MASTER THESIS REPORT

Realistic route choice modeling

UNIVERSITY OF TWENTE.

Author:

M.G. TELGEN BSc

Date:

June 17, 2010

Supervisors:

DHV:
Ir. M.C.VERKAIK-POELMAN

Civil Engineering and Management:
Prof. Dr. E.C. VAN BERKUM
Dr. R.PUEBOOBPAPHAN

Applied Mathematics:
Prof. Dr. R.J. BOUCHERIE
Dr. J.B.VINK-TIMMER
Dr. Ir. W.R.W. SCHEINHARDT

Acknowledgements

A human life is a chain of decisions, important decision, but also day-to-day decision like what to wear. This year, I made a lot important decisions. I decided to move to Utrecht and moved in with my boyfriend Niels. I decided to do my confession of faith, and this year I decide to which jobs I want to apply. All important decisions, which I could not make if I did not get the opportunities for them. I am grateful for all the opportunities I received. I would like to thank the people responsible for the opportunities I received in regarded to performing this master thesis project.

First of all I thank my parents for supporting me. Thank you for supporting me in my decisions, and for giving me all the opportunities to follow two study programs.

I would like to thank Muriel Verkaik-Poelman for supporting me during the master thesis project. You did not only represent the DHV vision on my research project, but you also kept in mind what is reasonable to expect from me, and helped me with the problems I encountered.

I would also like to thank my supervisors of the University of Twente. Eric van Berkum thank you for helping me finding a master thesis project, and advising during the research project. Rattaphol Pueboobpaphan thank you for supporting me during the project, and helping me focus the research goal in a more realistic research goal. Judith Vink-Timmer thank you for supporting me in my master thesis project, and helping me with the proofs. Werner Scheinhardt thank you for reading my master thesis project, and your comments.

Finally I would like to thank Niels for supporting and advising me in my decisions. Niels I want to thank you for helping me with the English language and listen to my stories, but more importantly I want to thank you for being there for me.

In the remaining of this report I will focus on the choices rather than on the opportunities.

Summary

Traffic engineers use transport models to predict future traffic streams. The route choice model is an important step of the transport model. We observe unrealistic routes in the results of the transport models used at DHV. But DHV has never calibrated or validated their transport model with regard to the route choices. From all the steps of the transport model we expect the biggest improvement of the transport model by improving the route choice model. DHV often uses the shortest route principle. We investigate literature for other route choice models.

We found several promising route choice models besides the shortest route principle. We select the most promising route choice models on the basis of: reality, computational time, and ease of use. The C-logit and the game theory based route choice models are realistic route choice models with reasonable computational time, and are easy to use due to earlier experience with these approaches. We select both approaches for further investigation. Within the C-logit we vary different parameter values, including the scale parameter. We vary the scale parameter, because we proved that the C-logit results depend on the utility scale. The games of game theory we use are the congestion game, the demon player game, the smooth fictitious play under the Multinomial logit rule, and the smooth fictitious play under the C-logit rule. In literature, the smooth fictitious play is only described under the Multinomial logit rule. We verify the use of the C-logit by a proof of convergence of the smooth fictitious play under the C-logit rule.

We formulate our research goal to investigate these two route choice model approaches:

Investigate whether route choice models based on C-logit and game theory are able to produce more realistic route choices than the shortest route principle, and investigate the behavior of the C-logit and game theory models.

We define the measure reality to express the extent in which a variant produces realistic route choices. The reality of a route choice model is the percentage of vehicles that chooses the same route according to the route choice model and loading method as registered by data, where we assume that the transport model generates the same amount of vehicles in the network as registered. We calculate in four different tests the reality scores of the C-logit and game theory variants with different parameter settings, and the reality score of the shortest route principle. We compare the reality scores with each other to determine the influence of the parameters on the C-logit and game theory, and whether the C-logit and game theory based route choice models are able to produce more realistic route choices. We only test on short trips (less than 11 km), therefore we cannot draw conclusions about our route choice models with long trips.

The tests results make clear that C-logit and game theory based route choice models can produce more realistic route choices than the shortest route principle. We expect that the C-logit and game theory based route choice models also result in more realistic route choices than the shortest route principle for other than our tested network, because the reality scores of the most C-logit and game theory variants where much larger than the reality scores of the shortest route principle on all the tested origin destination pairs.
The parameters β, θ, and the penalty function seem not to have much influence on the C-logit results when multiple origin destination pairs are used. We expect that this result also holds for other networks when using multiple origin destination pairs, because the tests show that other parameter settings are optimal for different origin destination pairs. These differences can level each other out if the C-logit is used on multiple origin destination pairs. The power of the travel time has very small influence on the C-logit results. The commonality factor has more influence, and the results indicate that commonality factor 4 is the most realistic commonality factor. We believe that commonality factor 4 is in general the most realistic commonality factor, because commonality factor 4 scores the highest reality score on all our tests with all parameter settings.
The distance fraction has not much influence on the route choices of the congestion game, unless we use unreasonably large distance fractions. The β, θ, commonality factor, and penalty function show similar behavior with the C-logit as with the smooth fictitious play. We are not able to model the demon player game in a dynamic assignment with reasonable computational time, therefore we did not test the demon player game in a dynamic assignment.

Besides the high reality scores, the C-logit and the game theory based route choice models have another advantage. The results of the C-logit and game theory based route choice models are less vulnerable for small errors in the travel time functions than the shortest route principle. This result does not depend on the network and the travelers, and therefore holds in general. The smooth fictitious play and the C-logit result in the highest reality scores in the final test, but the smooth fictitious play leads in the first test to low reality scores. Both the smooth fictitious play and the congestion game need more computational time than the C-logit. Therefore we recommend to use the C-logit instead of the shortest route principle.

Contents

1 Introduction **11**
1.1 Research subject . . . 11
1.2 Thesis setup . . . 11

2 Problem analysis **13**
2.1 Transport models . . . 13
2.1.1 The classical transport model . . . 13
2.1.2 Categories by the assignment step . . . 16
2.2 Equilibrium situations . . . 17
2.2.1 User equilibrium . . . 17
2.2.2 Social equilibrium . . . 17
2.3 Transport models used at DHV . . . 17
2.3.1 Questor . . . 17
2.3.2 Dynasmart . . . 18
2.3.3 Aimsun . . . 18
2.4 Problems . . . 19
2.4.1 Observed unrealistic routes . . . 19
2.4.2 Calibration and validation on route choices . . . 20
2.5 Conclusion . . . 20

3 Literature review **21**
3.1 Random utility based models . . . 21
3.1.1 Deterministic models . . . 21
3.1.2 Stochastic models . . . 22
3.1.3 Related Issues . . . 25
3.2 Other Approaches . . . 26
3.2.1 Prospect theory . . . 26
3.2.2 Fuzzy logic models . . . 27
3.2.3 Possibility theory . . . 28
3.2.4 Game theory . . . 28
3.3 Selection . . . 32

4 Research outline **33**
4.1 Research goal . . . 33
4.2 Research question . . . 33
4.3 Research approach . . . 34
4.3.1 Theoretical test . . . 34
4.3.2 Setup of the Enschede tests . . . 37
4.3.3 Enschede test 1 on 1 OD . . . 39
4.3.4 Enschede test 2 on 1 OD . . . 39
4.3.5 Enschede test 3 on 10 OD . . . 40
4.3.6 Advantages and disadvantages of the approach . . . 41

5 Route choice model variants **43**
5.1 C-logit variants . . . 43
5.1.1 Commonality factor functions . . . 43
5.1.2 The θ value . . . 44
5.1.3 Travel time power . . . 45
5.1.4 Commonality factor parameters . . . 45

5.1.5 Utility expression . . . 45
5.1.6 Penalty value . . . 46
5.1.7 Road category bias . . . 46
5.2 Game theory variants . . . 46
5.2.1 Congestion game . . . 47
5.2.2 Smooth fictitious play, θ values . . . 47
5.2.3 Smooth fictitious play, β values . . . 47
5.2.4 Demon player game . . . 50
5.2.5 Penalty value . . . 50
5.2.6 Road category bias . . . 50

6 Results theoretical test **51**
6.1 Static results . . . 51
6.1.1 C-logit . . . 51
6.1.2 Game theory . . . 59
6.1.3 Conclusion static results . . . 63
6.2 Dynamic results . . . 64
6.2.1 C-logit . . . 64
6.2.2 Game theory . . . 66
6.2.3 Conclusions dynamic results . . . 67

7 Results Enschede tests on 1 OD **69**
7.1 Comparison . . . 70
7.1.1 Travel time comparison . . . 70
7.1.2 Shortest route comparison . . . 70
7.1.3 Experienced/Dynasmart comparison . . . 70
7.2 C-logit results . . . 70
7.2.1 Commonality factor variants . . . 71
7.2.2 θ value variants . . . 71
7.2.3 Travel time power . . . 72
7.2.4 β value variants . . . 73
7.2.5 Penalty variants . . . 74
7.2.6 Conclusion C-logit . . . 75
7.3 Game theory . . . 75

8 Results Enschede tests on 10 OD **77**
8.1 C-logit results . . . 77
8.1.1 Commonality factor variants . . . 77
8.1.2 θ value variants . . . 78
8.1.3 β value variants . . . 78
8.1.4 Penalty variants . . . 79
8.2 Game theory results . . . 79
8.2.1 Congestion game, p-value variants . . . 80
8.2.2 Congestion game, penalty value variants . . . 80
8.2.3 Smooth fictitious play, θ value variants . . . 81
8.2.4 Smooth fictitious play, β value variants . . . 81
8.2.5 Smooth fictitious play, penalty value variants . . . 82
8.3 Conclusion . . . 82

9 Conclusions and recommendations **83**
9.1 Conclusions . . . 83
9.1.1 Comparison with shortest route principle . . . 83
9.1.2 Influence of the parameters . . . 83
9.1.3 Validity of the results . . . 83
9.2 Recommendations . . . 84
9.2.1 Usage of our route choice variant . . . 84
9.2.2 Further research . . . 84

A Matlab files **89**
A.1 Theoretical test . . . 89
A.1.1 The network . . . 89
A.1.2 The static assignment . . . 89
A.1.3 Dynamic assignment . . . 90
A.2 Shortest route algorithm . . . 95
A.2.1 Dijkstra algorithm . . . 95
A.2.2 k-shortest route algorithm . . . 97
A.3 Enschede Test on 1 OD with experienced travel times . . . 98
A.3.1 The network . . . 98
A.3.2 Determine route choices . . . 101
A.4 Enschede test on 1 OD with Dynasmart travel times . . . 103
A.4.1 Determine route choices . . . 103
A.5 Enschede test on 10 OD-pairs . . . 107
A.5.1 C-logit variants . . . 107
A.5.2 Congestion game variants . . . 114
A.5.3 Smooth fictitious play variants . . . 115

B Dynamic assignment demon player game **117**

Chapter 1

Introduction

This report describes the master thesis project of Marthe Telgen. Marthe Telgen performed her master thesis project at DHV. DHV is active in consultancy and engineering in Europe, Asia, Africa, and North America. DHV services on the markets transportation, water, building & industry, and spatial planning & environment. Marthe Telgen performed her project at the unit environment and transportation in Amersfoort. About 120 people work in this unit. The unit is divided into four different departments: transport models, urban & regional mobility, dynamic traffic management, and mobility consultancy. Marthe Telgen performed her thesis project at the department traffic models.

1.1 Research subject

The department transport models works with various kinds of transport models. A transport model is used to predict the influences of traffic measures, construction plans, or demographical changes on traffic streams. The transport models consist of several sequential steps. The route choice is one of the steps in a transport model. Errors in the route choice step will cause errors in the prediction of the total model. DHV recognizes that the transport models currently used at DHV do not always result in realistic routes. We believe that the route choice model has the most influence on the resulting route choices of the transport models. DHV uses often the shortest route principle as route choice model.
In this master thesis project we investigate whether a C-logit or game theory based route choice model could result in more realistic route choices than the shortest route principle, and the influence of the parameters of the C-logit and game theory models on the reality of the route choices. The reality of a route choice model is the percentage of vehicles that chooses the same route according to the route choice model and loading method as registered by data, where we assume that the model generates the same amount of vehicles in the network as registered. First we investigate the literature about C-logit and game theory models to understand these models, and to develop several C-logit and game theory route choice models. Second we test these route choice models on realistic routing and compare the resulting route choices with the route choice calculated with the shortest route principle.

1.2 Thesis setup

This thesis contains two parts: the research setup, and the research results. The first part consisting of chapter 1 to 4 contains the introduction in traffic theory, the problem description, the literature review, and the research outline. Chapter 2 describes the problems DHV encounters with their transport models. To understand the background of these problems, we first give a short introduction in traffic theory, and explain the assignment, and route choice approaches of the three most used models at DHV. With the problems in mind the literature is reviewed to find promising route choice approaches. Chapter 3 describes the literature review. Chapter 4, describes the research outline, consisting of the research goal, the research question and the research approach. The section research approach describes the setup of the tests.

The second part describes the results of the research project. We test different variants of the route choice model. These variants are described in Chapter 5. To answer the research questions of Chapter 4 we perform four different tests: a theoretical test, a test with experienced travel times, a test with travel times from Dynasmart, and a test on 10 origin destination pairs. We perform the theoretical test in a controlled environment, so that we have complete insight in what is happening. This first test will give us a good feeling of the influence of the different parameter values. Chapter 6 describes the results of this test. In the test with experienced

travel times, we calculate the route choices on the basis of experienced travel times. In the test with Dynasmart travel times, we calculate the route choices on the basis of the Dynasmart travel times between the same origin destination pair as in the test with experienced travel times. We compare the experienced travel times with the Dynasmart travel times, and the route choices based on experienced travel times with route choices based on Dynasmart travel times, to test the travel time function and the influence of the travel time function on the C-logit and game theory variants. This comparison is performed, because we want to know if improving the travel time function within the transport models is needed before we investigate different route choice models with these travel times. Chapter 7 describes the results of the two tests with experienced and Dynasmart travel times. In the test on ten origin destination pairs we investigate the influence of the parameters on the route choices of ten origin destination pairs. In the case of using the route choice model on more origin destination pairs the influence of coincidences of one origin destination pair is minimized. Chapter 8 describes the results of this test.
The results of these four tests and the literature review provide us the answers on the research question. The answer, the conclusions, and recommendations of this research project are given in Chapter 9.

Chapter 2

Problem analysis

DHV uses transport models to make predictions about future traffic flows. The model makes predictions with a model of a base year. DHV recognizes that the models of the base year, and especially the route choice part of the models are not completely in consensus with the reality. This inconsistency leads to improper predictions. Therefore we need to improve the models. We notice two route choice related problems in the transport models. First, the models often show travelers taking routes we do not expect that travelers take in reality. Second even if the route choice not seems unrealistic the route choice could be unrealistic, because the base year model is rarely calibrated or validated on route choice data. Section 2.4 describes these two problems.
First, this chapter gives an introduction to traffic theory, with an explanation of the structure of transport models and an explanation of two often modeled equilibrium situation. In this research project we focus on the assignment step of the transport model, therefore the assignment step of the transport models used at DHV are explained in section 2.3.

2.1 Transport models

Traffic managers are concerned with the optimization of road network performances, improving traffic conditions, reducing the congestion, and reducing emissions. A traffic flow consists of different individual drivers with their own individual behavior. Therefore traffic managers have to understand driver's behavior. For understanding driver's behavior traffic managers use transport models. The transport models predict the deviation of the traffic flows over the road network [1]. Examples of transport models applications are: investigating the effect of changing a traffic light protocol on an intersection, and predicting effects on traffic of development plans for a new Vinex location.

2.1.1 The classical transport model

Most transport models follow the structure of the classical transport model. The classical transport model is a four-step model [2]. The four steps of the classical model are: trip generation, trip distribution, modal split and the traffic assignment. The classical model is displayed in figure 2.1. Next to the four steps, the figure displays feedback loops at the left side of the steps, these loops present iterations within the transport model. The steps and the feedback arrows are explained below with the use of an example. *The example is printed italic.*

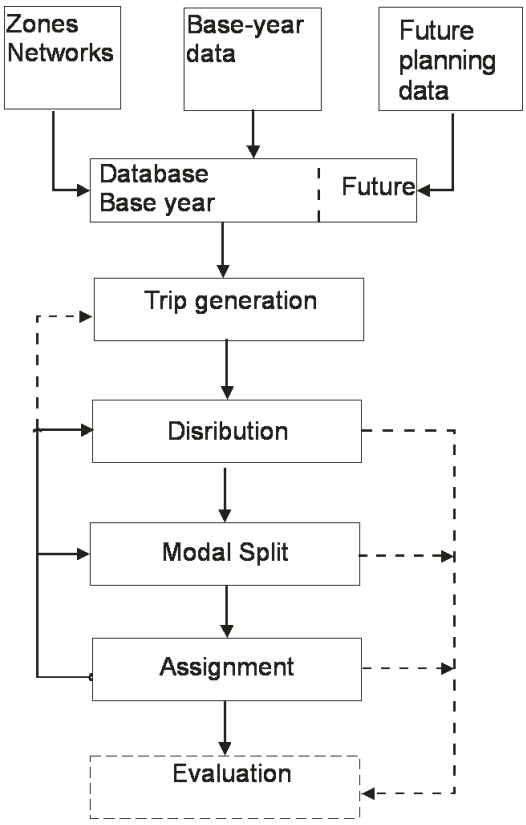


Figure 2.1: The classical transport model

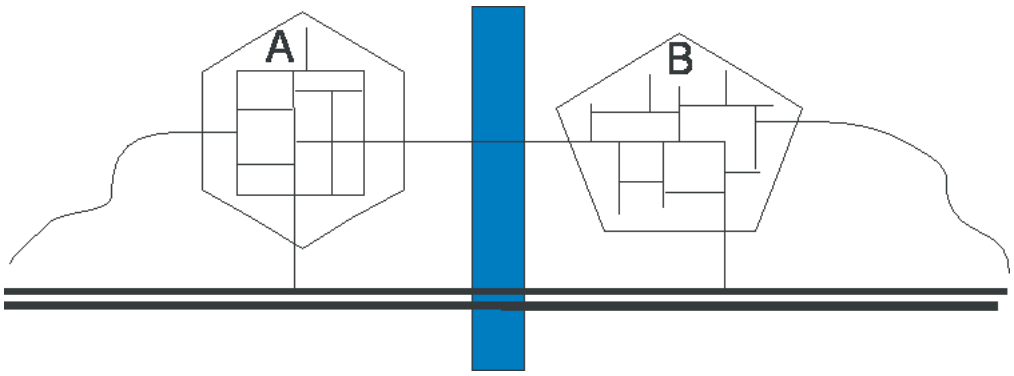


Figure 2.2: Plan of the example

Before starting the four steps, the network needs preparation. We divide the network into zones and collect, calibrate, and validate data about the zone and its inhabitants [2]. The number of zones depends on the required level of detail of the model, and on the level of detail of the available data. Within a zone we specify only one centroid. A centroid is a location within the zone, which is the origin of trips from this zone and the destination of trips to this zone.

Village A and B are separated by a canal. A sketch of the environment of the villages is displayed in figure 2.2. At the south side of the villages a highway connects the two villages. The centers of the villages are connected with a rural road. The city councils of villages A and B want to know how many people will use the bridge on the rural road in 2020. To predict this amount the city councils provided results of a traffic research made in 2000. The city councils expect that village A will have 25% more inhabitants (from 200 to 250) in 2020 than in 2000, further they expect village B to keep the same amount of inhabitants (250 inhabitants). For simplicity of the example, we divide the network into four zones. In figure 2.3 the deviation into zones is shown, the dotted line represent the zone border. The circle in each zone represents the centroid. We define two zones outside the village A and B, to account for the traffic from outside the study area. We also number the links.

After the preparation of the network the first step, the trip generation starts. In the first step the attraction and production of traffic in each zone is determined. The production and attraction of trips is determined with socio-economic properties. For instance the number of households, or number of inhabitants in a zone for production, and the surface of office blocks, or number of full time employees for attraction. In general the production and attraction of trips with different purposes is identified separately. Often used trip purposes for modeling are: working trips, education trips, shopping trips, social and recreational trips and other trips [2].

The trip attraction and production in the year 2000 is shown in table 2.1, this is a result of the traffic research of 2000. We expect that every inhabitant of village A in 2020 will produce 3 trips and attracts 2 trips and that the inhabitants of village B in 2020 will produce 2 trips and attract 3 trips. Further, we assume centroid C and D produce and attract 100

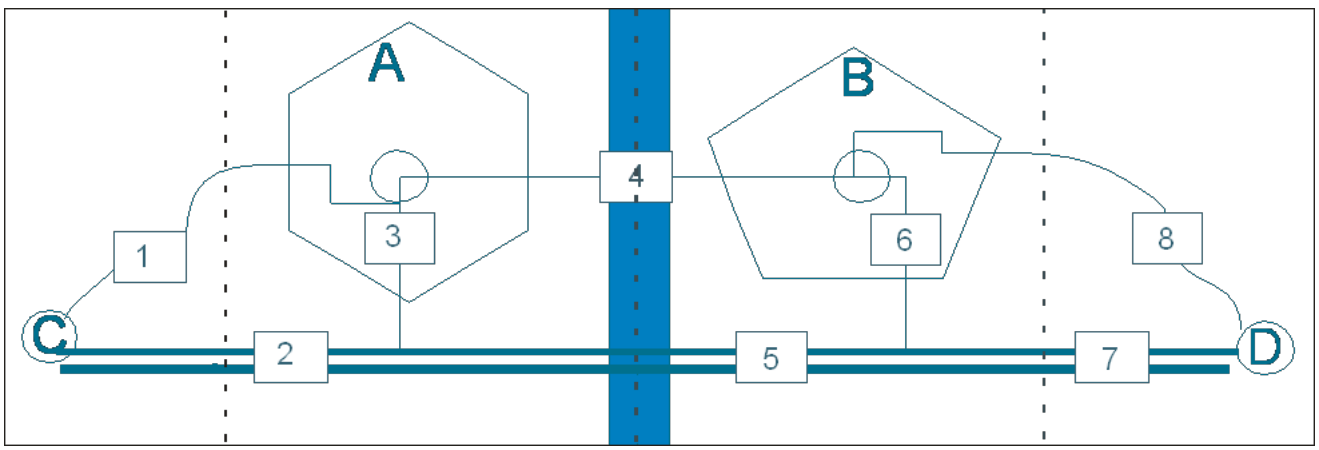


Figure 2.3: The example divided into zones

(a) 2000

Zone	Production	Attraction
C	100	150
A	600	400
B	500	600
D	100	150

(b) 2020

Zone	Production	Attraction
C	100	100
A	750	500
B	500	750
D	100	100

Table 2.1: The production and attraction figures

trips. The attraction and production numbers for the year 2020 are given in table b of table 2.1.

In the second step (the distribution step) an Origin-Destination matrix (OD matrix) is developed. The number in cell (i,j) of the OD matrix is the amount of trips with origin i and destination j. The number of trips that start and end in each zone (calculated in the first step) are the row and column sums of the OD matrix. The inner cells of the OD matrix can be estimated with many different methods, for example growth factor methods or the gravity model. Growth factor methods can only be used if a base year OD matrix is available of the study area, because this method consists of multiplication on the old matrix. We use and explain the growth factor method in the example below. The gravity model takes into account the distance (or travel costs) between zones to determine the OD matrix. The gravity model can be written as: $T_{ij} = \alpha O_i D_j f(c_{ij})$. Where T_{ij} is the number of trips from zone i to zone j, α a proportionality factor, O_i the amount of trips with origin i, D_j the amount of trips to destination j and $f(c_{ij})$ is a generalized function of the travel costs. The gravity model should be calibrated on a base year OD matrix, not necessarily of the study area [2].

(a) 2000

	C	A	B	D
C	0	30	70	0
A	100	100	350	50
B	50	250	100	100
D	0	20	80	0

(b) 2020

	C	A	B	D
C	0	31.39	68.65	0
A	71.5	149.55	490.64	38.55
B	28.5	298.00	111.73	61.45
D	0	21.07	78.99	0

Table 2.2: The OD matrices for 2000 and 2020

The OD matrix of the year 2000 is determined in the research of 2000, therefore we use the growth factor method. The OD matrix is given in table 2.2. The OD matrix for 2020 is calculated using the Furness method which is a growth factor method. Furness incorporates growth rates into new variables a_i and b_j: $T_{ij} = t_{ij} a_i b_j$ where t_{ij} is the number of trips from i to j in the OD matrix of 2000 and T_{ij} is the estimated amount of trips from i to j for 2020. Furness proposes to fix b_j with value 1 first and find the a_i that satisfies the amount of production found in the trip generation step. Next fix a_i on the identified value and find the value of b_j that satisfies the amount of attracted trips found in the trip generation step. Repeat these proceedings until the changes are sufficiently small [2]. After six steps the OD matrix given in table 2.2 is found.

The trips of the OD matrix are made by different transport modes. A trip can be made for example by bike, by car, by train, or by bus. Therefore a third step, the modal split is needed, the trips are divided among the available modes. Sometimes the modal split is simultaneously performed with the generation and distribution step.

In the example 90% of all the trips are made by car, the others by bike. Our interest is only the car trips. The OD matrix for car trips is given in table 2.3.

	C	A	B	D
C	0	28.25	61.78	0
A	64.35	134.60	441.57	34.69
B	25.65	268.20	100.56	55.31
D	0	18.86	71.09	0

Table 2.3: The estimated car OD matrix of 2020

The fourth step of the classical model is the assignment step. The assignment step consists of a route choice part, and a network loading part. In the most assignment methods the travel times are recalculated after the network loading part and a new iteration with route choice and network loading is started, see also figure 2.4.

In this research project, we focus on this assignment step, because we believe that the route choice results of a transport model are most influenced by the assignment step.

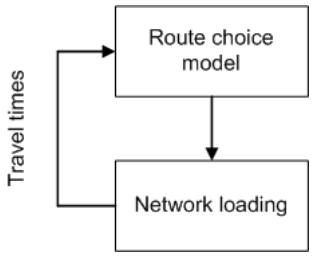


Figure 2.4: The process of the assignment step

We use a static assignment. The trips are assigned using an All or Nothing assignment, which means that the network loading is performed once and all the trips are assigned to the 'shortest' path of their OD pair. The link costs are given in table 2.4, we used the link numbering of figure 2.3. We will explain the route choice between A and D. Travelers between A and D can take four different routes: on links 3-5-7, links 3-5-6-8, links 4-8, or links 4-6-7 (Traveling via zone C is obvious longer so routes via C will not be considered). The travel cost on these routes are respectively 63, 78, 72 and 69. Therefore the route 3-5-7 is the shortest route and is chosen. The total link flow on the bridge with the rural road between village A and B is 442+268=710, because travelers from A to B, and B to A choose a route with link 4.

1	2	3	4	5	6	7	8
22	15	8	40	32	6	23	32

Table 2.4: The link costs

After the assignment step the model is evaluated. If needed the model has to be adapted and calculated again. It is possible to adapt every step or even to adapt just one step. In every step, estimation errors are made and the results of every step should be validated and/or calibrated. However, if there is data available to calibrate or validate the model, the data usually consists of link counts. Route choice data or real OD matrices are very rare. The following example explains that even when there is link count data available calibrating on link counts will not ensure that the model is correct.

Consider our example study area. Suppose in 2020 we measure a link flow of 700 on the rural road bridge, which is close to the estimated 710 vehicles. In the model the 710 vehicles consists of travelers from A to B and B to A, but maybe the counted 700 vehicles consists of vehicles from zone A to B and A to D. In that case, our model does give proper link flows, but results in wrong traffic movements. Unfortunately, we do not notice the wrong traffic movements by calibrating on the link counts. Also we have to be aware of the measure errors incorporated in link counts.

2.1.2 Categories by the assignment step

The transport models are categorized by two different properties of the assignment step. One of these properties is the scale of the assignment step. Transport models investigate traffic situations on three different assignment scales: macroscopic, mesoscopic, and microscopic. Macroscopic traffic flow research investigates the traffic flow in total on network level. Microscopic traffic flow research investigates the traffic flows at vehicle level, for instance the interaction between vehicles at corridor level [3]. Transport models with mesoscopic assignment steps operate between marcoscopic and microscopic models.
Categorization into static and dynamic assignment models is the second type of categorization. In dynamic models, the traffic flows vary in time due to different traffic demands for different time periods and other dynamic characteristics. Dynamic models are used to predict traffic flows during successive short periods (e.g. each quarter). Dynamic models can incorporate effects of congestion on route choice. Static models are used to determine the day flows and do not incorporate blocking back effects. The static models are computationally more easy than dynamic ones and run faster. The classical transport model can be used with assignment steps of different scale, dynamic assignment steps, and static assignment steps.

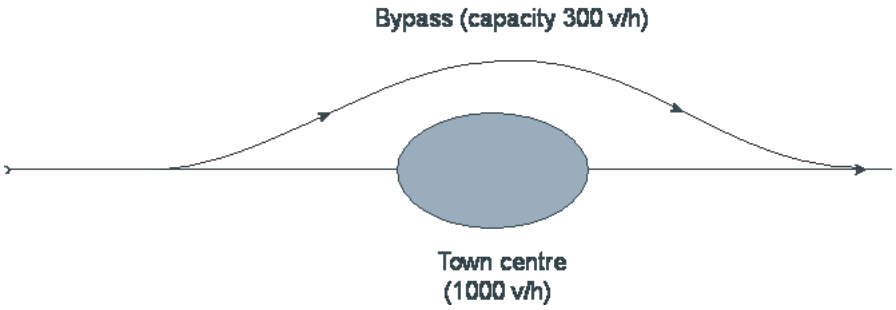


Figure 2.5: A town served by a bypass and a town center route

2.2 Equilibrium situations

In traffic research two special situations are often investigated: user equilibrium and social equilibrium. Traffic flows are stable in such situation. It is often tried to model an equilibrium situation because it is generally assumed that in normal conditions the reality approaches the user equilibrium, and because traffic managers want to realize the social equilibrium.

2.2.1 User equilibrium

Consider the town of figure 2.5. The road through the center has a capacity of 1000 vehicles per hour and the bypass has a capacity of 3000 vehicles per hour. Assume that during the morning peak 3500 vehicles per hour approach the town. Everyone would like the shortest route which is via the town center. It is clear that this road becomes congested when everyone chooses this road through the center. Many would opt for the bypass to avoid the long queues and delays. It is expected that drivers will vary their route on different days, until they find a more or less stable arrangement when none can decrease their travel time by switching to another route. This is a case of the user equilibrium, also called Wardrop's equilibrium after his first principle. Wardrop's first principle states: *Under equilibrium conditions traffic arranges itself in congested networks such that all used routes between an O-D pair have equal and minimum costs while all unused routes have greater or equal costs* [2] and [4]. This principle is based on the assumption that individual travelers are trying to minimize their travel costs.

2.2.2 Social equilibrium

The social equilibrium is based on Wardrop's second principle. His second principle states: *Under social equilibrium conditions, traffic should be arranged in congested networks in such a way that the average travel cost is minimized.* The social equilibrium is a target for traffic managers, who try to minimize travel costs. In principle traffic will not arrange itself according to the second principle, but the traffic will arrange itself following an approximation to Wardrop's first principle, the 'selfish' or users' equilibrium. To achieve a social equilibrium, travelers are assigned to links considering the marginal cost an additional traveler induces on link travel times [4].

2.3 Transport models used at DHV

DHV mainly uses three different computer programs as transport models: Questor, Dynasmart, and Aimsun. Questor is used for macroscopic static modeling, Dynasmart for mesoscopic dynamic modeling, and Aimsun for microscopic dynamic modeling. We believe that the assignment step of the transport model has the most influence on the route choice results, therefore we explain the assignment procedure of the three computer programs used by DHV.

2.3.1 Questor

In Questor travelers follow their perception of the cheapest route. The link costs are based on the travel time (delays can be incorporated) and travel distance. There are four static assignment methods possible: All or Nothing, Incremental, User Equilibrium and Stochastic User Equilibrium. In the All or Nothing assignment only one loading iteration is used. Every traveler is assigned to the shortest route, determined at the start of the assignment process. The latter three assignments are Capacity Restraint Assignments, they take congestion effects into account on road sections and intersections [5]. The Incremental assignment uses multiple

assignment iterations [6]. A fraction of the traffic is loaded on the network, the route costs are calculated and a smaller fraction is deloaded from the network. This process is repeated until the sum of the loaded minus the deloaded traffic is 100%. In the User Equilibrium assignment a fixed number of assignment iterations is used to approach the user equilibrium, at the end of each iteration step route costs are calculated. A fraction $1 - \alpha$ of the traffic is assigned according to the previous iteration step and a fraction of α is assigned according to the new derived route costs. If equal weights are used in each iteration step $\alpha^{(n+1)} = \frac{1}{n+1}$, than the Method of Successive Averages (MSA) is obtained. A more efficient approach to reach the local minimum, the user equilibrium, is the Frank-Wolve algorithm. In the Stochastic User Equilibrium assignment, perceived travel costs instead of real travel cost are used. A random term is added to the link costs [5].
DHV works both with the Incremental and the User Equilibrium assignment. The Incremental assignment is a loading method. This loading method is used in combination with the shortest route principle in Questor.

