that we left out the indicator variable, because we will only take into account the cost of used links. We also leave out c_a^f, because all costs in the implementation will be travel time related. Freight size, travel time and waiting time are now input variables that determine the cost for using a link. How these variables are determined will be discussed later, only note that $t_{d,a}(T)$ is dependent on the time of entrance T of link a.

$$C_a(t_{d,a}, t_{d,a}^{wait}, F_d, T) = F_d\Big(t_{d,a}(T) c_a^t + t_{d,a}^{wait} c_a^{t,wait}\Big). \tag{11}$$

Now we first look at the situation where no waiting occurs, because a classical shortest-path problem does not include the possibility of waiting. Also let w.l.o.g. $F_d = 1$. Then we have:

$$C_a(t_{d,a}, T) = t_{d,a}(T) c_a^t.$$

For an individual link a we define total cost $TC_a(IC, T)$ after using this link with initial costs IC and entrance time T. The initial costs are all costs that are made before entering link a.

$$TC_a(IC, T) = IC + t_{d,aT}(T) c_a^t. \tag{12}$$

For TC_a to be CICO-consistent we require that TC_a can only increase if IC increases. However, TC_a is not only dependent on IC but also on T. Therefore TC_a is not CICO-consistent. This implies that it is not guaranteed that A* can find an optimal solution to SCP.

This situation where we want to minimize costs, but have to deal with the time dimension causes our SCP to be different from a classical shortest-path problem. However, if we look at the SCP with only one transport modality and therefore only cost factor, then we can draw a different conclusion. This is shown in the next subsection.

5.1.3 Single modality SCP

We show that in case we have only transport modality that the total costs are CICO-consistent. To do this we define a path P of length n which denotes a series of connected links upto some link a_n, $P = \{a_1, ..., a_n\}$. The initial cost IC for a link a_i are replaced with the total cost $TC_{a_{i-1}}$ of the predecessing link a_{i-1}. Let vector T contain T_i defining the time of entrance of link a_i. Now we rewrite equation (12) with IC of a_1 equal to zero:

$$\begin{aligned} TC_{a_1}(T_1) &= t_{d,a_1}(T_i)c^t_{a_1}. \\ TC_{a_i}(TC_{a_{i-1}}, T_i) &= TC_{a_{i-1}} + t_{d,a_i}(T_i)c^t_{a_i}. \qquad i = 2, ..., n \end{aligned}$$

This means that we can calculate the total costs with the following sum if P, T, $t_{d,a}$ and c^t_a are known.

$$TC_{a_n}(T) = \sum_{k=1}^{n} t_{d,a_k}(T_k)c^t_{a_k} \tag{13}$$

In the situation of only a single modality there is only one time cost factor c^t_a and thus we have a constant $c = c^t_a$ for all $a \in A$. With t being the total travel time makes it even easier to determine the total travel cost of P.

$$\begin{aligned} t &= \sum_{k=1}^{n} t_{d,a_k}(T_k) \\ TC_{a_n}(T) &= \sum_{k=1}^{n} t_{d,a_k}(T_k)c = tc \end{aligned} \tag{14}$$

This equation shows that in the case of one modality the cheapest path is equal to the fastest path, because TC_a is minimal if t is minimal. As mentioned before determining the fastest path in a FIFO-consistent network is a classical shortest-path problem. This implies that A* can find the cheapest path for a single modality SCP.

This result allows us to use A* to determine the cheapest path for SCP if we have only one modality. However, in our setting SCP deals with more modalities, but we are going to use A* to find a solution. Although it is not guaranteed that we find an optimal path. In the remainder of this thesis we show that A* provides us with decent cheap paths for the multimodal SCP and ICP, but before doing this we take a look at the A* algorithm.

5.2 A* algorithm

A general description of the A* algorithm can be found in section 3.2.1. In this section we will present the pseudocode of the used algorithm. First we will pay attention to the heuristic function that is required for A*.

5.2.1 Heuristic function

To use A* algorithm a heuristic function that estimates the cost between two nodes is required, this cost estimation is used to direct the search towards the destination node.

It is required that the heuristic function is consistent and admissible and the link costs need to be positive for all links for the algorithm. If a monotonic(or consistent) function is used then the algorithm does not have to re-evaluate nodes. The heuristic is monotonic if the following equations are satisfied. Let h denote the heuristic function, $cost$ respresent the travel real travel costs between 2 nodes and let i, j, k be nodes in the network. Then the heuristic function must satisfy the following equations for all $i, j, k \in N$ to be monotonic.

$$\begin{aligned} &h(i,k) \leq cost(i,j) + h(j,k). \\ &h(k,k) = 0. \end{aligned} \tag{15}$$

In other words the heuristic function should not overestimate the real travel costs between any two nodes. The real travel costs are dependent upon the travel time and an amount of freight to be shipped. In the following A* algorithm the freight will be taken into account in the heuristic function, but the time dependence will be left out. The time dependence is left out to ensure the function never overestimates the real cost. The function is referred to as **heur**($startnode, goalnode, freight$), where $freight$ denotes an amount of freight.

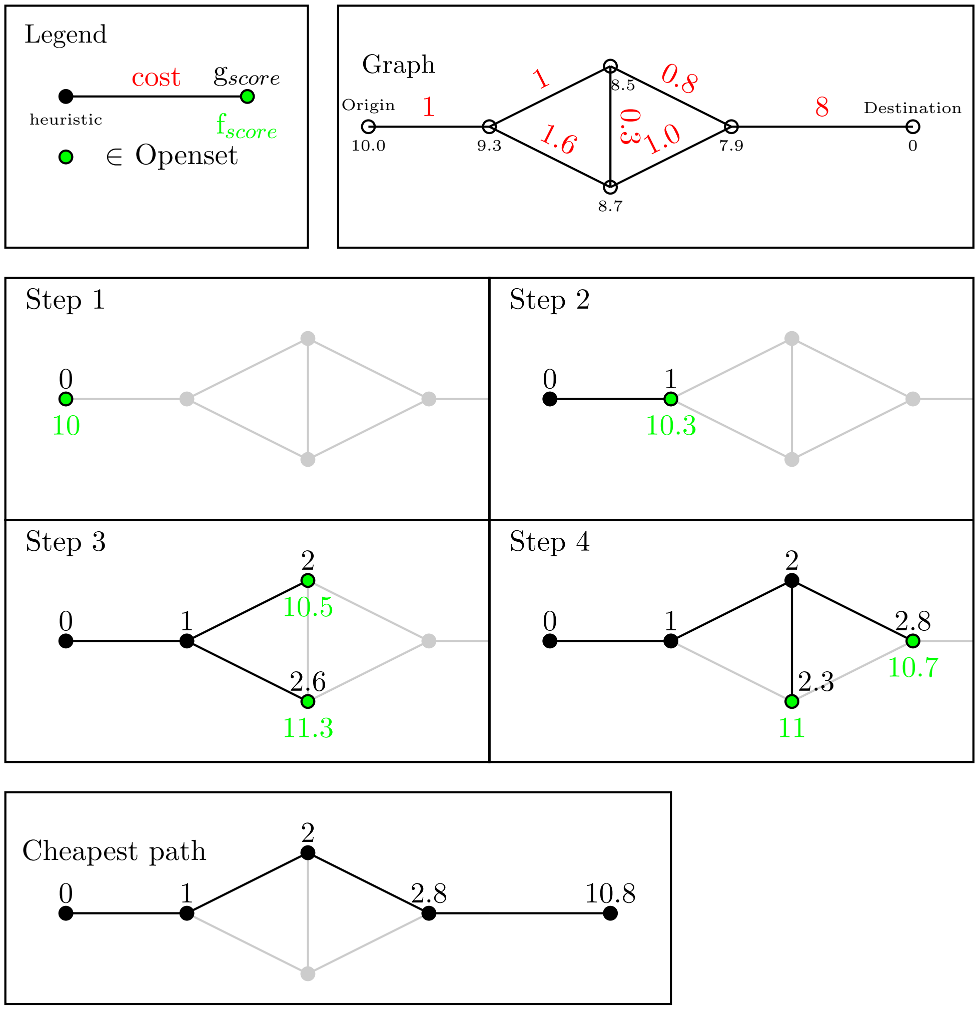
5.2.2 Basic A* in pseudocode

Below we have the used A* algorithm in pseudocode. The function *trans* determines the total travel time $\hat{t}_{time}$ and total travel cost $\hat{g}_{score}$. $\hat{t}_{time}$ is the total travel time after using the link defined by $(currrentnode, neighbor)$ with entrance time t_{time}. $\hat{g}_{score}$ is the total cost after travelling via that link between t_{time} and $\hat{t}_{time}$ with initial cost g_{score}.

```
1.  A*$\big(startnode, goalnode, timestart, freight\big)$
2.      Openset := $\{startnode\}$
3.      $\mathrm{g}_{score}[startnode] := 0$
4.      $\mathrm{t}_{time}[startnode] := timestart$
5.      $\mathrm{f}_{score}[startnode] := \mathrm{g}_{score}[startnode] + \mathrm{heur}(startnode, goalnode, freight)$
6.      While Openset $\neq \emptyset$ do
7.        $currentnode := \mathrm{argmin}_{n \in \mathrm{Openset}} f_{score}[n]$
8.        remove $currentnode$ from Openset
9.        If $currentnode = goalnode$ then do
10.           return ( $\mathrm{path}(startnode, goalnode, came\_from, t_{time})$)
11.       Else do
12.         For $neighbour \in \mathrm{Neighbours}[currentnode]$ do
13.           $\{\hat{g}_{score}, \hat{t}_{time}\} := \mathrm{trans}\big(currentnode, neighbour, \mathrm{g}_{score}, \mathrm{t}_{time}, freight\big)$
14.           If $neighbour \notin$ Openset or $\hat{g}_{score} < \mathrm{g}_{score}[neighbour]$ then do
15.             $\mathrm{g}_{score}[neighbour] = \hat{g}_{score}$
16.             $\mathrm{t}_{time}[neighbour] = \hat{t}_{time}$
17.             $\mathrm{f}_{score}[neighbour] = \hat{g}_{score} + \mathrm{heur}(currentnode, goalnode, freight)$
18.             $came\_from[neighbour] = currentnode$
19.             make sure $neighbour$ is in Openset
20.           end
21.         end
22.       end
23.     end
24. return ( failure)
```

This is the basic A* for solving shortest-path problems. So there is no waiting allowed and if the algorithm finishes succesfully then a cheap path will have been found, but we cannot be sure it is the cheapest. In the following we will describe in what situation of the SCP, A* can produce an optimal path and how to check if the resulting path is optimal.

Figure 1: Example of functioning of A*



5.3 Non-classical shortest-path problem

The main reason why A* does not guarantee an optimal path is that the costs per link are not CICO-consistent. This causes a problem for A*'s method of constructing a shortest-path tree.

The construction of this tree requires that a shortest path to a node n is also the start of the shortest path to all subsequent nodes in the tree. If the costs are CICO-consistent then that would be true, but the costs are not CICO-consistent. Therefore it is possible that there is a different path to node n which is a more expensive way to get to node n but it results in a cheaper path to a subsequent node. The following example describes the situation where the construction of the tree in A* can loose the shortest path.

Example 5.1
We want to solve a SCP for freight demand $d = (origin, destination, T_d^{release}, F_d)$ with A. Let some path **P** from the origin to some node **n** be in the tree constructed by the A* algorithm. Let **P'** $\neq$ **P** be a different path from the origin to node **n**, which has an earlier arrival time at node **n** but is more expensive then **P**.*

*Let the arrival of path **P** at node **n** be at time **T** and the arrival of **P'** at node **n** at time **T'**, **T'** $<$ **T**. Let the travel cost from node **n** to the destination, when leaving at time **T**, be higher then when leaving at time **T'**. This can be the case when a traffic jam occurs on a road link between node **n** and the destination at such a time that departure at time **T** causes the freight shipmet to encounter the traffic jam, but the traffic jam will not yet be there when departure is done at time **T'**.*

*In this situation it can happen that a path from the origin to the destination via node **n** with path **P'** is cheaper then travelling with path **P**. However node **n** was already in the tree of A* via path **P**, meaning that path **P'** will not be found by A* and A* would not produce an optimal path.*

In section 5.1.3 we stated that A* can be used to find the cheapest path on a single modality problem. In our setting of the SCP there are two main modalities: truck and barge. All freight originates at Rotterdam and also all barges have a departure at this location. This causes that the paths generally consist of only a path by truck or of first a path by barge and then a path by truck.

Lets assume we found a path using A*. In case the resulting path only travels by truck there is a single modality. Thus we can be sure it is an optimal path for the situation where only trucks can be used. Note that the freight shipment would have to pass one virtual terminal link to get onto the truck modality from the origin, but the path from the end of the virtual link to the destination is the optimal truck route and the virtual link is necessary to get out of the origin. It can also be that the resulting path is partly by barge and partly by truck. For all parts we can be sure that it is the cheapest path with the specific modality, but we cannot be sure that the entire path is the cheapest solution.

If it is not the cheapest then there must be some node in the tree constructed by A* where the cheapest path is lost. This can happen as described in Example 5.1 or a similar situation. These situations are quite specific, because the point of loosing the optimal path in the construction of the A* tree has at least the following two requirements. The first requirement is that there are branches in the tree which reach to the same node with different cost and different travel time. These differences leading to the loss of optimality can only be caused by branches using different modalities. The other requirement is that there must be a significant cost difference, thus a significant travel time difference between the node and the destination.