2.3.2 Dynasmart

In Dynasmart travelers also follow their perception of the cheapest route or the k-cheapest routes. The perception depends on the information available for the travelers. Dynasmart calculates the link costs on the basis of the travel times and toll costs. The user of the model defines the frequency of the route cost calculation. DHV chooses to calculate the cheapest route every simulation minute. A highway bias influences the link costs, to represent the preference for driving on highways. A highway bias of 0.15 is used to multiply the highway costs by 0.85. DHV uses a highway bias between 0.15 and 0.4. Two assignment procedures are possible in Dynasmart: One Shot assignment and Iterative Equilibrium assignment. In the One Shot assignment the travelers choose the shortest route on the basis of their perceived route costs and their behavior components. In the Iterative Equilibrium assignment the travelers are assigned to the network with the MSA. The MSA stops after a fixed number of iteration or when a preferred converge is reached.
In both assignment procedures Dynasmart assumes that the vehicles are fully informed about the link costs at the beginning of the trip. However not every traveler updates their calculation of the shortest route. Dynasmart has five different user classes of vehicles. Three user classes are used in the One Shot assignment. Within these user classes the vehicles calculate the shortest route at the same instance. These user classes are:

- User class 1: These vehicles calculate the shortest route at the start of the trip. They follow this trip unless a VMS **obliges** something else.
- User class 4: These vehicles calculate the shortest route at the start of the trip and every minute during their trip. Vehicles of user class 4 can change their route during a trip.
- User class 5: These vehicles calculate the shortest route at the start of the trip. They follow this route unless a VMS **advises** something else.

User classes 2 and 3 are used at the iterative equilibrium assignment. User class 2 consists of travelers following routes according to the social equilibrium. Travelers of user class 3 follow the routes according to the user equilibrium. In the equilibrium assignment the traffic is a mix of user class 2 and 3 users, the fractions of the mix are defined by a parameter [7].
The Iterative Equilibrium assignment takes much more computation time than the One Shot assignment [8], therefore DHV mainly performs One Shot assignments.

2.3.3 Aimsun

Different from Dynasmart and Questor, the route choice model in Aimsun can use beside a user equilibrium also discrete choice models. Aimsun accounts cost for each link just like the other models. The link costs are based on travel time, the road capacity and sometimes a turning penalty (the Aimsun user chooses of which elements the link costs consist). First the initial costs with free flow speeds are calculated, except when there is a warm up period. In case of a warm up period, the initial cost are calculated at the end of the warm up period.
In the discrete route choice model assignment a simulation starts for a interval defined by the model user. In the simulation the travelers choose between the k-shortest routes and known predefined routes, with a route choice model. Aimsun has four different discrete route choice models: Multinomial logit, C-logit, Proportional method, and user defined route choice. In the user defined route choice the user defines the probability of use for known routes and the k-shortest routes. In the proportional method the route is proportional chosen to the route costs. The probability of choosing alternative i for individual n in the proportional method is:

$$P_i = \frac{CP_i^{-\alpha}}{\sum_{j \in K_n} CP_j^{-\alpha}}$$

Where CP_j represents the route costs of route j and K_n the set of alternatives for individual n. The next chapter explains the Multinomial logit and the C-logit model. After assigning the predefined fraction, the cheapest paths are recalculated. Guided vehicles are provided with this new information and they are assigned to the new determined shortest route. Then the iteration starts again with simulating the next fraction of vehicles.
Aimsun uses the Method of Successive Averages in the dynamic user equilibrium case. In the first iteration of the MSA the total amount of traffic is assigned to the cheapest route. After the first iteration, at iteration n a fraction $1/n$ is removed from the network and reassigned to a recalculated cheapest route. The algorithm stops after a fixed number of iterations or when the difference between the cheapest and the used routes is below a predefined acceptance gap (the user equilibrium is reached if the gap is zero) [9].
DHV uses the discrete choice assignment with the default settings of the C-logit route choice model. The default settings are used, because it is not know which settings leads to the most realistic results.

2.4 Problems

We notice two problems of the transport models at DHV. First the transport models often show travelers taking routes that we do not expect that travelers take in reality. Second the transport models are never calibrated or validated on the route choices, therefore we expect that unrealistic routing takes more often place than we observed. This section describes both problems.

2.4.1 Observed unrealistic routes

Observed unrealistic route choices seems to happen only in congestion situation in the three programs. Below three examples of observed unrealistic route choice are described.

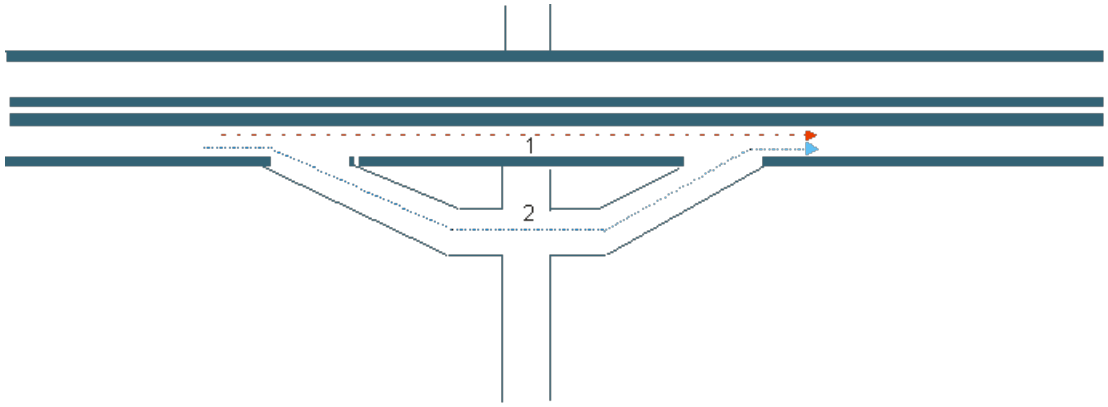


Figure 2.6: Situation sketch of possible problem point for transport models

Consider the network in figure 2.6, the figure shows a highway with its on-ramp and off-ramp. Suppose congestion occurs at the highway. Travelers are directed to the shortest route at the start of their trip. During congestion on the highway, the shortest route could be: from the highway to the off-ramp, immediately to the on-ramp, and again to the highway (route 2). When many travelers are directed via the off- and on-ramp, congestion occurs at the off- and on-ramp. Travelers starting their trip when congestion occurs at the off- and on-ramp, are directed on the highway. It takes some time between the first travelers directed to the off- and on-ramp and the first time that congestion on the the off- and on-ramp. In this time all travelers are directed to route 2, a major fraction of them reach the off-ramp when congestion already occurred. The size of the congestion on the on-ramp will increase due to these travelers, while maybe the congestion on the highway is already disappeared. Route 2 is an unrealistic route from the moment congestion occurs at the off- and on-ramp. A similar problem occurs at a short parallel road between two sequential off-and on-ramps. Small adaptations to the model could solve this unrealistic routing. Traffic on the off-ramp can be prohibited to drive to the on-ramp. Unfortunately, sometimes it is possible to go left or right at the end of the off-ramp and turn near by and go to the on-ramp. In that case the interference makes the problem worse. This problem occurs in all transport model programs, but most often in Dynasmart. Dynasmart takes into account the blocking back effects of congestion (in contrast to Questor) and assigns all the traffic to one route in one assignment iteration (in contrast to Aimsun). Therefore, Dynasmart and Aimsun define more early another route as the cheapest than Questor. Dynasmart assigns more traffic at once to this route than Aimsun.
A second problem only occurs in Dynasmart with the user class 4 travelers. These travelers can change their route during their trip. Some of these travelers change their route very often during their trip. These travelers

zigzag through the network which seems very unrealistic.
A third problem occurs at Aimsun. The inner lanes of multiple lane-roundabouts are just a little bit shorter than the outer lane. Therefore travelers choose for the left lane and the right lane is not used. This behavior causes an unrealistic traffic situation.

2.4.2 Calibration and validation on route choices

We do not know to which extend the models used at DHV result in realistic route choices. DHV calibrates the transport model by adapting the OD matrix until the results will fit (within some boundaries) the traffic counts. As mentioned in section 2.1, the origin and destination of counted travelers are unknown. Calibrating on traffic counts will not ensure that counted travelers have the same origin and destination as the travelers in the model on these links have. Besides that, the results of the transport model depend on more steps than the trip distribution and errors can occur at every step (and at counting the traffic). We expect that improving one of the steps will lead to more realistic transport models. The route choice process has a major influence in the assignment step and is therefore an important aspect. DHV has never calibrated or validated the route choice model and we observe some unrealistic routing. Therefore we expect the biggest potential improvement of the transport model by improving the route choice process.

2.5 Conclusion

DHV mostly uses the user equilibrium assignment or the incremental assignment with the shortest route principle in Questor, the One Shot assignment with the shortest route principle in Dynasmart, and the method of successive averages with the default settings of the C-logit in Aimsun. The default settings are used, because it is not know which settings result in the most realistic results. We observe that the current transport models of DHV result in unrealistic routes in congestion situations. We do not know to which extent the transport models result in more unrealistic routing, because the route choice models are never validated and calibrated. Therefore an investigation of route choices predicted by the transport models is needed. We expect that an improvement of the route choice model influences the route choices determined by the transport model the most. The shortest route principle is most often used as route choice model at DHV, we investigate in the next chapter other possible route choice models.

Chapter 3

Literature review

This chapter describes existing route choice models and approaches. Section 3.1 describes deterministic random utility based models, stochastic random utility based models, and related issues of the random utility based models. Section 3.2 describes other route choice models. In this master thesis project we investigate two of the described route choice models: the C-logit and game theory. Section 3.3 explains the choice for these two route choice models.

3.1 Random utility based models

The most common theoretical framework for generating discrete choice models is the random utility theory. In discrete choice models, travelers select from a finite set of alternatives, called a choice set [10]. The random utility theory is based on the following four properties:

- Individuals act rational, and are fully informed about the existence of links and their travel times.
- There is a set A of available alternatives, and a set X of vectors of measured attributes of the individuals and alternatives.
- Each alternative i has associated a net utility U_{in} for individual n. The modeler does not possess complete information about all the elements considered by the individual making the route choice decision. The modeler assumes that U_{in} can be represented by two parts, a deterministic and random part: $U_{in} = V_{in} + \epsilon_{in}$.
- The individual selects the alternative with maximum utility [2].

In general the utility is expressed in a cost function, with travel time as the most important component. Basically, the random utility based models can be divided into two groups: deterministic and stochastic route choice models. Deterministic route choice models always generate the same set of paths for an OD-pair. Most of the deterministic models can be made stochastic by using random generalized cost for the shortest path computations [11]. Stochastic methods generate an individual (or observation) specific subset. Stochastic route choice models based on the random utility theory are: Multinomial logit, C-logit, PS logit, Nested logit, and Cross Nested Logit.

3.1.1 Deterministic models

Deterministic route choice models assume that the travelers have full knowledge about the links and their 'costs' in the network. A group of uniform travelers will therefore contribute the same link cost to a link according to deterministic route choice models.

Shortest Path

Most models of the deterministic group are based on the shortest path principle. In the shortest path principle, travelers are assumed to minimize on one variable or a mix of variables, for instance travel time or distance. At Questor a mix of travel times, toll costs, and distance expressed in costs is minimized. All travelers will choose the 'shortest' route.

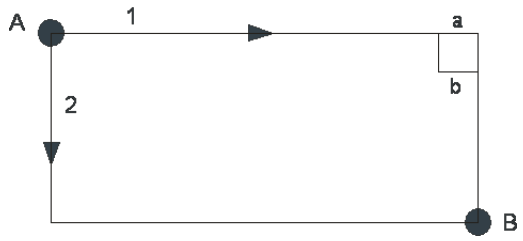


Figure 3.1: Simple network with a path choice problem

Labeling approach

The labeling approach is another deterministic route choice model [11]. The labeling approach assumes that different travelers minimize different attributes. Some travelers may wish to minimize travel time, while other travelers feel uncomfortable driving on dangerous roads and avoid curvy roads. Each of these criteria may correspond to a different road being preferred, and thus, each route can be labeled with a different criterion for which it is the optimum. Examples of labels are: time, distance, fuel, scenery, traffic lights and congested travel. The labeling approach is used in combination with the Nested logit model when paths have multiple labels [4].

3.1.2 Stochastic models

Most stochastic route choice models are a member of the Generalized Extreme Value (GEV) family. GEV members described in this section are: Multinomial logit model, the C-logit, the Path Size logit, the Nested logit, and the Cross Nested logit. Mc Fadden derived the basic GEV model from the random utility theory [12]. The GEV model assumes that the random term in the utility function follows the Gumbel distribution. In the GEV models a choice set is defined. The choice set C_n consists of all available alternative routes for individual n.

The Multinomial logit

The Multinomial logit model (MNL) is the most widely used choice model, due to its simple mathematical structure and ease of estimation [13]. According to the random utility theory the traveler choose his route from the choice set based on the net utility U_{in}. We as modelers do not have complete information about this net utility and try to describe the net utility by a deterministic part V_{in} and a random part ϵ_{in}. In the Multinomial logit is assumed that the random part can be described by the exponential Gumbel distribution. The probability of choosing alternative i from choice set C_n in the multinomial logit model is therefore described by:

$$P(i|C_n) = \frac{e^{\theta V_{in}}}{\sum_{j \in C_n} e^{\theta V_{jn}}}$$

Where θ is a scale parameter and V_i the utility of route i. The utility of the route V_{in} is mostly expressed by the negative travel time, Section 3.1.3 discusses the utility expression in more detail. The exponential in the probability distribution ensures that large differences in travel times are highly recharged in the route choice probabilities.
A special property of the MNL is the Independent of Irrelevant Alternatives (IIA) property. Independent of Irrelevant Alternatives means that the ratio of the probabilities of any two alternatives is independent of other alternatives, so: $\frac{P(i|C_1)}{P(j|C_1)} = \frac{P(i|C_2)}{P(j|C_2)}$ [12]. First, this property was considered as an advantage of the model, due to the property it is possible to forecast the share of a new alternative that is not present at the calibration stage if the attributes are known [2]. However this property has also a disadvantage, the disadvantage is explained below.

Someone is traveling from A to B in the network of figure 3.1. The travel times of route 1a, 1b and 2 are t minutes. Suppose that the utility is entirely based on travel time and that route 1b does not exist yet. According to the MNL the traveler will choose with probability $\frac{e^t}{2e^t} = 0.5$ *route 1a and with probability 0.5 route 2. Now the road b is finished and a new alternative is created. This new route has also travel time t. The new route choice probabilities are: 0.33 for all the three alternatives (Indeed the IIA property holds). Clearly we would expect probabilities around 0.25 for route 1a and 1b and a probability around 0.5 for route 2 [12].*

The C-logit

The basic idea of the C-logit is to deal with similarities among overlapping paths (like route 1a and 1b in figure 3.1) through an additional "cost" attribute named commonality factor (CF_i) [14]. Cascetta, Nuzzolo, Russo and Vitetta (1996) proposed the C-logit model, to maintain the computational simplicity of the logit form but produce more intuitive forecasts of route shares especially when path overlapping occurs. In contrary to the Multinomial logit consists the deterministic part of the net utility of two elements $V_{in} - CF_i$ instead of one V_{in}. The probability that an individual n chooses route i is formulated in the C-logit model by:

$$P(i|C_n) = \frac{e^{\theta(V_{in}-CF_i)}}{\sum_{j\in Cn} e^{\theta(V_{jn}-CF_j)}}$$

The scale factor, θ scales the effect that differences in systematic utilities ($V_{jn} - CF_j$) have on the travelers decision, the larger θ the more influence of the differences. The commonality factor (CF_i) of individual n for route i is proportional to the path overlap. Heavily overlapping paths have larger commonality factors and thus a smaller systematic utility with respect to similar, but independent paths. If path i is made up of links belonging exclusively to that path, then CF_i is equal to zero [14]. In literature, we found five different forms of the commonality factor:

$$CF_i = \beta \ln \sum_{j\in C_n} \left(\frac{L_{ij}}{\sqrt{L_i L_j}} \right)^{\gamma} \tag{3.1}$$

$$CF_i = \beta \ln \sum_{a\in\Gamma_i} \frac{l_a}{L_i} N_a \tag{3.2}$$

$$CF_i = \beta \sum_{a\in\Gamma_i} \frac{l_a}{L_i} \ln N_a \tag{3.3}$$

$$CF_i = \beta \ln \left(1 + \sum_{j\in C_n, j\neq i} \frac{L_{ij}}{\sqrt{L_i L_j}} \frac{L_i - L_{ij}}{L_j - L_{ij}} \right) \quad [4] \tag{3.4}$$

$$CF_{in} = \beta \ln \sum_{a\in i} w_{ai} N_a \quad [14] \tag{3.5}$$

Where β and γ are coefficients that weight the commonality factor, L_{ij} is the common length of path i and path j, Γ_i is the set of arcs in path i, and l_a the length of link a. N_a is the number of paths connecting the same OD pair which share link a, with $N_a = 1$ for centroid connectors [4], and w_{ai} is the proportional weight of link a for path i [14]. Formulation 3.5 is similar to formulation 3.2 if w_{ai} is chosen according to the distances. Larger values of β causes higher influence of the overlapping constant with respect to the utility. The influence of γ is smaller than β and has the opposite effect. The parameter γ is usually taken in the range [0,2] [9].
Formulation 3.1 is the only symmetric formulation. Symmetric means that the order in which routes i and j are considered, does not influence the value of the commonality factor. In the other formulations, differences in route length will lead to an asymmetry [15].

In literature is not described which form of the commonality factor is the best formulation. There is a lack of theory or guidance to which form of commonality factor should be used [4]. Formulation 1 shows the most similarity with the Probit model, which we suspect to be realistic, but suffers from computational difficulties. Cascetta, Nuzzolo, Russo and Vitetta (1996) published the results of a calibration of the C-logit model with formulation 3.5. They calibrated the C-logit model for heavy truck route choice on the Italian national network. The calibration, with 1471 observations, shows that the usage of the commonality factor leads to significant improvements. They also found that the C-logit performs better when a limited number of alternatives with comparable path costs is considered [14]. Therefore, we advice to use the C-logit model in combination with a path generation algorithm which produces a limited number of such paths.
At DHV C-logit can be used in Aimsun. Formulation 3.1 is used in Aimsun. The default values in Aimsun are: $\theta = 60$, $\beta = 0.15$, and $\gamma = 1$. DHV uses always the default values, although it is not know if other values result in more realistic results.

Path Size logit

The Path Size logit (PS logit) also copes with overlapping paths. Like the C-logit, PS logit adds a correction term, the path size PS, to the utility function. The probability function is of the PS logit model is:

$$P(i|C_n) = \frac{PS_{in} e^{V_{in}}}{\sum_{j\in Cn} PS_{jn} e^{V_{jn}}}$$

Where PS_{in} is the size of path i for individual n. Path-Size logit was introduced by Ben-Akiva and Ramming (1998), who presented the following formulation for the path size:

$$PS_{in} = \sum_{a \in \Gamma_i} \frac{l_a}{L_i} \frac{1}{N_{an}}$$

The term (l_a/L_i) is a weight corresponding to the fraction of path impedance coming from a specific link. The term N_a is like at the C-logit the amount of paths using link a. This term is not affected by the length or impedance of the paths using it. Therefore, Ben-Akiva and Ramming 's (1998) formulation of the correction term is not yet appropriate for long trips [4]. Ramming (2002) has developed a more general formulation:

$$PS_{in} = \sum_{a \in \Gamma i} (\frac{l_a}{L_i}) \frac{1}{\sum_{j \in Cn} \frac{G(L_i;\gamma)}{G(L_j;\gamma)} \delta_{aj}}$$

Where G is a function with parameter γ. The value of γ represents the impact of the overlapping path lengths; the higher γ the more impact of the lengths [4]. Frejinger, Bierlaire and Ben-Akiva (2009) developed an expanded PS logit model. The route choice set is based on sample data in the expanded PS logit model. The Expanded PS shows good results and outperforms models with the original PS formulation [11].

The (Cross) Nested logit

The Nested logit (NL) is an extension of the MNL to deal with correlations between alternatives. The NL is most often used in mode choice and in multi-dimensional choice, such as combined destination and mode choice. Additional to the nested logit model the Cross Nested logit is developed. Nested logit models divide the choice set C_n into m nests C_{mn}. The Cross Nested logit differs from the nested logit in that lower-level alternatives may belong to more than one nest [4]. The probability that individual n chooses alternative i consists of two parts:

$$P(i|C_n) = P(C_{mn}|C_n)P(i|C_{mn})$$

The choice probabilities of the Cross Nested logit are:

$$P(i|C_{mn}) = \frac{\alpha_{mi} e^{V_{in}}}{\sum_{j \in Cmn} \alpha_{mj} e^{V_{jn}}}$$

$$P(C_{mn}|C_n) = \frac{e^{V_{Cmn} + \mu_m I_{Cmn}}}{\sum_{l=1}^{M} e^{V_{Cln} + \mu_m I_{Cln}}}$$

where $I_{Cmn} = ln \sum_{j \in Cmn} (\alpha_{mj} e^{Vjn})^{\frac{1}{\mu m}}$ and α_{mi} represents the membership of alternative i in nest m. The Cross Nested logit model is difficult to estimate because of the large number of nesting parameters [11], this reduces the usage of the Cross Nested logit. The Cross Nested logit is used for route choice modeling in small networks. We are unaware of applications of the Cross Nested logit in moderate size city [4].

The Probit model

The Probit model is not a member of the GEV family. The error terms in the Probit model follow a multivariate normal distribution. The Probit model incorporates the correlation among alternatives. Therefore, the Probit model uses a vector notation for the utility function:

$$U_n = V_n + \epsilon_n$$

where U_n, V_n, and ϵ_n are $(J_n \times 1)$ vectors. The probability function of the Probit model is:

$$P(i|C_n) = P(U_{jn} - U_{in} \leq 0 \,; \forall j \in C_n) \text{ [12]}$$

The difficulty in implementing the Probit model is that no closed form exists for the multivariate normal distribution, so numerical techniques must be used. Numerical integration techniques are computationally feasible when the number of Gaussian variables (generally the number of alternatives less one) is small [4].

Mixed Logit models

Models with error terms distributed as a combination of normal and Gumbel distribution are: Mixed logit, Hybrid logit and Kernell logit. The general form of the Kernell logit model, in vector notation is given by Walker (2000):

$$U = X\beta + FT\xi + \nu$$

Where U is a J_n by 1 vector of utilities, β is a column vector of K unknown parameters; X is a J_n by K matrix of explanatory variables; ξ is a column vector of M i.i.d. standard Normal variables, which represent unobservable factors; F is a J_n by M factor loading matrix; T is an M by M lower triangular matrix of unknown parameters; and ν is a J_n by 1 vector of i.i.d. Gumbel variables with scale parameter μ. The Logit Kernel model suffers from the same computational difficulties as pure Probit [4].

The Implicit Availability/Perception Logit Model

The Implicit Availability/Perception logit model (IAP logit) seems similar to the Multinomial logit model, but the IAP logit does not use a choice set. In fact the IAP logit starts with all existing alternative routes. IAP logit uses a correction term to a path's share to reflect the possibility that travelers are unaware of that path, or unable to use it. The probability of using path i for individual n is:

$$P_n(i) = \frac{\mu_n(i)e^{V_i}}{\sum_{j \in M} \mu_n(i)e^{V_i}}$$

Where M is the set of all possible alternatives. We only know applications of the IAP logit with using variables related to availability and not with awareness. We expect that this route choice model will take too much computational time, because all existing alternative routes are considered.

3.1.3 Related Issues

Besides the parameters of the models, the utility function and the choice set major have influence on the route choice results of the random utility based models.

Utility function

It is clear that the deterministic utility term V_{in} has a major influence on the route choice results of the random utility based models. This term is often expressed by a combination of distance and travel time, because it is generally agreed that most travelers try to minimize travel time and distance for most types of journeys [16]. We also know that other elements like the amount of intersections, the amount of speed bumps, the view along the route, and the location of gas stations influence the route choice. The elements and the weights of the elements on which a route choice is based differ for every traveler. It is beyond the scope of this research project to investigate which elements and with which weights are the most realistic. We only consider different combinations of travel time and distance in the utility function. But we have to be aware that the route choice model could be improved by using other utility functions.

Choice set

All the random utility based route choice models make use of a choice set. The choice set should contain the routes that a traveler considers. Routes outside the route choice set can never be chosen by the random utility based route choice models, therefore it is important that the right routes are contained in the choice set. But the choice set should not contain all available routes. In realistic networks the number of alternative routes is an inordinately large number, but travelers may consider a much smaller number of attractive paths [4]. Therefore a route set generation algorithm is needed that selects a few of the available routes. The routes for the choice set can be selected on many different criteria: free flow travel time, free flow travel time and distance, minimal left turns, minimal number of links, minimal number of traffic lights, etc. Sometimes different choice sets are used for different drivers, which is called probabilistic choice set generation [17]. We only use choice sets based on free flow travel time. Therefore we should be aware that the results of the route choice models with choice sets could be improved by a using better choice set generation methods. The shortest route principle does not use a choice set and therefore is not influenced by the choice set generation.

3.2 Other Approaches

In the previous section we consider utility based models. This section discusses 4 other non utility based approaches. Non utility based models cope with the disadvantages of the random utility based models. Tversky and Kahneman (1986) have shown conflicts of the random utility theory with actual decision situation, therefore they developed the Prospect theory. Henn and Ottomanelli (2006) show that the consideration of randomness of traffic by drivers is hardly ever represented in random utility route choice models. These randomness of traffic by drivers is incorporated in Fuzzy logic. They also discuss that Possibility theory is more accurate in representing decision behavior under uncertainty than Probability theory which is used in the random utility theory. Game theory considers the traffic assignment from a totally different point of view. This theory considers the assignment as a game with multiple players. This section describes these four non utility based approaches and their applications in route choice modeling.

3.2.1 Prospect theory

Tversky and Kahneman (1979) developed the Prospect theory, because the random utility seems not to be the best theory to describe decision making. A first drawback is that the random utility theory is not developed from a psychological analysis of the decision making [18]. Second, the random utility theory does not take into account the certainty effect, the possibility effect, and the reflection effect. This lack causes conflicts of the random utility theory with actual decision processes. The example below shows the conflict with the certainty effect. In the example, Kahneman and Tversky asked students to give their preferences on two problems.

Problem 1: Choose between

A:	*2,500 with probability*	*.33,*	*B:*	*2,400 with certainty*	
	2,400 with probability	*.66,*			
	0 with probability	*.01;*			

Problem 2: Choose between

C:	*2,500 with probability*	*.33,*	*D:*	*2,400 with probability*	*.34,*
	0 with probability	*.67;*		*0 with probability*	*.66*

The majority of the 72 respondents, 82% of the respondents preferred situation B against 18% for situation A in problem 2. In problem 2, 83% of the respondents choose C. Each of these preferences is significant at the 0.01 level. According to the random utility theory, with $U(0) = 0$, the first preference implies:

$$\begin{aligned} U(2,400) &> 0.33U(2,500) + 0.66U(2,400) \\ 0.34U(2,400) &> 0.33U(2,500) \end{aligned}$$

While the preferences of problem 2 implies the reverse inequality [19]. In the light of the previous and other observations from Kahneman and Tversky (1979), they argue that the random utility theory is not an adequate descriptive model. They propose the Prospect theory, which accounts for choice under risk.
Prospect theory distinguishes two phases in the choice process: editing and evaluation. Editing consists of the application of coding, combining, segregation and cancellation. Coding is the formulation of the offered prospects. Probabilities associated with identical outcomes are combined to simplify prospects. Segregation occurs to segregate risk less components from the risky component. Cancellation involves the discarding of common outcome probability pairs. In the evaluation phase the decision maker is assumed to evaluate each of the edited prospects, and to choose the prospect of the highest value. The overall value of a prospect V is expressed in two scales, π and v. the first scale, π, associates with each probability p a decision weight $\pi(p)$. The second scale, v, assigns to outcome x a number $v(x)$, which reflects the subjective value of that outcome. Hence, v measures the value of the deviations from the reference point. The value of prospect which is neither strictly positive nor strictly negative is formulated as:

$$V(c, p; y, q) = \pi(p)v(x) + \pi(q)v(y)$$

where $v(0) = 0$, $\pi(0) = 0$, and $\pi(1) = 1$. The evaluation of strictly positive and strictly negative prospect is different. These prospects are in the editing phase segregated in a risk less component and a risky component. If $p + q = 1$ and either $x > y > 0$ or $x < y < 0$, then:

$$V(x, p; y, q) = v(y) + \pi(p)[v(x) - v(y)]$$

The Prospect theory is clearly more difficult to work with than random utility theory but would appear to have a much sounder behavioral basis [20].

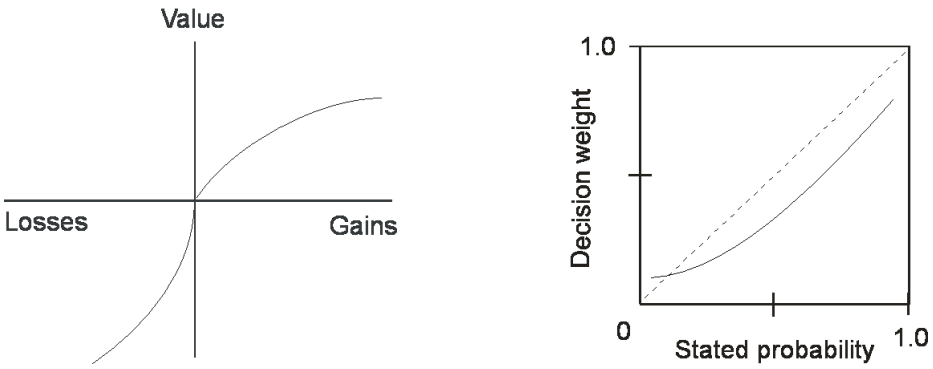


Figure 3.2: Hypothetical value and weighting function

Kahneman and Tversky (1979) proposed that the value function is defined on deviations from the reference point, generally concave for gains and commonly convex for losses, steeper for losses than for gains. Figure 3.2 displays a value function that satisfies these properties.

The weighting function is based on the overweighting property, subadditivity property, subcertainty property and subproportional property. Low probabilities are generally overweighted, this mean that $\pi(p) > p$ for small p, this is the overweighting property. The subadditivity property states that for small values of p $\pi(rp) > r\pi(p)$ holds. Subcertainty entails that π is regressive with respect to p, preferences are generally less sensitive to variations of probability than you would expect. The subproportional property causes that the ratio of the corresponding decision weights is closer to unity when the probabilities are low than when they are high. This mean that if (x, p) is equivalent to (y, pq) then (x, pr) is not preferred to (y, pqr), $0 < p, q, r \leq 1$. For example, suppose that a prospect of a gain of 40 with probability 0.5 is equal to a gain of 200 with probability 0.1, then the prospect a gain of 40 with probability 0.05 is not preferred to a prospect of 200 with probability 0.01. Figure 3.2 displays a weighting function that satisfies these four properties. The weighting function is not well-behaved near the end-points, because highly unlikely events are either ignored or overweighted, and the difference between high probability and certainty is either neglected or exaggerated.

Connors and Sumalee (2009) have successfully applied Cumulative Prospect theory on a two and a five link network. They used an equilibrium assignment, and derived the parameter values from choice modeling. Further research is needed to apply Prospect theory on real sized networks and to find meaningful parameters for the value v and decision weight π functions [21]. Avineri and Prashker (2002) propose the Cumulative Prospect Theory Learning (CPTL) model, because the CPT static model failed to predict feedback based decisions. The CPTL is a dynamic generalization of the CPT model. The model uses travel time frequencies instead of travel time distributions, so instead of p_i the frequency $fr_t(i)$ is used, for every time period t. They compare this model with three other models: Multinomial Logit model, the Cumulative Prospect Theory model, FL learning model and the Reinforcement Learning model. The experimental results show that travelers' behavior is better captured by learning models like the CPTL [22].

3.2.2 Fuzzy logic models

Henn and Ottomanelli (2006) show that the consideration of randomness of traffic by drivers is hardly ever represented in random utility route choice models. These randomness of traffic by drivers is incorporated in Fuzzy logic. Fuzzy logic is based on the concept of fuzzy sets. Fuzzy logic generalizes the notion of membership (to belong or not to belong to a set) to a continuous grade of membership (belonging more or less to a set). A fuzzy set is defined by its membership function, μ_a which takes values in the interval [0,1]. In classical logic nothing can be concluded from $A \Rightarrow B$ if A is not observed, while in fuzzy logic a conclusion can be drawn if nearly A is observed. Several route choice models using fuzzy logic are developed, with fuzzy linguistic rules such as "if travel time on route 1 is very short and travel time on route 2 is intermediate, then I will certainly choose route 1". In this linguistic rules are 'very short' and 'intermediate' fuzzy sets, which are based on expert knowledge.

One of the major problems of fuzzy logic models is that very few information is given on how the membership functions are built. The definition and the building of membership functions is still not completely solved. Route choice models based on fuzzy linguistic rules already exists. These models benefit from the tools of fuzzy control which are known to be easy to design, and robust to small variations. However, these models are only valid for one particular network [1]. A general model is developed by Henn and Ottomanelli (2006), but this model describes only the behavior of one individual traveler. We expected that a fuzzy logic based

route choice model for multiple travelers, with the current knowledge and computation possibilities will take much computational time.

3.2.3 Possibility theory

Recently, researchers in psychology have begun to study the decision making behavior under uncertainty, and they have found that it seems that the possibility theory is more close to the mental framework than the probability theory. Therefore possibility theory seems a promising theory for route choice modeling. The possibility theory is developed as an alternative way to probability in order to represent uncertainty. While probability is based on the additivity axiom, possibility is based on some max-axiom: Given two subsets A and B in the universe X, the probability and possibility measures are:
$Pr(X) = 1 \quad Poss(X) = 1,$
$Pr(\oslash) = 0 \quad Poss(\oslash) = 0,$
$Pr(A \cup B) = Pr(A) + Pr(B) - Pr(A \cap B),$
$Poss(A \cup B) = \max\{Poss(A); Poss(B)\}.$
The probability that traveler n chooses alternative i is supposed to be proportional to the possibility that a cost is smaller or equal to others. Applying the normality condition for probability, we get:

$$P_n(i) = \frac{Poss(\widetilde{C}_i^n < \widetilde{C}_j^n \; \forall j \neq i)}{\sum_i Poss(\widetilde{C}_i^n < \widetilde{C}_j^n \forall j \neq i)}$$

A few fuzzy based route choice model already use the possibility theory [1]. Like in the case of Fuzzy logic based models, we expect that possibility theory route choice models will take much computational time.