The first requirement can only be met by a limited amount of nodes where these branches using different modalities cross. The second requirement is not likely to be met at these nodes, because most destinations are in the hinterland where there is less traffic congestion, which would be the main cause of a significant travel time difference. These reasons create the situation of A* not finding an optimal path quite specific. Therefore we choose to use A* algorithm to find a cheap or even the cheapest path for SCP. To ensure that the resulting paths from the A* path do not differ to much from the cheapest paths a checking method was implemented. This method is described in the next section and it checks if a possible better path is lost during tree construction.

5.3.1 Check for suboptimality

To keep an eye on the possibility of the occurrance of A* finding a suboptimal path, checking algorithm **A*Check** was implemented.

First we take the following notation from the pseudocode of A*: $g_{score}(n)$ denotes the cost for travelling from origin to node n via the A* tree. $g_{time}(n)$ denotes the travel time of travelling to node n via the A* tree. For nodes a, b, time T and freight size F, let $cost(a, b, T, F)$ and $time(a, b, T, F)$ represent the cost and the travel time for travelling from node a to node b with departure time T and freight size F. These functions are part of the earlier mentioned function $trans(a, b, g_{score}, T, F)$. In words the checking algorithm does the following: Let Z_d be the tree constructed by A* for $d \in OD$, for each node n in Z_d, for all neighbors of n check if a cheaper path could exist. Let x be a neighbor of n. A cheaper path can exist via x if $t_{time}(x) + time(x, n, g_{time}(x), F_d) < t_{time}(n)$ and if $g_{score}(x) < g_{score}(D_d)$. If both these conditions hold compare the costs of travelling via x to the goal node with release time $t_{time}(x)$ to path initially found by A*.

```
1.  A*Check($P_d, Z_d, g_{score}, t_{time}$)
2.      $\forall n \in Z_d,\ \forall a \in A_n^{in},\ a = (i,n)$
3.       If $\big(\ t_{time}(i) + time(i,n,t_{time}(i),F_d) < t_{time}(n)$ and
4.          $g_{score}(i) < g_{score}(Des_d)\ \big)$ then do
5.              $cost(i, Des_d, t_{time}(i), F_d) := $A*$\big(\{i, Des_d, t_{time}(i), F_d\}, G(N,A)\big)$
6.              If $g_{score}(i) + cost(i, Des_d, t_{time}(i), F_d) \leq g_{score}(Des_d)$ then do
7.                  $\hat{g}(i) = g_{score}(i) + cost(i, Des_d, t_{time}(i), F_d)$
8.              end
9.      end
10. return($\hat{g}$)
```

In the verification section in Chapter 6 we show that this algorithm can find path improvements on paths resulting from the A* algorithm. This result gives us an indication in the difference between the paths resulting from A* and the possible cheaper paths. In the following section we take a look at the other issue that makes SCP a non-classical shortest-path problem, which is the waiting time at terminals.

5.4 Waiting-time expansion

In the model in Chapter 4 a variable was integrated to represent waiting time at a terminal. The A* algorithm is not able let a freight shipment wait at a terminal. This section describes a method to make A* find a path including waiting times.

To incorporate the variable waiting time at a terminal, the idea of time expansion is used. The idea of a Time-Expanded network is described in section 3.2.4. Instead of doing an expansion of the entire network only expansion is done on the tree made by A*. The algorithm is modified such that the tree can have several branches to the same node in different time expansions. Each time expansion represents a period of waiting time at a terminal. This allows a single node to occur on several places in the cheapest path tree. The different branches to that same node indicate a path with a different number of waiting time periods at specific terminals.

The tree constructed by A* must be modified such that it expands with a number of waiting-time expansions at each terminal. In the next paragraph this is explained in combination with the basic method of A*, subsequently a pseudocode addition for the A* algorithm is given. An illustration of the process can be found in the appendix in Figure 10 which is a time-expanded version of Figure 1.

A* adds a node n to the tree if it has the lowest possible score of a set of reachable nodes. The neighbours of this node n will then be added to the set of reachable nodes. If it is possible to wait at a terminal, then for each waiting period the neighbor node, with a time expansion index, is added to the set of reachable nodes. Then in the next step it is possible that a time expansion indexed node is added to the shortest-path tree. Finally when A* reaches the goal node it is possible that the constructed path includes a waiting period at a terminal.

Pseudocode of the time-expansion method

Let *expansion number* be the possible number of waiting periods at a terminal and let *maxexp* be a predefined maximum number of expansions for each branch of the tree. The pseudocode of the method is added to the peudocode of A*, the line numbers originate from the same pseudocode in section 5.2.2.

```
7.    $\{currentnode, currentexp\} := \text{argmin}_{n \in \text{Openset}, exp \in [1, maxexp]} f_{score}[n, exp]$
12.     For $neighbour \in$ Neighbours[$currentnode$] do
          For $expand \in$ [1 to $expansion\ number$] do
             $exp = currentexp + expand$
             If $exp \leq maxexp$ do
13.             $\{\hat{g}_{score}, \hat{t}_{time}\}$ :=trans($currentnode, neighbour$,$\text{g}_{score}$,$\text{t}_{time}$,$freight$,$exp$)
14.             If $neighbour \notin$ Openset or $\hat{g}_{score} < \text{g}_{score}[neighbour, exp]$ then do
15.                $\text{g}_{score}[neighbour, exp] = \hat{g}_{score}$
16.                $\text{t}_{time}[neighbour, exp] = \hat{t}_{time}$
17.                $\text{f}_{score}[neighbour, exp] = \hat{g}_{score}$+heur($currentnode, goalnode, freight$)
18.                $came\_from[neighbour, exp] = \{currentnode, currentexp\}$
19.                make sure $neighbour$ is in Openset
20.             end
             Else go to next $neighbour$
          end
21.     end
```

Adding this method of time expansion to A* implies that for each terminal the number of nodes in the tree can get k times larger, where k is the number of waiting periods possible at a terminal. If there are many terminals this implies that the size of the cheapest path tree can expand rapidly, which strains the computation time of the algorithm. The amount of expansions can be bounded by *maxexp*, to prevent the tree becoming too large. The larger the tree the more computations are required before A* finishes.

5.5 ICPA*

In the previous section the approach of solving the SCP with waiting at terminals and the possibility of A* not finding an optimal path is described. However, our main goal is to solve the ICP to find a solution to the DTA problem.

As described in Chapter 4 we do this by iteratively solving SCP and in between the iterations add the load resulting from the solution of SCP onto the network. This means that after each iteration some available capacity of the network is claimed. It remains to determine the order in which to solve the SCP, this will be discussed furtheron in this section. First we give pseudocode of the used ICP solving algorithm, this algorithm will be called ICPA*. For this, let σ define the order in which the SCP will be solved. The function *Load* defines the capacities that are required for path P_d with freight F_d; the load that is required for the network update.

5.5.1 Pseudocode of ICPA*

```
1. ICPA*$\big(G(N,A), OD, \sigma)\big)$
2.     For $d \in \sigma(OD)$ do
3.         $P_d := \mathbf{A*}\big(\{O_d, D_d, T_d^{release}, F_d\}, G(N,A)\big)$
4.         Update network: $G(N,A) = G(N,A) + Load(P_d, F_d)$
5.     end
6.  return($\mathbf{P}$)
```

5.5.2 Order of assigning

Note that there are limited capacities in the ICP. Updating these capacities after assigning freight shipments causes that later assigned shipments cannot use this capacity. This means that earlier solved SCP can choose from more feasible paths and can choose the cheapest feasible paths. Therefore, the order of assigning σ influences the cost for individual shipments and therefore also the total cost. Thus different orders of iteration can result in a different total cost.

One possible policy that could be used to define an order is First-Come First-Served(FCFS), which might be seen as most fair method as the earliest in line is first to claim capacity. However, to make efficient use of capacities it might be wise to first assign large batches of demand or if you want to reduce waiting times for limited capacity barges it might be wise to assign with Last-Come First-Served(LCFS) policy. The effects of different policies will be discussed in Chapter 6.

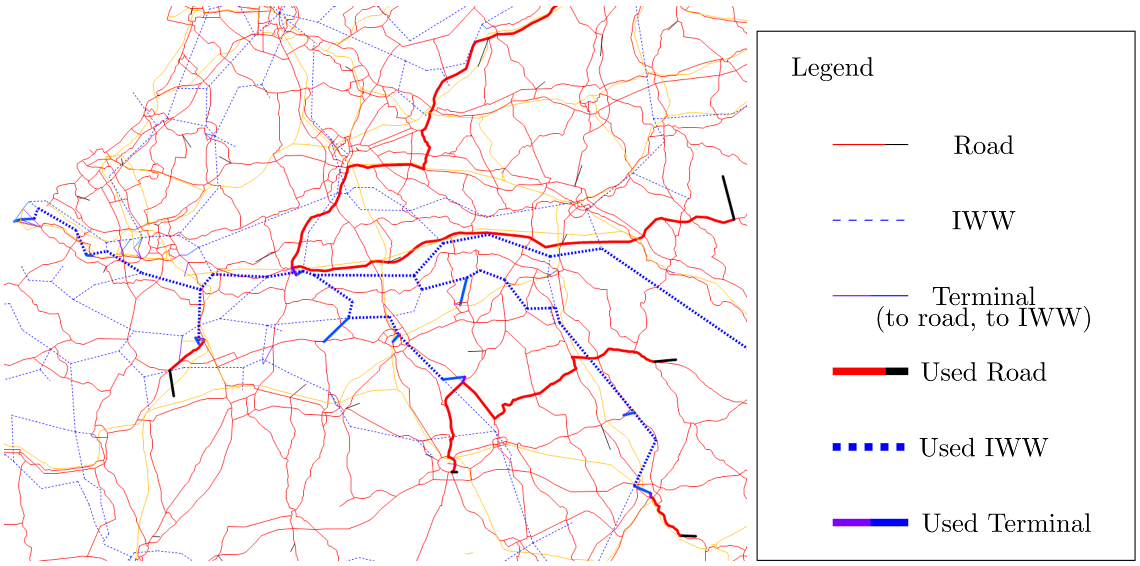
6 Implementation

This section deals with the implementation of the ICPA* algorithm in TransCAD and its verification. Next to analyzing if the algorithm does what it should, calibration of parameters is done. No validation of results will be done as not enough data is available to do a significant validation.

6.1 TransCAD

The ICPA* algorithm has been implemented in the developers environment of TransCAD. TransCAD is created by the Caliper Cooperation. It is an extensive software system that can be used for all kinds of transportation problems. TransCAD supports transportation and route optimization models and includes tools for creating and displaying traffic networks. An example of a route map in TransCAD can be seen in Figure 2. Next to the implemented functions and tools it is possible to create macros in the Geographic Informations System Developer's Kit(GISDK). With these macros it is possible to program algorithms that use the network structures, network data and tools of TransCAD.

Figure 2: Example of TransCAD routemap



6.2 ICPA* implementation

For the implementation of the ICPA* algorithm we first define the setting and requirements. For the A* algorithm we need a network, the barges, the heuristic function and the costs per link. The costs per link are time and modality dependent and the calculation of the travel time for the different modalities is also discussed. Next to this setting for the A* algorithm we also require freight demand requests to assign to the network.

6.2.1 Freight demand

A set OD with all freight requests is required. No real freight traffic information is available, but a set OD can be randomly generated. A random generator is developed by Zhang [38] and was made to create a set of freight demand that represents a day of container freight originating from Rotterdam. The random generation is based upon data from Panteia and TNO [26].

The generator creates a realistic day of freight demand if the created set has a size around 1000 freight demand requests. It is also possible to use the random generator to create a smaller OD set with less requests. Such a smaller set will be used for the verification and calibration. For each individual $d \in OD$ a cheap path will be generated with the A* algorithm.

6.2.2 A* implementation

For the implementation of the A* algorithm we need to define the heuristic function and the cost function. Pseudocode of the A* algorithm can be found in section 5.2.2.

The heuristic function

In the pseudocode of A* we mentioned the function **heur**($startnode, goalnode, freight$). This function needs to make an estimation for the costs of transporting $freight$ from a *startnode* to a *goalnode*, but it should not overestimate the real costs.

The estimation is therefore based upon the costs of a cheapest path in a static situation. Determining these costs for a static situation can be done faster than for a the dynamic situation. We choose the following safe and fast method to ensure that the estimation is not higher than the real costs.

The shortest-path function of TransCAD is used to determine these cheapest paths in a static situation, where the costs for travelling along a link are determined by travelling with maximum speed. This maximum is the speed limit set for the specific road link, or maximal sailing speed in case of barges. The speed being maximal implies that travel times are minimal and therefore the time related costs are never overestimated by this heuristic. Note that this shortest-path function of TransCAD does not take into account vehicles or capacities, this implies that there are no transfer costs or waiting costs in the estimation. .

Costs per link

Another requirement for using A* is defining the weights per link, where the weights represents the costs. From Chapter 5 we have the following cost function (11) for defining a cost per link a.

$$C_a(t_{d,a}, t_{d,a}^{wait}, F_d, T) = F_d\Big(t_{d,a}(T)c_a^t + t_{d,a}^{wait}c_a^{t,wait}\Big),$$

The value of the different cost factors can be found in the next paragraph. How the travel times are determined is explained furtheron in this section.

Costs of transport

Costs of the different transport modalities and other transport costs per minute can be found in Table 1 below. The costs are gained from several reports on freight transport in the Netherlands [25] [30] [31]. Costs for loading and unloading a barge is around €33 per TEU [25]. A loading time of 3 minutes per TEU results in a cost of €11 per minute for handling a TEU. Note that Table 1 contains a containercost c_d, this time related cost represents the costs of renting shipping container. This cost makes every minute of travel or waiting from origin to destination more expensive.

Table 1: Costs in Euro per TEU per minute for the different transportation modalities.

Specification	Notation	Euro/(TEU·minute)
Truck	$c_a^t, a \in R$	€0.86
Barge ship	$c_a^t, a \in W$	€0.015
Waiting cost	$c_a^{t,wait}, \forall a$	€0.005
Handling cost	$c_a^t, a \in V$	€11
Container cost	c_d	€0.055

6.2.3 The network

The next element we need for determining cheapest paths is a network $G(N, A)$. The main structure of the road and inlandwaterway network is from TransTools, ECJRCIPTS(2010) Trans-Tools V2. Destinations and terminals for freight transport are added to the network by Zhang [38]. For this network we have to define the travel speed for road links, extend the network with a barge schedule and determine the capacities of the virtual terminal links.

6.2.4 Road travel speeds

Rijkswaterstaat(RWS), the executive branch of the Dutch Ministry of Infrastructure and Environment, supplied data upon travelling speeds on road segments for the Netherlands. The data set consists out of Dutch road map with speed averages per road segment and for the averages are provided for three periods a day. The three periods are the morning rush hour, evening rush hour and the remaining parts of the day.

The road map from RWS does not entirely match with the map from TransTools. The RWS map only covers the Netherlands and some roads across the border in Germany and Belgium. Also the maps have a different level of precision. The transtools map defines road links between major intersections and the RWS map is more precise with smaller road segments. Road links in the TransTools map are mostly represented by several links in the RWS map. The map might be better of use because of its precision, but the graph lacks in connectivity and does not have the terminals and the related connections to road and waterways. It would require extensive work to combine the RWS map to the TransTools map to make it useful for our situation. Therefore only the speed averages are used and translated to the TransTools network.

Translating speed averages from the RWS road segments to the TransTools road links is done by relating a road link to a path of road segments, if possible in the RWS map. If it was not possible, then the links speed limit is used as speed for the road link. A weighted average of the speeds of the segments was determined to represent the average speed of the road link, where the weighing factor was travel time. This travel time is defined by the length of the road segment divided by the average speed in the specific time period.

The data consists out of averages from real measurements of traffic speeds on the road segments in the year 2010. The traffic that was observed in these measurements includes freight traffic originating from Rotterdam.

The road travel times are determined with these speeds as proposed by Ichoua [18] and described by equation (6). The method sums distances travelled per time interval until the total length of the link has been achieved. The width of the time interval influences the level of exactness of the travel time, smaller intervals lead to a more exact calculation of the travel time. However, the amount of required calculations increases linearly when the interval gets smaller.

6.2.5 Barge ships

As described in Chapter 4 the network will be extended with a schedule of barges. To make this barge schedule a set of observations was provided by TNO. The set contains a number of standard routes sailed by barge ships. Each route denoted the stops it made, a frequency the route was sailed in a week and the size of barge ship that sailed the route.

The frequencies are translated to departure times at the port of Rotterdam. With these departure times the A* algorithm is used to find the cheapest path along the given route with a stopping time at terminals of 1 hour. Output is a path with arrival and departure times along the given route and this path is set as virtual network links as described in section 4.2.4.

To determine these paths with A* it is assumed that all barges sail with an average of 9 kilometres an hour. General sailing speed of most barges is higher [2]. However, this lower average is used because bridges, sluices, currents and other obstacles delay the trip of barge ship. As barge journeys from Rotterdam to the inland are mostly upstream implies currents are also slowing down the barges.

6.2.6 Container Terminals

The barges can only be loaded and unloaded at terminals. The capacities and opening hours for all terminals in the network were available within TNO. This data is put into the network, such that the terminal nodes can transfer freight according to the given information.

Terminal handling capacity is defined by the capacity of freight that can be handled per hour and the operating hours of the terminal. The capacity per hour of the terminals is considered to be linearly dependent on the number of cranes available at the terminal. Each crane can move up to 20 containers per hour, therefore we defined an handling speed of 3 minutes per TEU. Terminals have varying opening hours, some work 24 hours a day and some only during daytime. Closing hours are integrated by setting the capacity at these hours equal to zero.

The handling time of a single freight demand is linearly dependent on the amount of containers. Assuming a barge can only be loaded or unloaded by one crane at a time, this might be different in reality. The freight orders have a minimal handling time at a terminal, therefore each shipment has a handling time of at least 30 minutes and the handling time increases with 3 minutes per container.

A freight order can only be loaded onto a barge if loading is finished before barge departure as planned by barge schedule.

The amount of containers that can wait on a terminal has not been limited, because most terminals have more than enough storage capacity. At least more capacity than can be shipped by barges that visit the terminal according to the barge schedule.

6.3 Verification

To ensure the ICPA* algorithm is implemented correctly verification is done. In this section we first verify the implementation of ICPA* by testing the influence of the parameters on the outcome of the algorithm. Then we verify A*Check by checking if there are possible better paths than the paths resulting from ICPA*. After that the difference between static and dynamic path assignment is shown and at the end of this section the implemented waiting-time expansion for the A* algorithm is verified.

6.3.1 Verificatifion ICPA* implementation

Assigning the same set OD with different values for the heuristic function and for parameters should have some expected effect on the resulting paths. For example increasing the costs of travel by barge should increase the use of trucks instead of barges. We only verify the effect of different values on the outcome. No quantative analysis is done to specify the sensitivity of the different parameters. This analysis does not treat all parameters, because some input parameters are calibrated in the next section. These parameters are the step sizes for the waiting time and the Ichoua step size and the calibrations also verify the correctness of their implementation.

The parameters that are analysed in this section are the costs and the heuristic function. The sensitivity of the heuristic function is analysed to verify the performance of A*. After verifying that A* is implemented correctly, the implementation of ICPA* with different modalities is verified with the sensitivity of the different modality costs.

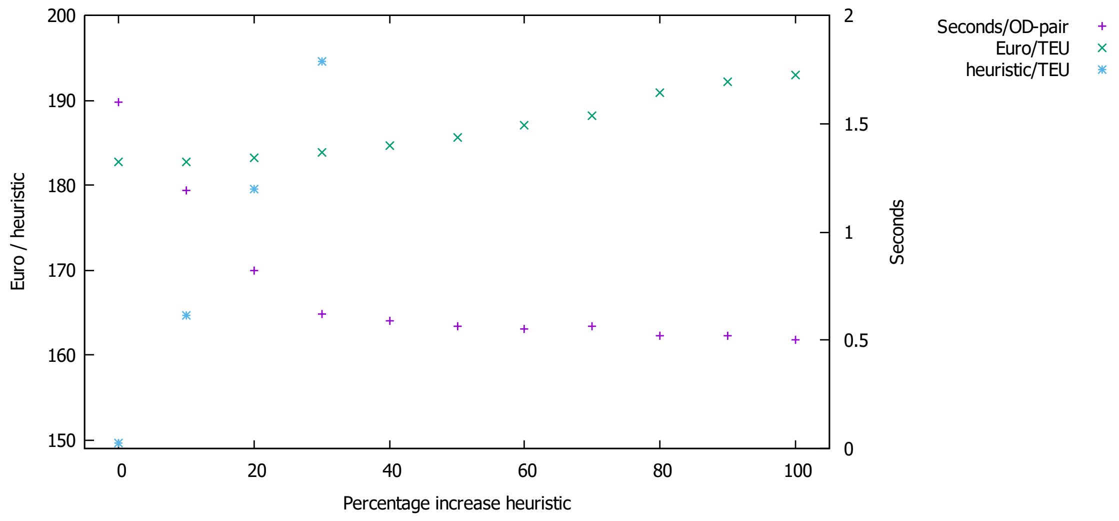
Influence of heuristic function

Now lets take a look at the influence of the heuristic function on the performance of A*. The heuristic function should not overestimate the real costs, but the higher the value the faster the search is directed to the destination. Thus a higher heuristic value should decrease the computation time of A*, but if the costs are overestimated then there is the possibility that the resulting path is not the cheapest that can be found by A*.

We should have a heuristic function that never overestimates the costs and we are also interested in the result of increasing the value of the heuristic function. Therefore the same OD is assigned for increasing heuristic values. Assignment is done for increasing heuristic values, from 0% increase until 100% increase with steps of 10%. The results are shown by the average cost per TEU, the average heuristic value per TEU and the average computation time per $d \in OD$. An OD set with 105 freight requests is assigned by ICPA* in FCFS order with the costs from Table 1.

Figure 3 shows the results. This figure shows the expected response upon changing the heuristic value. The computation time per path decreases if the heuristic value increases and the average cost per TEU increases when the heuristic value overestimates the cost. Note that the average heuristic value is below average TEU cost at 10% and 20% increase. However, in the original data we can see that for some of the 105 paths the costs are overestimated. At 10% increase the costs are overestimated for 3 paths and at 20% increase for 33 paths. With no increase the heuristic did not overestimate the costs of any path.

Figure 3: Average cost per TEU and average heurstic value per TEU for path from origin to destination, OD with 105 shipments and increasing value for the heuristic function.



Cost influence

Now lets take a look at the sensitivity of the cost. If the cost of a transportation modality changes then the amount of shipments using that modality should also change, because the cheapest path should use the cheapest modality. For this analysis we used the same OD set with 105 freight requests. First paths were assigned with the basic input costs from Table 1. The resulting paths can be seen in Figure 4.

The same set OD was assigned with different costs for barge transport. Figures 5 and 6 show that indeed the used modality changes when prices change. If the barges get more expensive, then more shipments will go by truck. Other way around, if the barges are cheaper, then more shipments will go by barge. The same has been done for the truck costs and the terminal costs, these results can be found in the appendix in Figures 11 to 14. Note that trucks are always required to get to the final destinations.

Figure 4: Route map of 105 assigned shipments with standard costs: terminal cost per minute €11, barge cost per minute €0.015 and truck cost per minute €0.86.

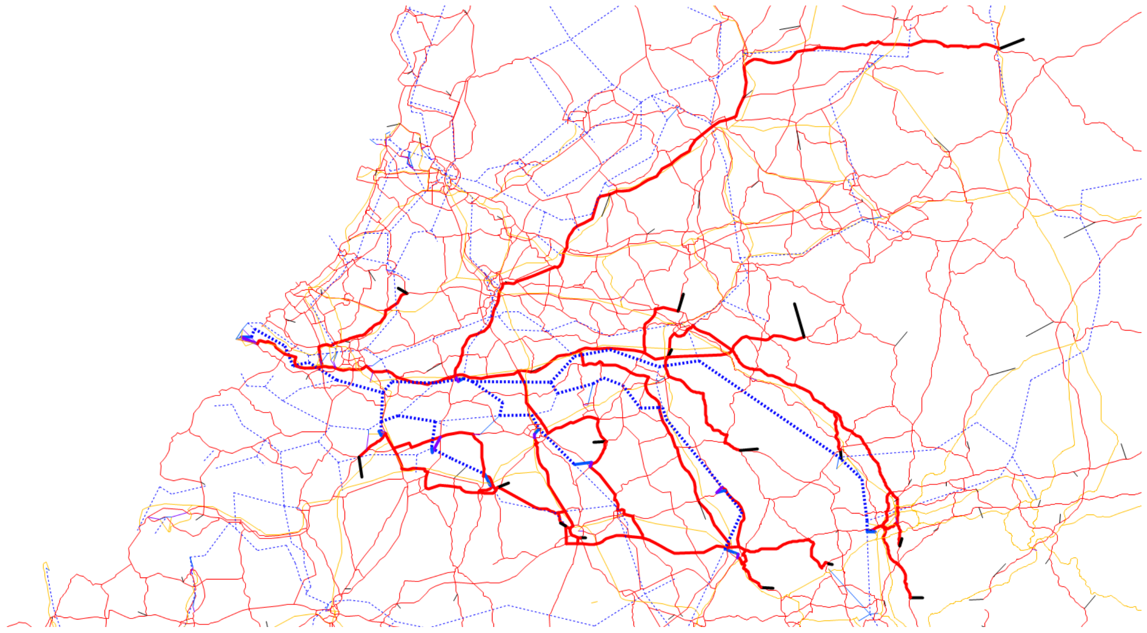


Figure 5: Route map of 105 assigned shipments with standard costs, but the barge cost per minute equals €0.001.