3.2.4 Game theory

Game theory is a mathematical theory to analyze situations with competition and cooperation between several parties. This is a broad definition of game theory, but is in consensus with the broad spectrum of applications of game theory. The applications range from strategic questions in warfare to understanding economic competition, and also include traffic situation. The problems of these applications are displayed as a game. The decision makers of the application are the players in the game. The players of the game can make a move, which we call an action. The result of the game is the final payoff, which is the optimal gain if every player plays his optimal strategy [23]. The game will reach an equilibrium situation if every player plays according to their optimal strategy.

Two travelers use the network of figure 3.3, a tractor and a car. Both travelers are going to the gas station, the tractor favors the northern route and the car the southern route. The tractor arrives first at the network and decides if he takes the northern route or the southern. The car arrives later and cannot see which route the tractor has taken. If the car takes the same route as the tractor, the car will be delayed due to the slower tractor, and the tractor will get nervous of the car behind him. The southern route is the most smallest route and they will bother each other the most on this route. The gain for each decision situation is displayed in table 3.1, the tractor chooses a row and the car a column. We translate this situation into a two player non cooperative game. The players are the tractor and the car, both want to maximize their gain (utility). By choosing the northern route the tractor is always at least as well off as by choosing the southern route. So the optimal strategy of the tractor is choosing the northern route. The car, being able to perform this same kind of reasoning for the prediction of the action of the tractor, has the southern route as optimal strategy. Since the southern route is the best reply to the choice of the northern route by the tractor.

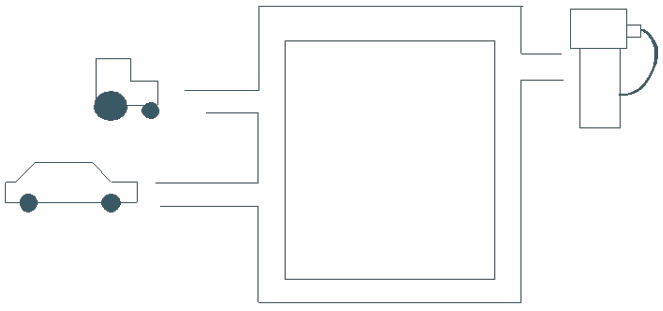


Figure 3.3: An example of a network with two 'players'.

	northern route	southern route
northern route	4	5
southern route	4	1

Table 3.1: Gain of the decision situations

The previous example is a very simple traffic situation with only two travelers and a small network. But this example can easily be extended to a more complex network with more (even infinite) travelers. Often, the solutions are not that straight forward as in the example. The optimal strategy of the example is a pure strategy. In pure strategies, players choose always exactly one route; the tractor always chooses the northern route and the car always chooses the southern route. In mixed strategies, players choose with a probability distribution over m parallel routes in a mixed strategy.

Congestion game

In literature several different games with applications in traffic situation are described. The n-player congestion game is the most simple game described. A congestion game is a non cooperative game, meaning that players cannot make agreements with each other [23]. The payoff of the players depends only on the player 's own strategy and on the number of other players choosing the same or some interfering strategy. A player represents one vehicle of the network.
We describe the congestion game with travel times as negative payoff by the following mathematical model. Let $P = \{1, ..., N\}$ denote the set of players. Each player $i \in P$ maximizes its own payoff, they minimize their expected travel time.

$$\min \sum_{j \in A_i} (TT_j(\overline{x}) \cdot s_i(j))$$

where A_i is the action set of which player i selects an action (a route), $TT_j(\overline{x})$ the travel time of route j with loading $\overline{x}$, s_i the mixed strategy of player i, and $s_i(j)$ the probability that player i chooses route j according to his strategy. The travel time of route j consists of the sum of all link travel times of route j:

$$TT_j(\overline{x}) = \sum_{l=1}^{L} (LTT_l(x(l)) \cdot PL(j,l))$$

where L is the total amount of links in the network, $LTT_l(x(l))$ the travel time of link l with loading $x(l)$ on link l, and $PL(j,l) = 1$ if link l is an element of path j otherwise $PL(j,l) = 0$. The amount of traffic on the links is calculated by:

$$x(l) = \sum_{i \in P} \sum_{j \in J} (s_i(j) \cdot PL(j,l))$$

The mixed strategy s_i is conditioned by two restraints:

$$\sum_{j \in J} s_i(j) = 1 \quad and \quad s_i(j) \geq 0$$

The optimal strategy of the players of this game can be determined with an iterative scheme. Figure 3.4 displays this iterative scheme of the congestion game. The process starts with initiating the counter of the number of iteration, and the SocialCost. In each iteration the routes with minimal travel time are selected by the new strategy. The Strategy s_i is a combination of the old strategy (StrategyO) and the new strategy (StrategyN). The vehicles are assigned to the routes according to their strategy. The travel times and social costs are recalculated with this new load on the network. The social costs SC represents the sum of all experienced

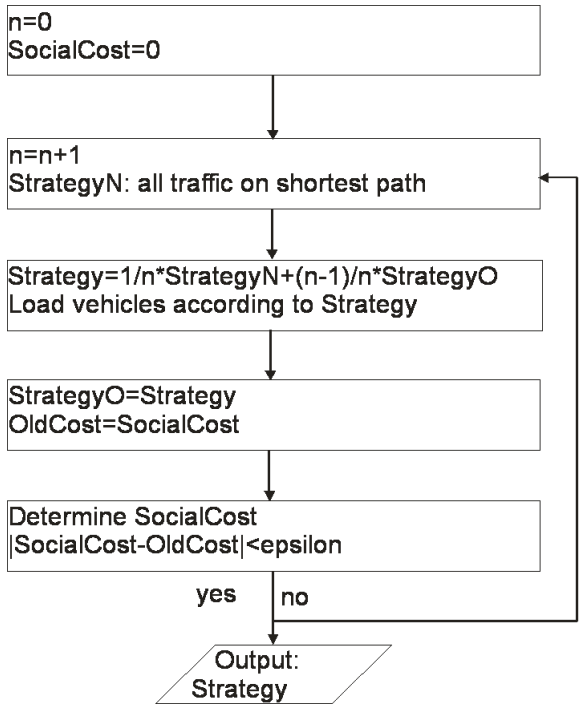


Figure 3.4: The process of the congestion game

path times: $SC = \sum_{l=1}^{L} LTT_l \cdot x(l)$ with L the amount of links, and LTT_l the travel time of link l. This process is repeated until the differences between the SocialCost and the OldCost is within some predefined bounds. The output of this iterative scheme is the mixed strategy of the players to reach the flow of the final iteration. Players of the same origin destination pair will have the same optimal mixed strategy.
The congestion game always reaches a Nash equilibrium of pure strategies. The process of the congestion game will always converge [24].

Smooth fictitious play

Garcia, Reaume & Smith (2000), Cominetti, Melo & Sorin (2008) use the perception of travel times instead of real travel times to model traffic routing. Both use a fictitious play to incorporate a learning effect in the perception of travel times. Fictitious play is an iterative procedure in which at each step, players compute their best replies based on the assumption that the opponents' decisions follow a probability distribution in agreement with the historical frequency of their past decision [25]. Garcia et. al. (2000) describe the results of Monderer and Shapley (1996) who demonstrate that fictitious play convergences in some sense, when players share a common objective function. Therefore, Garcia et. al. (2000) assign the players with a common objective function instead of a selfish. The player's objective is to minimize average trip time of all travelers. This game is called the Fictitious Dynamic Traffic Game [25]. In reality people do not act that socially, people rather want to minimize their own travel time than the average travel times of all vehicles. Cominetti et. al. (2008) base their model on this thought, assuming that the travel times of other vehicles are unknown. Their iterative fictitious play is based on the vehicles own travel times, and is called the smooth fictitious play.
The game of the smooth fictitious play is iteratively played, each stage can be described by the same mathematical model. Therefore we describe just a single stage. Again let $P = \{1, ..., N\}$ denote the set of players. Each player $i \in P$ maximizes their own payoff, the negative perception of their travel time under the assumption that the other players play the action (the routes) whose probability distribution is given by historical frequency's of past plays [26].

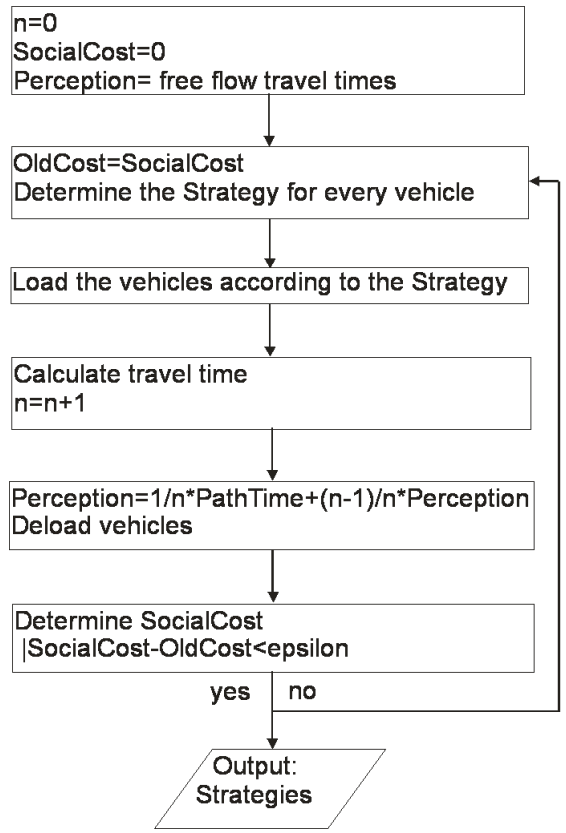


Figure 3.5: The smooth fictitious play

$$\min \sum_{j \in A_i} (PTT_i^n(j) \cdot s_i^n(j))$$

where $PTT_i^n(j)$ the perception of player i at stage n of the travel time of route j, and $s_i^n(j)$ the probability that player i selects route j at stage n. The perception of the travel times is updated at every stage. We use the free flow travel times as the perception of the travel times in the first stage. The perceptions of the routes at other stages are updated by the following rule:

$$PPT_i^{n+1}(j) = \begin{cases} (1 - \frac{1}{n}) \cdot PTT_i^n(j) + \frac{1}{n} \cdot TT^n(j) & \text{if } j = a_i^n, \\ PTT_i^n(j) & \text{otherwise} \end{cases}$$

Where $TT^n(j)$ is the travel time of route j at stage n, and a_i^n the chosen action (route) at stage n by player i. The action a_i^n is chosen at random from their mixed strategy s_i^n. The mixed strategy is determined with the Multinomial logit rule at the smooth fictitious play:

$$s_i^n(j) = \frac{e^{-\theta PTT_i^n(j)}}{\sum_{k \in A_i} e^{-\theta PTT_i^n(k)}}$$

The solution of this game can be derived with a iterative process, figure 3.5 shows this iterative process. The process starts with the initiation of the counter, the SocialCost, and the Perception of the travel times. The Strategy ($s_i^n(j)$) of each player is determined with the Multinomial logit. The Perception is used as utility in the Multinomial logit. The vehicles select an action according to their strategy and are loaded on the route of their action. The payoff of the chosen alternative is then observed and is used to update the perception of the payoff for that particular move for the vehicle. The iterative process stops if the absolute difference between the social cost and the old social cost is

smaller than a predefined ϵ. The output of this iterative process is the final strategy of each player. The strategies of players of the same origin destination pair will approach each other if ϵ goes to zero. It is proven that this process converges almost surely towards a stationary state which is characterized as an equilibrium [27].

Demon player game

Bell and Cassir (2002) propose to represent the traffic network with a $n + m$ player game, which is called the demon player game. In this game with m OD pairs, n travelers seek their best route and m specific demons penalize the network users as much as possible. A demon player is OD specific and damages the network on one and only one link. This game results in a risk-averse user equilibrium.
We describe the demon player game by the following mathematical model. We define the set of players by two subsets: $P = P_1 \cup P_2$. The subset $P_1 = \{1, ..., n\}$ represents the 'normal' players who represent the vehicles, this is a set of homogeneous players. The subset $P_2 = \{1, ..., m\}$ represent the demon players. The objective of the players of subset P_1 is to maximize the payoff of all players of P_1, which means minimize expected travel times of all players of P_1:

$$\min \sum_{i \in P_1} \sum_{j \in J} ETT(j) \cdot s_i(j)$$

Where $ETT(j)$ is the total expected travel time of route j, and $s_i(j)$ the probability that player i chooses route j under strategy s_i. The objective of the demon players is to maximize the total travel of all travelers on their OD:

$$\max \sum_{i \in P_1} \sum_{j \in J} ETT(j) \cdot s_i(j) \cdot OD_k(j) \quad \forall k = 1, ..., m$$

Where $OD_k(j) = 1$ if route j connects origin destination pair k, otherwise $OD_k(j) = 0$. The expected travel time of route j is the sum of all its expected link travel times. The expected link travel time is a combination of the normal link travel time and the failure link travel time:

$$ELTT(l) = (1 - fp(l)) \cdot LTT(l) + fp(l) \cdot LFTT(l)$$

Where $fp(l)$ is the probability that link l is selected by the demon to damage, $LTT(l)$ the travel time of link l under normal conditions, and $LFTT(l)$ the travel time of link l when this link is damaged by the demon. The vector of link failure probabilities of one demon is the mixed strategy of this demon. The demon has the freedom to select their failure probabilities in such a way that the following three conditioned are met:

$$\begin{aligned} fp(l) &\geq 0 \quad \forall l \\ \sum_{l \in LK} fp(l) &= 1 \\ fp(l) &= 0 \quad \forall l \notin LK \end{aligned}$$

Where LK is the set of all links which are contained in the routes between origin destination pair k.
It is difficult to find the equilibrium of this game, therefore an approximation of the solution is obtained [28]. Figure 3.6 displays the iterative process of this approximation. The approximation starts with initiating a counter, the strategies of the 'normal' players (StrategyO) and the demon players (DStrategyO), and the SocialCost. We determine the optimal strategy of the 'normal' players in the next step. The players select the path with minimal expected travel time, and set the route choice probability of this route on one in StrategyN. We load the players on the network according to the route choice probabilities of Strategy. The demon players determine their optimal strategy with these loads. Their optimal

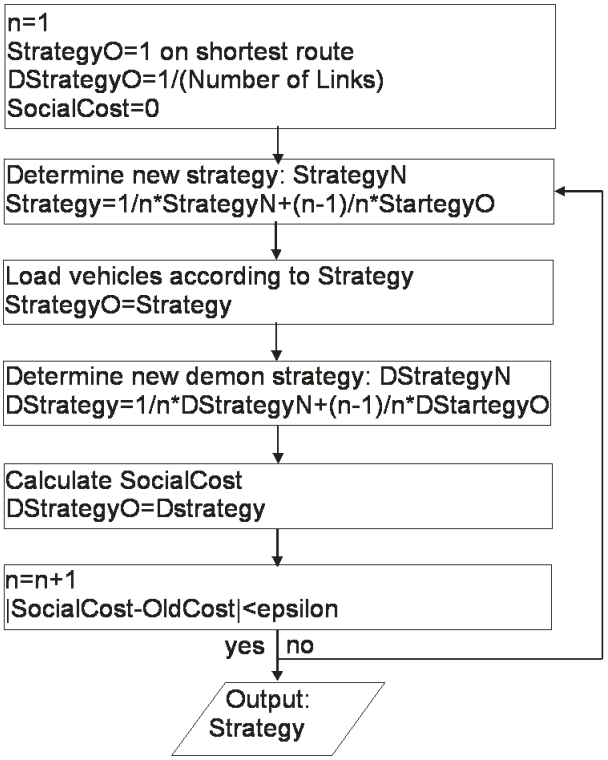


Figure 3.6: The demon player game

strategies are the link failure probabilities. We determine the demon's optimal strategy by the follow procedure. We start with a link failure probabilities of zero on all links. Next we set one for one the link failure probability of one link to one. We calculate the expected travel times with the new loads and new link failure probabilities. The demon selects the scenario that causes the highest expected travel times for the travelers on his origin destination pair, and set the link failure probability of this link to one in DStrategyN. A link damaged by the demon looses a certain fraction a of its capacity. The lowered capacity influences the travel times of all vehicles using that link, but in this game only the expected travel times of the travelers with the same origin destination pair as the demon are adapted to this lowered capacity. The iteration process stops if the difference between the SocialCost and the OldCost is within some bound. The output of this iterative process is the strategy of the 'normal' players. There is no guarantee for convergence of this iterative process [28].

3.3 Selection

We select two route choice approaches to investigate the abilities of these approaches in comparison with the shortest route principle. We base our selection of route choice approaches on three elements: reality, computational time, and ease of use. First the route choice approach should be realistic. Second the route choice approach should not take to much computational time, because route choice models with exceptional long computational time will not be used. And third the route choice approach should be easy to use, and direct usable so no further research should be needed to develop a route choice variant of this approach.
Both the labeling approach and the Multinomial logit are not realistic route choice models. The labeling approach is not realistic, because this model assumes that travelers are fully informed about the routes and their costs. The Multinomial logit is not realistic in most networks, because it suffers from the IAA property. Previous research on the C-logit and the path size logit approaches indicates these approaches are fairly realistic route choice approaches, with reasonable computational time and easy to use. We do not investigate both the PS-logit and the C-logit, because these models are very similar. We choose to investigate the C-logit instead of the PS-logit, because C-logit is already used at DHV (at Aimsun). Notice that it is possible to express the Path size logit in terms of the C-logit model. The Cross Nested logit also shows realistic results in reasonable computational time, but only in small networks. The large number of nesting parameters which need to be calibrated makes this model not easy to use. Also no applications of the Cross Nested logit are known for networks of large or even moderate size. The Probit model and the Mixed Logit model are both realistic route choice models, but suffer from the fact that no closed form of the multivariate normal distribution exist. This causes high computational time. Also the implicit availability logit model suffers from high computational time, because the utility of all the available routes has to be calculated.
Next to C-logit we also want to select a non-random utility based approach because of the disadvantages of the random utility theory. The Prospect theory is a realistic model with reasonable computation time. Further research is needed to apply Prospect theory on real sized networks and to find meaningful parameters for the value and decisions weight functions. Such further research is beyond the scope of this master thesis project, but required to make the Prospect theory easy to use. Therefore the Prospect theory is not easy to use at this moment. Both the Fuzzy logit and the Possibility theory take to much computational time. Certain games from Game theory are realistic route choice model, have reasonable computational time, and because of early experience with game theory in traffic application game theory is easy to use. Therefore we select the game theory approach.

Chapter 4

Research outline

In this chapter, we draw a conclusion based on previous chapters. The previous chapters define problems in the field of route choice modeling from literature and the experiences at DHV. This chapter specifies the direction of our research project to deal with these problems. We describe the research goal in Section 4.1, the research questions in Section 4.2, and the research approach in Section 4.3.

4.1 Research goal

From Chapter 2 we know that DHV uses transport models, and that these transport models result in unrealistic route choices. DHV wants to investigate the possibilities to improve the transport model, such that the model results in more realistic route choices. We believe that the route choice step has the most influence on the route choices within the transport model. Therefore we focus on the route choice model, but we keep in mind that the route choices in transport models are also a result of the loading method.
DHV often uses the shortest route principle as route choice model. In Chapter 3, we conclude that there are several route choice models and approaches beside the shortest route principle. We select two promising route choice approaches: C-logit and game theory. We make our selection on three criteria: reality, computational time, and ease of use. C-logit and Game theory score positive on these three criteria.
Our research goal joins the two conclusions of the previous chapters; the need for a better route choice model and the possibilities of the C-logit and the game theory models. Our research goal is:

Investigate whether route choice models based on C-logit and game theory are able to produce more realistic route choices than the shortest route principle, and investigate the behavior of the C-logit and game theory models.

We make this goal more explicit by giving a definition for some of the terms in this research goal. The route choices resulting from the total assignment step (the route choice model and the loading method) are meant when we speak of route choices. A route choice model produces more realistic route choices than the shortest route principle if the reality of that route choice model is higher than the reality of the shortest route principle. The reality of a route choice model is the extent to which the route choices of the route choice model in combination with the loading method resembles the real route choices. Suppose we have a network with two routes, route a and b. We measure that 50% of the vehicles chooses route a and 50% of the vehicles chooses route b. The shortest route principle determines a route choice of 100% for route a, then the shortest route principle has a reality score of 50%. Suppose that our variant assigns 40% on route a and 60% on route b. Then, our variant assigns only 10% of the vehicles on the wrong route and has a reality score of 90%. In the second part of our research goal we use the term behavior. We mean with behavior of the C-logit and game theory the infuence of the parameters on the reality score of the route choices.
We have to keep an eye on the computational time of the route choice models we investigate, because a realistic route choice model with very high computational time will not be used.

4.2 Research question

We formulate our research question to meet the research goal. The research question is:

Are C-logit and game theory based route choice models able to produce more realistic route choices than the shortest route principle, and what is the influence of the C-logit and game theory parameters on the reality score of the route choices?

We answer this research question in Chapter 9. We develop sub questions for both C-logit and game theory to find an answer to the research question. The first two sub questions about C-logit and the first two sub questions about game theory are already answered in Chapter 2 and Chapter 3. Other sub question will be answered in the remaining part of this report. All the sub questions support the research question, therefore we also mention already answered questions. The sub questions about the C-logit route choice model are:

- How does the C-logit route choice model work?
- Which parameter values and commonality factor does DHV use at Aimsun for the C-logit route choice model?
- Which parameter values and commonality factor are expected to be realistic for the C-logit from literature?
- Can a C-logit based route choice model produce more realistic route choices than the shortest route principle?
- What is the influence of the C-logit parameters on the reality score of the route choices?

Game theory is not yet used at DHV for route choice modeling. The sub questions about game theory are:

- What is game theory?
- How can game theory be used for route choice modeling?
- Can a game theory based route choice model produce more realistic route choices than the shortest route principle?
- What is the influence of the game theory parameters on the reality score of the route choices?

4.3 Research approach

Our research question can be divided into two questions: "Are C-logit and game theory based route choice models able to produce more realistic route choices than the shortest route principle?" and "What is the influence of the C-logit and game theory parameters on the reality score of the route choices?".
It is easy to find an answer to the first question, we only have to show one case for which a C-logit model result in more realistic route choices than the shortest route principle, and one case for which a game theory model result in more realistic route choices than the shortest route principle. Therefore we need to develop C-logit and game theory variants, and compare the outcomes of the variants with actual made route choices and results from the shortest route principle. We use information from literature and from interviewing to develop the C-logit and game theory variants. We interview several DHV employees and Henk Taale from Rijkswaterstaat. Henk Taale used game theory to find an optimal integrated anticipatorial control of roads networks. We compare the results of the route choice variants with actual made route choices from a license plate survey to determine the reality of the variant. We determine the reality of the shortest route principle at the same manner, and compare the reality of the variant with the reality of the shortest route principle. We develop a transport model in which all steps are fixed except for the route choice in the assignment step, to have an equal comparison between the shortest route principle and the variant.
The second subgoal is much harder to achieve, because the route choice results of a transport model are not only depended on the route choice model and loading method, but also on the network, the travelers, and the travel time functions in the transport model. It is impossible to describe the influence of the parameters on the reality score in all circumstances, but we can get a strong presumption by an analytical investigation of the variants and by comparing the results of the route choice models with different parameter settings with the route choice data. Clearly if we test the variants at once at a network with multiple origin destination pairs we do not get a feeling of what is happening. Therefore we decide to test the variants in a smaller network first. In total we perform four tests, we explain the setup of these tests in the remaining of this chapter.

4.3.1 Theoretical test

The first test is a theoretical test, because we use a non existing theoretical network. This network consists of three routes, and we have total control over the travel times and background traffic in the network. The total control makes it possible to trace back the influence of the parameters. We use this test to get a first feeling about the variants and their parameters, and to select the most promising variants for the more complicated tests.

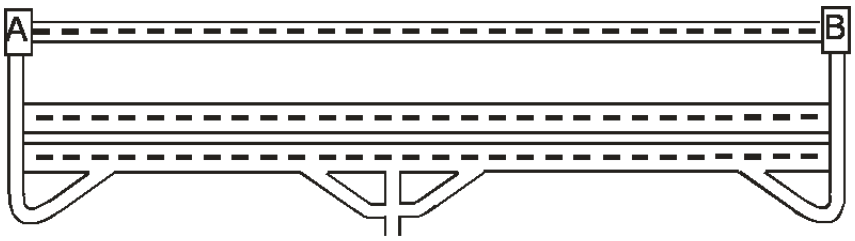


Figure 4.1: The network used for the theoretical test

Test network

Figure 4.1 displays the traffic network of the theoretical test. This network consists of three routes connecting two villages. The upper road is a rural road, the double road below is a highway, and the roads without road marks are the on- and off-ramps. Table 4.1 displays the average free flow speed and capacities of the different road categories.

	Capacity	Average free flow speed
Rural road	1800	80 km/h
Highway	4400	120 km/h
On- and off-ramp	1600	60 km/h

Table 4.1: The characteristics of each road category

The travel time functions of the road section depend on the road capacity and the amount of traffic on the road. The travel time function of a road section is defined by:

$$TravelTime(i) = \frac{60 * RoadLength(i)}{FreeFlowSpeed(i)} \left(1 + 0.6 \left(\frac{Intensity(i)}{Capacity(i)}\right)^4\right)$$

Where $TravelTime$ is the travel time in minutes, $RoadLength$ the link length in km, $FreeFlowSpeed$ the free flow speed in km/h, $Intensity$ the amount of vehicles on the road, and $Capacity$ the capacity of that road. A transport model expert of DHV, Wim van der Hoeven, recommended this travel time function.
The first route in the network of figure 4.1 consists of a rural road of 10 km long. Route 2 consists of an on-ramp, the highway and an off-ramp, this route is 77 meters longer than route 1. Route 3 is the longest route and is a small variation of route 2. The first half of route 3 is the same as route 2, but halfway the highway route 3 follows an off-ramp and an on-ramp from where route 3 goes back and follows route 2 again (see figure 4.1). Route 3 is 10,154 meters long, and is the unrealistic route we discussed earlier in Section 2.4. In a real world situation there would be traffic with other origins than the two villages on the rural road and the highway. Therefore, and to ensure congestion on the highway, we assign vehicles on the highway and on the rural road with a fixed route. We call these vehicles the background traffic. The other vehicles are assigned from village A to B and have a free route choice. These vehicles follow the routes determined by the route choice model variants.
This network is a non existing network, and we do not have any route choice data to determine the reality of the variants. Therefore we designed the network in such a way that it is not realistic that someone takes the third route. We cannot say anything about the division of the vehicles over the first two routes, and do not investigate the route choices on these two routes.

Static assignment

We first perform the theoretical test in a static assignment, because the static assignment is less complicated than a dynamic assignment. We expect that the static assignment gives us a clear overview of the effects of the different variants. We use an incremental assignment with equal step sizes in the static assignment. We model the static assignment in MATLAB. Figure 4.2 displays the content of the MATLAB file for the static assignment in pseudo code. First, the MATLAB file calculates the travel times in a empty network (the free flow travel times). The second step differs for all route choice variants, in this step the route choice probabilities (p_1, p_2, and p_3) are calculated using the selected variant. Now 10% of the total demand is loaded to the network according to these probabilities. Suppose $p_1 = a$ than the load of this iteration on route 1 is $a \cdot 1/10 \cdot demand$. The demand is 2000 vehicles. At the same time, 10% of the background traffic is loaded to the network. We assign in total 5000 background vehicles on the highway, and 2200 background vehicles on the rural road. The travel times are calculated again with the new loads and the process starts all over again with the calculation of

the new route choice probabilities. After ten iterations the total demand is loaded to the network. The results of the MATLAB file are the loads on the routes. The code of the MATLAB file is added to appendix A.

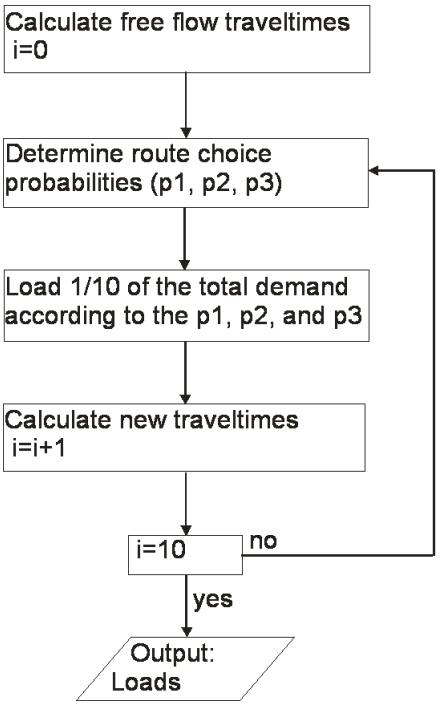


Figure 4.2: Schematic overview of the static assignment

Dynamic assignment

We test the most promising variants of the static assignment also in a dynamic assignment. A dynamic assignment works with more realistic traffic properties like blocking back effects which influences the travel times. The results of the theoretical test in a static assignment do not have to hold for a dynamic assignment, because the route choices depend on the travel times.
Dynasmart and MATLAB together run the dynamic assignment. We use Dynasmart, because it is possible to specify the routes of the vehicles in Dynasmart, and Dynasmart is able to run a dynamic assignment. Before we run the dynamic assignment we change the static demand of 2000 vehicles with free route choice in a dynamic demand profile. We equally divide the demand on four quarters and Dynasmart assigns the vehicles to a departure minute within the quarter in a random process, which results in a total of 2044 vehicles of free route choice. We also change the background traffic and the network in comparison to the static assignment, to ensure that shortest route principle assigns traffic on the unrealistic third route. Because the shortest route based principle should produce some unrealistic results, in order to test if our variants produce more realistic results. We change the background traffic of 5000 vehicles on the highway in 4500 vehicles per hour, and the background traffic of the rural road from 2200 vehicles per hour to 2500 vehicles per hour. Dynasmart assigns the background traffic with a dynamic profile, like the vehicles with free route choice. Further we make a little change in the network. One kilometer after the off-ramp on the highway, the capacity of the highway is cut in half over a length of 500 meter, see figure 4.3.

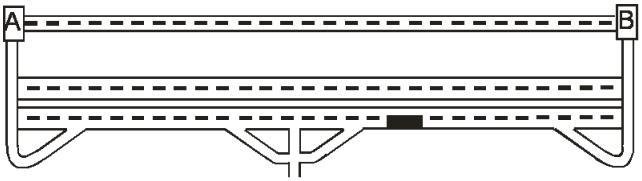
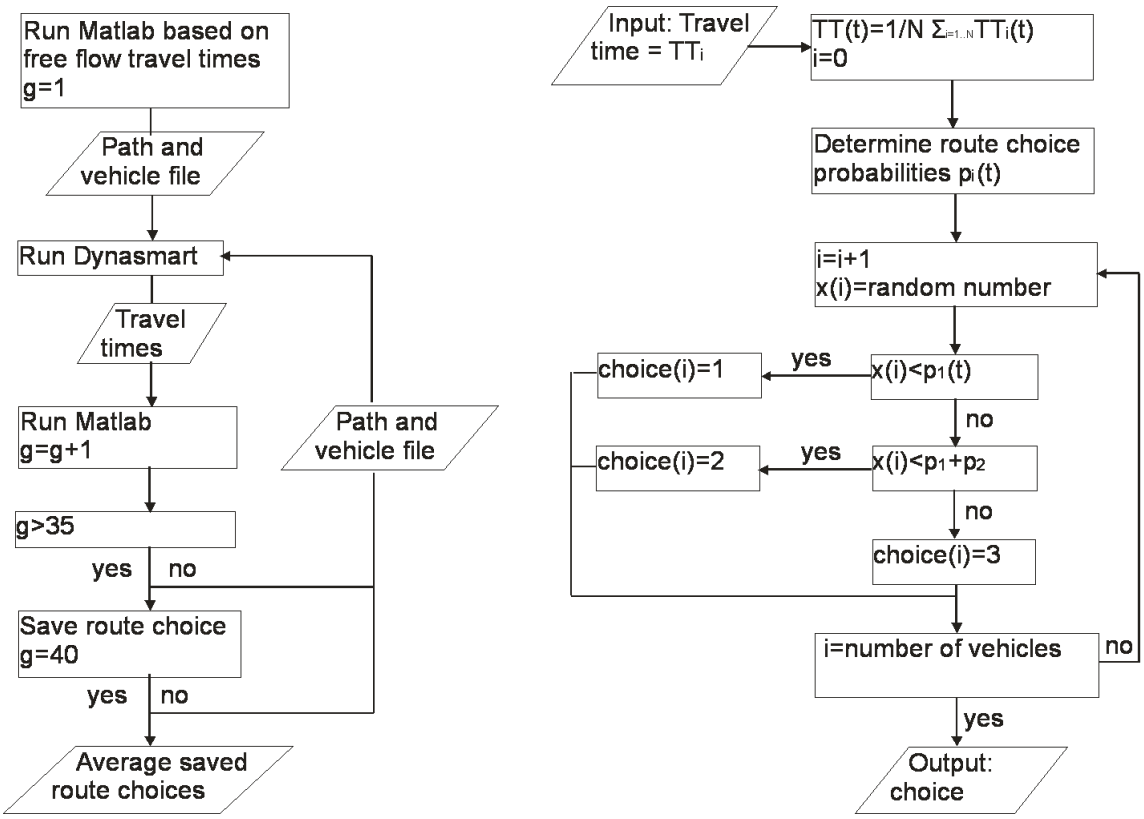


Figure 4.3: The network used for the theoretical test in a dynamic assignment

Due to this adaption congestion will occur and the congestion blocks back on the highway to the second on-ramp and further back. We run Dynasmart 2 simulation hours to ensure that the most vehicles reach their destination. Figure 4.4(a) displays the process of the dynamic assignment and figure 4.4(b) the process of the MATLAB file. MATLAB calculates the route choices and Dynasmart the dynamic travel times. We develop for every variant a different MATLAB file, but the basic idea of the MATLAB files are all the same.
The assignment method in the dynamic assignment is an approximation of an equilibrium assignment. In the first step of the dynamic assignment, we calculate the route choices on the basis of free flow travel times in MATLAB. MATLAB writes the route choices in a vehicle and path file. These two files are the input files of a Dynasmart run. The vehicles follow the route specified in the vehicle and path file in this Dynasmart run. Dynasmart calculates the average travel times for every 2 minutes and writes these travel times in a travel time document. This document is the input for a new MATLAB run. MATLAB averages the travel times found in this iteration with the travel times found in the previous iterations, such that a file is created with the average travel times over the iteration for every link, and every 2 minutes interval. The routes choice probabilities $p_i(t)$ are calculated for every two minutes interval t on the basis of these average travel times and the selected route choice variant. Each vehicle receives a random number x between 0 and 1. We know from the demand profile, in which 2 minutes interval a vehicle will depart. If the route choice probability of the two minutes departure



(a) The process of the dynamic assignment

(b) The MATLAB file of the dynamic assignment

Figure 4.4: Schematic overview of the dynamic assignment

interval is smaller than the sum of the route choice probabilities 1 to i, and larger than the sum of the route choice probabilities 1 to $i-1$ then route i is chosen:

$$\sum_{j=0}^{i-1} p_j(t) \leq x \leq \sum_{j=0}^{i} p_j(t) \;\; then \;\; choose \;\; route \;\; i$$

With $p_0(t) = 0$. MATLAB writes the route choices in the vehicle and path file, these documents are the input for a new Dynasmart run. This process of a Dynasmart run and a MATLAB run is repeated 40 times, because all variants converge within 40 iteration. The route choices of the 36th until the 40th iteration are averaged, these average route choices are the results of the dynamic assignment. The code of the MATLAB file is added to Appendix A.