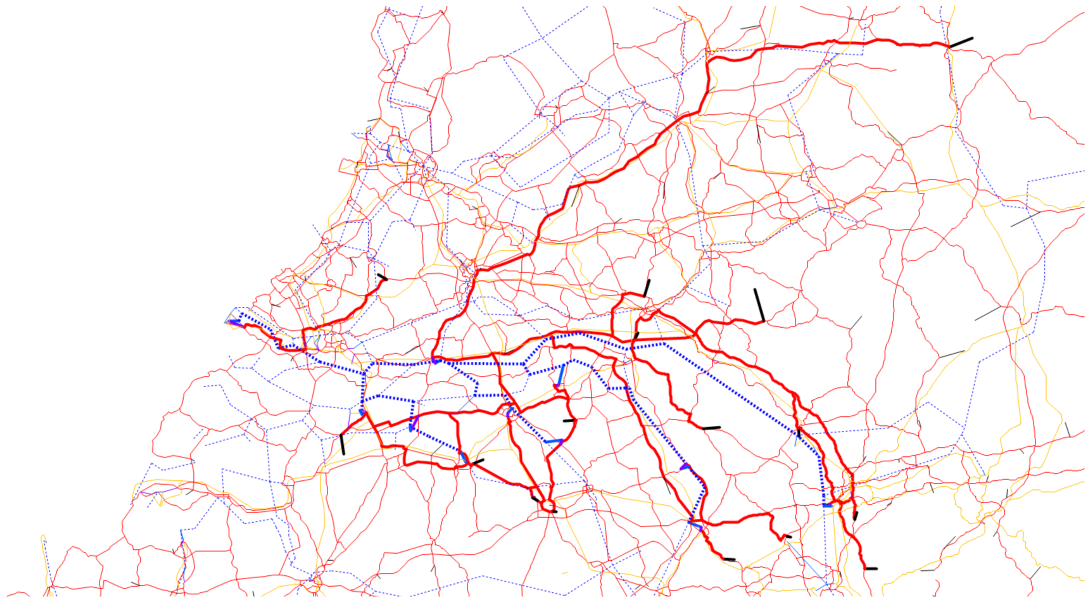
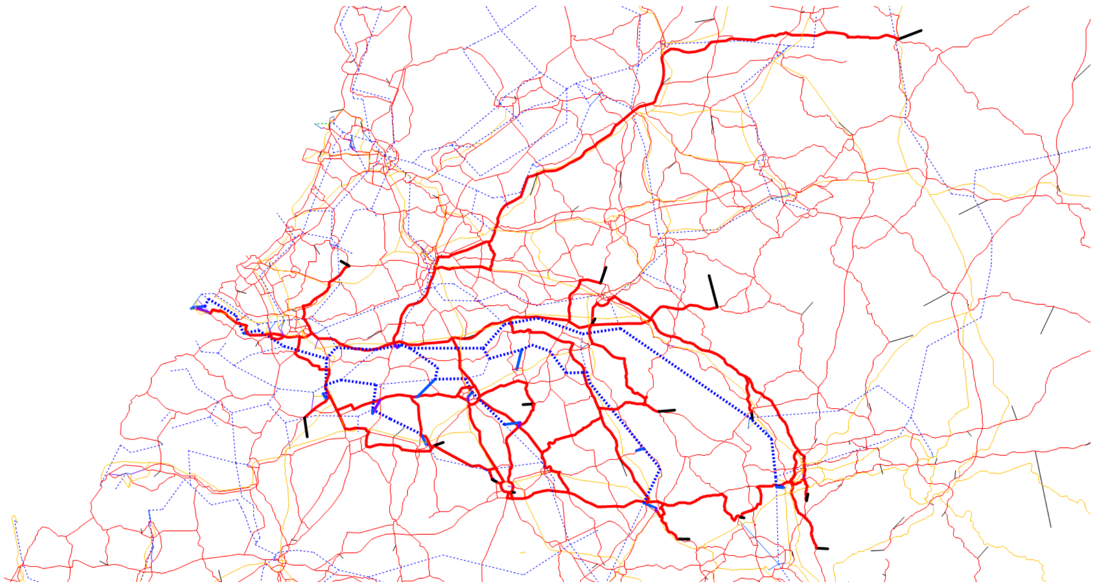


Figure 6: Route map of 105 assigned shipments with standard costs, but the barge cost per minute equals €0.30.



6.3.2 Cheapest paths

As mentioned in the previous chapter it can happen that A* does not produce the cheapest path. We now present the results which show the difference between the initially produced paths and possible cheaper paths. The A*Check algorithm, which can be found in section 5.3.1, is verified.

This verification was done with a large OD with 1011 freight requests and the ordering policy is FCFS. The same set is used in the use case of Chapter 7. The check was done for this set OD to show the possible improvement for the use case. Note that the better paths resulting from CheckA* are also not guaranteed to be optimal. Also note that the A*Check can find multiple path improvements for one original path, only the best improvement is taken into account.

As can be seen in the table only small fraction of paths is not optimal and the paths that are not optimal can only be improved a fraction. Only 24 out of the 1011 paths can be improved and the average improvement is less than one percent. Therefore the possible improvement on the total costs is very small. This means that the resulting total costs from ICPA* without CheckA* approach the minimum total costs for this order of assignment.

Table 2: Result of looking for cheaper paths with set of 1011 shipments, runtime of 48.11 seconds per shipment.

Initial A* total costs for 1011 paths	€3548170.31
Total costs with cheaper paths	€3547435.01
Actual difference	€735.30
Relative difference	0.02 %
Number of improvable paths	24
Average cost improvement with a cheaper path	€30.63
Average improvement percentage	0.75 %
Weighted average improvement percentage, weighing factor: initial cost per path	0.82 %
Maximum improvement percentage	2.2 %

The possible improvement is so small that we from now on assume that the result from ICPA* without A*Check is the best solution we can get. This assumption is also done to reduce computation time. Assigning with A*Check increases the computation time to 48 seconds per OD-pair instead of less than 3 seconds without A*Check.

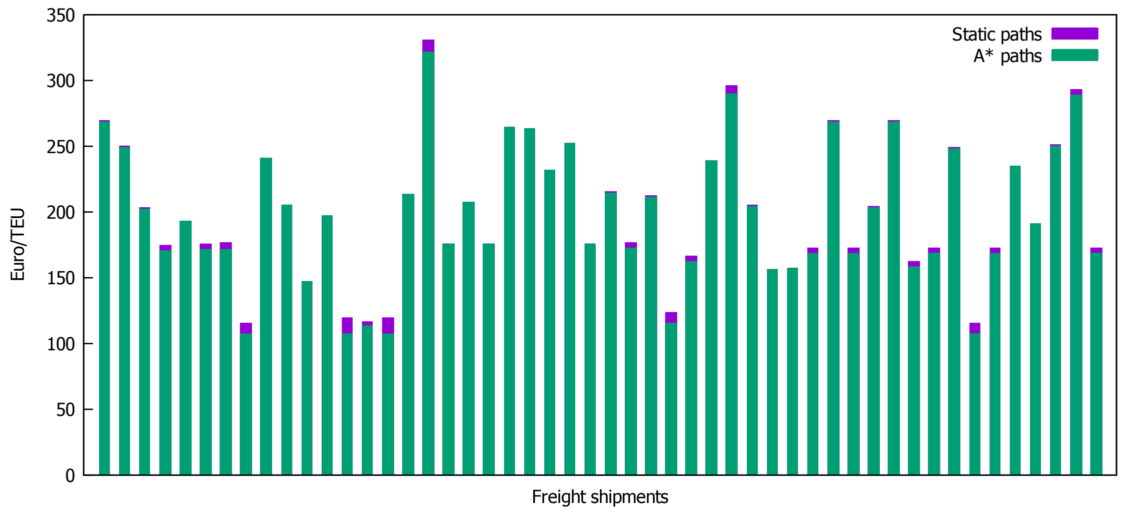
6.3.3 Verification of dynamic assignment

The next element of the verification is to show that dynamic assignment has an advantage over static assignment. To do this we assigned freight orders from the same OD set with both methods. The A* algorithm with dynamic network properties is compared to the static assignment of TransCAD. Where the static assignment is done with input speed average of one day. The assigning has been done without a barge schip schedule, as the schedule could not be comparibly translated to a static situation. Thus freight was only assigned via road links.

The output of both methods is a set of paths. Comparing the different set of paths is done by determining the travel costs of both path sets on the dynamic network, this means that the costs of travelling the paths from static assignment via the dynamic network are determined.

50 Freight shipments requests are assigned, the result can be seen in Figure 7. The differences are small because the road travel speeds only vary over 3 periods of the day and not every hour. It can be seen that some OD-pairs get an identical path assigned by both methods and some a different one. The paths determined by A* were all just as cheap or cheaper then the paths resulting from static assignment.

Figure 7: Verification result. The costs of all 50 paths from static assignment and A* dynamic assignment of the same set OD. Each bar represents the costs per TEU of a shipment.



6.3.4 Waiting-time expansion

The last element of the verfication is to verify the implementation of the time-expansion in A*. The same freight is assigned for several sizes of time-expansion to see the effect of the time-expansion on the running time of the algorithm and the cost reduction for this specific test case. An amount of time expansions per terminal has to be set and a maximum number of time expansions for a path.

The input data for all road travel speeds does not contain high levels of traffic congestion. So there probably is no situation where waiting at a terminal is cheaper with these travel speeds. So to test this method several traffic jams were manually created. The speed on 2 necessary road links is reduced to 5 km/h for a period of 3 hours and the following result shows that waiting a terminal produces cheaper paths.

The result in Table 3 shows that waiting at a terminal generates a cheaper path, but it is a significant drag on the computation time.

Table 3: The influence of the stepsize of the time expansion on computation time and average cost per assigned TEU for the same *OD* with 50 shipments. With 1 hour waiting per terminal, 2 hours maximum for each path and manually added traffic jams and no barges.

Time step size	Max expansions	Nr. expansions	Comp. time per d	Cost per TEU
No expansion	1	0	3.53 s	€207.99
30	4	2	14.25 s	€203.35
20	6	3	19.83 s	€201.05
10	12	6	43.31 s	€199.28

6.4 Calibration

The ICPA* algorithm uses several parameters that required calibration. These parameters are: step size for road travel time calculation, step size of waiting for capacity, step size for the time expansion. These parameters influence the exactness of the solution and the computation time of the algorithm. A higher level of exactness usually requires more calculations and thus more computation time.

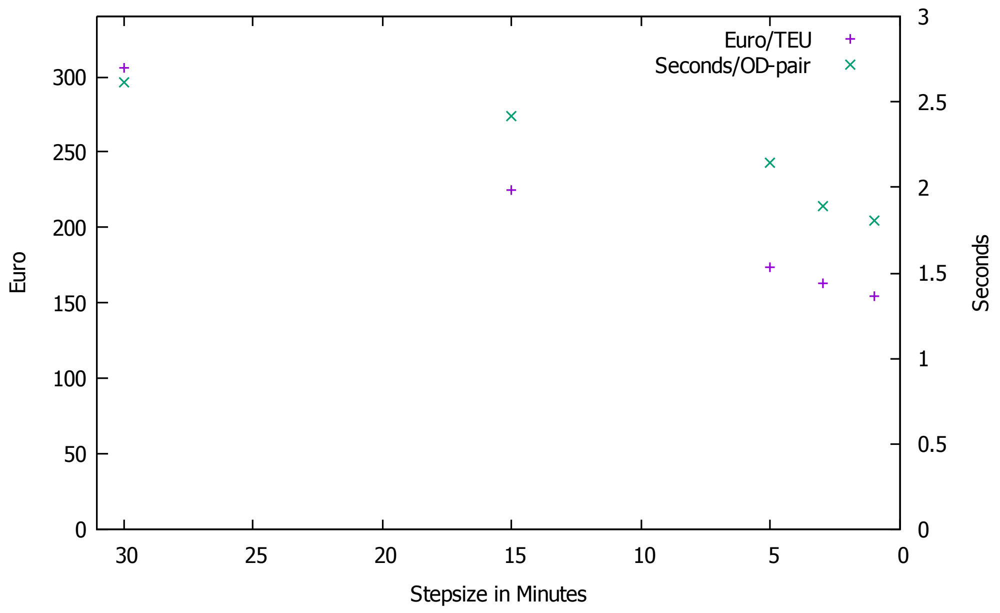
All simulations were done on the same machine with a i5-3230M processor and 8 GB of memory. Note that in the results in this chapter can have inconsistent fluctutations in computation time because of simultaneous simulations or varying machine settings.

6.4.1 Ichoua step size

Determining travel times for a road link is done in steps of a chosen step size. The size of the steps influences the level of precision of the estimated travel time. A step size of 5 minutes means that the resulting travel times are multiplications of 5. So the smaller the step size the higher the precision, but more iterations have to be done to calculate the travel time. A lot of travel times have to be calculated to create the cheapest path tree in the algorithm. Doing a lot of calculations for many freight orders with more iterations could mean a significant increase in travel time.

Several different step sizes are compared to know which step size would provide the best balance between level of precision and computation time. The algorithm has been run with the same set OD and the same network with the different step sizes. This lead to some surprising results and these can be found in Figure 8. The expectation was that a smaller step size would increase the computation time, this result shows the opposite.

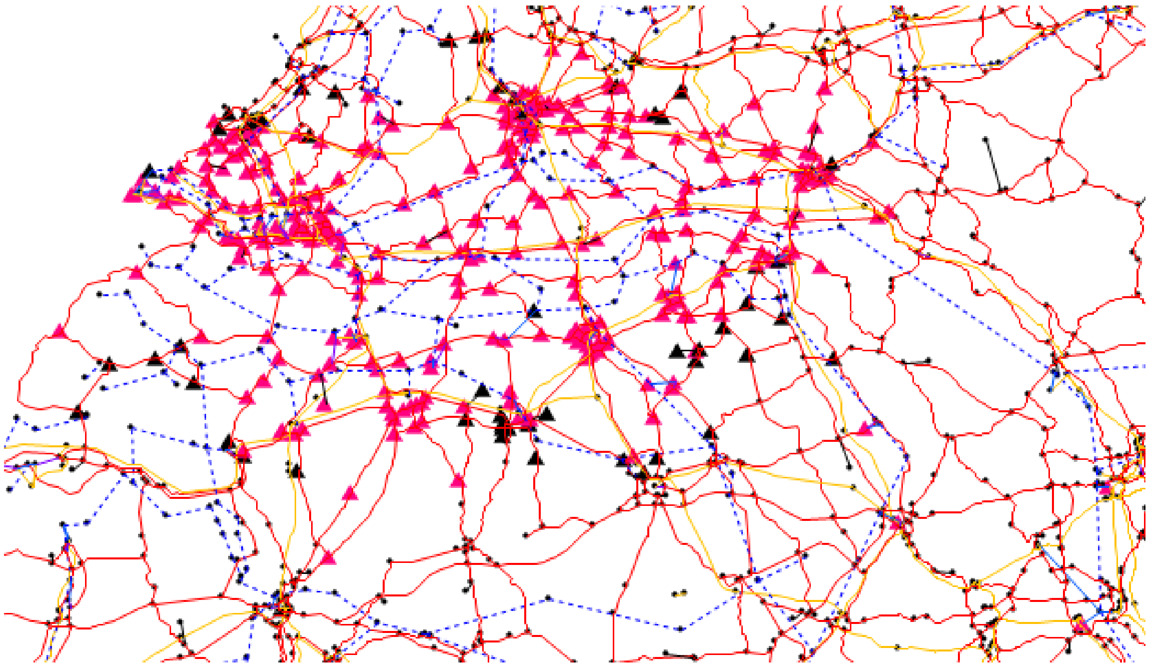
Figure 8: Influence of choice of step size on computation time and cost estimation. Horizontal axis denotes step size in minutes and the vertical axis the computation per shipment in seconds. OD contains 50 shipments.



This result was unexpected at first, but some further investigation resulted in the explanation of this effect. Taking a better look at the algorithm process revealed that the tree created by the A* algorithm differed in size for the different step sizes, see Figure 9. The smaller the step size the smaller the cheapest path tree, this explains why the computation time decreased for a decreasing step size. This result also implies that the size of the tree has more influence on the computation time than the amount of iterations done for calculating link travel times.

The cheapest path tree evaluated by A* being smaller is a consequence of the functioning of A*. As explained earlier, A* uses a heuristic function to guide the cheapest path tree towards the destination, the better the heuristic function estimates the costs of travelling, the smaller the evaluated tree gets. A smaller step size in the algorithm does not influence the heuristic function, but it does influence the costs. The larger the step size the larger the overestimation of the travel time, so also the costs will be overestimated. Thus the smaller the step size the closer the costs get to the heuristic function and less nodes will put in the tree before finding the destination node.

Figure 9: Difference in the cheapest path tree that has been evaluated by A* for different Ichoau step size. The purple triangles represent the nodes in the tree when the step size is 3 minutes and the black triangles the extra nodes in the tree when the step size is 5 minutes.



This result makes the choice of which Ichoua step size to use not very difficult. The algorithm gets a better precision and a reduction in computation time with a smaller step size. Note that the exactness of the calculated travel times is not only dependent on the step size, but also on the input for road link length and travel speed.

6.4.2 Order of iteration

The influence of the order of iteration on the dynamic traffic assignment is desctribed in this section. The order assignment influences the total costs. How big is this influence and is one order really cheaper than another?

The order of assignment has influence on costs for different freight shipments. For each individual freight shipment the order is of importance for its invidual tranport costs. However, we only look at the effect of the order on the total costs.

The algorithm has been run with 2 different OD sets in different iterative orders. The policies are chosen upon properties that can be directly related from the shipment requests $d \in OD$. Some policies have OD-pairs with the same priority values, there can be e.g. several shipment requests of 1 TEU and they would be indifferent for a shipment-size driven policy. These policies are therefore made more consistent by adding a second ordering policy. The results are shown in table 4.

The table shows that the the cheapest policy for the first OD is not the cheapest policy for the second OD. The ordering of the policies in costs differs per OD. The most remarkable is that the cheapest results comes from assignment in random order. Also the effect of the secondary policy is represented with the different shipment-size ascending policies.

Table 4: Cost per TEU for different ordering policies for sets $OD1$ and $OD2$.

Primary policy	Secondary policy	OD1:Cost/TEU	OD2:Cost/TEU
Shipment size, ascending	LCFS	€226.73	€186.48
LCFS		€226.91	€186.38
Shipment size, ascending	Random	€227.64	€187.84
Heur(O,D), descending	FCFS	€228.30	€193.18
Length, descending	FCFS	€228.61	€187.09
Shipment size, ascending	FCFS	€229.85	€191.03
Shipment size, descending	FCFS	€231.18	€190.61
FCFS		€233.64	€192.22
Heur(O,D), ascending	FCFS	€236.74	€190.45
Length, ascending	FCFS	€237.60	€190.81
Random		€225.36	€186.59
Random		€225.99	€186.93
Random		€226.09	€186.26

So if randomized is not considered to be a policy then the most cost effective policy seems to be ordering by ascending shipment size and the second place is for LCFS. That Last-Come First-Serve results in the lower costs is not surprising. As this policy makes sure that the order that is closest to the departure time of a barge is the first to claim some capacity. It is best to minimize waiting times and let the shipments with the potential longest waiting times for a barge go by truck.

Why the shipment size ascending order gives a cheaper solution than only LCFS is less trivial. It probably has to do with the input OD. With this policy all the smaller shipments will be assigned first. This probably lead to a cheaper result because barge capacities are more effectively used by the smaller shipments and it might be that the majority of the smaller shipments have to travel longer distances.

6.4.3 Waiting-time step size

In the algorithm there are two different step sizes for waiting time. One of them is the step size for the time expansion and this step size will be discussed in the following subsection. The other step size is the waiting time at a limited capacity link.

A limited capacity link has a limited capacity for a certain period of time, in the current situation the capacity is denoted per hour. If the ICPA* algorithm finds a link with not enough available capacity in the current period, then it is possible that in the next period there will be enough capacity available. Finding a subsequent period with available waiting time is done by adding a number of waiting time steps. The size of the step influences the number of required steps and the exactness of the waiting time estimation. It is possible to make the step the size of the periods, one hour. In that case one step is enough to check if there is available capacity in the next period, but it could be that the next period could be reached by waiting only 10 minutes instead of 1 hour. An insight in the effect of the size of this stepsize on the performance of the ICPA* can be found in Table 5.

Table 5: The influence of the waiting-time step size on computation time and average cost per assigned TEU for the same OD.

Waiting time step size	Comp. time per d	Cost per TEU
1	3.63 s	€188.69
10	3.22 s	€188.78
30	3.19 s	€188.27
60	3.43 s	€188.31

As can be seen in the table this step size does not have a significant influence on the computation time. The varying cost per TEU stands out. The costs probably vary because the different step sizes cause some shipments to get onto a different barge or not onto a barge. Allowing some other shipment to use the capacity that is left open on the barges and causing this slight different in average cost per TUE.

6.4.4 Time-expansion step size

The method of time expansion is described in section 5.4. For this method several parameter have to be defined: the time step size, the amount of expansions per terminal and the maximum number of expansions for one path. Therefore we take a look at the effect of these parameters on the computation time of the algorithm and the cost reduction that can be achieved by waiting.

Again some manual traffic jams were created to see the effect of waiting at a terminal. First we take a look at the effect of the stepsize. We want the possible waiting time at a terminal to equal one hour and the maximal waiting time in for one path to equal two hours for different stepsizes. This means that different numbers of expansions per terminal and maximal expansion numbers must be set. These different parameters and the results of assigning 50 freight shipments can be found in Table 3.

The result in Table 3 shows that not all freight shipments have to wait 2 hours to skip the traffic jam. The smaller step size ensures that the different freight shipments get a more specific cost-reducing waiting time. However, a lot more computations are required to find the paths as can been seen in the significantly rising computation time.

Thus a smaller step size can cause a better cost reduction, but it requires a lot of computation time to get a large window of possible waiting time. A larger time window is needed to allow all freight to skip the entire manually added traffic congestion, which lasts for 3 hours. There must be some number of expansions that allows all freight demand to skip the traffic congestion.

A search is done with a large time step size of 30 minutes to find the largest amount of waiting time that would maximize cost reduction. The results can be found Table 6, where the maximal reduction is reached after 10 expansions per terminal and maximal 20 expansions. Allowing a waiting time of 5 hours per terminal. This is a lot more then the 3 hours of traffic congestion. The traffic congestion occurs at a little distance from the terminal of origin. This distance causes some freight shipment to have to wait 4,5 or 5 hours to skip the traffic congestion that occurs for 3 hours.

Table 6: The influence of the stepsize of the time expansion on computation time and average cost per assigned TEU for the same OD with 50 freight demand requests. With manually added traffic jams and highly expensive barges.

Time step size	Max expansions	Nr. expansions	Comp. time per d	Cost per TEU
No expansion	1	0	3.53 s	€207.99
30	4	2	18.48 s	€203.35
30	8	4	39.85 s	€199.80
30	12	6	71.86 s	€195.41
30	16	8	72.32 s	€193.98
30	20	10	90.17 s	€193.92
30	24	12	133.81 s	€193.92
10	4	2	15.14 s	€199.37
10	8	4	35.19 s	€199.32
10	12	6	83.39 s	€199.28
10	16	8	153.54 s	€198.14
10	20	10	195.25 s	€196.94
10	24	12	202.15 s	€195.87
10	60	30	501.99 s	€192.70
10	72	36	700.84 s	€192.70

From the resultsin in Table 6 we conclude that it is better to use a larger time step size to allow a larger waiting time. Using a smaller step size is a real strain on the computation time, which is to be expected because the tree in A* will get larger with more expansions.

7 Use case

In this section a use case is created to give an example of an application possibility of the developed algorithm. To be able to analyse an interesting case a simulation program is created, to simulate one day of freight transport on a dynamic network with probabilistic disturbances. At first the method Discrete Event Simulation is used to simulate the assignment of freight during a day with probabilistic disturbances. Secondly a currently more practical method is used for the use case.

The ICPA* algorithm can be used to get a Dynamic Traffic Assignment for a set of freight demand. Such an assignment results in a path for each shipment request and an estimation of the costs for each path. The goal of this use case is to show that the algorithm can be used to determine the cost effect resulting from a change in the network. This means that this use case is on a strategic or tactical level. The use case is not on a operational level, because there is not enough specific information available to make a case. With specific information we mean more detailed data on freight demand, barge ships and road travel speeds; interesting information for scheduling and planning on an operational level.

So the case is on a higher level, because the used network is a realistic resprentation and therefore the influence of network changes can be an interesting study. What happens to the total estimated costs of the DTA if something changes in the network? In this case we look at reducing the probability of a shiplock or bridge breaking down. E.g. if Rijkswaterstaat (RWS) wants to do maintenance on a ship lock and therefore some waterway could no longer be used. What effect would closing this waterway have on the total costs of freight transport from Rotterdam. Another effect could be that RWS wants to do the maintenance to reduce the probability of a breakdown of the shiplock, what effect would the reduction in probability have. Determining such an effect can be done by comparing the resulting costs from a DTA for the different network situations.

For the situation with breakdown probalities it might be interesting to look at random occuring breakdowns. In reality there is some possibility that some event occurs randomly during the day, such as a traffic jam or a ship lock breakdown. When modelling with different random events it becomes quite difficult to analytically determine the effect of one probability changing. To be be able to get an insight in the effect of one probablity changing a simulation method can be used. A way simulating with such different random events is Discrete Event Simulation(DES). This method of simulation requires a lot of computation time with the current DTA algorithm, therefore results for the use case are generated in a little less random situation. For the use case we will first define parameter choices that are required for ICPA* and then we will define the network disturbances for the use case. We also do a DES of a case with less freight shipments.

7.1 Parameter choices

The following parameter choices have been used in the algorithm. The choice for the step size in the method of Ichoua is determined in section 6.4. The choice of waiting time step size is based upon a consolidation between computation time and the effect on estimated costs, and the something plausible in realistic schedules. This use case was done without using the time expansion method, because allowing the waiting times would significantly increase computation time and the used road travel speeds do not contain significant traffic congestion that would make it cheaper for freight to wait at a terminal.

Table 7: Algorithm parameters for the use case.

Parameter	Value
Ichoua step size	1 minute
Waiting-time step size	10 minutes
Time-expansion step size	10 minutes
Time expansions per terminal	0
Maximum expansions	1

7.2 The case

This section will describe the specifics of the use case. The possible breakdowns for the system will be based upon network disturbances, which have happened in real life.

So we can make a randomly generated set OD and the parameter choices for the algorithm have been made. Now we define different network situations for comparance. The differenct network situations will be determined by some network disturbances. We want to compare the total costs of traffic assignments on the network with different disturbances happening and also look at the effect of the disturbances occuring with some probability.

For this use case a situation is created where it is interesting to know the effect of changing a disturbance probability. Let there be some disturbance that occurs regularly and this disturbance causes the freight traffic from Rotterdam to take other and therefore more expensive paths. The Port of Rotterdam and RWS want to do an investment to reduce the probabilty of this disturbance occuring. Before doing this investment they would like to know what the benefit of this investment would be.

7.2.1 Disturbances

The disturbances used for this case are situations which really occured over the past few years. The first disturbance is a blockade of a river by a barge ship, in the last few years it has happened several times that ship got stuck in the German part of the Rhine river. Everytime this lead to a blockade for other barge ships for at least a day. In this case the Rhine will be blocked just after the river passed Nijmegen, this means if blocked that barge ships can not reach Nijmegen via the Rhine. The second disturbance is a broken shiplock, the ship locks of Weurt near Nijmegen were broken down in August 2015, reported by Koninklijke BLN-Schuttevaer. The third and last disturbance in this use case can happen at the Botlek bridge, this moveable bridge has been broken down several times over the past few years. This means that it can no longer open and barge ships cannot pass the waterway crossed by this bridge.