4.3.2 Setup of the Enschede tests

The three Enschede tests use the same route choice data and are performed on the same network. This section describes the route choice data and the network of the Enschede tests.

Route choice data

The three Enschede tests compare the route choices with route choice data from Enschede. Witteveen+Bos and Dufec performed a license plate survey at the order of the municipality of Enschede. More than 50,000 license plates were registered on Tuesday the 4th of March and Saturday the 8th of March in 2008 at the roads of Enschede. The license plates where registered on Tuesday from 7:00 to 9:00 hour and from 14:00 to 18:00 hour, and on Saturday from 13:00 to 15:00 hour. The register points where cited at the ring roads and major roads through the city, see also figure 4.5. In the license plate survey the first four characters of the license plates and the time (in minutes) of crossing the register point is registered. We use the dataset of Tuesday from 14:00 to 18:00 hour in the tests on realistic routing. Tom Thomas, an employee at the University of Twente, used this data before. We interview him for tips about this data set.

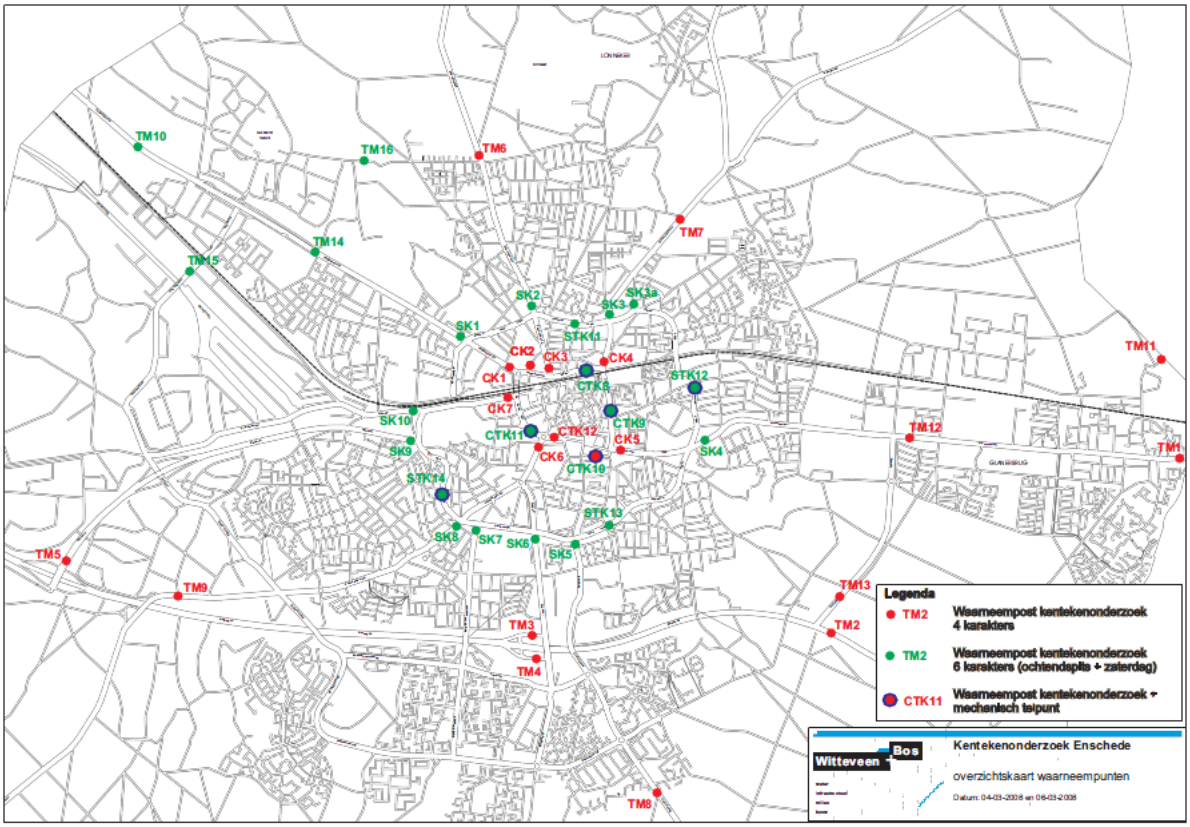


Figure 4.5: The register points of the license plate survey in Enschede

Test network

Our research network does not consist of all roads in Enschede, because we only have data about the main roads. We build our research network in two steps. First we create a network consisting of all the roads with a register point. Second we add often used roads that do not have a register point. The often used roads are determined in the following way: every pair of register points found in the data is searched on a map of Enschede. We select combinations of register points that cannot be reached from driving on the roads with register points only, and that are registered for at least 10 vehicles. The most logical route between the selected combinations is added to the network, this is the group of often taken routes. Figure 4.6 displays our research network.

Trips

Our research network does not include all the register points of the dataset. Therefore, the route choice data consists of trips inside and outside our research network. The trips outside our research network are not considered in our tests. Some registered trips partly took place in our network and partly outside the network.

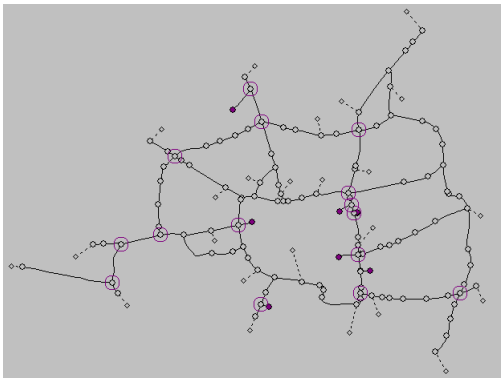


Figure 4.6: The network used for the Enschede tests

We consider only the parts inside our research network. A few trips go from our research network outside the research network and back again inside the research network. We only consider the first part of these trips. Almost half of the remaining trips passed only one register point. We cannot assign vehicles to only one register point, because each register point is represented by one zone and every trip has to have a different destination zone than origin zone in our transport model (we do not use inter zonal trips). The register point of the trips registered at one register point is defined as the origin zone. We define a set of possible destination zones by the nearest register points. We adapt this set by calibrating on the counts at the register points. For each trip consisting of one register point, we select at random one register point from the set of possible destination zones, and define this register point as the destination zone.
The data set describes the sequential passing register points of every vehicle. The routes between the register points remain unknown. We assume that the vehicles follow the shortest route between the register points. The shortest routes are calculated with the Dijkstra algorithm.

4.3.3 Enschede test 1 on 1 OD

We derive the route choices of our variants on the basis of experienced travel time. The route choice data provides us information about the real experienced travel times. If the route choices of the variants react differently in the case where we use these experienced travel times than in the case where we use travel times calculated by Dynasmart, then we could argue that it is far more important to improve the travel time functions before making a recommendation about the route choice model. Therefore we test our variants once with the real experienced travel times, and once with the travel times of Dynasmart. The variants calculate the route choices of the vehicles of only one selected origin destination pair, to have a clear picture of what is happening. We test on origin, registration point TM7, and destination, registration point CK6, because this origin destination pair is the most registered origin destination pair where multiple routes are possible. We assign the vehicles with other origin destination pairs than TM7-CK6 on the routes that registered by the data set.

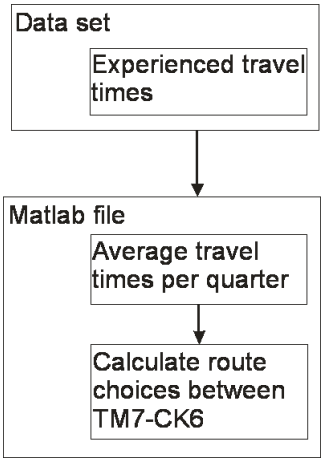


Figure 4.7: Schematic overview of Enschede test 1

We perform the test with experienced travel times in MATLAB. Appendix A describes the code of the MATLAB file. We use the data set of Tuesday the 4th of March from 14:00 to 18:00 hour, and do not use a transport model. The MATLAB file calculates the averaged experience travel times between each pair of register points in a time interval of 15 minutes. We use the free flow travel times for a pair of register points if no vehicle passed these register points in the 15 minutes time interval. We base the calculation of route choice on the travel times of the previous time interval. This process is displayed in figure 4.7. The route choice of our variants are compared with the data and with the route choices of Enschede test 2 on 1 OD.
All our route choice variants use a choice set. The choice set consists of the 10 shortest routes in free flow conditions.

4.3.4 Enschede test 2 on 1 OD

In this test we derive the route choices of our variants on the basis of Dynasmart travel times. We compare the route choices based on the experienced travel times with the route choices based on the travel times of Dynasmart. Also we compare the Dynasmart travel times with the experienced travel times.
The travel times in Dynasmart consist of link travel times and intersection travel times. The intersection delays at intersections with traffic lights depends on the traffic light protocol. The protocol is based on traffic streams from a static assignment in Questor. We use the OD matrix and the departure times of the Tuesday data from 14:00 to 16:00 hour for this static model. The link travel times are calculated with a travel time function depended of intensity, road capacity and free flow speed.
First, we assign all the vehicles on the routes registered in the data set and compare the Dynasmart travel times with the travel times of the data set. Second we determine the route choices of the selected origin destination pair for our variants. We assigns the background traffic (traffic with an other origin destination pair TM7-CK6) on the routes and departure time of the data set. The vehicles with origin and destination pair TM7-CK6 depart at the departure time registered in the dataset. Dynasmart uses a departure time in tenth of a minute precise,

our data set uses departure times in minutes precisely. We use a random number to divide the trips over the minute from the data in a tenth of a minute precisely. The route choice are determined for the vehicles between TM7 and CK6 and compared with the route choice in the data.

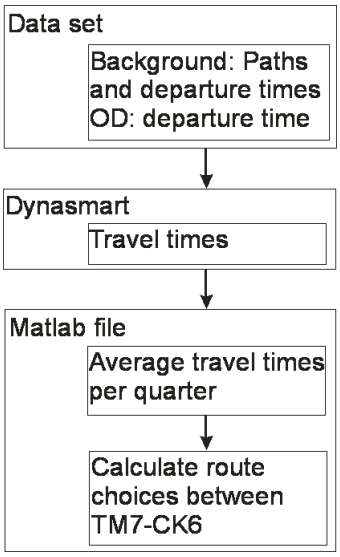


Figure 4.8: Schematic overview of Enschede test 2

We calculate the route choices of the vehicles between TM7-CK6 with the process displayed in figure 4.8. Dynasmart calculates the travel times on every link and gives the travel times as input to the Matlab file. First, the Matlab file calculates the average travel times for every quarter from the travel time file of Dynasmart. Second, the Matlab file calculates the route choice probabilities with the average travel time of the previous 15 minutes interval, like in the test with experienced travel time. The process need not to be repeated, because there is not much traffic with origin destination pair TM7-CK6, wherefore the travel times is not influence much by the route choice of this traffic.
In this test we use travel times from Dynasmart, while we use experienced travel times in the first test. Both travel times have disadvantages and advantages. By using the travel times of the data set we introduce an error in the travel times. The registered vehicles do not have to be average drivers, with average driving behavior, and they may have had a little stop on their way (long stops are already removed from the dataset). Second the passing times at register points are noted in minutes precise, so our travel times are also in minutes precise. The ten shortest routes between OD-pair TM7-CK6 are approximately 12 minutes long, so rounding up on whole minutes introduces a big error in the data set. The Dynasmart travel times have also multiple error sources. An error source in the Dynasmart travel times is the network used in Dynasmart. We only use a part of the network, in our network not all side-roads are incorporated and thus not all intersections. Another source of errors is the travel time function, which is only an estimation of the real travel time. A third source of errors is the amount of vehicles in our network, we make an estimation of the vehicles by the dataset. An advantage of the travel times from Dynasmart in comparison to the travel times from the data set is that these travel times are average travel times over time and not average experienced travel times. All the possible errors in the travel times causes that the travel times calculated in Dynasmart differ from the experienced travel time.
We assign the vehicles of the first 15 minutes according to the route choice model with free flow travel times as input. The first quarter is a warm up period for the Dynasmart model. We do not use the route choices of this first quarter in the comparison with the data. The code of the MATLAB file is described in Appendix A

4.3.5 Enschede test 3 on 10 OD

The previous tests investigate the route choices of our variants on only one origin destination pair. To lower the possibility that we select one origin destination pair for which the route choices are coincidentally well described by our variants, we test our variants on 10 origin destination pairs. We select 10 origin destination pairs with routes that consist of interfering links, such that the route choices on one origin destination pair influences the travel times of another origin destination pair. We compare the route choices of the 10 selected origin destination pairs with the route choices of the data. The reality measure on every origin destination pair is weighed by the total amount of vehicles between that specific origin destination pair. The measure is weighted by the total amount of vehicles, to ensure that the route choices on busy origin destination pairs have more importance in the test than origin destination pairs with little traffic. Such a measure is appropriate for investigating the behavior of the route choice model. This measure is not appropriate to calibrate a base year model, which we want to use to make predictions of future development plans. Links with little traffic receive small weights and are therefore not important in the calibrating process, while maybe due to the development plans these links becomes very important in the future model. Calibrating on the reality measure of the base year model will not give much information on these links, therefore the reality measure is not appropriate for such applications.
We use an iterative process with the MATLAB file and Dynasmart runs to run the test. Figure 4.9 displays this iterative process. We use the paths and departure time of the background traffic (the vehicles with

other origin destination pairs than the 10 selected origin destination pairs). The MATLAB file calculates the route choices between the ten selected origin destination pairs for our route choice model variants. These vehicles depart at the departure time of the data and travel between the origin destination pair from the data. Dynasmart uses in the first run the background traffic to calculate the travel times on the links. These travel times are the input for the MATLAB file. Like in the two earlier Enschede tests the MATLAB file averages the travel times per quarter and calculate the route choices for the selected OD pairs. Then a new iteration starts with a new Dynasmart run based on the background traffic and the route choices calculated with MATLAB. The MATLAB file averages these new travel times per quarter and over the earlier Dynasmart runs. After 10 iterations a stable situations occurs and the iterations stops. This process of 10 iterations is done 2 times for each variant. The first time for the first two hours and the second time for the last two hours. The travel times after ten iteration of the last quarter of the first 2 hours time are the input for the first quarter of the last two hours.

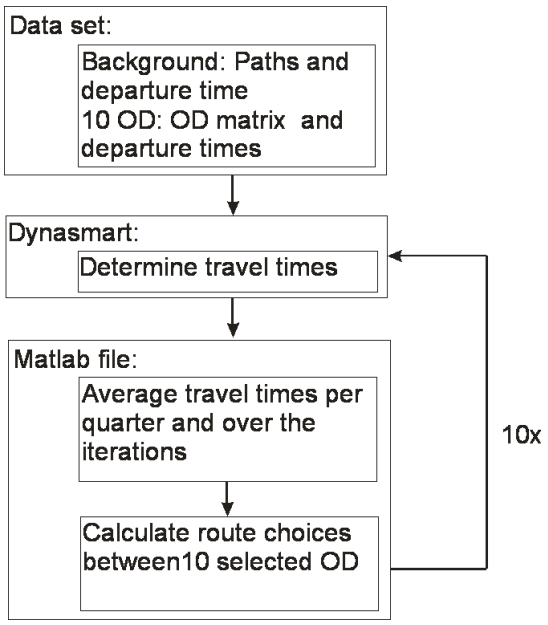


Figure 4.9: Schematic overview of Enschede test 3

In this test, we use a choice set of the 5 shortest routes in free flow conditions for the 10 selected origin destination pair.

4.3.6 Advantages and disadvantages of the approach

The research approach has some advantages and disadvantages, we discuss both the advantages and disadvantages in this section.

One major disadvantage of our approach is that we compare the results of our route choice variants with only one dataset. We use the data of Enschede to measure the reality of our variants, but it is possible that we get other results if we use data of other cities. In Enschede, congestions do not occur often, therefore we do not know if the same results hold for a congestion situation. Further, we test our route choice variants for short trips, but it is generally agreed that route decisions for long trips are different from route decisions for short trips. Therefore our results also do not have to hold for long trips. We get a more general result at the last test, by testing on multiple OD's, but we still have to deal with the fact that the travelers in Enschede could make other route choices than travelers in other regions. Another disadvantage of our research approach is that our measure, reality, is based on an outcome of more processes than the route choice step. The route choices resulting from a transport model are a result of the network in the model, the travel time functions, the origin destination matrix estimation, the loading method, and the route choice model. Errors in other steps than the route choice modeling influence the result of our tests. Therefore we do not know if our results also hold for other loading methods, or for instance in other networks. Of course, we cannot test with all possible loading methods to investigate all possible influences of the loading methods, but we use different loading methods to get a good sense of the possible influences of the loading method. In the Enschede tests we assign all vehicles in a 15 minutes interval according to the same probability distribution. We choose to aggregate the vehicles in a 15 minute interval, because there is not much traffic, and it is hard to assign 1 vehicle for 30% to one route and for 70% to another route. The aggregation of the vehicles influences the results of the tests, we expect that the shortest route principle is more accurate if the shortest route is calculated in smaller time intervals. In Chapter 3 we mentioned two other shortcomings of our research approach. We do not investigate other utility functions than the travel time, predicted travel time and combinations of distance and travel time. Also we do not investigate choice set generation methods, although we know that the choice set and the utility function strongly influence the route choice results. An investigation of these two elements is beyond the scope of this master thesis project.

A huge advantage of our research approach is that DHV is able to use our results directly. We test the route choice models in combination with DHV's current transport models in such a way as which they would use the route choice model. Therefore, these tests give a good overview of the possibilities of our route choice model variants. An other advantage of our approach is that we also test if the route choice model is in fact the problem for DHV, or that an improvement of the travel time function is a better investment. Two other

advantages of our research approach are practical issues. First, we need route choice data only at the end of the research, when we test in Enschede, which give us time to find route choice data. Second, we perform the tests in the order from simple to complicated. This order makes the results of the test more understandable and errors in the tests are observed earlier.

Chapter 5

Route choice model variants

This chapter describes the C-logit and game theory variants which we test. Section 5.1 describes the C-logit variants, and Section 5.2 the game theory variants.

5.1 C-logit variants

We test C-logit models that vary on the following elements:

- Commonality factor functions
- The θ value
- Power of the travel time
- The β of the commonality factor (and γ for the first commonality factor)
- Utility functions
- Penalty for usage of the on and of ramp
- Bias for different road categories

This section describes these variants of the C-logit and why we have selected them. Next to the variants in which one of the elements above are varied we also test variants in which two elements are varied.

5.1.1 Commonality factor functions

The literature proposes four different variants of the commonality factor, see Chapter 3 for the definitions of all variables. We use these four variants and an additional fifth form which is derived from the Path Size logit route choice. The Path Size logit is also described in Chapter 3. The five tested commonality factor forms are:

$$
\begin{aligned}
CF_i &= \beta \ln \sum_{j \in C_n} \left(\frac{L_{ij}}{\sqrt{L_i L_j}} \right)^{\gamma} \\
CF_i &= \beta \ln \sum_{a \in \Gamma_i} \frac{l_a}{L_i} N_a \\
CF_i &= \beta \sum_{a \in \Gamma_i} \frac{l_a}{L_i} \ln N_a \\
CF_i &= \beta \ln \left(1 + \sum_{j \in C_n, j \neq i} \frac{L_{ij}}{\sqrt{L_i L_j}} \frac{L_i - L_{ij}}{L_j - L_{ij}} \right) \quad [4] \\
CF_i &= \beta \ln \left(\sum_{a \in \Gamma_i} \frac{l_a}{L_i} \frac{1}{N_a} \right)
\end{aligned}
$$

5.1.2 The θ value

The results of the C-logit approach depend on the scale of the utility, we show this by an example. Suppose that we calculate the route choice between two routes. In the first case we express the utility in minutes and in the second case in seconds. The travel time of the first route is two minutes and the travel time of the second route is one minute. There is not any overlap between the two routes and we use a θ of one. The route choice probability of route 1 in the first case is:

$$\frac{e^{-2}}{e^{-2}+e^{-1}} \approx 0.2689$$

While the route choice probability in the second case, where we use seconds instead of minutes is:

$$\frac{e^{-120}}{e^{-120}+e^{-60}} \approx 0.0000$$

This example shows that route choice probabilities for the same route differ if different utility scales are used. A proof of the dependency on the utility scale of the C-logit model is given in the box below. We vary the θ value to test the influence of the scale.

We first show that the results of the more simple model, the Multinomial logit depends on the scale of the utility. Next the dependency of the C-logit model on the utility scale is shown.

Multinomial logit

We show that the Multinomial logit depends on the scale of the utility by determining the probability distributions of two networks. The only difference between the networks is the scale, so if we proof that the probability distributions differ for these networks we proof the dependency of scale. In this section we will proof Theorem 5.1.

Theorem 5.1. *The results of the Multinomial logit model depends on the scale of the utility.*

Proof. According to the Multinomial logit distribution, the probability of choosing alternative i for individual n is given by:

$$P(i|C_n) = \frac{e^{V_i}}{\sum_{j \in C_n} e^{V_j}}$$

We assume that route j of network 1 has utility $x + \delta_j$, with $\delta_i = 0$, so:

$$P(i|C_n) = \frac{e^{x}}{e^{x+\delta_1} + ... + e^{x+\delta_n}} = \frac{1}{1 + e^{\delta_1} + ... + e^{\delta_{i-1}} + e^{\delta_{i+1}} + ... + e^{\delta_n}}$$

Now consider network 2 which is a stretched version of network 1. The utility of the alternatives of this network is given by $V_g i = mx + m\delta_i$. Suppose that scale does not matter, than the probability distributions of the two networks should equal each other, than $P(i|C_n) = P(g_i|C_n)$.

$$\begin{aligned} P(g_i|C_n) &= \frac{e^{mx}}{e^{mx+m\delta_1} .. + e^{mx+m\delta_{i-1}} + e^{mx+m\delta_{i+1}} + ...e^{mx+m\delta_n}} \\ &= \frac{1}{1 + e^{m\delta_1} + ... + e^{m\delta_{i-1}} + e^{m\delta_{i+1}} + ... + e^{m\delta_n}} \\ &\neq \frac{1}{1 + e^{\delta_1} + ... + e^{\delta_{i-1}} + e^{\delta_{i+1}} + ... + e^{\delta_n}} \quad for \;\; m \neq 1 \end{aligned}$$

A contradiction, so the results of the Multinomial Logit model depends on the scale of the utility. □

C-logit

We use the results of the Multinomial logit for the proof of the C-logit model. We proof that the C-logit model with the commonality factors we use is dependent of scale. First, we show that none of the used commonality factors depends on the scale. Second, we proof that that the C-logit model with commonality factors which are independent of scale result in probability distributions that are dependent of scale.

Theorem 5.2. *The results of the C-logit model with one of the following five commonality factor functions are depended of the scale of the utilities.*

$$
\begin{aligned}
CF_i &= \beta \ln \sum_{j \in C_n} \left(\frac{L_{ij}}{\sqrt{L_i L_j}} \right)^{\gamma} \\
CF_i &= \beta \ln \sum_{a \in \Gamma_i} \frac{l_a}{L_i} N_a \\
CF_i &= \beta \sum_{a \in \Gamma_i} \frac{l_a}{L_i} \ln N_a \\
CF_i &= \beta \ln \left(1 + \sum_{j \in C_n, j \neq i} \frac{L_{ij}}{\sqrt{L_i L_j}} \frac{L_i - L_{ij}}{L_j - L_{ij}} \right) \\
CF_i &= \ln \left(\sum_{a \in \Gamma_i} \frac{l_a}{L_i} \frac{1}{N_a} \right)
\end{aligned}
$$

Proof. The five commonality factors only depend on the fractions of overlapping route lengths, they do not depend on the scale of the route length or the utility scale.
Now consider the two networks of the previous section (network 2 is a stretched version of network 1). The probability of choosing route i in network 1 is given below, where C_i is the commonality factor of route i.

$$P(i|C_n) = \frac{e^{x+\delta_i - C_i}}{e^{x+\delta_1 - C_1} + .. + e^{x+\delta_i - C_i} + ... + e^{x+\delta_J - C_J}} = \frac{1}{e^{\delta_1 - C_1 + C_i} + ... + e^{\delta - C_J + C_i}}$$

Now remember that route i of network 1 and route g_i of network 2 are similar except for the scale of the route. Therefore the commonality factor of route i of network 1 is the same as the commonality factor of route g_i of network 2, due to the independence of scale of the commonality factor. The probability of choosing route g_i from network 2 is defined by:

$$P(g_i|C_n) = \frac{e^{mx+m\delta_i - C_i}}{e^{mx+m\delta_1 - C_1} + ... + e^{mx+m\delta_J - C_J}} = \frac{1}{e^{m\delta_1 - C_1 + C_i} + ... + e^{m\delta_J - C_J + C_i}}$$

The C-logit with one of the five mentioned commonality factors is independent of scale if $P(i|C_n) = P(g_i|C_n)$, but:

$$e^{\delta_1 - C_1 + C_i} + ... + e^{\delta - C_J + C_i} \neq e^{m\delta_1 - C_1 + C_i} + ... + e^{m\delta_J - C_J + C_i} \quad for \;\; m \neq 1$$

A contradiction, so we haven proven that the C-logit is dependent of the scale of the utility and thus we have proven Theorem 5.2. □

5.1.3 Travel time power

Thomas and Tutert (2009) fit aggregated data of a license plate survey in Enschede to travel time. They fitted a power of 0.7 to the travel time [29]. To test if the travel time with this power fits better to route choice data we added a power ι to the travel time. The utility equals now: $V_{in} = -(TT_i)^{\iota}$. function. For travel times larger than 1, a power lower than 1 assigns less weight to travel time differences, and a power larger than 1 assigns more weight to travel time differences.

5.1.4 Commonality factor parameters

All five commonality factor functions are expressed with the parameters β and the first one also with parameter γ. These parameters influence the value of the commonality factor, and therefore the effect of the commonality factor on the result. We use variants with different β and γ values to test the influence of these parameters.

5.1.5 Utility expression

In Chapter 3 we mention the importance of the utility expression. The utility can be expressed in different variables. We test the following expressions of the utility function:

- Travel time
- A combination of travel time (weight 0.75) and distance (weight 0.25)

- Travel time+

We test travel time, because travel time is the most often used utility function at DHV. It is possible to use a combination of travel time and distance as utility function in Questor, we are curious of this utility function is more accurate than travel time and we test this variant. People make risk averse route choice decisions, therefore we also want to incorporate risk in the route choice models. Travel time+ is the travel time with an additional load. We only test travel time+ in the static incremental assignment of the theoretical test, because developing a method how to apply travel time+ in a dynamic assignment is beyond the scope of this research project. Figure 5.1 shows the assignment procedure for the predicted travel time variants. The utility travel time+ is calculated by using an additional hypothetical incremental assignment of five steps. Before each assignment step a hypothetical incremental assignment is performed. We assign in total half the demand in the hypothetical incremental assignment. After this hypothetical incremental assignment we calculate the travel time. We call these travel times travel time+. We do not use other variables than time and distance to describe the utility, because it is generally agreed that most drivers attempt to minimize travel time and distance [16].

5.1.6 Penalty value

Another possible variant is the use of penalties in the utility function of the unrealistic route. A penalty corrects discrepancies in the utility value. We test two types of penalties. The first type observes the links of a route and decreases the utility if it contains a pair of on- and off-ramps. An on- and off-ramp is called a pair if they are located at the same intersection, thus if it is possible to drive from the off-ramp directly to the on-ramp. We do not oblige that the off- and on-ramp are sequentially included in the route, to avoid the detour mentioned in Section 2.4. We use this penalty function in the theoretical test. We use the second penalty type in the Enschede tests. The second penalty function penalizes or favorizes one or multiple predefined routes.

5.1.7 Road category bias

Another form of penalizing contains the use of road type biases. The travel times are multiplied by a bias. A bias below one will lower the travel time in the utility function, such a bias value is used to state a preference for this type of road. The highway bias is the lowest bias and the ramp bias the highest, because people prefer to drive on the highway.

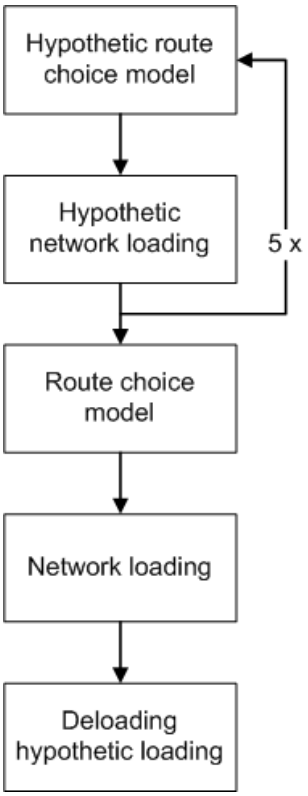


Figure 5.1: Overview of the assignment for travel time+ variants

5.2 Game theory variants

The game theory variants we test are:

- Congestion game with different objectives
- Smooth fictitious play with different θ values
- Smooth fictitious play with different β values
- Demon player with different damaging capacities
- The three games with penalty functions
- The three games with road category bias

This section describes these variants and why we selected them. The three mentioned games are extensively explained in section 3.2.4.

5.2.1 Congestion game

The simplest form of traffic applications in game theory is the atomic congested game. We test different variants of the congestion game. The influence of the length of the route is tested by varying the objective. The players minimize a combination of their travel time and their route distance, this combination is calculated by the following formula: $p * distance + (1 - p) * TravelTime$. We vary the p value in this test.

5.2.2 Smooth fictitious play, θ values

Almost 60% of the car trips in the Netherlands consists of trips shorter than 10 km [30]. These trips are made near home, and we may assume that these trips are distributed on a small subset of the roads. Therefore we expect that people drive often on the same roads and a learning effect can be seen. Moreover experimental results of Avineri (2002) showed that travelers' behavior is better captured by learning models, see Section 3.2.1. The learning effect is incorporated in the fictitious play. We investigate the smooth fictitious play of Cominetti et. al. (2008). Cominetti et. al. (2008) use the Multinomial logit as map from perceptions of pay-offs to mixed strategies. Before we have shown that the Multinomial logit depends on the scale, therefore we test different θ values for the smooth fictitious play.

5.2.3 Smooth fictitious play, β values

The Multinomial logit can be seen as a C-logit with β value zero. Comminetti et. al. (2008) proof the convergence of the smooth fictitious play under the Multinomial logit rule, but they did not proof the convergence under the C-logit rule. We proof the convergence of the smooth fictitious play with the C-logit under certain conditions in the box below. In the box is also described when these conditions are met in the theoretical test. We test the smooth fictitious play with different β values for the C-logit.

Notation

We describe the smooth fictitious play under the Multinomial logit rule in section 3.2.4. We change the used notation of section 3.2.4 a bit to describe the smooth fictitious play under the C-logit rule. Still, let $P = \{1, ..., N\}$ denote the set of players. Each player i select an action a_i^n at stage n from the pure strategy A_i according to their mixed strategy s_i^n. The mixed strategy is determined by the C-logit rule, which is strictly positive:

$$\pi_{ij} = s_{ij}(\widetilde{V}_i) = \frac{e^{\theta \widetilde{V}_{ij}}}{\sum_{k \in A_i} e^{\theta \widetilde{V}_{ik}}}$$

Where $\widetilde{V}_{ij}$ is the corrected perception of utility of route j for player i. The utility is corrected by the commonality factor. We calculate the corrected perception of the utility by: $\widetilde{V}_{ij} = (-PTT_i(j) - CF(j))$, where $CF(j)$ is the commonality factor. The perception of the corrected utility is updated at every stage of the game by the following rule:

$$\widetilde{V}_{ij}^{n+1} = \begin{cases} (1 - \frac{1}{n}) \cdot (-PTT_i^n(j) - CF(j)) + \frac{1}{n} \cdot (-TT^n(j) - CF(j)) & \text{if } j = a_i^n, \\ -PTT_i^n(j) - CF(j) & \text{otherwise} \end{cases}$$

The payoff is no longer the negative travel time of the chosen route, but $g_i^n = -TT^n(j) - CF(j)$ under the C-logit rule. We denote the payoff function by $G_i : A_i \times A_{-i} \to \mathbb{R}$ where $A_{-i} = \prod_{i \neq j} A_j$. Now suppose that the mixed strategies of all players except player i are given by π_{-i}, then $G_i(j, \pi_{-i})$ describes the expected payoff of player i by selecting route j . We denote the expected payoff function of player i by: $F_{ij}(\pi) = G_i(j, \pi_{-i})$. We could also write the expected vector payoff as a function of the state, the state is described by the corrected perception of the utility vector. We denote $C(\widetilde{V})$ the map of the expected vector payoff as a function of the state and: $C(\widetilde{V}) = F(S(\widetilde{V}))$, where $S(\widetilde{V})$ is the function of the mixed strategies of the players at state $\widetilde{V}$.
We use two norms and the inner product in the proof below. The inner product in the proof is denoted by: $\langle \cdot \rangle$. The maximum difference between the elements of vectors a and b is given by the infinity norm: $\|a - b\|_\infty = \max_i |a(i) - b(i)|$. The one norm represents: $\|a - b\|_1 = \sum_i |a(i) - b(i)|$. The map A is a $\| \cdot \|_\infty$ - contraction if $|A(x) - A(y)| < \|x - y\|_\infty$ holds.