All these disturbance examples lasted longer than one day and they would be probably last longer than one day if they would happen again. Therefore the effect on the generated set of freight demand OD could be determined for the three disturbances lasting all day. It can however also be interesting to determine the effect for the disturbances occurring during a day.

In the latter situation, the disturbance could be seen as a random event happening during freight assignment. Then it might be useful to use Discrete Event Simulation (DES) to determine the effect of the disturbances. In the other situation let there be some probability that a disturbance occurs on a specified time or the entire day. For each disturbance some probability of occurance p would be known. With these probabilities it is possible to determine the probability of a combination of the three disturbances occuring. Having the probabilites of the different disturbance combinations and being able to determine the costs for each disturbance combination, allows us to calculate an expected amount of costs for the initial distributions. Thus for a different disturbance probability we can also calculate an expected cost and there for an expected change in cost can be determined for a change in probability.

This situation with three disturbances implies that only 8 (2 options for 3 disturbances leads to 8 unique combinations of disturbances being possible: $2^3 = 8$) different network situations have to be used for the DTA. The results of assigning for these 8 situations can be used to calculate the change in cost for different disturbance probabilities. Where determining the effect of during the day disturbances with DES would require many more runs than 8. In the situation where the disturbances can happen randomly during the assignment, there are a lot more than 8 unique situations. To deal with such randomness the simulation technique DES is used.

7.2.2 Discrete Event Simulation

A method of simulation with probabilistic events is Discrete Event Simulation. Discrete Event Simulation is a way of simulating a situation where distinct events take place in time and the process between the events is assumed to be known. So the simulation is determining the status of all processes in the system at each distinct event, where each event has some influence on the processes after the occurance of this event.

Three types of events are considered, the release of a freight shipment $d \in OD$ at its predefined release time, a network disturbance at some randomly generated time and rescheduling of a freight shipment which is hindered by a disturbance. An event of release of a freight shipment requires that a path is assigned to the shipment and the freight is loaded onto the network, which can be done by the DTA algorithm. A disturbance event is executed by removing a link from the network at the time of disturbance. The link that is broken could have been part of assigned paths, this means that shipments that were going to use the link after the event time have to be rescheduled. This is done by retracing the freight shipments that were assigned to the specific link after the disturbance time, and put these shipments back in set OD such that they can be scheduled again.

Putting a shipment in the to be assigned freight set OD again requires an origin, destination, releasetime and amount of freight. We first define this process for shipments that are on a truck. The amount of freight and the destination obviously stay the same. The origin is set to be the next node, in time, in the original path of the freight shipment. This means that freight shipments which arere on the broken link at the time of disturbance are not hindered. The release time is set to be the time of arrival at the specified next node.

For shipments that are on a barge a slightly different approach is used. As an entire barge ship has to be rescheduled when a waterway link is disrupted. This is done similar to the initial creation of the barge schedule. Current position will be the starting positions and the barge stops have to be visited are known.

This rescheduling of a barge implies that all the shipments on that barge get a different path. The first part of their new path is determined by the barge they are already on, because they can only leave the barge at the next terminal. They will be added to the set OD with the next terminal as origin, and the arrival time at that terminal as release time. We have to take care that the handling the costs at next terminal are added correctly if the shipment leaves the barge and that they are not taken into account if the shipment stays on the barge.

The event time of rescheduling events is now also defined as the new release times which were added together with the shipment to OD. A rescheduling event is almost the same as a initial assignment event, because it is just assigning a $d \in OD$.

Having the DTA algorithm and the event description we are almost ready to do DES. Only the distribution of event time and the probability of a disturbance event have to be set. The disturbances have been described above. The random event of such a disturbance occuring will be defined by a probility of occurance and a random event time. The random event time will be defined by a uniform distribution, defined by a mean and a time interval

As mentioned before the DES should be done a suitable number of runs, such that the average of the results can be said to be a representative result of the simulated situation. Simulating with input probability distributions requires enough simulations runs to get a trusthworty resulting average. To give an idea of the reliability of the resulting average a confidence interval can be used.

7.2.3 Confidence interval

This section describes how a confidence interval is set up around the results of the DES. We are interested in the expected total costs for the simulated setting. Every simulation run produces a total cost of assignment, simulating a number n runs therefore results in a sample of n total costs. It is generally known that the average of a sample with unknown mean and variance is distributed according to the Student t-distribution. This distribution is used to determine a confidence interval. The confidence interval is an interval which encloses the expectation of the total cost with a chosen probability. This means that the chosen probability is the probility that the expectation of total costs lies in the confidence interval.

Let the sample of simulating n runs be $(X_1, X_2, ...X_n)$. The sample mean $\bar{X}$, variance S^2 and the confidence interval, with t^* the critical value of the t-distribution, are defined as follows.

$$\bar{X} = \frac{\sum_{i=1}^{n} X_i}{n}$$

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(X_i - \bar{X}\right)^2$$

$$\left(\bar{X} - t^* \frac{S}{\sqrt{n}}, \bar{X} + t^* \frac{S}{\sqrt{n}}\right) \tag{16}$$

Critical value t^* determines the level of confidence (C). It defines that with some probability α the expectation of the total costs lies in the confidence interval. The critical value t^* can be found in the critical value table $t_\alpha(r)$ with r being the degrees of freedom and $\alpha = \frac{1-C}{2}$. A small confidence interval requires the value n to be high enough. However simulation with a large amount of runs n requires a lot computation time.

7.3 Results

In this section the results of a discrete event simulation with a very small set OD and the results of simulating the use case for a larger set OD are given. DES has only be done with two freight shipment requests due to the required computation time. The results from this DES are presented to show that it is possible to use the DES method to do a study if more computational power is available or time is not an issue. Lets first present the more practical results of the use case.

7.3.1 Results of Discrete Event Simulation

This section gives the results of a DES, to show the possibility of running a DES with the developed Dynamic Traffic Assignemnt algorithm with possible rerouting. The simulation has only been done for OD set with only 2 freight requests to be to simulate 10000 runs in a reasonable amount of time.

This amount of runs was required to be able to draw a reasonable confidence interval around the resulting total cost average. For each run the total cost of transporting the 2 freight shipments was determined by the algorithm. This total cost was influenced by the disturbances happening in a run with a probability of 5 % and at a uniformly distributed time with specified average and width. This randomized disturbance time caused the shipments to be rescheduled at different points on their routes resulting in different total costs.

Table 8: DES result for 10000 runs with 2 freight shipments and 2 possible disturbances with probablity of 5%.

runs, n	10000
No disturbance total cost	€7040.31
Sample average, $\bar{X}$	€7525.07
Sample variation S^2	4232861

Confidence level, C	95 %
Critical value t^*	1.96
Confidence interval	$\left(\bar{X} - t^* \frac{S}{\sqrt{n}}, \bar{X} + t^* \frac{S}{\sqrt{n}}\right)$ $\left(7484.74, 7565.40\right)$

Runtime

This simulation took more than 10 hours, with 1,5 second per OD-pair and 20000 assignments, and also time was required for rescheduling the boats and the corresponding shipments in case of disturbances. This amount of computation time is why no simulation of a larger OD set has been done, it would require a lot more computation time. The computation time could be reduced by making a look-up table for assigned shipments in the situations where no disturbances have yet occured. The situations where disturbances occur with random event time would require new assignments or rerouting for shipments. If these situations are plentyful it would mean that the total computation time would still be large.

To make a use case in which the computation time is more comprehensible a part of the randomness was dropped. This lead to the use case described in the next section.

7.3.2 Results for the use case

Doing a DES with a larger OD set was troublesome because of the required computation time to get a decently small confidence interval. DES was used to deal with situations with several random disturbances, making the disturbances less random allows a more simple method to determine their effect. Hereby the results for this use case with the disturbances as described in section 7.2, where the disturbances occur the entire day instead of randomly during the day.

This case with the predetermined disturbance times has a lot less possible network situations compared to the situation with uniform random disturbance times. Having the 3 possible disturbances only gives 8 possible network states for which the OD set can be assigned. Next to this small amount of possible situations we are also able to determine the possibility of each situation happening, if the individual disturbance probabilities are known.

Having this limited amount of states and their probability of occuring allows us to determine the expected amount of total costs, if we know the total costs of each state. Only the effect of the disturbances happening somewhere random during the assignment is now lost. The predetermined event times could chosen somewhere during the day to give some indication. At least this method could determine the costs effects for these disturbances during an entire day for a large OD.

Having the total costs of each possible combination of disturbances and the probabilities allows us to calculate an expected total costs. So it also allows us to determine the effect on the total cost of changing one of these probabilities.

The total costs for the eight different combinations can be found in table 9. It is interesting to see that if the Rhine and Botlek are both disturbed that the total costs are less compared to the situation where only the Rhine is disturbed. This is probably caused by barges that are sailing a different route as they cannot pass the Botlek bridge and cause a cost reduction for other shipments.

For all three disturbances we consider a probablity p that the disturbance occurs, for the different disturbances we add index r for the Rhine disturbance, w for the disturbance at Weurt and b for the Botlek bridge. With these three probabilities it is possible to determine the probability of a combination of disturances occuring, because the stochastic disturbances are independent. This can be done with the probability formula's in the following table, let p_i with $i = 1, 2, ..., 8$ define the probabilities of the combinations of disturbances occuring and $costs_i$ define total costs for combination i with $i = 1, 2, ..., 8$.

Table 9: Total costs for the different combinations of disturbances. A 0 indicates that the disturbance does not occur and a 1 that it does.

Rhine	Weurt	Botlek	Total costs	Probability formula
0	0	0	€3456915.89	$p_1 = (1 - p_r)(1 - p_w)(1 - p_b)$
0	0	1	€3471962.94	$p_2 = (1 - p_r)(1 - p_w)p_b$
0	1	0	€3489111.36	$p_3 = (1 - p_r)p_w(1 - p_b)$
0	1	1	€3508415.18	$p_4 = (1 - p_r)p_w p_b$
1	0	0	€3509790.20	$p_5 = p_r(1 - p_w)(1 - p_b)$
1	0	1	€3507183.94	$p_6 = p_r(1 - p_w)p_b$
1	1	0	€3540747.47	$p_7 = p_r p_w(1 - p_b)$
1	1	1	€3539293.94	$p_8 = p_r p_w p_b$

Having the costs and the probabilities for the different situations gives the expected total cost for a situation. Let the initial situation be determined by the following disturbance probabilties and have a resulting expected total costs.

Table 10: Disturbance probabilities for the disturbances described in section 7.2 and resulting expected total transport costs for 1011 freight shipments.

Disturbance	Probability
Rhine	$p_r = 0.05$
Weurt	$p_w = 0.05$
Botlek	$p_b = 0.10$

$$\mathbb{E}[\textit{total costs}] = \sum_{i=1}^{8} p_i costs_i = €3462603.22.$$

Now imagine tat the Port of Rotterdam thinks that the 10% probability of the Botlek bridge breaking down is too high and they want to do an investment to reduce this probability by doing maintainance on the bridge. They would like to know what cost effect such an investment would have on the freight transported originating at the Port of Rotterdam. If their investment would decrease the probability to 5% then the cost effect can be determined as follows, first we determine the new expected costs.

Table 11: Disturbance probabilities for the disturbances described in section 7.2 after an imaginery investment by the Port of Rotterdam and resulting expectation of total transport costs for 1011 freight shipments.

Disturbance	Probability
Rhine	$p_r = 0.05$
Weurt	$p_w = 0.05$
Botlek	$p_b = 0.05$

$$\mathbb{E}[total\ costs] = \sum_{i=1}^{8} p_i costs_i = €3461884.76.$$

Now we can calculate the expected cost difference for one day of freight transport, for the cost effect for an entire year we multiply this difference with 250 work days a year, representing that there are 250 days a similar set of freight is shipped. For a better indication it might be required to determine the cost effect for more randomly generated OD sets. The result of this calculation can be seen as an expected cost reduction of applying the improvement to the Botlek bridge and reducing breakdown probability. The cost effect of the bridge being in maintenance and unusable is not taken into account.

$$\begin{aligned} Cost\ reduction =& 250 \cdot \big(\mathbb{E}[total\ costs|p_b = 0.10] - \mathbb{E}[total\ costs|p_b = 0.05]\big) \\ =& 250 \cdot (3462603.22 - 3461884.76) \\ =& €179618.29. \end{aligned}$$

This means that it is to be expected to save €180k per year if the bridge would only break down 5% of the time instead of 10%. If this were a real situation and if the savings would be benificial for the Port of Rotterdam, then this could be a profitable investment.

This was the use case for a DTA solution from the ICPA* algorithm. The next chapter states the conclusion of this research and recommendations.

8 Conclusion and recommendations

This section will describe the conclusions that can be drawn from the results of this research and the recommendations for improvements and further research.

8.1 Conclusion

In the following we consider the research questions one by one and give answers to them.

How to find dynamic routes fast enough to make real time decisions?

The algorithm A* can find a cheap path for a freight shipment in matter of seconds if waiting time is not allowed. If waiting time at terminal is allowed the computation time for one path becomes a little larger than a minute. If A* is used to find a path for a single user who wants to respond to real-time information, then a few seconds or a minute is a reasonable amount of computation time. However, finding many dynamic routes with the developed algorithm takes a quite a lot of time. Subsequently, it takes too much time to do more extensive simulation studies with the algorithm.

How to optimally assign freight demand requests to limited barge capacity?

The problem of optimally assigning freight to barge capacity has not been answered in this research. An assumption is done and with this assumption a method of assigning freight shipments to specific barges is created.