Proof

We follow the proof of [27] about the convergence of the smooth fictitious play under the Multinomial logit rule. Cominetti et. al. (2008) prove that the smooth fictitious play converges if the cost function $C(\cdot)$ is a

$\|\cdot\|\infty$ -contraction. Therefore we have to proof that the function $C(\cdot)$ is also under the C-logit rule a $\|\cdot\|\infty$ -contraction. To proof this we need the following Lemma:

Lemma 5.3. *For each $i \in P$ and $j \in A^i$ the C-logit rule satisfies:*

$$||\nabla s_{ij}(\widetilde{V}_i)||_1 = 2\theta s_{ij}(\widetilde{V}_i)(1 - s_{ij}(\widetilde{V}_i))$$

Proof. Note from the previous section that $s_{ij}(\widetilde{V}_i)$ under the C-logit is described by:

$$s_{ij}(\widetilde{V}_i) = \frac{e^{\theta \widetilde{V}_{ij}}}{\sum_{k \in A_i} e^{\theta \widetilde{V}_{ik}}}$$

The partial derivative of s_{ij} to $\widetilde{V}_{ij}$ gives

$$\begin{aligned} \frac{\partial s_{ij}}{\partial \widetilde{V}_{ij}} &= \frac{\theta e^{\theta \widetilde{V}_{ij}} \sum_{k \in A_i} e^{\theta \widetilde{V}_{ik}} - \theta e^{\theta \widetilde{V}_{ij}} e^{\theta \widetilde{V}_{ij}}}{(\sum_{k \in A_i} e^{\theta \widetilde{V}_{ik}})^2} \\ &= \theta s_{ij} - \theta (s_{ij})^2 \end{aligned}$$

and the partial derivative of s_{ij} to $\widetilde{V}_{it}$ for $t \neq j$ gives

$$\begin{aligned} \frac{\partial s_{ij}}{\partial \widetilde{V}_{it}} &= \frac{-\theta e^{\theta \widetilde{V}_{it}} e^{\theta \widetilde{V}_{ij}}}{(\sum_{k \in A_i} e^{\theta \widetilde{V}_{ik}})^2} \\ &= -\theta s_{it} s_{ij} \end{aligned}$$

Thus

$$\frac{\partial s_{ij}}{\partial \widetilde{V}_{it}} = \theta s_{ij}(\delta_{jt} - s_{it})$$

with $\delta_{jt} = 1$ if $j = t$ and $\delta_{jt} = 0$ otherwise, from which we get:

$$||\nabla s_{ij}(\widetilde{V}_i)||_1 = \theta s_{ij} \sum_{t \in A_i} |\delta_{st} - s_{it}| = 2\theta s_{ij}(\widetilde{V}_i)(1 - s_{ij}(\widetilde{V}_i))$$

□

Before we start the proof that $C(\cdot)$ is a $\|\cdot\|_\infty$ contraction, we need to define the parameter η. This η is an upper bound for the impact over a player's payoff when a single player changes her move, namely

$$|G^i(j, x) - G^i(j, y)| \leq \eta$$

for each player $i \in P$, every pure strategy $j \in A_i$, and all pairs $x, y \in A_{-i}$ such that $x_j = y_j$ except for one $j \neq i$. Comminetti et. al. (2008) need to define $\omega = \max_{i \in P} \sum \theta_i$, but we use the same θ for every player in our game so we do not need to define a variable ω.
Next to the variable η we also need the mean value theorem in the proof. The mean value theorem states: let f(x) be differentiable on the open interval (a, b) and continuous on the closed interval $[a, b]$. Then there is at least one point c in (a, b) such that $f'(c) = \frac{f(b)-f(a)}{b-a}$.

Theorem 5.4. *Under the C-logit rule, if $2\theta\eta < 1$ then $C(\cdot)$ is a $||\cdot||_\infty$ -contraction.*

Proof. We will proof this theorem in two parts, first we proof that

$$\|s_i(\widetilde{V}_i) - s_i(\widetilde{U}_i)\|_1 \leq 2\theta \|\widetilde{V} - \widetilde{U}\|_\infty$$

second we will proof that

$$|C_{ij}(\widetilde{V}) - C_{ij}(\widetilde{U})| \leq \eta \sum_{l \neq i} \|s_n(\widetilde{V}_l) - s_l(\widetilde{U}_l)\|_1$$

Combination of these two statements gives:

$$|C_{ij}(\widetilde{V}) - C_{ij}(\widetilde{U})| \leq 2\eta\theta\|\widetilde{V} - \widetilde{U}\|_\infty$$

and thus if $2\theta\eta < 1$ then $C(\cdot)$ is a $||\cdot||_\infty$ -contraction.

Take for the first part of the proof $w_i \in \mathbb{R}^{Ai}$ with $||w_i||_\infty = 1$ and $||s_i(\widetilde{V}_i) - s_i(\widetilde{U}_i)||_1 = \left\langle w_i, s_i(\widetilde{V}_i) - s_i(\widetilde{U}_i)\right\rangle$. Then according to the mean value theorem there is a $\widetilde{W}_i \in [\widetilde{V}_i, \widetilde{U}_i]$ with

$$\|s_i(\widetilde{V}_i) - s_i(\widetilde{U}_i)\|_1 = \sum_{t\in A_i} w_i^t \left\langle \nabla s_{it}(\widetilde{W}_i), \widetilde{V}_i - \widetilde{U}_i\right\rangle \leq \sum_{t\in A_i} \|\nabla s_{it}(\widetilde{W}_i)\|_1 \|\widetilde{V}_i - \widetilde{U}_i\|_\infty. \tag{5.1}$$

By using Lemma 5.3 and the fact that $s_i(\widetilde{W}_i) \in S_i$ we get:

$$\|s_i(\widetilde{V}_i) - s_i(\widetilde{U}_i)\|_1 \leq \sum_{t\in A_i} 2\theta s_{jt}(\widetilde{W}_i)\|\widetilde{V}_i - \widetilde{U}_i\|_\infty \leq 2\theta\|\widetilde{V} - \widetilde{U}\|_\infty.$$

This result will be used in the second part of the proof.
In the second part of the proof we consider the difference $C_{ij}(\widetilde{V}) - C_{ij}(\widetilde{U})$ for a fixed player $i \in P$ and pure strategy $j \in A_i$. Remember from the first section of this appendix that we can write $C_{ij}(\widetilde{V}) - C_{ij}(\widetilde{U}) = F_{ij}(S(\widetilde{V})) - F_{ij}(S(\widetilde{U}))$. We can write this difference as the telescopic sum of $\Lambda_l = F_{ij}(\pi_l) - F_{ij}(\pi_{l-1})$ for $l = 1, ..., N$ where

$$\pi_l = (s_1(\widetilde{V}_1), ..., s_l(\widetilde{V}_l), s_{l+1}(\widetilde{U}_{l+1}), ..., s_N(\widetilde{U}_N))$$

Since $F_{ij}(\pi)$ only depends on π^{-i} & j, and not on π^i we have $\Lambda_i = 0$. We can express the remaining terms Λ_l as the difference between expectations conditioned on choosing actions from mixed strategies $s_l(\widetilde{V}_l)$ and $s_l(\widetilde{U}_l)$: $\Lambda_l = \left\langle B_l, s_l(\widetilde{U}_l) - s_l(\widetilde{U}_l)\right\rangle$ where for $t \in A_l$ we put:

$$B_l^t = \sum_{x\in A_{-i}, x^l = t} G^i(j, x) \prod_{k\neq i,l} \pi_l^{kx^k}$$

Because $s_l(\widetilde{V}_l)$ and $s_l(\widetilde{U}_l)$ are both probability distribution we may write:

$$\begin{aligned}
\left\langle B_l - B_l^r, s_l(\widetilde{V}_l) - s_l(\widetilde{U}_l)\right\rangle &= \sum_{t\in Al}(B_l^t - B_l^r)(s_{lt}(\widetilde{V}_l) - s_{lt}(\widetilde{U}_l)) \\
&= \sum_{t\in Al} B_j^t(s_{lt}(\widetilde{V}_l) - s_{lt}(\widetilde{U}_l)) - \sum_{t\in Al} B_l^r(s_{lt}(\widetilde{(V)}_l) - s_{lt}(\widetilde{U}_l)) \\
&= \Lambda_l - B_l^r \sum_{t\in Al}(s_{lt}(\widetilde{V}_l) - s_{lt}(\widetilde{U}_l)) \\
&= \Lambda_l
\end{aligned}$$

So $\Lambda_l = \left\langle B_l - B_l^r, s_l(\widetilde{V}_l) - s_l(\widetilde{U}_l)\right\rangle$ for any fixed $r \in A_l$. Now because we have chosen η such that $|G^i(j, x) - G^i(j, y)| \leq \eta$ we can state the following:

$$\begin{aligned}
|B_l^t - B_l^r| &= |\sum_{x\in A_{-i}, x^l = t} G^i(j, x) \prod_{k\neq i,l} \pi_l^{kx^k} - \sum_{x\in A_{-i}, x^l = r} G^i(s, x) \prod_{k\neq i,l} \pi_l^{kx^k} \\
&= |\sum_{x\in A_{-i}} (G^i(s, x|x^l = t) - G^i(s, x|x^l = r)) \prod_{k\neq i,l} \pi_j^{kx^k} \\
&\leq |\sum_{x\in A-i} \eta \prod_{k\neq i,l} \pi_l^{kx^k}| \\
&\leq \eta \sum_{x\in A_{-i}} \prod_{k\neq i,l} \pi_l^{kx^k} \\
&\leq \eta
\end{aligned}$$

So $\Lambda_l \leq \eta||s_l(\widetilde{V}_l) - s_l(\widetilde{U}_l)||_1$, therefore we deduce

$$|C_{ij}(\widetilde{V}) - C_{ij}(\widetilde{U})| \leq \eta \sum_{l\neq i} \|s_l(\widetilde{V}_l) - s_l(\widetilde{U}_l)\|_1$$

This combined with (5.1) gives:

$$|C_{ij}(\widetilde{V}) - C_{ij}(\widetilde{U})| \leq 2\eta\theta\|\widetilde{V} - \widetilde{U}\|_\infty$$

□

Convergence in our tests

Now, we know that the smooth fictitious play under the C-logit converges if $2\theta\eta < 1$ holds with $|G^i(j,x) - G^i(j,y)| \leq \eta$. If we find the η value of the test values then we can determine the allowed θ values for the test case scenarios. The value of G consists of the travel times and the commonality factor. The travel time depends on the amount of vehicles on the route, and thus the strategy of the players, but the commonality factor is a constant. Therefore $|G^i(j,x) - G^i(j,y)| = |TT_i(j,x) - TT_i(j,y)| \leq \eta$ where $TT_i(j,x)$ is the travel time of player i who chooses action i and the other players divided on the routes according to x.
The highest possible intensity/capacity ratio in the theoretical test, is the intensity/capacity ratio on the rural road if all players choose route 1. The influence of one player changing his route is the highest on the travel time of route 1 in this scenario. The travel time is lowered by 0.127 minute if one player changes his route from the rural route to another route. Thus $\eta = 0.127$ in this test, and θ should be chosen such that $\theta \leq \frac{1}{0.254}$. This means that we cannot proof that the game converge by θ values of 4 and larger in the theoretical test. Unfortunately the travel time calculations in Dynasmart are a black box for the user. Therefore we cannot calculate what the maximum difference in travel time is when one traveler changes his route in the Enschede tests.

5.2.4 Demon player game

We expect that travelers act risk avoiding, therefore we investigate the game theory variant with a demon player. We test demon player game variants in which the demon lowers the capacity of a link with different percentages. A demon is active for vehicles of one specific origin destination pair only. The demon damages the link that increases the total travel time of all travelers on his origin destination pair the most. If the network consists of links with different lengths the link with the longest link will be selected, because damaging this link will influence the total travel the most. To deal with this effect we divide the links in smaller links of equal length. Dividing a link in several smaller links has no further influence on the static result, but has influence on the dynamic results, see Appendix B.

5.2.5 Penalty value

The penalty value can as easily as with the C-logit variants be used at the game theory variants. The penalty value is subtracted from the utility at the C-logit variants, the game theory variants do not use utilities but objective functions. The utility is a negative value and the objective of the games a positive value, therefore we add the penalty value to the objective function:

$$\min_{i \in Cn} (TravelTime(i) + PenaltyValue(i))$$

With $TravelTime(i)$ the travel time of route i, and $PenaltyValue(i)$ the penalty value of route i. The set C_n is the route choice set for player n. We use the penalty function at the congestion game, the smooth fictitious play, and the demon player game.

5.2.6 Road category bias

The road category bias is also used at the game theory variants. The objective of the players in a game with a road category bias becomes:

$$\min_{i \in Cn} \left(\sum_j LinkTime(j) \cdot Bias(j) \cdot PathLink(i,j) \right)$$

where $PathLink(i,j) = 1$ if link j is contained in path i, and $Bias(j)$ is the road category bias value of link j. We use the road category bias at the congestion game, the smooth fictitious play, and the demon player game.

Chapter 6

Results theoretical test

The results of the theoretical test give us a first impression whether C-logit and game theory variants can result in more realistic route choices than the shortest route principle. Further, this test provides us information about the influence of the parameters on the route choices. We test in a controlled environment, in which the influence of the parameters is traced back more easy. We do not have any route choice data of the theoretical test network, but we design our test network in such a way that it is not realistic to assume that much traffic chooses the third route. Therefore we assume that variants with low possibilities of choosing route 3 are more realistic than route choices with many choices for route 3. The reality of the route choice variants is defined as the extent in which it does not assign vehicles to the third route. We investigate the influence of the parameters on this reality score. We are not trying to find a route choice model that minimizes the traffic on the third route, because then we would remove the third route from our choice set. We have to comment that in the current traffic models the choice set can only be specified by the k-shortest routes for some positive integer k.
Section 6.1 describes the results of the C-logit and game theory variants using a static assignment, and Section 6.2 describes the results of the most promising variants using a dynamic assignment. The previous chapter describes the used C-logit and game theory variants.

6.1 Static results

We use an incremental assignment of ten equal steps for the static assignments, see Chapter 4. We compare the results of the route choice variants with the shortest route based principle. We use the same incremental loading method for the shortest route principle as for the C-logit variants. In total 2000 vehicles have a free route choice in this network, the other vehicles are background traffic. The shortest route based principle results in 200 vehicles on route 1, 1400 vehicles on route 2 and 400 vehicles on route 3, of the 2000 vehicles with free route choice. Route 3 is an unrealistic route. So if our variants assign less vehicles than 400 on route 3, then has our variant a higher reality score for this test than the shortest route principle.

6.1.1 C-logit

This section discusses the expected results and the obtained results for the C-logit variants in the static assignment of the theoretical test. A table describes the used parameter settings for every test. The tested and varied parameter is printed **bold** in the tables. The categories in the table are utility, commonality factor, θ value, power, β value, γ value, penalty value, and bias values. The category utilities describes which of the following utility expressions is used: travel time, travel time and distance, or travel time+. The commonality factor describes the used commonality factor function. The θ value influences the scale of the utility, the third column describes this value. The power is the value of ι in the function: $V_{in} = -(TT_i)^{\iota}$. We keep $\iota = 1$ in all variants except for the power variants. The β and γ values are parameters values in the commonality function. The penalty value describes the used penalty value for route 3. This value is subtracted of the utility of route 3. The road category bias is described in three numbers. The first number describes the highway bias, the second the rural road bias, and the third the ramp bias used in the test.
We start our analysis of the C-logit with a partial derivative of the route choice probability of route 3. The sign of the derivative give us insight in which direction the specific parameter will influence the probability distribution. A positive partial derivative means that if we increase the specific parameter the probability of choosing route 3 will increase. The value of the partial derivative shows the extent in which the parameter influence the route choices. We do not derive the exact value of the partial derivative, because this value depends on many different factors and changes a lot during the assignment step.

Commonality factor functions

The five commonality factor functions produce almost the same commonality factor values, but there are some small mutual differences. To get an idea of the influence of the commonality factor values we derive two partial derivatives of the route choice probability of route 3. The route choice probability of route 3 is given by:

$$P_3 = \frac{e^{\theta(V_3 - CF_3)}}{\sum_{i=1}^{3} e^{\theta(V_i - CF_i)}}$$

Where V_i stand for the utility of route i. In general the utility represents the negative travel time of a route. The partial derivative to CF_3 is:

$$\frac{\partial P_3}{\partial CF_3} = \frac{-\theta e^{\theta(V_3 - CF_3)}(e^{\theta(V_1 - CF_1)} + e^{\theta(V_2 - CF_2)})}{(h(CF_3))^2}$$

Where $h(CF_3) = \sum_{i=1}^{3} e^{\theta(V_i - CF_i)}$. Note that $h(CF_3)^2 \geq 0$, $e^x \geq 0$ for all x, and $\theta \geq 0$ for all i. Therefore the sign of the partial derivative of P_3 to CF_3 is negative, which means that the probability of choosing route 3 is lower for higher commonality factor values of path 3. This result is obvious, since the commonality factor corrects for overrated routes. From the previous result it is easy to see that the partial derivative $\frac{\partial P_3}{\partial CF_i}$ for $i = 1, 2$ is positive. This means that the commonality factor variant that has the highest commonality factor for route 3 in compare to the commonality factor for the other two routes assigns the least traffic to route 3.
The commonality factor values of the five commonality factor functions do not differ much, therefore we expect that the commonality factor has not much influence. The commonality factor of route 3 is the highest for all commonality factor functions. Commonality factor function four has the largest difference between the values for route 1 and 2 and the value of route 3. Therefore we expect that this commonality factor scores the best on reality. The parameter values of this test are given in table 6.1.

Utility	**Commonality factor**	θ value	Power	β value	γ value	Penalty value	Bias values
all	**all**	1.33	1	1	1	0	[1 1 1]

Table 6.1: The used parameter values for the commonality factor variants

Figure 6.1 shows the results of the theoretical test with different commonality factors. We see that choosing an other commonality factor function influences the reality of the route choice model a little. The fourth commonality factor function scores the best as we expected. The figure also shows that travel time+ scores the best for the parameter settings used in this test.

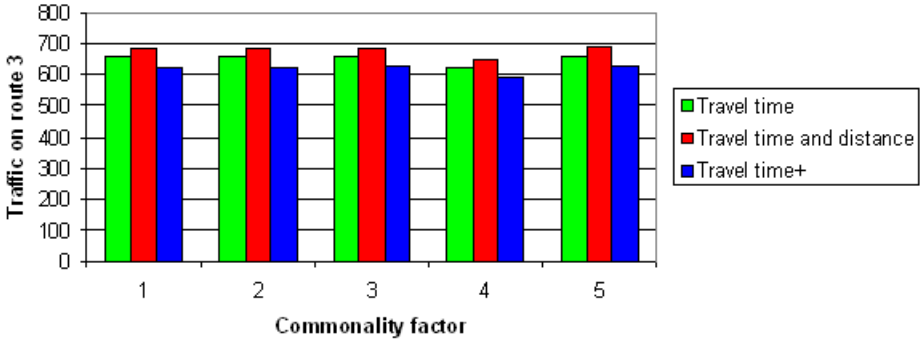


Figure 6.1: The results for different commonality factor functions

θ value variants

We vary the θ value to investigate the influence of the utility scale on the route choices. To calculate the partial derivative of $P(3)$ to θ we define a, b, and c, the systematic utilities $(V_i - CF_i)$ of route 1, 2, and 3. The partial derivative is:

$$\frac{\partial P_3}{\partial \theta} = \frac{ce^{\theta(a+c)} + ce^{\theta(b+c)} - ae^{\theta(a+c)} - be^{\theta(b+c)}}{(h(\theta))^2}$$

Where $h(\theta) = \sum_{i=1}^{3} e^{\theta(V_i - CF_i)}$. Again note that $h(\theta) \geq 0$ and that $a, b, c \leq 0$. The partial derivative does not give us a strict conclusion as with the commonality factor function, but we can conclude some things from this

partial derivative. We see that the partial derivative is positive if the systematic utility of route 3 is largest, this means that in that case the probability of choosing route 3 increases when θ is increased. We can also draw a conclusion for the opposite case, if the systematic utility of route 3 is the smallest, then the probability of choosing route 3 decreases when the value of θ increases. The systematic utility of route 3 is smaller than the systematic utilities of route 1 and 2, at the start of the assignment. Unfortunately we cannot predict if this stays the same during the total assignment step. And we cannot draw conclusions for cases between $c \geq a, b$ and $c \leq a, b$ from the partial derivative, but we know the following. In a Multinomial logit with a very low utility scale, the traffic is equally divided among the three routes. The traffic is assigned according the shortest route principle when a very high utility scale is used in the Multinomial logit. The C-logit reacts more or less the same as the Multinomial logit. The only difference between the C-logit and the Multinomial logit is the commonality factor. The commonality factor lowers the utility relatively the same for every scale. Due to the fact that the C-logit is based on absolute differences the C-logit is most influenced by its commonality factor for high scales. Therefore we expect that increasing the scale has a decreasing effect on the amount of traffic on the unrealistic route, and a very large scale will result in less traffic on route 3 than the shortest route based assignment. From the previous chapter we know that the results of the C-logit are influenced by the scale of the utility but we do not know anything about the extent in which the results are influenced by the scale. The test with the parameter values of table 6.2 shows the extent in which the results are influenced by the utility scale.

Utility	Commonality factor	θ **value**	Power	β value	γ value	Penalty value	Bias values
all	all	**0...20**	1	1	1	0	[1 1 1]

Table 6.2: The used parameter values for the θ variants

Figure 6.2 shows the results for commonality factor 4. The results of the other commonality factor functions produce a slightly higher route share on route 3. The utility function variant travel time+ has not got results for θ values larger than 50, because e^{-x} for large x goes to zero, and division by zero is not possible. The travel time+ grows faster than the other utility functions because it uses an additional load to the network to calculate the travel times. In the figure we see that the utility scale has a major influence on the route choice results. Increasing the θ value from 1 to 5 cuts in half the route choices of the unrealistic routes. The C-logit scores better than the shortest path based assignment for θ values of 4 and larger. Figure 6.2 also shows that there is not one utility expression that scores the best on realistic routing for every utility scale.

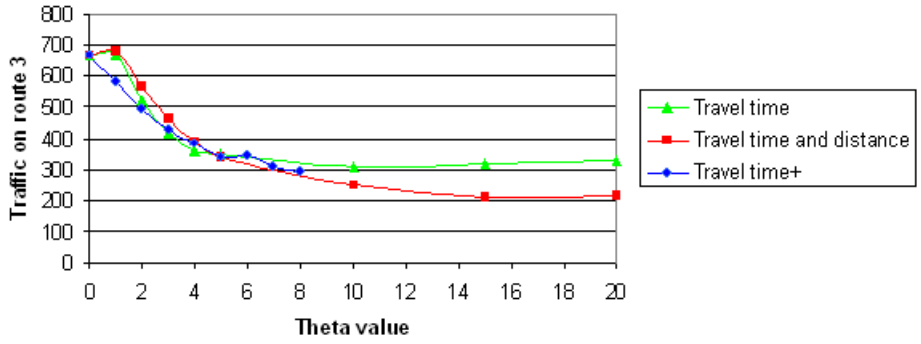


Figure 6.2: The results for the θ variants with commonality factor 4

Power variants

At this variant we do not keep the ι value at one but vary this value. We investigate this variant analytical by looking at the partial derivative of the route choice probability for route 3. We need to write the utility in terms of travel time because the power is a power of the travel time:

$$\frac{\partial P_3}{\partial \iota} = \frac{-\theta e^{-\theta(TT_3^\iota + CF_3)} \cdot TT_3^\iota \cdot ln(TT_3) \cdot (e^{-\theta(TT_1^\iota + CF_1)} + e^{-\theta(TT_2^\iota + CF_2)})}{(h(\iota))^2} + \frac{(\theta e^{-\theta(TT_1^\iota + CF_1)} \cdot TT_1^\iota \cdot ln(TT_1) + \theta e^{-\theta(TT_2^\iota + CF_2)} \cdot TT_2^\iota \cdot ln(TT_2)) \cdot e^{-\theta(TT_3^\iota + CF_3)}}{(h(\iota))^2}$$

Where $h(\iota) = \sum_{j=1}^{3} e^{-\theta(TT_j^\iota - CF_j)}$ and $(h(\iota))^2 \geq 0$. Further we know that the travel times of the three routes are always larger than one minute and thus $ln(TT_j) \geq 0$ for all j. Therefore we can conclude that the partial

derivative is positive if:

$$(\theta e^{-\theta(TT_1^\iota+CF_1)} \cdot TT_1^\iota \cdot ln(TT_1) + \theta e^{-\theta(TT_2^\iota+CF_2)} \cdot TT_2^\iota \cdot ln(TT_2)) \cdot e^{-\theta(TT_3^\iota+CF_3)} > \\ \theta e^{-\theta(TT_3^\iota+CF_3)} \cdot TT_3^\iota \cdot ln(TT_3) \cdot (e^{-\theta(TT_1^\iota+CF_1)} + e^{-\theta(TT_2^\iota+CF_2)})$$

The sign of the partial derivative is negative if the opposite hold. Clearly the sign of the partial derivative $\frac{\delta P(3)}{\delta \iota}$ depends highly on the travel time of the three routes. We do not have a priori knowledge of the development of the travel times during the assignment step. Therefore we do not make a prediction of the influence of ι on the total assignment procedure by the partial derivative. But we know that a higher power assigns more weights to differences in the travel times, (for travel times higher than 1) therefore we expect a decrease on the traffic on route 3 for higher powers. Table 6.3 shows the parameter values of the tested power variants.

Utility	Commonality factor	θ value	**Power**	β value	γ value	Penalty value	Bias values
all	all	1.33	**0...2**	1	1	0	[1 1 1]

Table 6.3: The used parameter values for the utility power variants

We see similar effects on the five commonality factor. A power of one leads to the highest route choice probabilities for the third route. The power of the utility has clearly less influence on the route choices than the θ value. The recommendation of the power of 0.7 from previous research cannot be concluded from this test. There is not one utility expression that scores the best for all utility powers.

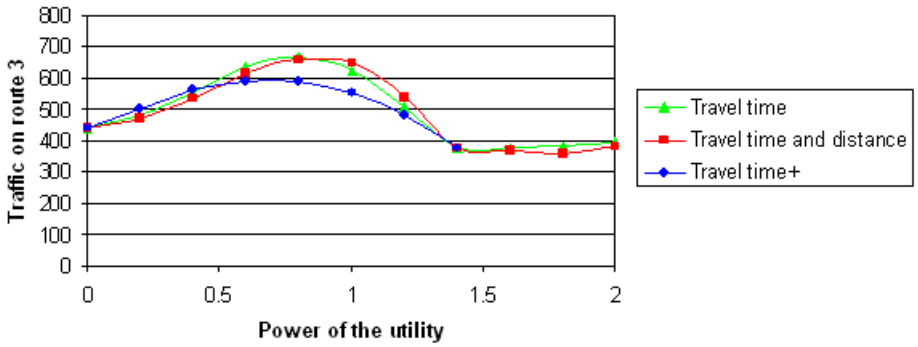


Figure 6.3: The results for power variants with commonality factor 4

β and γ value variants

The parameter β is incorporated in all commonality factor types and can be seen as the scaling factor of the commonality factor. From the partial derivative $\frac{\partial P(3)}{\partial CF}$ we know that the larger $CF_3 - CF_2$ and $CF_3 - CF_1$ the smaller the probability of choosing route 3. Because $CF_3 > CF_1, CF_2$ we expect that increasing β will decrease the probability of choosing the route 3.
The parameter γ is only used at the first commonality factor function and is the power over the fraction $\frac{L_{ij}}{\sqrt{L_i L_j}}$. This fraction is always smaller than 1 because $i \neq j$. Increasing the γ value will therefore decrease the commonality function value. Following the same reasoning as with the β value we know that increasing the commonality factor decreases the probability of choosing route 3, the unrealistic route. Therefore, increasing the γ value increases the probability of choosing route 3. The subtables of table 6.4 show the tested parameter values for the β variants, and the γ variants.

(a) The used parameter values for β variants

Utility	Commonality factor	θ value	Power	β **value**	γ value	Penalty value	Bias values
all	all	1.33	1	**0...20**	1	0	[1 1 1]

(b) The used parameter value for γ variants

Utility	Commonality factor	θ value	Power	β value	γ **value**	Penalty value	Bias values
all	1	1.33	1	1	**0...10**	0	[1 1 1]

Table 6.4: The parameter of the commonality factor variants settings

The results show that the traffic on the unrealistic route decreases as the β value increases. We see that the β

value has a strong influence on the route choices. The results are similar for the five commonality factor forms, but the fourth commonality factor function is influenced the most by the value of β. This is obvious from the results of the commonality factor variants. A higher β value leads also to less traffic on route 2, which is possible unrealistic. Commonality factor 4 suffers the less of this problem, due to the relative big difference of the commonality factor value for route 2 and 3 for this factor. It is not unambiguous which utility expression scores the best for different β values, see figure 6.4. This figure shows the results for β variants with commonality factor 4.

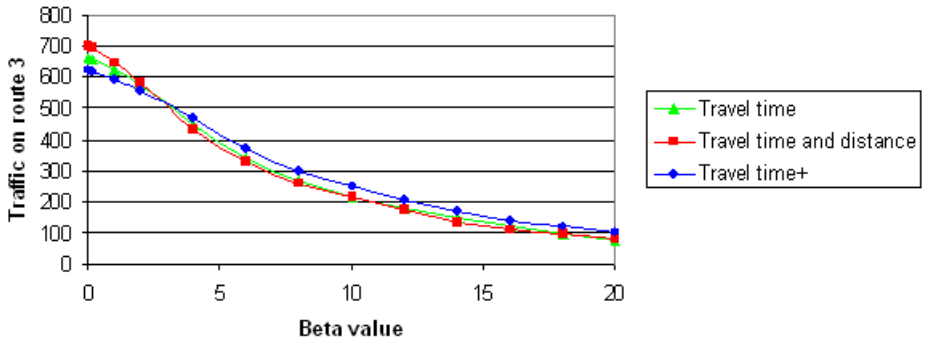


Figure 6.4: The results of the β variants with commonality factor 4

Commonality factor 1 is the only factor that uses parameter γ. The parameter γ has not much influence. Like we expected lead small γ values to the smallest number of traffic on route 3. Figure 6.5 presents the results for variations in γ with $\beta = 1$ and a θ value of 1.

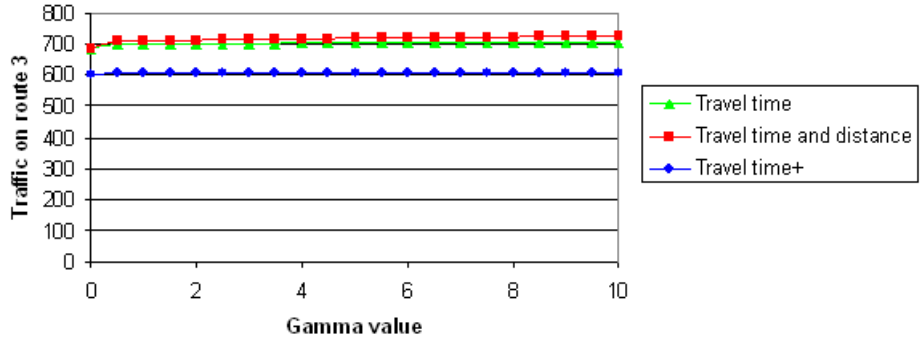


Figure 6.5: The results for γ variants with commonality factor 1

Penalty variants

We expect good results from the penalty function, because the penalty value is able to decrease only the utility of one selected route. We only penalize a route if it consists of a paired off- and on-ramp. By introducing a penalty function for route 3 we lower the utility of route 3. To analyze the influence of the penalty function we derive the partial derivative of the probability function of route 3 with respect to the utility function of route 3.

$$\frac{\partial P_3}{\partial V_3} = \frac{\theta e^{\theta(V_3 - CF_3)} \cdot (e^{\theta(V_1 - CF_1)} + e^{\theta(V_2 - CF_2)})}{(h(V_3))^2}$$

Where $h(V_3) = \sum_{i=1}^{3} e^{\theta(V_1 - CF_1)}$, and $(h(V_3))^2 \geq 0$. So the sign of the partial derivative is positive, which means that increasing the penalty and thus decreasing the utility will decrease the probability of route 3. We expect that the influence of the penalty value is huge, because there are small differences between the travel times of the three routes. The used penalty values and the other parameter settings are displayed in table 6.5.