The results of assigning freight demand in different orders gives some insight in the optimal assigning to barges in the assumed situation. The order of assigning defines which shipments can first claim limited capacities of terminals and barge ships. This means that the order of assigning influences which shipments are assigned to which barge. The effect of this order has been studied and the results show that costs can be saved when assigning is done in a different order than FCFS.

What insight in network behaviours could a dynamic traffic assignment model give when network disruptions occur?

As shown by the use case it is possible to determine the cost effects for changes in the network. The cost effect can be determined including random disturbances, also a simulation method was set up to incorporate disturbances randomly during freight assignment. Such a simulation can give better insight on an operational level. However, the algorithm needs a lot of time to finish a complete simulation for a large set of freight demand.

Note that more aspects that can be studied with a DTA model, for example the effect of the transport costs on the choice of path and modality can also be studied with the model. Also the effect of growing freight demand on the use of network capacity can be studied.

The research question

How can a dynamic traffic assignment model effectively assign dynamic freight demand on inland waterways and roads dealing with disruptions and congestion?

The algorithm developed in this research can assign freight demand onto a dynamic network. It can deal with the time dimension in the assignment of freight transport. This also means it can deal with dynamic disturbances such as congestion and disruptions. When simulating a day of freight assignment it can also reschedule an assignment if a disruption occurs.

The main goal of this master thesis was to create a Dynamic Traffic Assignment model. The algorithm implemented in TransCAD can succesfully do this. Some assumptions had to be made to accomplish this result, but the result is a method to dynamically assign freight traffic onto a multimodal network. The algorithm does require quite a lot of computation time, which can be troublesome if a lot of freight shipments have to be assigned.

8.2 Recommendations

In this section the recommendations for further research based upon the done research are discussed. In this research several assumptions and choices are made to make the problem compehensible. In the following paragraphs ideas are sketched on how to deal with problems which were circumvented by these choices.

8.2.1 Trucks

Currently the assumption is done that trucks are always available at the port of Rotterdam to transport containers. In reality there are not infinitely many trucks and it might therefore be more realistic if the availability trucks is also taken into account.

This assumpton of trucks always being available caused the neglection of delivery deadlines. In reality there is a delivery deadline for some freight shipments. These deadlines come into scope if they are close to the release time or in case there is a lack of capacity for transporting all freight demand. There is no lack of capacity because of the unlimited amount of trucks and no influence of the trucks on the travel speed.

8.2.2 Barge schedule

A main assumption to be able to dynamically assign freight was that the barge ships sail some predetermined schedule. It would be interesting to also determine the barge schedule based upon the given freight demand, such that the barges could be used for the paths where they are most useful. This could be implemented by integrating the scheduling of the barges into the algorithm, which would however add more decision variables to the problem, making it more difficult to find a solution.

One way of constructing such a barge schedule could be by first assigning all freight demand without vehicle constraints. The resulting paths would give insight in which paths could be beneficial to be sailed by barges.

8.2.3 Normative cost function

The objective function used for finding a DTA is a direct cost function and is a so called normative function, because it focuses on direct costs. In reality it happens that the choice for which path to take is not always directly based upon costs. It could be that there is some preference for using some transport modality, a preference which is not based upon costs. Such a preference can be added to the objective function by adding a so called soft decision variable. There is the challenge to determine the value of this variable. A common technique to determine the value is to add the variable to the objective function and calibrate the objective function including this variable. A complete observation of a path assignment is needed to be able to do such a calibration.

Adding this soft decision variable in the objective can make the results of the algorithm more realistic, because integrating such a choice variable would allow the algorithm to assign paths which would better represent realistic path choices.

8.2.4 Waiting time at terminals

The time expansion in the algorithm that allows a variable waiting time at a terminal requires a lot of computation time. This method calculates all paths possible for many waiting possibilities.

Instead of doing all these calculations in advance it might be a lot faster to do it afterwards. If ICPA* with no time expansions finds a cheap path with no significant slow travel speed on road links, then it will not be cheaper to wait at a terminal. If the path does contain traffic jams on road links then it might be cheaper to wait at some terminal, and in that case the effort of doing the time expansion could be useful.

8.2.5 Splitting shipments

Freight shipments can consists of several containers. It can be that the order size is larger than the remaining capacity on a barge, or even that an order size is larger then the total capacity of a barge. In those cases it might be cheaper to split the order and send a part by barge and the other part via the road or a different barge. In reality this can and does happen to make optimal use of available barge capacities.

Splitting of shipment is troublesome for ICPA*, because it would require splitting a shipment during the construction of the cheap path. It is not merely splitting a shipment when the algorithm finds a barge with not enough capacity. There would be several points where splitting could be done and it is not sure that it is the cheapest or a decently cheap point to split the order.

It is possible to split all shipments at the origin, for example by reducing all shipments to shipments of a single container. Then for each single container a cheap path can be determined and all barges can be fully loaded if necessary. The number of paths that have to be determined becomes a lot larger. However, the number of paths to be determined could be reduced. It is possible to determine a path for one container and to also assign this path to other containers from the same original shipment until some capacity limit is reached.

References

[1] Richard Bellman. On a routing problem. *Quart. Appl. Math.*, 16:87–90, 1958. [15]

[2] Bureau Voorlichting Binnenvaart. Scheepstypen, 2015. [43]

[3] Malachy Carey. Nonconvexity of the dynamic traffic assignment problem. *Transportation Research part B*, 26B(2):127–133, 1992. [13]

[4] Malachy Carey and Mark McCartney. Behaviour of a whole-link travel time model used in dynamic traffic assignment. *Transportation Research part B*, 36:83–95, 2002. doi:10.1016/S0191-2615(00)00039-4. [12]

[5] Malachy Carey and Eswaran Subrahmanian. An approach to modelling time-varying flows on congested networks. *Tranportation Reserach Part B*, 34:157–183, 2000. [11, 16]

[6] Kenneth L. Cooke and Eric Halsey. The shortest route through a network with time-dependent internodal transit times. *Journal of mathematical analysis and applications*, 14:493–498, 1966. doi:10.1016/0022-247X(66)90009-6. [15]

[7] C.F. Daganzo. *Fundementals of transportation and traffic operations.* Emerald Inc, 1997. ISBN 0080427855, 9780080427850. [12]

[8] Edsger Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959. [14]

[9] S.E. Dreyfus. An appraisal of some shortest-path algorithms. *Operations Research*, 17:395–412, 1969. doi:10.1287/opre.17.3.395. [14, 15]

[10] Omar Drissi-Kaitouni. A variational inequality formulation of the dynamic traffic assignment problem. *European Journal of Operational Research*, 71:188–204, 1993. doi:10.1016/0377-2217(93)90048-R. [11, 13]

[11] Bernard Fleischmann, Martin Gietz, and Stefan Gnutzmann. Time-varying travel times in vehicle routing. *Transportation Science*, 1:160–173, 2004. doi:10.1287/trsc.1030.0062. [13]

[12] Michael Florian, Michael Mahut, and Nicolas Tremblay. Application of a simulation-based dynamic traffic assignment model. *European journal of operational research*, 189:1381–1392, 2008. doi:10.1016/j.ejor.2006.07.054. [11]

[13] Terry L. Friesz, Davind Bernstein, R.L. Tobin, and S. Ganjalizadeh. Dynamic network traffic assignment considered as a continuous time optimal control problem. *Operations Research*, 37(6):893, 1989. [11]

[14] Ron Gutman. Reach-based routing: a new approach to shortest path algorithms optimized for road networks. *In Proc. 6th International Workshop on Algorithm Engineering and Experiments*, page 100, 2004. [15]

[15] R.W. Hall. The fastest path through a network with random time-dependent shortest paths. *Transportation Science*, 20(3):182–188, 1986. doi:10.1287/trsc.20.3.182. [15]

[16] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *Transactions of systems science and cybernetics*, (2):157–183, 1968. [15]

[17] Mark E T. Horn. Efficient modeling of travel in networks with time-varying link speeds. *Networks*, 36(2):80–90, 2000. [15, 16]

[18] Soumia Ichoua, Michel Gendrau, and Jean-Yves Potvin. Vehicle dispatching with time-dependent travel times. *European journal of operations research*, 144:379–396, 2003. URL `http://works.bepress.com/soumia_ichoua/4/`. [13, 24, 42]

[19] Bart Jourquin. A multi-flow multi-modal assignment procedure applied to the european freight transportation networks. *Nectar Cluster 1r*, 2005. doi:10.13140/2.1.2742.4008. [17]

[20] Jurgen Lerner, Dorothea Wagner, and Katharina A. Zweig. Algorithmics of large and complex networks. 1973. doi:10.1007/978-3-642-02094-0. [15]

[21] L. Li, R.R. Negenborn, and B. De Schutter. A sequential linear programming approach for flow assingment in intermodal freight transport. *16th Internation IEEE Conference on Intelligent Transportation Systems*, 2013. doi:10.1109/ITSC.2013.6728399. [12]

[22] Hani S. Mahmassani. Dynamic network traffic assignment and simulation methodology for advanced system management applications. *Networks and spatial economics*, 1: 267–292, 2001. [11]

[23] Deepak K. Merchant and George L. Nemhauser. A model and an algorithm for the dynamic traffic assignment problems. *Transportation Sciences*, 12(3):183–199, 1978. doi:10.1287/trsc.12.3.183. [11, 12]

[24] A. Nagurney. *Network Economics: A variational inequality approach.* Springer, New York, 1998. [11]

[25] NEA. Kostenkengetallen binnenvaart, 2008. [41]

[26] NEA and TNO. Basisbestanden goederenvervoer 2004: Eindrapport rijswijk, 2004. [40]

[27] Artyom Nhapetyan and Siripohng Lawphongpanich. Discrete-time dynamic traffic assignment models with periodic planning horizon: system optimum. *Journal of global optimization*, 38, 2007. doi:10.1007/s10898-006-9082-4. [16]

[28] Ariel Orda and Raphael Rom. Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length. *Journal of the ACM*, 37(3):607–645, 1988. [14, 15]

[29] Stefano Pallotino and Maria Grazia Scutellá. Shortest path algorithms in transportation models: classical and innovative aspects, 1998. technical report. [16]

[30] Panteia/NEA. Kostenontwikkeling, 2015. [41]

[31] Panteia/NEA. Ontwikkeling kostenniveau binnenlands vrachtautovervoer, 2015. [41]

[32] Srinivas Peeta and Hani S. Mahmassani. System optimal and user equilibrium time-dependent traffic assignment in congested networks. *Annals of Operations Research*, 60:81–113, 1995. doi:10.1007/BF02031941. [12]

[33] Srinivas Peeta and Athanasios K. Ziliaskopoulos. Foundations of dynamic traffic assignment: the past, the present and the future. *Networks and Spatial Economics*, 1: 233–265, 2001. [6, 11]

[34] Hayssam Sbayti, Chung-Cheng Lu, and Hani S. Mahmassani. Efficient implementation of method of succesive averages in simulation-based dynamic traffic assignment models for large-scale network applications. *Transportation Research Record Journal*, 2007. doi:10.3141/2029-03. [11, 12]

[35] Yosef Sheffi. *Urban Tranportation Networks: Equilibrium Analysis with mathematical programming methods*. Prentice-Hall, Inc., Englewood Cliffs, 1985. [17]

[36] C.O. Tong and S.C. Wong. A predicitive dynamic traffic assignment model in congested capacity contrained road networks. *Transportation Research Part B*, 2000. [11]

[37] J.G. Wardrop and J. I. Whitehead. Correspondence. some theoretical aspects of road traffic research. *ICE Proceedings: Engineering Divisions 1*, 5:767, 1952. doi:10.1680/ipeds.1952.11362. [10]

[38] Mo Zhang, J. Janic, and L.A. Tavasszy. A freight transport optimization model for integrated network service and policy design. *Transportation Research Part E*, 77:61, 2015. [40, 41]

[39] Athanasios Ziliaskopoulos and Whitney Wardell. An intermodal optimum path algorithm for multimodal networks with dynamic arc travel times and switching delays. *European journal of operational research*, 125:486–502, 2000. [16]

List of symbols

$\mathbb{T}$	Time horizon
T, t, τ	Time, T being a point time and t, τ respectively an amount of time
G	Graph representing traffic network
N, n	Set of nodes, single node
A, a	Set of links, single link
R	Roads, subset of A
W	Inland Waterways, subset of A
V	Virtual transfer links, subset of A
B	Virtual links representing barge trips
N_n^{in}	Neighbouring nodes with link towards node n
N_n^{out}	Neighbouring nodes with link from node n
OD	Set of freight demand with Origin, Destination, release time and shipment size
d	Single freight demand in set OD
P	Path in G
δ	Indicator variable denoting if a link is used for some freight demand
F	Amount of flow, in TEU
c	Cost of travel, per link and per time.
$\mathbb{Z}^+$	Set of all positive integers
p	Some probability

List of abbreviations

CICO	Cheapest-In Cheapest-Out
DTA	Dynamic Traffic Assignment
FIFO	First-In First-Out
FCFS	First-Come First-Serve
ICP	Iterative Cheapest-paths Problem
IWW	Inland Waterways
LCFS	Last-Come First-Serve
MNL	Multinomial Logit Model
MSA	Method of Succesive Averages
RWS	Rijkswaterstaat
SCP	Single Cheapest-path Problem
SO	System-Optimal equilibrium
SPP	Shortest-Path Problem
TNO	Nederlandse Organisatie voor Toegepast-natuurwetenschappelijk onderzoek
TEU	Twenty-foot Equivalent Unit
UE	User Equilibrium
VI	Variational Inequality