Utility	Commonality factor	θ value	Power	β value	γ value	**Penalty value**	Bias values
all	all	1.33	1	1	1	**0...5**	[1 1 1]

Table 6.5: The used parameter values for the penalty variants

The results of this test show that the penalty value has a major influence, we can adapt the penalty value such

that a desired amount of traffic on route 3 is reached. For instance, to reach a smaller amount than 100 vehicles on the unrealistic third route for every utility function and commonality factor, we have to increase the penalty value to 2.5. The major part of the reduce of traffic on route 3 goes to route 2. Figure 6.6 presents the amount of traffic on route 3 for the penalty variants with commonality factor 4.

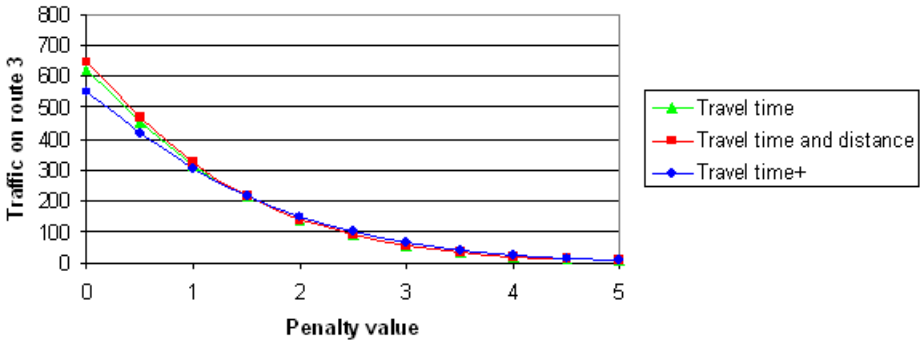


Figure 6.6: The results for penalty variants with commonality factor 4

Road category bias variants

The road category bias is multiplied by the link travel times. We saw in the previous section that the route share of route 3 decreases if the utility of route 3 decreases and the other utilities are stable. We must multiply the link times of route 3 with higher bias values than the other routes to decrease the utility of route 3. Route 3 has the most on and off ramps of the three roads. Therefore, increasing the ramp category bias will influence the route shares on route 3 the most. We expect that an increase in the ramp bias leads to less traffic on the unrealistic route 3. We expect that an increase in the ramp bias also leads to a decrease of traffic using route 2. This effect can be compensated by a low highway bias, but we expect that a low highway bias increases the amount of traffic on route 3 back to the initial value. Table 6.6 displays the used parameter settings. We do not test ramp bias values which are lower than the highway bias, because we do not want to favor ramp links above highway links.

Utility	Commonality factor	θ value	Power	β value	γ value	Penalty value	**Bias values**
all	all	1.33	1	1	1	0	**[0.85 0.9 0.85...4] & [0.6 0.9 0.6...4]**

Table 6.6: The used parameter values for the road category bias variants

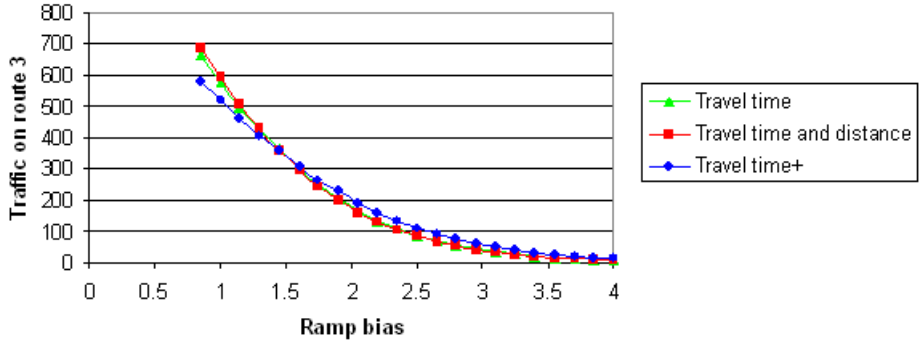


Figure 6.7: The results of the road category bias variants with bias [0.85 0.9 0.85...4]

Fortunately the road category bias can result in less traffic on route 3 and much traffic on the highway. Adapting the highway bias has more influence on the amount of traffic on route 2 than on the amount of traffic on route 3, for high ramp biases. For instance for a highway bias of 0.85 and 0.6, an urban road bias of 0.9 and a ramp bias on 4 we found the following result: The traffic on route 3 is low for both the highway biases: 29 and 27 vehicles, but there is a significant difference between the amount of traffic on route 2: 1183 and 1342 vehicles (commonality factor 4). Figure 6.7 displays the result for commonality factor 4 with a highway bias of 0.85, a rural road bias of 0.9 and different ramp bias values. In compare to the penalty function has the road

category bias more influence, because it influences the route choice probabilities of all routes. This is also a disadvantage of the road category bias, because the road category bias is hard to calibrate.

Combinations of θ and β variants

We expect that the decreasing effect on route 3 choices due to increasing β value exists for all θ values. We expect that the influence of the β value is higher in combination with high θ values, because the commonality factors are also multiplied by the θ value. So an increase in the θ value leads indirect to an increase in the scale of the commonality factors. Table 6.7 shows the used parameter values for the β and θ combination variants.

Utility	Commonality factor	θ **value**	Power	β **value**	γ value	Penalty value	Bias values
all	all	**0...20**	1	**1...5**	1	0	[1 1 1]

Table 6.7: The used parameter values for the combination of θ and β value variants

The results of this test show that the decreasing effect of the β value on the traffic on route 3 indeed exists for all different utility scales. The decreasing effect of high beta values is obvious the highest for large utility scales. Figure 6.8 shows the results for commonality factor 4 and utility expression travel time.

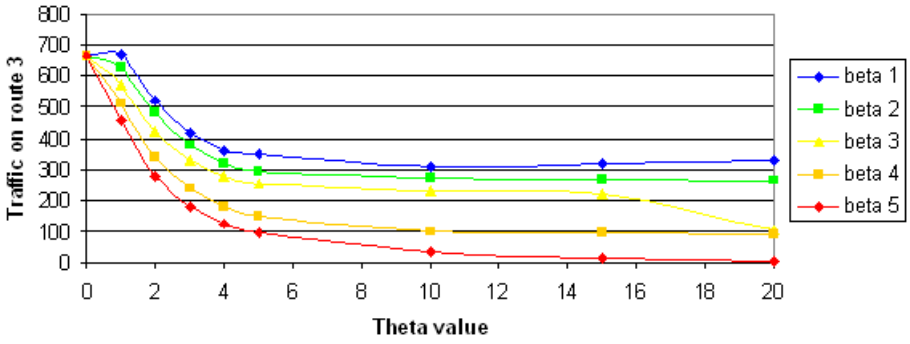


Figure 6.8: The results for a combination of θ and β variants with commonality factor 4

Combination of θ and penalty variants

The penalty variant has a strong decreasing effect on the unrealistic route choices. We expect that the penalty value has the most influences on the lowest scale value and thus lowest θ value, simply because a penalty value of 2 means more to a utility of scale 1 than to a utility of scale 100. The tested parameter values for the combination of scale and penalty are given in table 6.8.

Utility	Commonality factor	θ **value**	Power	β value	γ value	**Penalty value**	Bias values
all	all	**0...20**	1	1	1	**0...5**	[1 1 1]

Table 6.8: The used parameter values for the combination of θ and penalty variants

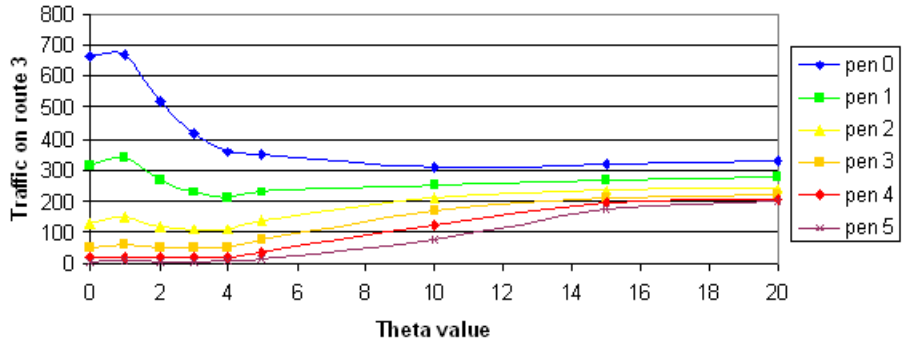


Figure 6.9: The results for a combination of θ and penalty variants with commonality factor 4

We observe the expected effect, see figure 6.9. This figure shows the results of the utility function travel time

for commonality factor 4. Clearly, the decreasing effect of the penalty value is active for all utility scales, and is the strongest for small utility scales. The increasing line for higher θ values shows that the influence of the penalty value is stronger than the influence of the θ value.

Combination of θ and road category bias variants

The road category multiplies a constant with the utilities. We expect that the road category bias has more impact on higher θ values, because the C-logit model is based on absolute differences. Table 6.9 shows the tested parameter values.

Utility	Commonality factor	θ **value**	Power	β value	γ value	Penalty value	**Bias values**
all	all	**0...20**	1	1	1	0	**[0.85 0.9 0.9...4]**

Table 6.9: The used parameter values for the combination of θ and bias variants

Figure 6.10 pictures the results for commonality factor 4 and the utility expression travel time. The highway bias is 0.85 and the rural road bias 0.9 in this test. We need to remember that a ramp bias below the 0.9 will favor route 3 with the ramps. Therefore, we only ramp bias values above 0.9. We see in the figure that the decreasing effect of the scale and the road category bias strengthen each other. The higher θ, the higher the influence of the road category bias.

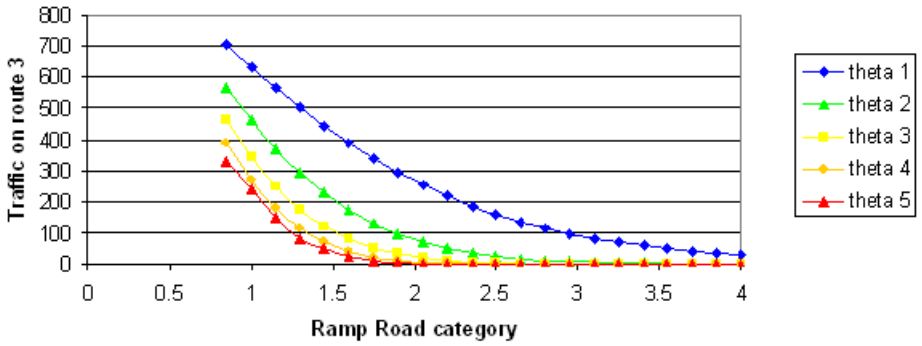


Figure 6.10: The results for a combination of θ and bias variants with commonality factor 4

Combination of β and penalty variants

As well the penalty variant as the β variant show a decreasing effect on the amount of traffic on route 3. Therefore we expect that a combination of these two variant will show a stronger decrease. The tested parameter values are displayed in table 6.10.

Utility	Commonality factor	θ value	Power	β **value**	γ value	**Penalty value**	Bias values
all	all	1.33	1	**1...5**	1	**0...5**	[1 1 1]

Table 6.10: The used parameter values for the combination of β and penalty variants

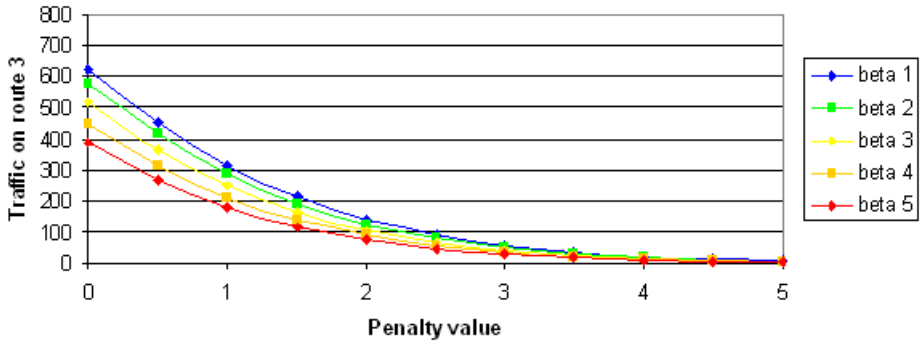


Figure 6.11: The results for a combination of β and penalty variants with commonality factor 4

In fact the results show that the combination of β and penalty variants have a strong decreasing effect. Figure 6.11 shows the results for commonality factor 4 and utility expression travel time.

Combination of β and road category variants

We expect to see the decreasing effect of both the road category and the β value, like at the combination with β value and the penalty variant. Table 6.11 describes the tested parameter values for this combination.

Utility	Commonality factor	θ value	Power	**β value**	γ value	Penalty value	**Bias values**
all	all	1.33	1	**1...5**	1	0	**[0.85 0.9 0.9...4]**

Table 6.11: The used parameter values for the combination of β and road category variants

Figure 6.12 shows the results for commonality factor 4 and utility expression travel time, the highway bias is set on 0.85 and the rural bias on 0.9. The expected effect is clearly pictured in this figure.

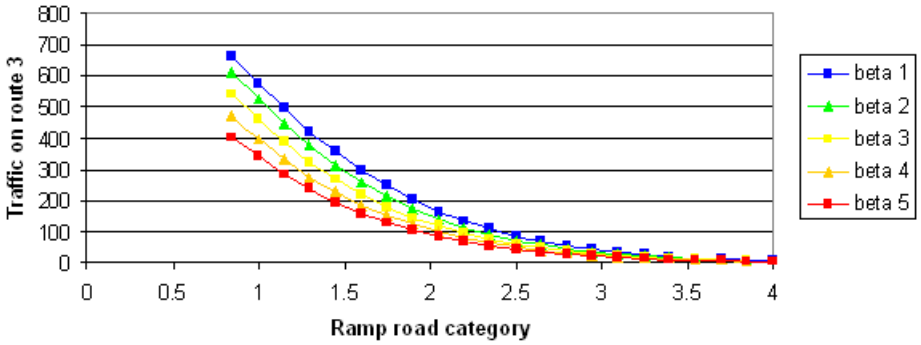


Figure 6.12: The results for a combination of β and bias variants with commonality factor 4

6.1.2 Game theory

This section describes the expected and realized results of the game theory variants in the static assignment. Chapter 5 describes the different game theory variants. Again, tables describe the used parameter settings of the variants. The categories of these tables are: game, p value, β value, θ value, damaging fraction, penalty value, and the bias values. The category game describes which type of game is tested. Three game types are possible: the congestion game, the smooth fictitious play, and the demon player game. In the congestion game, we use a combination of travel time and distance as objective, expressed by: $p*distance+(1-p)*TravelTime$. The category p value describes the value of p for this formula, the p value is only used for congestion games. The smooth fictitious play uses the Multinomial logit or the C-logit as mapping function. The columns β and θ values describes the values of these parameters in the Multinomial logit and C-logit rule. We use commonality factor 4 in the C-logit mapping function and do not describe the commonality factor function in the table. The demon player in the demon player game damages the capacity of one link. The fraction of the capacity that is damaged is described by the damaging fraction. We use the damaging fraction only at the demon player game. The categories penalty value and bias value are just the same as in the previous section.
All three games are seeking for an equilibrium situation in every loading step. It is therefore hard to give analytical expectations about the route choices.

Congestion game, p value variants

The equilibrium of the congestion game with a p value of 0, is similar to the user equilibrium. In every iteration of the incremental assignment a new user equilibrium with a tenth of the demand is loaded. The load of the congestion game will approach the user equilibrium. The third route is not the shortest and fastest route after the shortest route based assignment. Therefore, we expect that the congestion game will result in a lower amount of traffic on the third route than the shortest route based assignment. We expect that an increase of the p value will lead to less traffic on the third route, because the third route is the longest route. We do not expect that small differences in the p value will influence the route choices much, because the route lengths of the three routes do not differ much. The used parameter settings for the congestion game with p-value variants are displayed in table 6.12.

Game	p **value**	β value	θ value	damaging fraction	Penalty value	Bias values
Congestion	**0...1**	-	-	-	0	[1 1 1]

Table 6.12: The used parameter values for congestion game variants

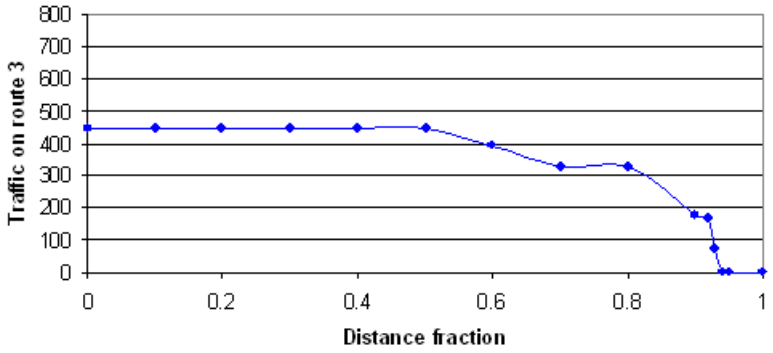


Figure 6.13: The results of the congestion game variants with different p values.

Figure 6.13 shows the decreasing effect of larger p values, like we expected. There seems to be a turning point in the p value, first a variation in the p value has no to little influence but from a p value of 0.5 the p value seems to have much influence. We have to comment that it is generally assumed that people prefer fast routes more than short routes, therefore we do not expect that a p value above 0.5 leads to more realistic results in general. For this specific case the congestion game with a p value above 0.6 leads to more realistic results than the shortest route principle.

Smooth fictitious play, θ value variants

We expect that the θ value has the same influence on the smooth fictitious play as on the regular C-logit model. At the regular C-logit model, the scale had a strong decreasing effect on the amount of traffic on the third route. Table 6.13 shows the used parameter settings for this test. We use both a β value of zero and one. C-logit with a β value of zero is similar to the Multinomial logit.

Game	p value	β value	θ **value**	damaging fraction	Penalty value	Bias values
Smooth fictitious	-	0 & 1	**0...20**	-	0	[1 1 1]

Table 6.13: The used parameter values for the smooth fictitious play with scale variants

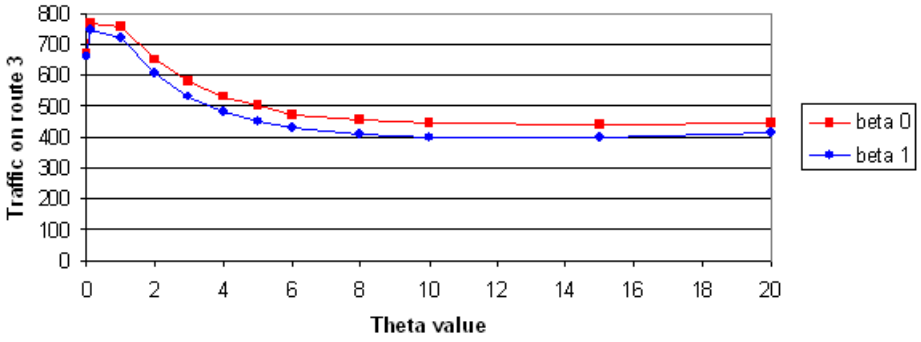


Figure 6.14: The results of the smooth fictitious play with scale variants.

Figure 6.14 displays the results of the smooth fictitious play θ variants. The results of the smooth fictitious play for β values of zero and one do not differ much, but like we expected a decreasing effect and similar effect as with the normal C-logit model is shown. The graph also shows that the θ value has a stronger effect on the C-logit than on the smooth fictitious play. A larger scale increases the absolute differences of the alternatives, and ensures that the Multinomial logit becomes more like the shortest route based principle. The Multinomial logit and C-logit based fictitious play reaches the shortest route principle for large utility scales.

Smooth fictitious play, β value variants

We expect the same effect of the β variants at the smooth fictitious play as with the regular C-logit models, namely that the amount of traffic on the unrealistic route will decrease for larger β values. We expect that the influence of the β value at the smooth fictitious play is as strong as at the C-logit. The settings of the smooth fictitious play with β variants are given in table 6.14.

Game	p value	β **value**	θ value	damaging fraction	Penalty value	Bias values
Smooth fictitious	-	**0...20**	1.33	-	0	[1 1 1]

Table 6.14: The used parameter values for the smooth fictitious play with β value variants

Figure 6.15 shows the results of the smooth fictitious play with different β values. We use a θ value of 1.33 and commonality factor 4. We see that the β value has a slightly stronger effect on the C-logit than on the smooth fictitious play, but the differences are small. The direction of the influence of the β value is obvious the same, the amount of traffic on the unrealistic route decreases for larger β values.

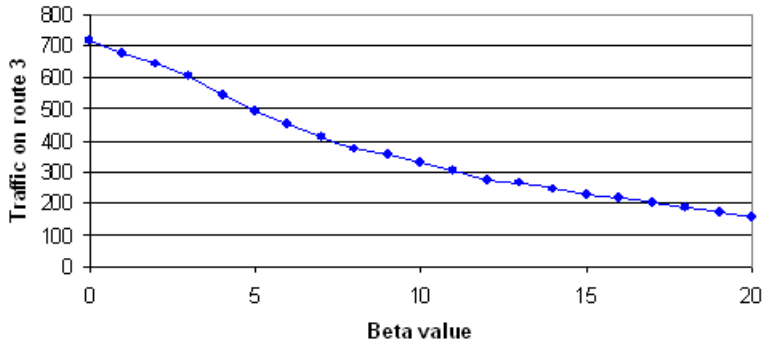


Figure 6.15: The results of the smooth fictitious play with different β values.

Demon player, damaging fractions

The demon player game assigns less traffic to routes which are vulnerable for failure. We expect less traffic on route 3 with the demon player game than with the shortest path based incremental assignment, because the on- and off-ramps of route 3 are vulnerable for vailure with their low capacities. We expect that an increase in the damaging fraction will decrease the amount of traffic on route 3. The tested parameter values are given in table 6.15. We do not use a damaging fraction of one, because a route gets blocked if the damaging fraction is one.

Game	p value	β value	θ value	**damaging fraction**	Penalty value	Bias values
Demon player	-	-	-	**0...0.95**	0	[1 1 1]

Table 6.15: The used parameter values for the demon player game with damaging fraction variants

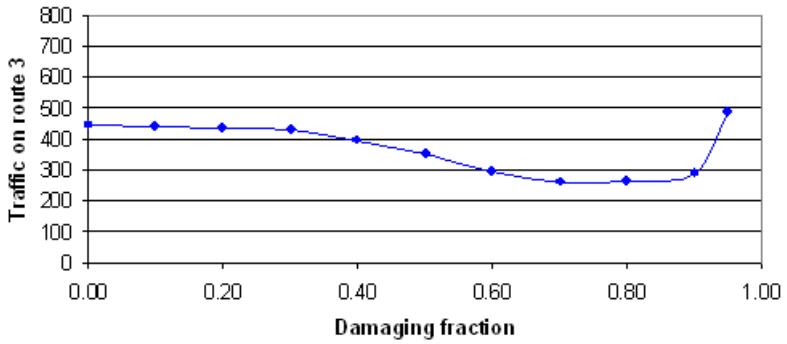


Figure 6.16: The results of the demon player game with damaging fraction variants.

Figure 6.16 displays the results of the demon player game with different damaging fraction. A damaging fraction of 0.2 on link i means that, 20% of the capacity of i link is damaged. The figure shows clearly that less than 400 vehicles on route 3 is loaded when the demon damages more than 40% of the link capacity. A higher influence of the demon, by damaging more of the capacity, lowers the amount of traffic on the unrealistic route. For damaging fractions above 0.8, the route choices of route 3 increases when the damaging fraction increases any further. The damaging fractions are at that moment so high, that damaging the main roads and therefore directing all the vehicles on the more vulnerable roads which get congestion is the most effective for the demons. While the demons select the most vulnerable roads at small damaging fractions. The goal of the game is to incorporate risk averse routing, therefore it is not appropriate to use such high damaging fractions.

Game theory, penalty function variants

A penalty value can easily be incorporated in the three game theory models. We use travel time as objective in the congestion game, a damaging factor of 0.2 in the demon player game, and the Multinomial logit with $\theta = 1$ in the smooth fictitious game, see table 6.16. We expect that the penalty value has a similar lowering effect on the amount of traffic on route 3 for all the three games.

Game	p value	β value	θ value	damaging fraction	**Penalty value**	Bias values
Congestion	0	-	-	-	**0...5**	[1 1 1]
Smooth fictitious	-	0	1.33	-	**0...5**	[1 1 1]
Demon player	-	-	-	0.5	**0...5**	[1 1 1]

Table 6.16: The used parameter values for the three games with a penalty variants

Figure 6.17 shows a similar effect on the congestion game and the demon player game of the penalty value. The difference is that the demon player assigns a bit lower amount of traffic on the unrealistic third route. The smooth fictitious game assigns a lot more traffic to the unrealistic route and shows a more gradually effect of the penalty functions. This more gradually decrease is caused by the manner of assigning vehicles to a route. In the smooth fictitious play every single vehicle is considered seperately. The perceptions of the route performance differs for different vehicles. In the congestion game and the demon player game are all vehicles with the same origin and destination considered as one group. The total group has the same optimal strategy. The vehicles are scattered on the different routes, because the optimal strategy is an mixed strategy. The approach of the congestion game and the demon player game, where the vehicles are considered as one group causes the stepwise reaction on the penalty function. The penalty function has a similar but a little bit stronger effect on the C-logit model as on the smooth fictitious play.

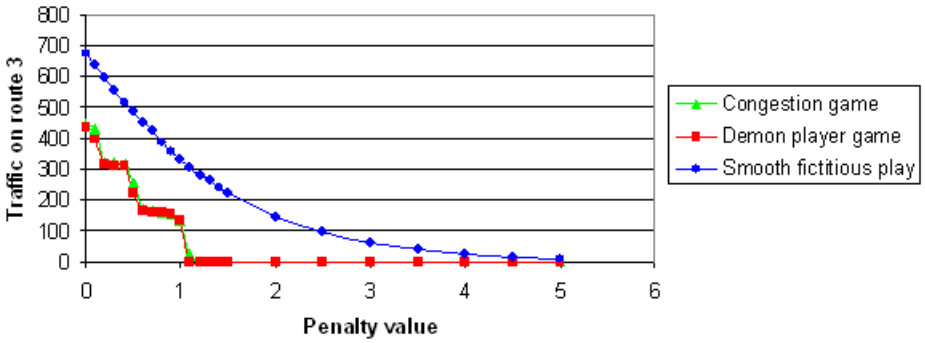


Figure 6.17: The results of game theory variants with the penalty functions

Game theory, road category bias variants

Beside the test on the influence of the penalty function we also test the influence of the road category bias on the congestion game, the demon player game, and the smooth fictitious play. Figure 6.18 displays the results of game theory models with a road category bias. Again we use travel time as objective for the congestion game, a damaging fraction of 0.2 in the demon player game, and the Multinomial logit with a θ value of 1.33 in the smooth fictitious play. This figure shows the results for a highway bias of 0.85 and a rural road bias of 0.9. We vary the bias of the on- and off-ramps between 1 and 5, see also table 6.17. We expect a decreasing effect of the road category bias on the amount of traffic on the unrealistic route.
We see that the road category bias has a similar effect on the game variants as the penalty value. Again the

Game	p value	β value	θ value	damaging fraction	Penalty value	**Bias values**
Congestion	0	-	-	-	0	**[0.85 0.9 0.85...4]**
Smooth fictitious	-	0	1.33	-	0	**[0.85 0.9 0.85...4]**
Demon player	-	-	-	0.5	0	**[0.85 0.9 0.85...4]**

Table 6.17: The used parameter values for the three games with road category bias

congestion game and the demon player game encounter the same effects and the smooth fictitious play a more gradually effect. This is explained by the same reasoning as at the penalty function variants. Also the road category bias has a stronger effect on the C-logit than on the smooth fictitious play.

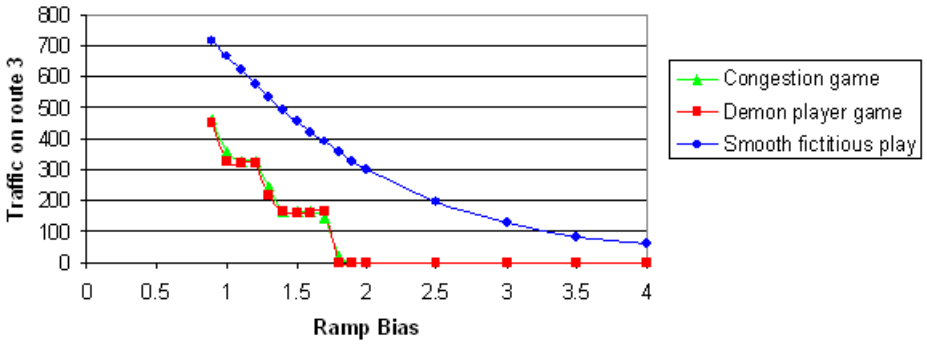


Figure 6.18: The results of game theory variants with the road category bias

6.1.3 Conclusion static results

The theoretical tests in a static assignment show promising results for both the C-logit and the game theory variants. The C-logit variants show that they are able to score better on reality than the shortest route principle. The choice of the commonalty factor function has not much influence on the route choices in this theoretical network. The θ value, the θ value has a strong influence on the route choices. The route choice probability of route 3 decreases if the θ value increases. The β value has also a strong decreasing influence on the route choice probability of route 3. The utility expression travel time+ shows promising results, it scores the best on realistic routing for the most but not all parameter settings. Further research towards the possibilities of this utility expression in a dynamic equilibrium assignment is strongly recommended. We will use the utility expression travel time in the remaining of this research project because we did not investigate the possibilities of the travel time+ usage in a dynamic equilibrium assignment. Another variant we do not test in the dynamic assignment although it shows promising results is the road category bias. This variant suffers too much of computational difficulties in realistic networks and difficult calibrating abilities. In contrary to the road category bias, the penalty function does not suffer from computational difficulties. The penalty function variant shows a strong decreasing effect for increasing penalty values.
The power of the utility and the γ value for commonality factor function 1 has not much influence on the results. The shown effect of the power of the utility is caused by the change in utility scale and not the power of the utility. The tests with combinations of two variants show that the effect of the single variants are strengthen by each other.

All three games are able to produce less traffic on the unrealistic route 3 than the shortest route based assignment. The congestion game scores better on realistic routing with p-values higher than 0.6 and with penalty function. It is not realistic to assume that people base their route choice more on distance than on travel time, therefore we do not test congestion game variants with high p-values in the remaining of this research project. The smooth fictitious play is only able to score better than the shortest route principle with extreme parameter settings. Therefore we do not test the smooth fictitious play in a dynamical assignment of this theoretical test. The parameters of the C-logit have a stronger effect on the C-logit than on the smooth fictitious play. The demon player game is able to produce less traffic on the third route with realistic parameter settings. The penalty function and the road bias category have a strong decreasing influence on the amount of traffic on the third route. But like at the normal C-logit model the road category bias is complex for realistic networks. Therefore we only test the penalty value at the games in the dynamic assignment.

6.2 Dynamic results

The shortest route principle in a dynamic assignment results in 611 vehicles on route 1, 830 vehicles on route 2, and 603 vehicles on the unrealistic third route. We use the average of the last five iterations as the result (iteration 36 until iteration 40). In the dynamic assignment a total of 2044 traffic with free route choice leaves in the first hour. In the static assignment, 2000 vehicles have a free route choice. Because of the difference in number and network we compare the static with the dynamical results by relative route choices compared to the shortest route principle. A result of 0%, means that the variant results in exactly the same amount of traffic on the third route as the shortest route based principle and a result of -100% means that the dynamic assignment variant result in zero traffic on the third route. So a negative percentage means that our variants has a higher reality score than the shortest route principle.

6.2.1 C-logit

We test the variants with different β values, θ value, and penalty values. Commonality factor 4 is used, because this commonality factor gives the most promising results for the static assignment. We use the utility expression travel time. The C-logit variants converge within 40 iterations. The highest squared coefficient of variation between the amount of traffic on route 3 for the last 5 iterations is derived for a penalty value of 4. The amount of vehicles are rounded up to whole numbers, and the amount of traffic on route 3 is the lowest for the variant with a penalty value of 4. It is therefore not surprisingly that the variant with penalty value 4 has the highest squared coefficient of variation. The squared coefficient of variation for this variant is 0.01.
We expect that the influence of the variants is stronger on the static assignments than on the dynamic assignments, first because we use an equilibrium assignment instead of an incremental assignment in the dynamic assignment, and second because the travel times are more accurate in the dynamic assignment: If a variant defines route 3 unattractive, less traffic is assigned on route 3, which causes lower travel times and makes the route more attractive. Therefore, the influence of the parameters on the variant 's results will be smaller in an equilibrium assignment. The travel times are averaged per minute in the dynamic assignment, while in the static assignment the travel times are averaged over the total planning horizon. Therefore the dynamic assignment has more accurate travel times. This effect on the travel times slows down the influence of the variants.

β value variants

The used parameter settings for the comparison between the β variants in a static and dynamic assignment are shown in table 6.18. We expect that the β value has a same but smaller decreasing effect on the amount of traffic on route 3 for the dynamic assignment than in the static assignment.

Utility	commonality factor	β **value**	θ value	Penalty value	Bias values
Travel time	factor 4	**0,1,2 & 5**	1.33	0	[1 1 1]

Table 6.18: The used parameter values for the comparison of dynamic and static assignment of the β variants

Figure 6.19 shows the results of the comparison between the static and the dynamic assignment for different β values. The figure shows the expected effect. The results of the dynamic assignment do not differ much for β values between 0 and 2.