Mathematical program - Single Cheapest-path Problem (SCP)

$$\min_{\delta_{d,a}, t^{wait}_{d,a}} \quad F_d\big(\sum_{a\in A} \delta_{d,a} c^f_a + t_{d,a} c^t_a + t^{wait}_{d,a} c^{t,wait}_a\big) \tag{A1}$$

Time constraints

$$\begin{aligned}
\sum_{k=T^{in}_{d,a}}^{T^{in}_{d,a}+t_{d,a}} \left[\Delta k v^I_a(k)\right] &\geq \delta_{d,a} L_a && \forall a \in R \\
t_{d,a} &= \delta_{d,a} \alpha^t_a F_d && \forall a \in V \\
t_{d,a} &= \delta_{d,a}(T^{arrival}_a - T^{departure}_a) && \forall a \in B \\
t^{wait}_{d,a} &= 0 && \forall a \in R \\
T^{in}_{d,a} &\leq \delta_{d,a} M && \\
T^{in}_{d,a} &\geq t^{wait}_{d,a} + T^{release}_d - (1-\delta_{d,a})M && \forall a \in A^{out}_{Ori_d} \\
T^{in}_{d,a} &\geq t^{wait}_{d,a} + T^{in}_{d,a'} + t_{d,a'} - (1-\delta_{d,a})M && \forall i \in N, a \in A^{out}_i, a' \in A^{in}_i \\
t^{wait}_{d,a} &\geq T^{in}_{d,a} - T^{in}_{d,a'} + t_{d,a'} - (2 - \delta_{d,a} - \delta_{d,a'})M && \forall i \in N, a \in A^{out}_i, a' \in A^{in}_i
\end{aligned}$$

Capacity constraints

$$\begin{aligned}
\delta_{d,a} F_d &\leq F_a(T) && \forall a \in V,\ T \in [T^{in}_a, T^{in}_a + t_{d,a}) \\
\delta_{d,a} F_d &\leq F_a && \forall a \in B
\end{aligned}$$

Path constraints

$$\sum_{a\in A^{in}_i} \delta_{d,a} - \sum_{a \in A^{out}_i} \delta_{d,a} = \begin{cases} -1 & i = Ori_d \\ 0 & \forall i \in N\ /\{Ori_d, Des_d\} \\ 1 & i = Des_d \end{cases}$$

Variables

$$\begin{aligned}
\delta_{d,a} &\in \{0,1\} && \forall a \in A \\
t^d_a, t^{wait}_{d,a}, T^{in}_{d,a} &\in \mathbb{Z}^+ && \forall a \in A
\end{aligned}$$

Figures

Figure 10: Example of functioning of A* with time expansion. Let there be 1 terminal, 1 time expansion possible, leading to cost reduction on later links. Let the cost for waiting 1 time expansion period equal 0.1.

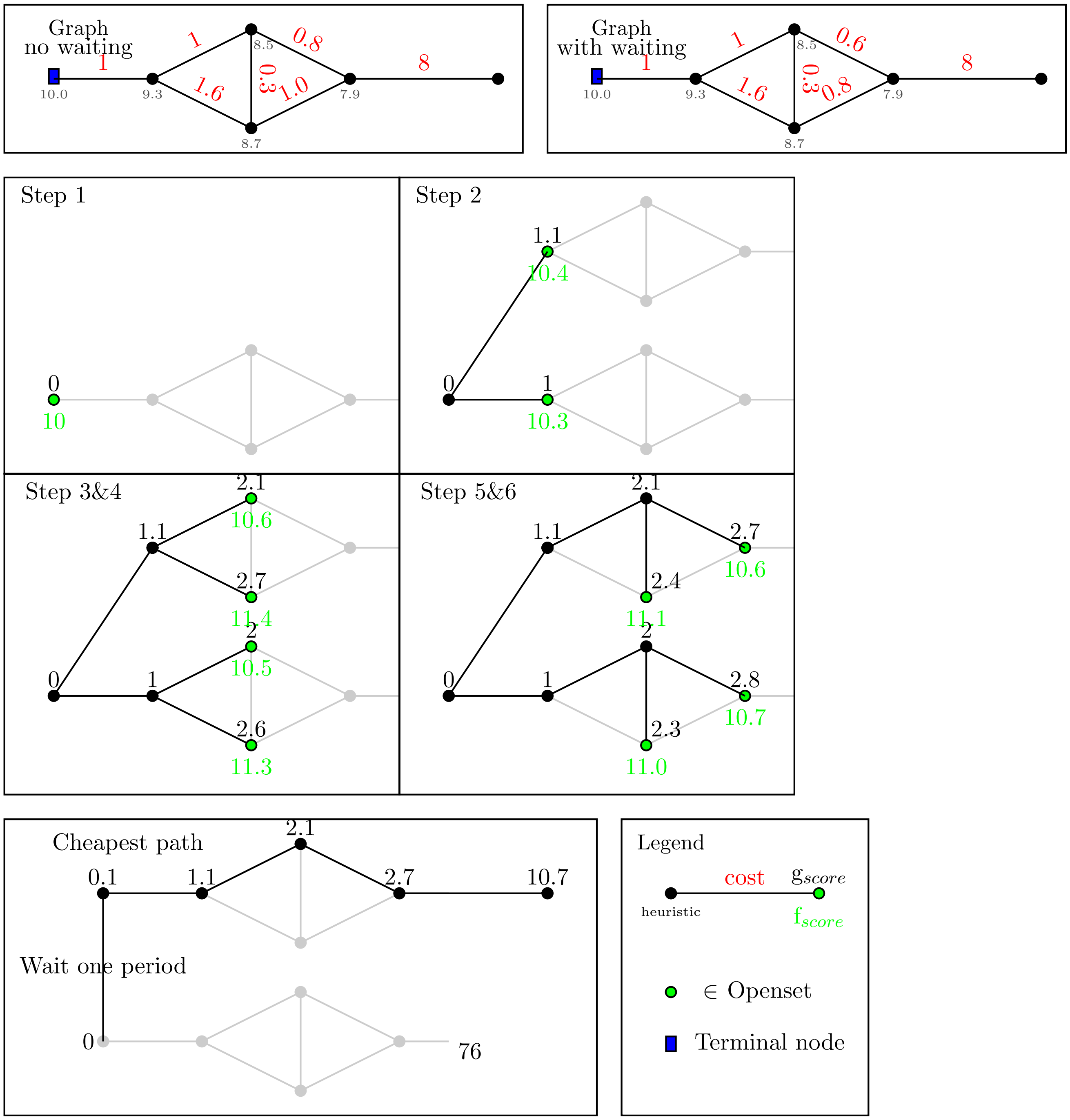


Figure 11: Route map of 105 assigned OD-pairs for sensitivity truck costs, these costs per minute equal €0.50, for section 6.3.1.

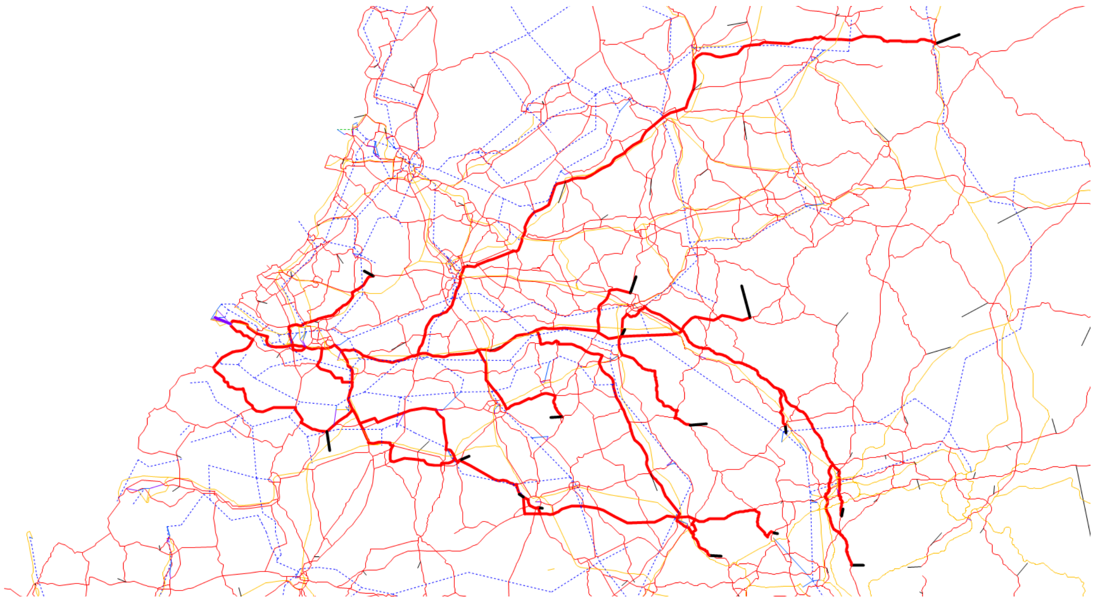


Figure 12: Route map of 105 assigned OD-pairs for sensitivity truck costs, these costs per minute equal €1.50, for section 6.3.1.

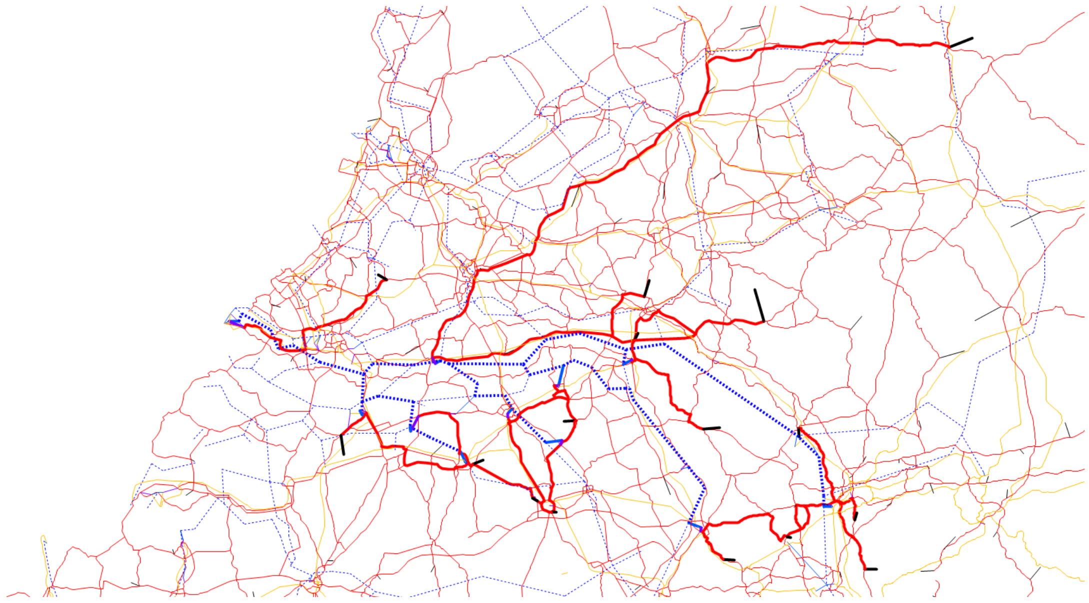


Figure 13: Route map of 105 assigned OD-pairs for sensitivity terminal costs, these costs per minute equal €15, for section 6.3.1.

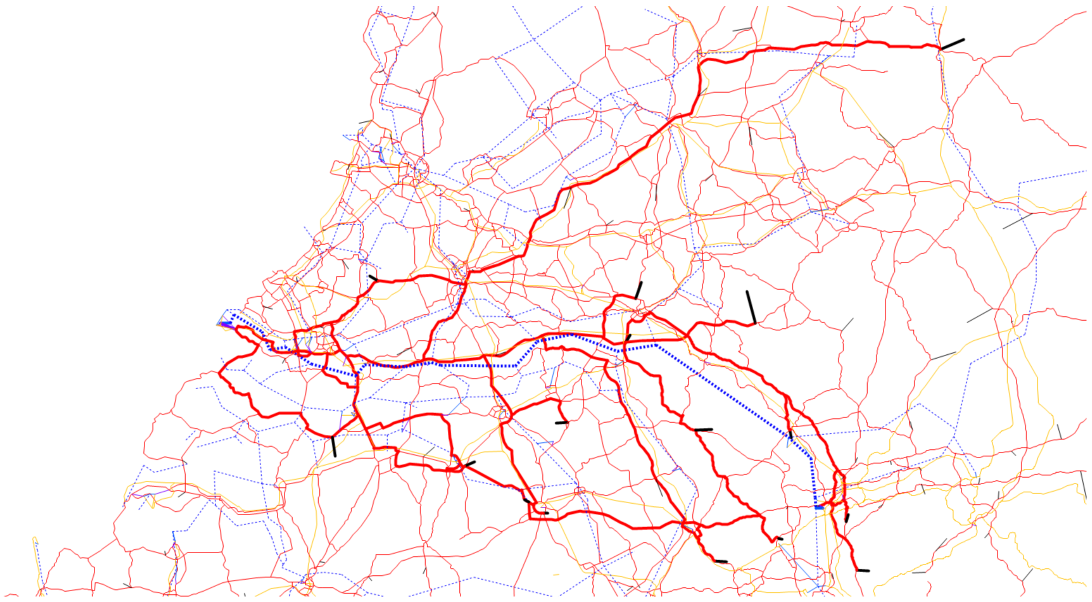


Figure 14: Route map of 105 assigned OD-pairs for sensitivity terminal costs, these costs per minute equal €5, for section 6.3.1.