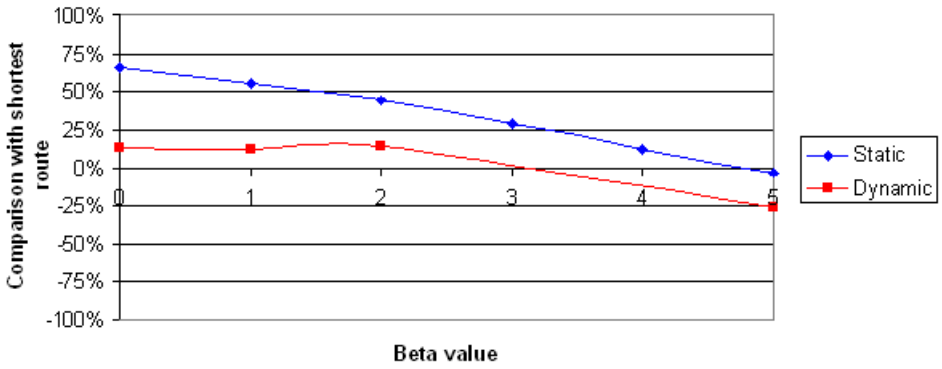


Figure 6.19: The comparison of the β variants in a dynamic and static assignment.

θ value variants

The amount of traffic on route 3 decreases if the θ value increases in a static assignment, first because route 3 has a low utility, and second because the C-logit is based on absolute differences. In the dynamic equilibrium assignment, the C-logit with a low scale will show an equal deviation among the three routes, and the results for the C-logit with a large scale will tend to a equilibrium assignment based on the shortest route. Therefore we also expect that the dynamic assignment will show a decreasing effect. Table 6.19 describes the used parameter settings.

Utility	commonality factor	β value	**θ value**	Penalty value	Bias values
Travel time	factor 4	1	**0.13...13.33**	0	[1 1 1]

Table 6.19: The used parameter values for the tests of the θ variants in the dynamic assignment

Figure 6.20 pictures the results for the dynamical assignment and the static assignment of the C-logit model with different θ values. The C-logit seems not to response on θ changes in our dynamic assignment. But in fact the θ variant result in different route choice per 2 minute interval, but the total route shares on route 3 are coincidental more or less the same for different scales.

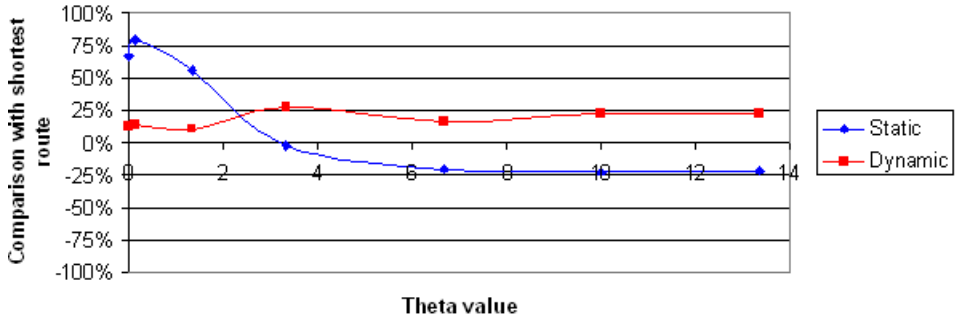


Figure 6.20: The comparison of the θ C-logit variants in a dynamic and static assignment.

Penalty variants

We expect that the dynamic assignment reacts the same on the penalty variants as the static assignment, but with a smaller decreasing effect. Table 6.20 presents the used parameters to test this presumption.

Utility	commonality factor	β value	θ value	**Penalty value**	Bias values
Travel time	factor 4	1	1.33	**0...3**	[1 1 1]

Table 6.20: The used parameter values for the penalty C-logit variants in a dynamic assignment

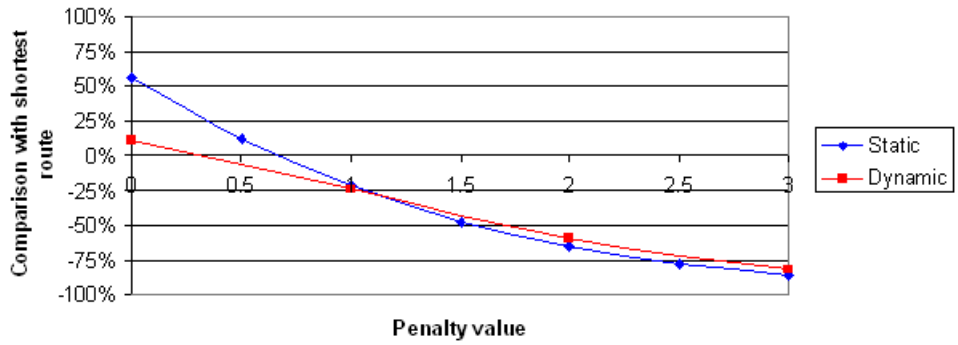


Figure 6.21: The comparison of the penalty C-Logit variants in a dynamic and static assignment.

Figure 6.21 displays the comparison for different penalty values in the C-logit between the static and the dynamic assignment. The static assignment reacts stronger on the penalty value, like we expected. But there is not much difference between the influence of the penalty value on the static and the dynamic assignment.

6.2.2 Game theory

The most promising game theory variants of the theoretical test in a static assignment are: the congestion game with different p-values, the demon player game with different damaging fractions and the demon player game and the congestion game with penalty function. We do not test the demon player game in a dynamic assignment. Appendix B explains why we did not test this variant in a dynamic assignment.

Congestion game, p value variants

The p-values between 0 and 0.5 influence the route choices of the static assignment a little, because most of the time the shortest route is also the fastest route in the network of our static assignment. The network and the background traffic differ in the dynamic assignment from the static assignment, the changed background traffic causes higher travel times for route 1 in the dynamic assignment. The lengths are not changed in the network, route 1 is still the shortest route. Therefore we expect that an increase in the p-value will cause an increase in the traffic on route 1 which causes decrease in traffic on route 3. Table 6.21 shows the used parameters settings for the test in a dynamical assignment.

Game	p **value**	damaging fraction	Penalty value
Congestion	**0, 0.25 & 0.5**	-	0

Table 6.21: The used parameter values for congestion game variants in a dynamic assignment

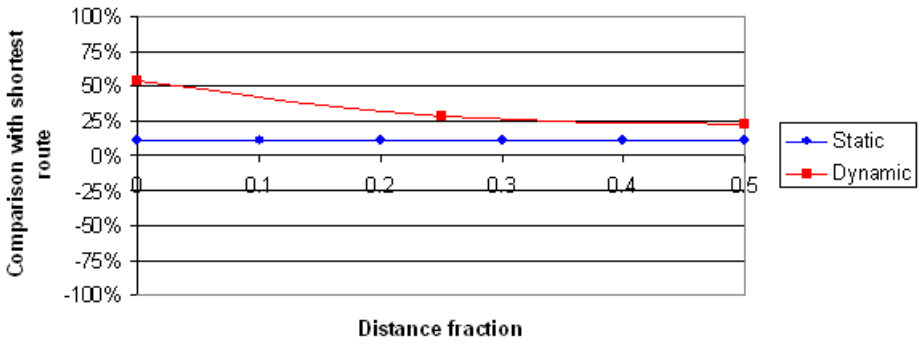


Figure 6.22: The comparison of the congestion game variants in a dynamic and static assignment.

Figure 6.22 shows the results for route 3 of this test. The expected effect is observed in this test, the amount of traffic on route 1 increases as the p-value increases, and therefore decreases the amount of traffic on route 3. The decreasing effect of the p-value on the traffic on route 3 is smaller for higher p-values.

Congestion game, penalty value variants

The penalty value is an effective measure in the static assignment of the congestion game. We expect that the penalty value is also an effective measure in the dynamic assignment of the congestion game. The penalty values and other used parameter values of this test are given in table 6.22. We do not test a penalty value larger than 1.25, because in the static assignment a penalty of 1.25 already causes a no traffic on the third route. Like we expected the penalty value is an effective measure in the dynamic assignment. The step wise

Game	p value	damaging fraction	**Penalty value**
Congestion	0	-	**0...1.25**

Table 6.22: The used parameter values for the congestion game with a penalty variants

pattern of the congestion game with penalty value in a dynamic assignment looks like the step wise pattern of the static assignment. Figure 6.23 displays the comparison between the static and dynamic assignment for the congestion game with penalty functions. The penalty function seems to have a stronger effect on the congestion game in the dynamic assignment than in the static assignment.

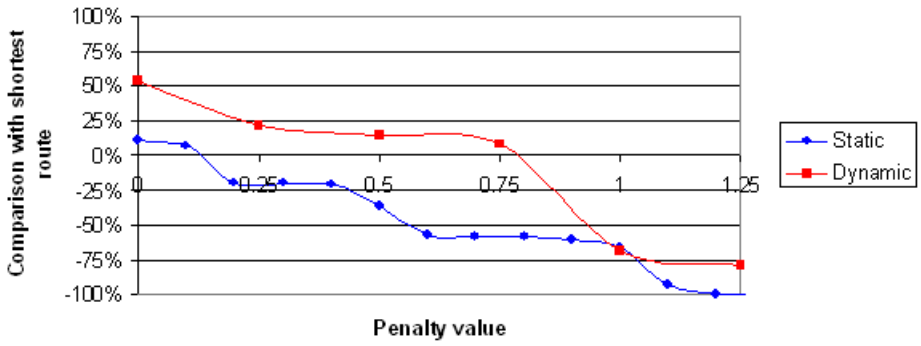


Figure 6.23: The comparison of the congestion game variants with the penalty function.

6.2.3 Conclusions dynamic results

The tests in the dynamic assignment show promising results, both the C-logit variants and the game theory variants are able to score better than the shortest route principle on realistic routing. The parameters of the C-logit influence the route choices more in a static assignment than in a dynamic assignment, we expect that this is realized by the different loading methods. In the static assignment we use a incremental loading and in the dynamic assignment an equilibrium loading. We see an opposite result for the congestion game variants, the parameters of the game theory models influence the route choices of the dynamic assignment more than the static assignment.
The β value, the θ value and the penalty value influence the route choices of the C-logit variants the most. The p-value of the congestion game has more influence on the dynamic assignment results than on the static assignment results, but still the influence is very small. The penalty value has a strong influence on the congestion game route choices.

Chapter 7

Results Enschede tests on 1 OD

This chapter describes the results of the two Enschede tests on 1 origin destination pair. The theoretical tests show promising results from both the C-logit and the game theory variants. We want to know if these variants also produce realistic route choices in a real network, therefore we compare the results with real route choice data. To compare the results with the route choice data, we need to know the travel times in the network. Dynasmart calculates the travel times in the network for us, but there are some errors in the travel time functions of Dynasmart. Therefore we investigate to which extent the route choice results of our variants are affected by the travel time function errors. We compare the route choices of our variants based on experienced travel time with route choices based on travel times from Dynasmart. At the same time we investigate the influence of the parameters in this model environment.
Both the C-logit as the game theory variants make use of a choice set. In our theoretical test, 3 routes are available. But in our network of the Enschede tests more than 20 routes are possible between register points TM7 and CK6. We define our choice set by the 10 fastest routes in free flow conditions. Figure 7 shows the ten shortest routes. These 10 routes do not contain all routes registered at the license plate survey. At the license plate survey 6 different routes are registered between TM7 and CK6. One of them is not included in the choice set. This causes that our route choice model will never assign all the traffic at the same manner as was registered.

Figure 7.1: The 10 shortest free flow routes between TM7 and CK6

We score the variants by the reality score defined in Chapter 4. The reality score represents the percentage of

vehicles that is loaded on the same route as registered in the data. Because of the restrictions of our choices set is 97.6% the highest possible score on reality for our variants.

7.1 Comparison

In this chapter we make three comparisons. One comparison between the experienced and the Dynasmart travel times, one between the shortest route principle route choice and the route choice of our variants, and one between the route choices of our variants based on experienced travel times and the route choices of our variants based on Dynasmart travel times.

7.1.1 Travel time comparison

Before we test the influence of the differences in travel times on the route choice we investigate the differences between the experienced travel times and the travel times from Dynasmart. We compare the travel times in a run of Dynasmart with the registered travel times. In this Dynasmart run we assign all the traffic according to their registered route (and the shortest route between the register points). We compare experienced travel times of the vehicles in Dynasmart with the data, because the data only consists of experienced travel times. We select ten intensive used pair of register points, the Dynasmart travel times between these ten pairs of Dynasmart are compared with the travel times from the data. The ten pairs of register points are: SK3-CK4, CTK11-CK7, CK1-CTK11, CTK8-CK1, CK4-CTK9, CK2-CK7, CK5-CTK10, CTK9-CK4, CTK12-CTK11, and CK1-CTK8. The first quarter of the first hour is used as a warm up period for Dynasmart and is not included in the comparison. The travel times of Dynasmart are on average 15% shorter than the real experienced travel times. From the experiences at DHV we know that 15% is just a small deviation from the reality and that our model is pretty good. The next section discusses the influence of this small deviation on the route choices.

7.1.2 Shortest route comparison

To investigate whether the C-logit and game theory variant result in more realistic route choices than the shortest route principle, we calculate the route choices of the shortest route principle. The shortest route principle assigns 29% of all the vehicles between TM7 and CK6 on the same route as registerd in the data when using the experienced travel times. Even small difference between the experienced travel times and Dynasmart travel times could be disastrous for the shortest route principle, because all the traffic of on departure interval is loaded on one route. A small error in the travel time could causes the wrong route to be the fastest, if the travel times are similar for different routes. This is what happens at this network when we use Dynasmart travel times. In Dynasmart route 4 is the fastest in all time intervals, but route 4 is only chosen twice of the 85 movements in the data set. So the shortest route principle scores with Dynamsmart just 2.4% on reality, instead of the 29% we found in the test with experienced travel times. We compare these scores with the scores of our variants.

7.1.3 Experienced/Dynasmart comparison

We compare the reality scores of the variants based on experienced travel times and the variants based on Dynasmart travel times on four elements: the height of the reality score, the difference between the lowest and the maximum score, the development of the reality score in time, and the order of the reality score. The difference between the lowest and maximum score at the end of the four hours indicates the influence of the parameters on the reality score. What we mean with the order of the reality score is explained by an example. Suppose a β of 1 scores the highest for the β variants based on experienced travel times, and $\beta = 2$ and $\beta = 3$ have the second and third best score. Now suppose that $\beta = 1$ also scores the highest, $\beta = 2$ the second highest, and $\beta = 3$ the third highest for the β variants based on Dynasmart travel times. Then the order of the reality scores is the same at the experienced and Dynasmart travel times.

7.2 C-logit results

We test C-logit variants with different β values, θ values, commonality factor functions, and different penalty values. Tables describe the used parameter settings. The tables contain the column β value, θ value, commonality factor, power, penalized paths, and penalty value. The column penalized path, describes the path that is penalized. The column penalty value describes the used penalty value for these paths/ this path. We show the result of every variant in a graph. The graph shows the reality score of the variant.

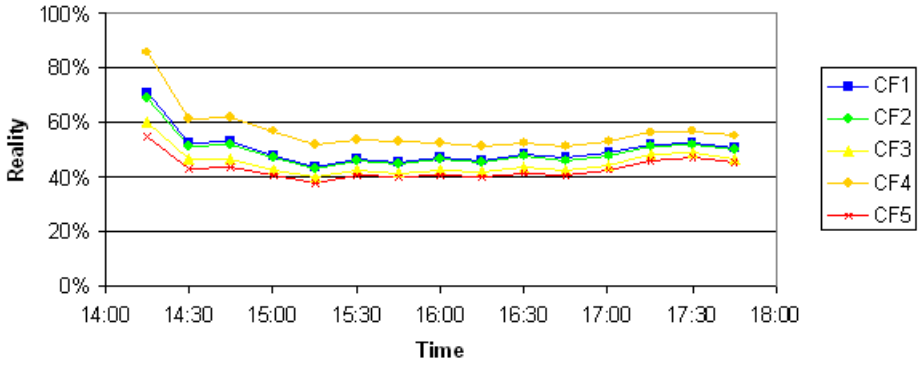
7.2.1 Commonality factor variants

In the theoretical test the choice of the commonality factor influences the route choices a little, because the commonality factor values of the different functions were similar. In this network the commonality factor values differ more, therefore we expect a larger influence of the commonality factor function choice on the route choices. The used parameter settings of this test are described in table 7.1.

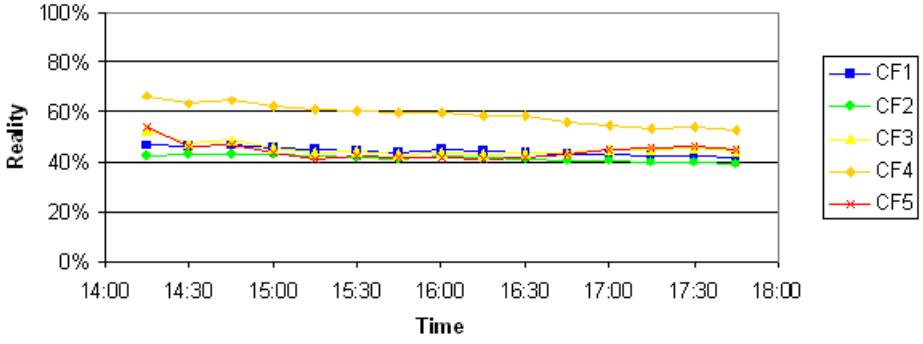
β value	θ value	**Commonality Factor**	Power	Penalized paths	Penalty value
5 & 8	1	**1, 2, 3, 4, and 5**	1	-	-

Table 7.1: The used parameter values for the commonality factor variants

The results of this test for C-logit variants with a θ value of one, and a β of eight are given by figure 7.2. This figure shows similar results for the experienced travel times and the travel times from Dynasmart. The variants score between the 40% and 60% with both travel times, and in both cases the commonality factor 4 scores the best. Also the difference between the score of commonality factor four after four hours and the lowest score after four hours are similar for the case with experienced travel times and the case with Dynasmart travel times. A difference between the two results is the developement of the reality score over time. The reality score increases in the last hour for the experienced travel times and the reality score decreases in the last hour for the Dynasmart travel times. This could indicates that after a four hour run the travel times becomes inaccurate, but the influence on the route choices is still small. We see similar behavior when we use a β value of five instead of eight. All the commonality factor functions have more realistic route choices than the shortest route principle for our selected origin destination pair.



(a) Experienced travel times



(b) Dynasmart travel times

Figure 7.2: The results of the commonality factor variants

7.2.2 θ value variants

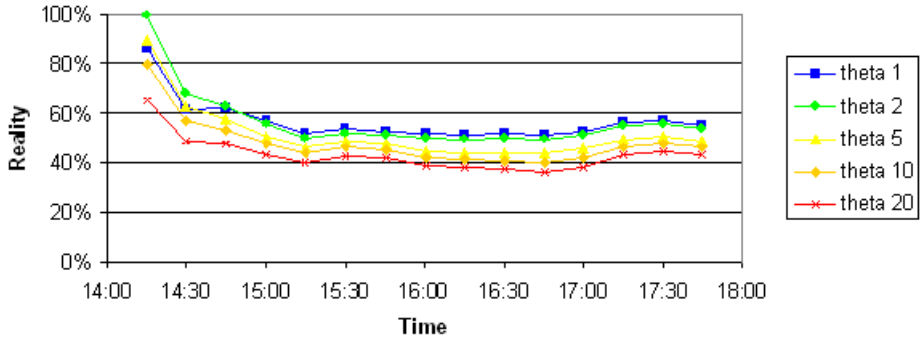
In the theoretical test we saw that a higher θ value in the C-logit takes care for a higher distinction between the different routes. If the travelers choose more like the shortest route principle, a high θ value is appropriate. We expect that route choices in traffic is not always comparable with the shortest route principle, simply because the travelers do not have the knowledge of all the current traffic properties. The ten routes do not have the same order from fastest to longest route in the case of experienced travel time as in the case of Dynasmart

travel times, therefore we expect different results for the experienced travel times and the Dynasmart travel times. The used parameter are described in table 7.2.

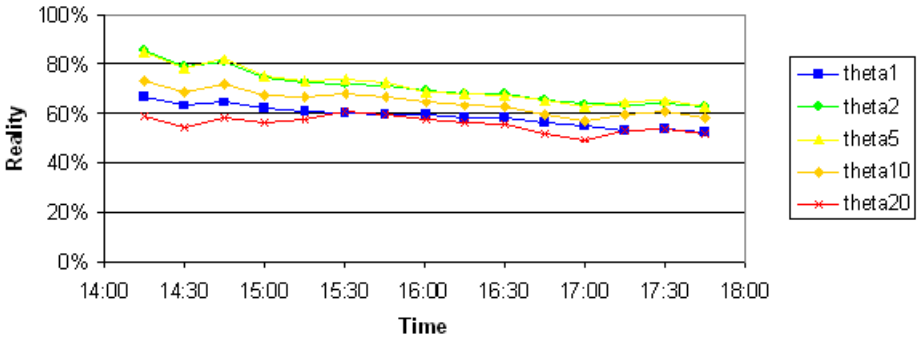
β value	θ **value**	Commonality Factor	Power	Penalized paths	Penalty value
5 & 8	**1...20**	4	1	-	-

Table 7.2: The used parameter values for the θ variants

Figure 7.3 shows the results for the θ values from 1 to 10, with a β value of eight and commonality factor function four. In these graphs, the decreasing reality score for the Dynasmart travel times in the last hour is again displayed, while the reality score increases the last hour for the experienced travel times. The graphs also show that in both cases the influence of the θ value is not very high, because of the small difference between the route choices for different θ values. Therefore it is not strange that the order of the scores for the different θ values differ for the experienced travel times and Dynasmart travel times. This causes that the variants score for some θ values higher when using Dynasmart travel times and some θ values score higher when using experienced travel time. The relatively high scores for low θ values indicate that the shortest route principle does not suit the data very well. The scale variants score better on realistic routing than the shortest route principle with both the travel times.



(a) Experienced travel times



(b) Dynasmart travel times

Figure 7.3: The results of the θ variants

7.2.3 Travel time power

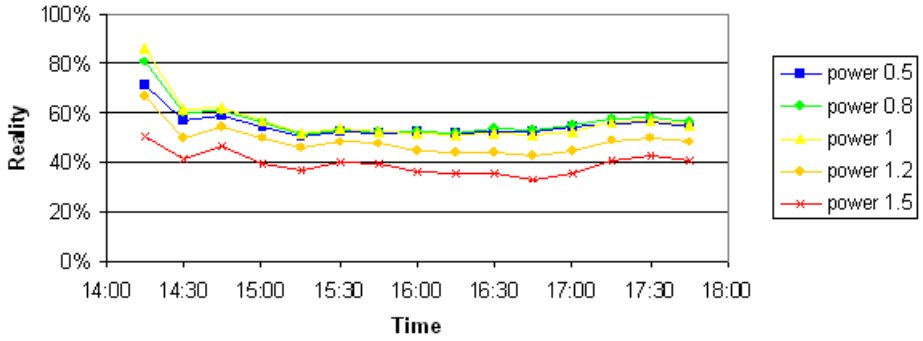
The power of the travel time did not have much influence on the results of the theoretical test. We will not investigate this variant on 10 origin destination pairs, if we can confirm this lack of influence in these two Enschede tests on 1 origin destination pair. We test the power variants with the parameter values of table 7.3.

β value	θ value	Commonality Factor	**Power**	Penalized paths	Penalty value
8	1	4	**0.5...1.5**	-	-

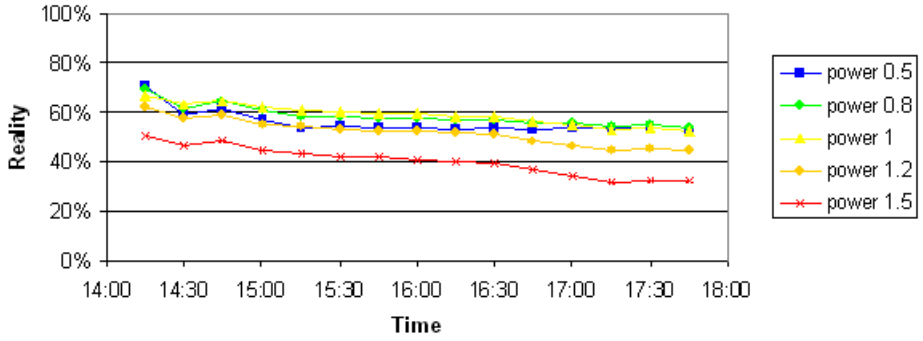
Table 7.3: The used parameter values for the power variants

Figure 7.4 shows the results of this test. We clearly see that the differences in the travel times have very little influence on the route choice results. It can clearly be seen that no matter which travel times are used, there are

almost no differences between the results for a power between the values 0.5 and 1. The height of the reality score, the order of the reality score and the differences between the highest and lowest reality score is only influenced a little by the change in use of Dynasmart travel times instead of experienced travel times. Again we see that the development of the reality over time lowers a bit for the Dynasmart travel times. A power larger than 1 seems to give less realistic results. This observation together with the lack of influence on the results on the theoretical tests makes us to decide to do not investigate the power variants in the test on ten origin destination pairs.



(a) Experienced travel times



(b) Dynasmart travel times

Figure 7.4: The results of the power variants

7.2.4 β value variants

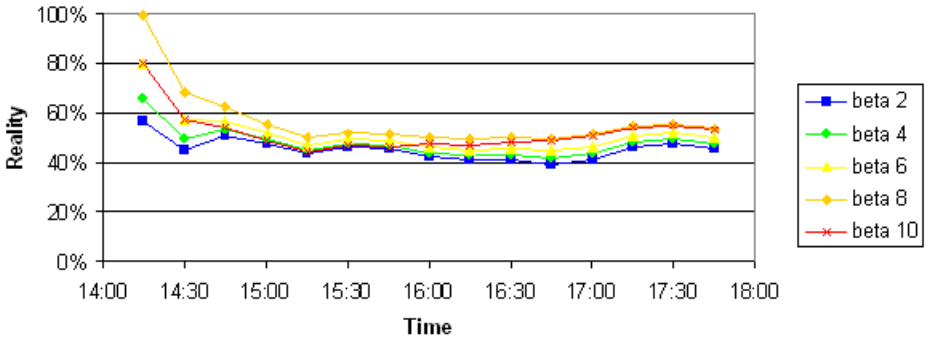
The β value is the scaling factor of the commonality factor. High overlapping paths will receive a lower route share in the C-logit route choice models with high β values. In reality travelers do not consider two highly overlapping paths as two different path, therefore less people will choose these routes than would be calculated by a Multinomial logit model. Therefore we expect that a higher β value leads to more realistic results. But we also do not want to increase the β too much, because the travel time is still much more important than path overlap in route choice decisions. We do not have to consider this limitation of the β value in the theoretical test. In order to lower the traffic on the unrealistic third route in the theoretical test we increase the β value. The tested parameter values are described by table 7.4.

β value	θ value	Commonality Factor	Power	Penalized paths	Penalty value
0...10	1,2,5	4	1	-	-

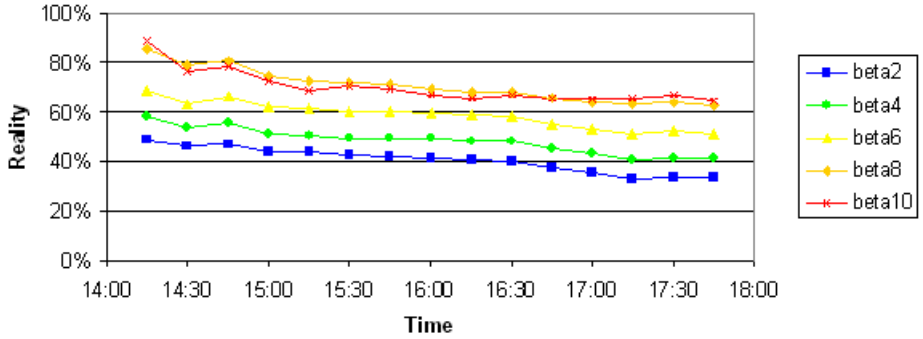
Table 7.4: The used parameter values for the β variants

Figure 7.5 displays the results for a θ value of 2, commonality factor 4, and different β values. We observe in these graph the expected behavior, increasing the β leads to more realistic results, but increasing the β value too much does not increase the reality score. The graphs also show that the β value has less influence on the results when we using the experienced travel times and more when we are using the Dynasmart travel times. But the order in the reality scores of the β variants is the same for both the experienced and the Dynasmart travel times. The height of the reality score is not the same with the experienced and the Dynasmart travel times for all β values, because of the much larger influence of the β value in the case with Dynasmart travel

times. Again the development of the reality score over time shows a decreasing curve for the Dynasmart travel times.



(a) Experienced travel times



(b) Dynasmart travel times

Figure 7.5: The results of the β variants

7.2.5 Penalty variants

It is not interesting to investigate the differences between a penalty variant based on experienced travel times and a penalty variant based on Dynasmart travel times, because the penalty function can be seen as a correction term for systematic errors in the travel time function and the experienced travel times suffers from different errors than the Dynasmart travel time. We are interested in the influence of the penalty function on the route choice results. In the theoretical test we saw that the penalty function is an effective measure to minimize traffic on a certain route. In this network we do not want to minimize traffic on a certain route but make corrections for possible errors in the travel time function. Therefore we test the penalty function on the Dynasmart travel times. In Dynasmart route 4 is the fastest route, while this route is only taken twicein the data. We also observe for a C-logit variant with $\theta = 1$, commonality factor 4 and $\beta = 8$, that the travel time of route 8 is systematically overestimated. Therefore, we penalize route 4 and favorize route 8. A route is favorized by subtracting the penalty value from its utility function. Table 7.5 describes the used parameter settings.

β value	θ value	Commonality Factor	Power	**Penalized paths**	**Favorized paths**	**Penalty value**
5	2	4	1	**4**	**8**	**0...2**

Table 7.5: The used parameter values for the penalty variants

Figure 7.6 shows the results of this test with a penalty value of 1 and 2. We see that a penalty value of 1 already has a positive influence on the results. We see that increasing the penalty value further to 2, has not got a positive influence on the route choice of every interval. The penalty function over-compensates the utilities in these time intervals. Therefore we conclude that the penalty function need only be used when there is a systematically over or under estimation of the utility. We see that the influence of the penalty function is not much at the route choices of the variants, but it does lower the traffic on route 4 and increase traffic on route 8.

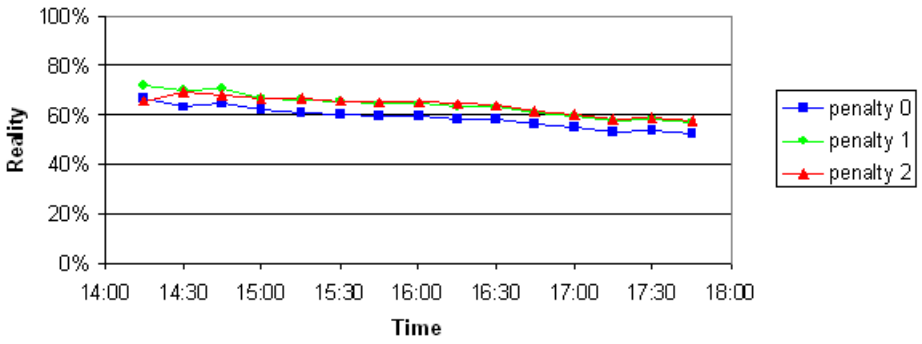


Figure 7.6: The results of the penalty variants with Dynasmart travel times

7.2.6 Conclusion C-logit

The tests show that the differences in the travel times have influences on the route choices of the variants, but that the influence is small. The most variants have the same order in reality score for using experienced travel times and for using Dynasmart travel times. Also the most variant have similar reality scores for using experienced travel times and Dynasmart travel, wherefore the difference between the lowest and the maximum reality score after four hours simulation is almost the same. The difference between the lowest and maximum reality score after four hours deviates for the β variants with experienced travel time from the β variants with Dynasmart travel times. But still all the β variants score higher than the shortest route principle. The decreasing trend in the reality scores over time for the cases with Dynasmart travel times indicates that the travel time function of Dynasmart suffer from errors. The shortest route principle is far more affected by these errors in the travel times. This is a huge advantage of the C-logit in compare to the shortest route principle. Further, all tested C-logit variants have higher reality scores than the shortest route principle.

7.3 Game theory

Game theory variants work towards an equilibrium situation in the travel times. It is not interesting to test the game theory variants on one origin destination pair, because the travel times are not much influenced by the traffic on only one origin destination. For instance, if we assign all traffic between TM7-CK6 on the same route, then the travel time is maximal increased by 4 seconds. Therefore we do not test the game theory variants with real travel times (the outcomes of the congestion game and the smooth fictitious play would be the same as the shortest route principle).

Chapter 8

Results Enschede tests on 10 OD

The Enschede tests on 1 origin destination show that the differences in the experienced travel times and travel times of Dynasmart do not lead to high deviations in the route choices. Therefore a further investigation about the influence of the route choice parameters is justified. We are interested if the results of the previous tests are specific results of the selected origin destination pair, or that these results hold for more origin destination pairs. Besides, we are interested if the results still holds if we use the route choice variants on multiple origin destination pair with conflicting routes. Therefore we test on 10 different origin destination pairs and compare the route choice results of this test with the results of the previous tests on two elements: the difference between the maximum and minimum reality score, and the order of the reality score among the variant. These two criteria are also used in the previous test.
In general we expect that the influences of the parameters are smaller on 10 origin destination pair together than on 1 origin destination pair, because we think that different parameter sets are optimal for different origin destination pairs. We think that the differences between the origin destination pairs will level each other. This causes that the difference between the maximum and minimum score is smaller than in the previous tests.
We determine a choice set for every origin destination pair. The choice set consists of the five fastest routes in free flow conditions. In total 526 trips were registered between our 10 origin destination pairs, and 85 trips did not use a route from our choice set. Therefore the maximum reality score that our variants can obtain is 83.8%. The shortest route principle does not use a choice set, and has a reality score of 32.5% in this test.

8.1 C-logit results

We test all C-logit variants of the previous test except for the power variant, because the previous tests has shown the lack of influence of the power on the route choices.

8.1.1 Commonality factor variants

The previous tests show that commonality factor 4 result in the highest reality score of all commonality factor functions. We are interested if commonality factor 4 has coincidentally the highest score due to characteristics of the origin destination pair TM7-CK6, or that commonality factor 4 has also the highest score for other origin destination pairs. To test this we used the parameter settings of table 8.1.

β value	θ value	**Commonality Factor**	Penalized paths	Penalty value
5	1	**1, 2, 3, 4, and 5**	-	-

Table 8.1: The used parameter values for the commonality factor variants

The scores of the different commonality factors do not differentiate more than 10% at both the Enschede tests on 1 origin destinaiton pair. Figure 8.1 displays the results for the commonality variants of the test on 10 origin destinaiton pairs. Clearly the difference between the results of the different commonality factors is more than 10%, and clearly commonality factor 4 has the highest reality score. This indicates that commonality factor function 4 is optimal for the most origin destination pairs within our selected 10 origin destination pair. The order of the scores of the other commonality functions has changed, but that is not realy interesteing because we only have to use one commonality function. The influence of the commonality function is very large. Changing the commonality factor function from function 4 to function 1 reduces the reality score by more than a half.

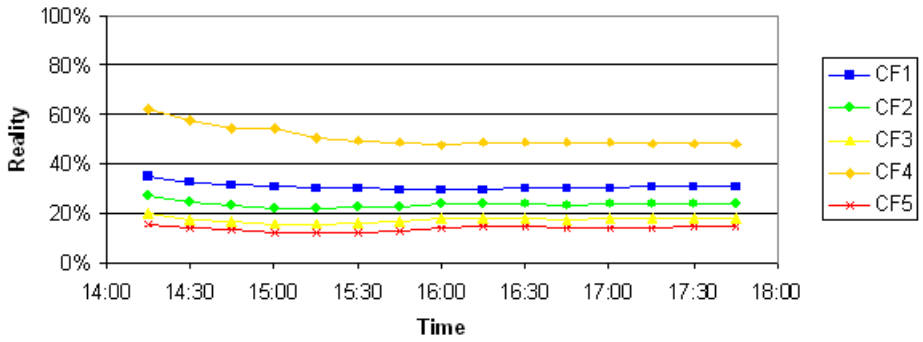


Figure 8.1: The results of the commonality factor variants

8.1.2 θ value variants

The previous tests do not give us a clear picture of the influence of the θ value. But the tests show that the θ value has influence on the C-logit results, and that θ values larger than 5 seems to produce less realistic route choices. To investigate the influence of the θ value on 10 origin destination pairs we used the parameter settings of table 8.2.

β value	**θ value**	Commonality Factor	Penalized paths	Penalty value
5	**1, 3, 5**	4	-	-

Table 8.2: The used parameter values for the θ variants

Figure 8.2 displays the results for the scale variants. The difference between the results of the different scale variants is comparable with the differences in the previous tests. The influence of the scale variants with θ values between 1 and 5 is small. It is not outstanding that the order of the reality score for the θ variants differ between the different tests, because of the small differences between the θ variant scores. All our tested θ variants have a higher reality score than the shortest route principle.

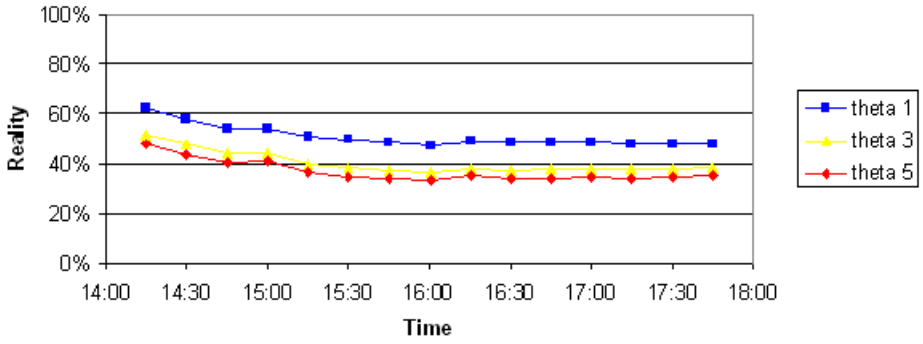


Figure 8.2: The results of the θ variants

8.1.3 β value variants

The previous tests show a high influence of the β values when we use Dynasmart travel times. We expect that this influence is less on 10 origin destination pairs than on 1 origin destination pair, because different β values will be optimal for different origin destination pairs and the differences will level each other out. Table 8.3 describes the used parameter values.

β value	θ value	Commonality Factor	Penalized paths	Penalty value
0,2,5 & 8	1	4	-	-

Table 8.3: The used parameter values for the β variants

Figure 8.3 displays the results for the β variants. Clearly the influence of the β value is much smaller on multiple origin destination pairs together, than on the origin destination pair TM7-CK6 alone. The scores of the β variants have a comparable order at this test as at the test on 1 OD, only the optimal β value is a bit

smaller (5 instead of 8 or 10). All our tested scale variants have a higher reality score than the shortest route principle.

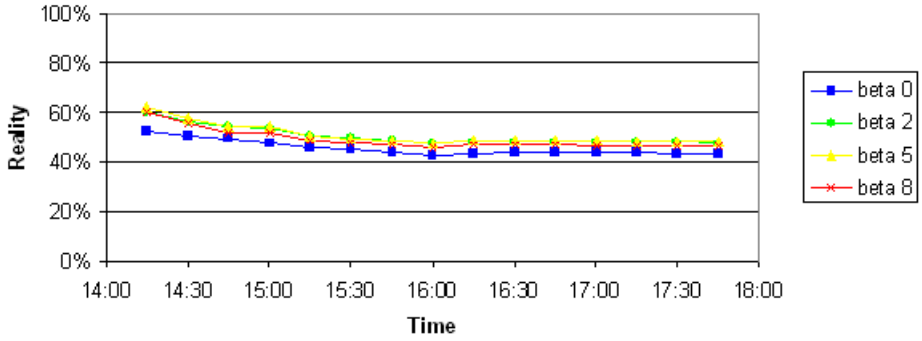


Figure 8.3: The results of the β variants

8.1.4 Penalty variants

The theoretical test shows promising results for the penalty function, while the Enschede tests on 1 OD show that the penalty function has little influence. We are interested if the influence of the penalty function is larger when we used the penalty function on multiple origin destination pairs. From the Enschede test we know that we need to penalize and favorize only routes that are systematically over or under estimated. The utility of the routes 4, 7, 12, 36, and 49 are systematically over estimated, we penalize these routes. The utilities of the route 6 and 37 are systematically under estimated, we favorize these routes. Table 8.4 describes the used parameter values.

β value	θ value	Commonality Factor	**Penalized routes**	**Favorized routes**	**Penalty value**
5	1	4	**4,7,12,36 & 49**	**6 & 37**	**0,1 & 2**

Table 8.4: The used parameter values for the penalty variants

The results of the penalty variants are given in figure 8.4. Clearly the influence of the penalty variant is small. We conclude that the penalty function is an effective measure to minimize traffic on a specific route, but is less effective to correct errors in the route choice model, or travel time function.

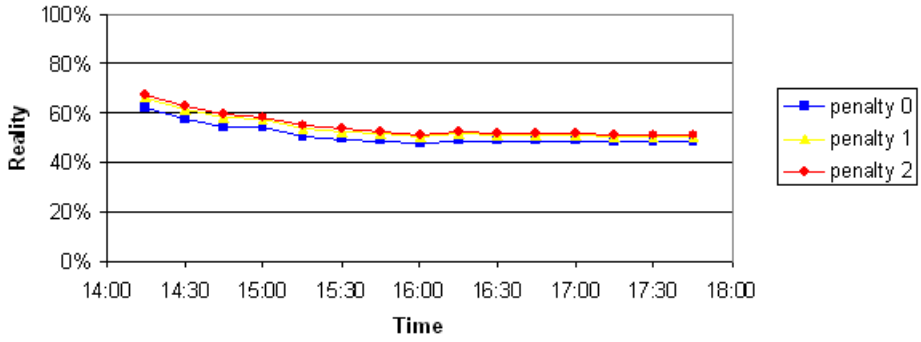


Figure 8.4: The results of the penalty variants

8.2 Game theory results

We test the congestion game and the smooth fictitious play. The smooth fictitious play has the lowest reality scores in the theoretical test. We still test this game in the Enschede test on 10 OD's, because we want to know if the bad results are a result of properties of the theoretical network or a result of the incapabilities of the smooth fictitious play. We do not test the commonality factor variants of the smooth fictitious play, because we have seen that the C-logit and smooth fictitious play lead to similar results and all commonality factor variants show the same influences of the commonality factor. Therefore, we only use commonality factor four. We also do not test the demon player game, because this test cannot be used in a dynamic assignment.

8.2.1 Congestion game, p-value variants

The congestion game results of the theoretical test show that varying the p-values between 0 and 0.5 has not much influence on the route choice results. In the Enschede network the lengths of the routes of the choice sets differ more than the route lengths in the theoretical network. Therefore we expect that the p-value has more influence on the route choices. Table 8.5 gives the tested parameter values.

Game	p **value**	θ value	β value	Penalty value	Penalized routes
Congestion	**0, 0.25 & 0.5**	-	-	0	-

Table 8.5: The used parameter values for congestion game variants

The results of this test are pictured in figure 8.5. It is remarkable that a p-value of 0.5 leads to more realistic results, than a p-value of 0.25, while it is generally agreed that people base their route choices more on travel time than on distances. We do not think that further research is necessary towards a high p-value, because the differences in the route choice are very small. The congestion game variants seem not to score very well on realistic routing, but the congestion game variants have still a higher reality score than the shortest route principle.

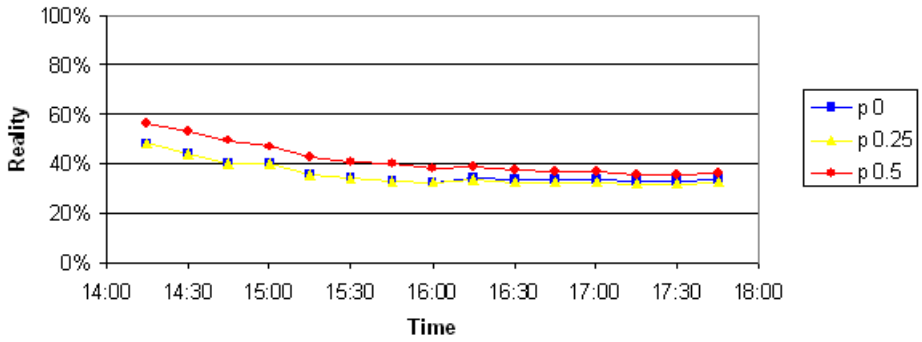


Figure 8.5: The results of the congestion game p-value variants

8.2.2 Congestion game, penalty value variants

In the theoretical test the congestion game is highly influenced by the penalty function. From the results of the C-logit variants we know that the penalty function is an effective measure to minimize traffic on a route, but a less effective measure to correct possible errors of the route choice model and the travel time function. We test with the parameter settings of table 8.6 if this statement also holds for the congestion game.

Game	p value	θ value	β value	Penalty value	Penalized routes	Favorized routes
Congestion	0.5	-	-	**0, 0.5 & 1**	**1, 7, 12, 26, 36 & 49**	**2, 11, 27, 37 & 47**

Table 8.6: The used parameter values for congestion game with penalty variants

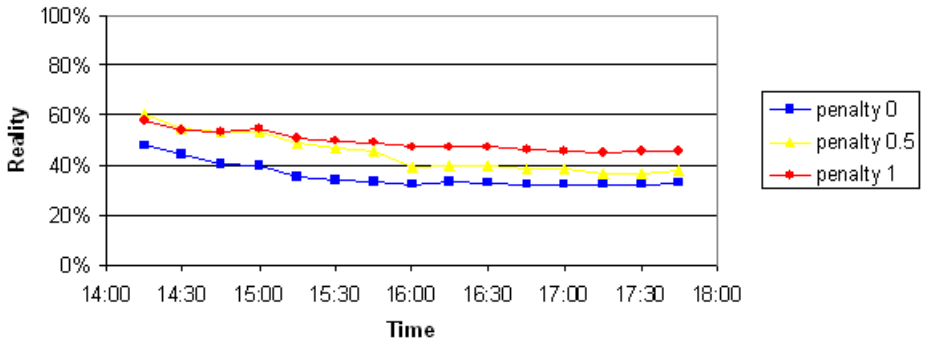


Figure 8.6: The results of the congestion game with penalty variants

Figure 8.6 shows the results of the penalty variants of a congestion game. The influence of the penalty function

is positive, and higher than the influence on the C-logit results. But if we use a higher penalty value than one we do not see an improvement in the reality score in compare to a penalty value of one.

8.2.3 Smooth fictitious play, θ value variants

In the theoretical test the smooth fictitious play show a similar but softer reaction on parameter setting as the C-logit variants. We investigate if this is also the case with the Enschede network. We test the smooth fictitious play scale variants with the parameter settings given in table 8.7.

Game	p value	θ **value**	β value	Penalty value	Penalized routes
Smooth fictitious play	-	**1, 3 & 5**	5	0	-

Table 8.7: The used parameter values for smooth fictitious play θ variants

Figure 8.7 pictures the results of the smooth fictitious play θ variants. The θ value has a similar but smaller influence on the smooth fictitious play as on the C-logit variants, like with the theoretical test. The order of the reality scores of the smooth fictitious play θ variants are similar for the theoretical test and this test. Different from the theoretical test is that the smooth fictitious play variants have higher reality scores than the C-logit variants. Obvious, all tested smooth fictitious play scale variants score higher than the shortest route principle on realistic routing.

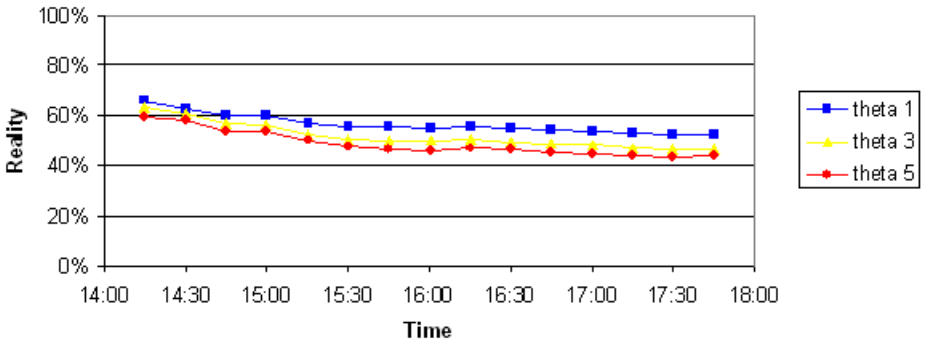


Figure 8.7: The results of the smooth fictitious play θ variants

8.2.4 Smooth fictitious play, β value variants

The β value had little to no influence on the C-logit variants, therefore we expect that the β value has also little influence on the smooth fictitious play. The parameter settings used in the test with the β variants are described in table 8.8.

Game	p value	θ value	β **value**	Penalty value	Penalized routes
Smooth fictitious play	-	1	**0, 5 & 8**	0	-

Table 8.8: The used parameter values for smooth fictitious play with β variants

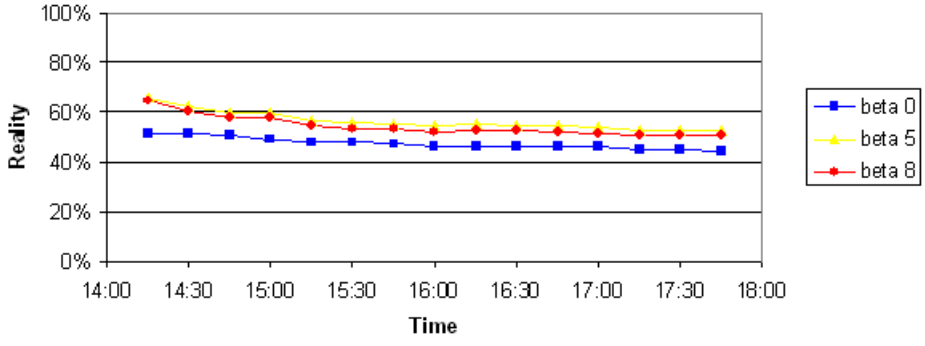


Figure 8.8: The results of the smooth fictitious play with β variants

The results of the test are given in figure 8.8 The β value has a bit stronger influence on the smooth fictitious play than on the C-logit, but again the influence on the results is not large. The order of the reality score of the variants is the same for the C-logit as for the smooth fictitious play variants. All tested smooth fictitious play scale variants score higher than the shortest route principle on realistic routing.

8.2.5 Smooth fictitious play, penalty value variants

The same as for the congestion game with penalty variants holds for the smooth fictitious play with penalty variants. We are interested if in a smooth fictitious play the penalty function is less effective as measure to correct possible errors of the smooth fictitious play and the travel time function, than as a measure to minimize traffic on one specific route. We penalize and favorize routes that are systematically under and over estimated. Table 8.9 describes the used parameter settings.
Figure 8.9 displays the results of the test with penalty variants. Like at the C-logit and the congestion game,

Game	p value	θ value	β value	**Penalty value**	**Penalized routes**	**Favorized routes**
Smooth fictitious play	-	1	5	**0, 1 & 2**	**7, 12, 36, 45 & 49**	**6, 37 & 47**

Table 8.9: The used parameter values for smooth fictitious play with penalty variants

the improvement caused by the penalty function is small. However, the smooth fictitious play with a penalty value of 2 has the highest reality score of all tested variants. Note that the end score of 61.3% is really a high score if we take into account that the highest score possible with our route choice set is 83.8%.

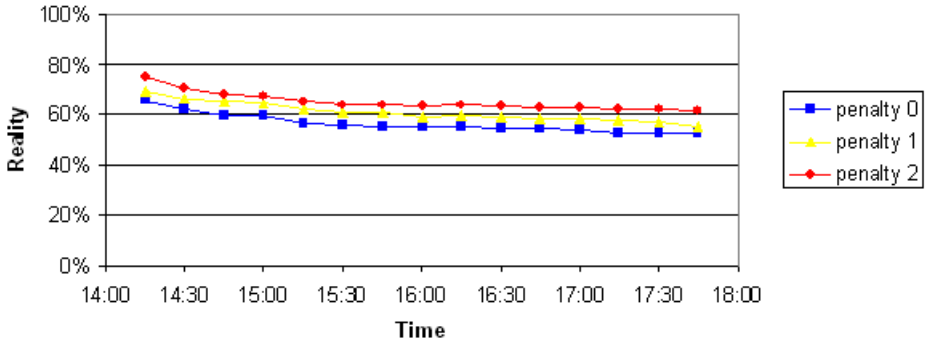


Figure 8.9: The results of the smooth fictitious play with penalty variants

8.3 Conclusion

All the C-logit variants tested on ten origin destination pairs have a higher reality score than the shortest route principle. The results also show that varying the parameter settings (within some bounds) do not have much influence on the reality score of the route choices between the ten origin destination pairs, except for the commonality factor function. Commonality factor four scores a lot better than the other commonality factor functions. The different orders of the reality scores of the θ and β variants at the different tests indicates that there is not one θ or one β value that scores the best on all origin destination pairs. The small influence of the θ and β value indicates that using a β of 5 instead of 8 has not much influence on the results, therefore there is not much need for one optimal β and θ values.
The θ and β value show similar behavior at the smooth fictitious play as at the C-logit, therefore we expect that there is not also one β and one θ value with an optimal reality score for the smooth fictitious play variants. The smooth fictitious play scores the highest of all tested variants. The congestion game scores also higher on reality than the shortest route principle. We saw that the parameters did not have much influence on the route choices of the smooth fictitious play and the congestion game.
The penalty function worked on the C-logit, congestion game and the smooth fictitious play. The penalty function seems to be an effective measure to minimize traffic on a specific route, but is less effective as correction term for errors in the route choice models and travel time functions.

Chapter 9

Conclusions and recommendations

This chapter describes the conclusions and the recommendations of this research project.

9.1 Conclusions

In this section we answer the research question and we discuss the validity of the test results in other networks. The research question is:

Are C-logit and game theory based route choice models able to produce more realistic route choices than the shortest route principle, and what is the influence of the C-logit and game theory parameters on the reality score of the route choices?

9.1.1 Comparison with shortest route principle

Our test results make clear that C-logit and game theory based route choice models can produce more realistic route choices than the shortest route principle. In fact all the C-logit and game theory variants result in more realistic route choices than the shortest route principle in the Enschede tests. The smooth fictitious play variants result in the highest reality score of all our tested variants, but this game also leads to the lowest scores on realistic routing in the theoretical test. Therefore we are not sure that a smooth fictitious play results in realistic route choices on other networks. The congestion game results in more realistic route choices than the shortest route principle, but the scores are not much higher than the shortest route principle. Next to the higher reality scores of the C-logit and game theory variants we also discover that the shortest route principle is more vulnerable to small errors in the travel time function than the C-logit and game theory based route choice models. This is a big advantage of the C-logit and game theory models, since we always use a route choice model in combination with estimated travel times and thus have to cope with errors in the travel times.

9.1.2 Influence of the parameters

The power of the travel time has little influence on the C-logit route choice results of one origin destination pair. The parameters β, θ, and the penalty function seem not to have much influence on the C-logit results of multiple origin destination pairs. The commonality factor has more influence, and the results indicate that commonality factor 4 is the most realistic commonality factor.
The p-values has not much influence on the route choices of the congestion game, unless we use unreasonably large p-values. The β, θ, commonality factor, and penalty function show similar behavior with the C-logit as with the smooth fictitious play.

9.1.3 Validity of the results

We test only one network with data, the network of Enschede. It is possible that we would derive different results if we test in another network, because people of other regions or in other networks could make different route choices. Still a majority of our conclusions hold for different networks or we have strong feelings that these conclusions also hold for other networks. The observation that the shortest route principle is more vulnerable for small errors in the travel time function than the C-logit and the game theory models does not depend on the network or the travelers, therefore this observation holds in general. We also see that the β, θ, and penalty value do not have much influence on the C-logit and smooth fictitious play, which is also a general observation. We cannot prove that the C-logit and the game theory model result in more realistic route choice also holds in other networks. Also we do not know whether commonality factor function 4 is the most realistic commonality factor in other networks. But we have a strong feeling that the C-logit results in more realistic

route choices for short trips for the C-logit, because the results of the tests show large differences between the C-logit and shortest route principle reality score. In more urban areas the intersection delays are a lot larger than in our Enschede network. Therefore, we expect that in more urban areas the number of intersections is an important component in the route choice decision. The current form of the shortest route principle is not able to incorporate the number of intersection in its calculations, while the C-logit is able to incorporate the number of intersections in the route choice results by a choice set generation method or utility function. Therefore we expect that the C-logit will also score better than the shortest route principle in more urban areas with an appropriate route set generation method. We also believe that commonality factor function 4 is the most realistic commonality factor for the C-logit in other networks with short trips, because commonality factor 4 has the highest reality score in all our test with all parameter combinations. In section 4.3.6 is mentioned that the shortest route principle might score higher on reality if we use a smaller aggregation interval, because the travel times may variate within the 15 minute interval. However we see that the travel times within the 15 minute interval do not change much. Therefore, a smaller aggregation interval will not lead to significant higher reality scores of the shortest route principle in our Enschede test. We cannot draw conclusions about our route choice models for long trips, because we only test our models on short trips.

9.2 Recommendations

We already mentioned several recommendations in this report. This section summarizes these recommendations and complete the recommendations with other new recommendations. We describe two kinds of recommendations: recommendations for the usage of our route choice variants, and recommendations for future research.

9.2.1 Usage of our route choice variant

The tests indicate that C-logit is a more realistic route choice model than the shortest route principle. Usage and experience with the C-logit can give more clearance if this also holds for other networks. The tests also indicate that the C-logit model does not have to suffer much from calibrating difficulties, because the influence of the β and θ value on the route choice are not very high. We expect that this holds also in other networks when multiple origin destination pairs are used, because the test results show that different β and θ value are optimal for different origin destination pairs. These differences level each other out when the route choices of multiple origin destination pairs are calculated. We already mentioned the big advantage that the C-logit and game theory models are less vulnerable for errors in the travel time function than the shortest route principle. Due to this advantage and the high reality scores the C-logit, smooth fictitious play and the congestion game are promising route choice models.
A disadvantage of the smooth fictitious play is that of all the players the previous move and their perception has to be remembered. This storage condition makes the computational time of the smooth fictitious play long. Also the congestion game suffers from long computational times, because the congestion game searches for an equilibrium situation. Therefore we recommend to use the C-logit. We advise to use a travel time power of 1, commonality factor function 4, θ values between 1 and 5, and β value between 2 and 8. We already found in literature that the C-logit should be used in combination with a choice set generation algorithm which is likely to produce a limited number of paths.
The theoretical test results show that the usage of travel time+ leads to better results in the most cases. Therefore we recommend the use of travel time+ instead of travel time, if possible.

9.2.2 Further research

In our research project we find that further research is still needed on many elements of route choice modeling. We recommend further research of the following six elements:

- Realistic route choice model starting from route choice data
- Realistic travel time functions
- Travel time+ functions
- The right link length
- Choice set generation algorithms
- The possibilities of Prospect theory

Investigation starting with route choice data

In his research project we start with a literature review, where we identify two promising route choice models. Next we investigate whether these two route choice approaches could result in more realistic route choices. The choice for starting the research project by literature review has some practical reasons. First, in this approach route choice data is only needed at the end of the research project, and second literature describes several promising route choice models. With the growing amount of available route choice data (think for example at mobile phone data) it would be interesting to start a research project at the other side of the problem. In such a research project, first route choice data needs to be investigated to find a 'pattern' in which the route choices can be described. Then the 'pattern' need to be linked to a route choice model from literature or a new route choice model will be developed.

Realistic travel time functions

We notice that the travel time functions have a huge impact on the the results of the traffic model, especially when the shortest route principle is used as route choice model. Further we see a decreasing trend in the reality score with the Dynasmart travel times. Therefore, we expect that there is still room for improvement of the travel time functions. An improvement in the travel time function can be derived by very simple investigation. Setting up one road section with a camera already gives a good impression of the travel time function for one road type. We believe that the total traffic model becomes more realistic with better travel time functions. The route choice decisions of travelers become more clear if we know the actual travel times.

Traveltime+ functions

Above, we mention that the results of our tests show that the usage of travel time+ leads to better route choices. We only test travel time+ in a static assignment of the theoretical test, because defining travel time+ in a dynamic assignment is out of the scope of this research project. Further research in the field of travel time+ functions in a dynamic transport model is therefore recommended.

The right link length

In a transport model the network is divided in links and nodes. The manner of deviation of the network in links influences the travel times (see also Appendix B). For instance, if we divide one congested link in two links, the sum of the travel times of the two links is smaller than the travel time of the one long link. In the case with two links only the first link is congested, and the outflow of the first link equals the capacity of the first and second link. Therefore the inflow on the second link equals the link 's capacity and no congestion occurs on the second link. The restrictions on the outflow of the first link cause that the total travel time on the two links is smaller than the travel time on the long link. This example shows how the link length affects the results of a transport model. Research in this field has not be done, but is needed to develop more realistic travel time function. Therefore we recommend research in the field of finding the right link lengths in a transport model network.

Choice set generation algorithms

In the C-logit route choice model, the route choice probability is strictly positive. A strictly positive route choice probability means that every route from the choice set has a probability of being chosen. Therefore, a wrongly chosen route in the choice set receives an undeserved traffic share. Clever chosen choice sets avoid observable unrealistic routing, by selecting only realistic routes for the choice set. These two examples show the importance of proper choice set generation. Unfortunately about choice set generation little is known, we therefore recommend further research in realistic choice set generation.

The possibilities of Prospect theory

In Chapter 3 we saw that Prospect theory is a promising approach for route choice modeling. We recommend further research in the field of Prospect theory, because there is not much known about Prospect theory in traffic situations.

Bibliography

[1] V. Henn & M. Ottomanelli. Handling uncertainty in route choice models: from probabilistic to possibilistic approaches. *European journal of operational research*, 175:1526–1538, 2006.

[2] J. de D. Ortuzar & L.G. Willumsen. *Modelling Transport*. John Wiley & Sons, third edition, 2005.

[3] M.F.A.M. Maarseveen M.H.P. Zuidgeest & K.M. van Zuilekom. *Dictaat Verkeer*. Universiteit Twente, 2004.

[4] M.S. Ramming. Network knowledge and route choice. Master's thesis, Massachusetts institute of technology, 2002.

[5] M.C. Verkaik Poelman. *Questor assignment*. DHV, 2007.

[6] DHV infrastructuur en milieu. *Handleiding Questor 7.3*, 2004.

[7] T. Giessen. *Dynamische verkeerssimulaties Dynasmart en Questor modelinterface*, 2009. Power Point presentation.

[8] H.S. Mahmassani & H. Sbayti. *Dynasmart-P version 1.4 User's guide*, 2007.

[9] Transport Simulation Systems. *Microsimulator and Mesosimulatior in Aimsun 6 User's Manual*, 2007.

[10] P.H.L. Bovy. On modeling route choice sets in transportation networks: a synthesis. *Transport Reviews*, 29:43–69, 2009.

[11] E. Frejinger M. Bierlaire & M. Ben-Akiva. Sampling of alternatives for route choice modeling. *Transportation Research Part B*, xx:xx, 2009.

[12] M. Ben-Akiva & M. Bierlaire. Discrete choice models with applications to departure time and route choice. In *Handbook of Transportation Science*, chapter 2. Second edition, 2003.

[13] C.H. Wen & F.S. Koppelman. The generalized nested logit model. *Transportation research part b*, 35:627–641, 2001.

[14] E. Cascetta A. Nuzzolar F. Russo & A. Vietta. A modified logit route choice model overcoming path overlapping problems. *Transportation and Traffic Theory Reviews*, pages 697–711, 1996.

[15] S.Hoogendoorn-Lanser R. van Nes & P.H.L. Bovy. Path size and overlap in multimodal transport networks. In *Transportation and traffic theory*, chapter 4. Elsevier, 2005.

[16] P. Bonsall. The influence of route guidance advice on route choice in urban networks. *Transportation*, 9:1–23, 1992.

[17] J. Swait & M. Ben-Akiva. Incorporating random constraints in discrete models of choice set generation. *Transportation Research Part B*, 21:91–102, 1987.

[18] A. Tversky & D. Kahneman. Rational choice and the framing of decision. *The journal of business*, 59(4):s251–s278, 1986.

[19] D. Kahneman & A. Tversky. Prospect theory: an analysis of decision under risk. *Econometrica*, 47(2):263–291, 1979.

[20] A. Tversky & D. Kahneman. Traveller behavior: Decision making in an unpredictable world. *Journal of Intelligent Transportation Systems*, 8(1):45–60, 2004.

[21] R.D. Connors & A. Sumalee. A network equilibrium model with travellers' perception of stochastic travel times. *Transportation Research Part B*, 43:614–624, 2009.

[22] E. Avineri & J.N Prashkar. Sensitivity to uncertainty: The need for a paradigm shift. *Transportation Research Record*, 1854:90–98, 2002.

[23] H. Peters. *Game theory*. Springer-Verlag, 2008.

[24] I. Milchtaich. Congestion games with player-specific payoff functions. *Games and economic behavior*, 13:111–124, 1996.

[25] A. Garcia D. Reaume & R. L. Smith. Fictitious play for finding system optimal routings in dynamic traffic networks. *Transportation Research Part B*, 34:147–156, 2000.

[26] M. Benaim. Dynamics of stochastic approximation algorithms. *Seminaire de probabilities (Strasbourg)*, 33:1–68, 1999.

[27] R. Cominetti E.Melo & S. Sorin. A payoff-based learning procedure and its application to traffic games. *Games and Economic Behavior*, 2008.

[28] M.G.H. Bell & C. Cassir. Risk-averse user equilibrium traffic assignment: an application of game theory. *Transportation Research Part B*, 36:671–681, 2002.

[29] T. Thomas & B. Tutert. *Een nieuwe kijk op route keuze? Bevindingen uit het kentekenonderzoek Enschede*. Witteveen+Bos, 2009.

[30] CBS. *Mobiliteit per regio naar vervoerwijzen en algemene kenmerken*, 2008. Retrieved 16th of December 2009 from: http://www.cbs.nl/nl-NL/menu/themas/verkeer-vervoer/cijfers/default.htm.

Appendix A

Matlab files

A.1 Theoretical test

A.1.1 The network

%Initializing variables
A=[0 1 0 0 0 0 1; 0 0 1 0 0 0 0; 0 0 0 1 1 0 0; 0 0 0 0 1 0 0; 0 0 0 0 0 1 0; 0 0 0 0 0 0 1; 0 0 0 0 0 0 0];
C=[0 15; 0.5 14.8; 4 14.8; 4.5 14.6; 5 14.8; 9.5 14.8; 10 15]; %Coordinates nodes
NoN=length(Coordinates); %Number of Nodes
NoL=length(row); % Number of Links
NoP=3; % Number of paths
Demand=[2000];
Capacity=[1600 4400 1600 4400 1600 4400 1800 1600];
LinksPath=[0 0 0 0 0 0 1 0; 1 1 0 1 0 1 0 1; 1 1 1 0 1 1 0 1];
speed=[60 120 60 120 60 120 80 60];
RoadCat=[3 1 3 1 3 1 2 3]; % 1=highway, 2=urban road, 3=on and off-ramp
.[r, c]=find(A); *%Number links*

%Calculate matrix with link number from node i to node j LinkMatrix=zeros(NoN,NoN);
for k=1:length(row)
    LinkMatrix(row(k),col(k))=k;
end
for i=1:NoL
    $LengthLink(i) = \sqrt{(C(r(i),1) - C(c(i),1))^2 + (C(r(i),2) - C(c(i),2))^2}$;
end

A.1.2 The static assignment

m=SimpleNetwork; *%Load the network file*

% Initializing
NoI=10; *% Number of intervals*
m.NoP=3; *% Number of paths*
x=[0 0 0 0 0 0 0 0]; *% initial flow*
PathTime=zeros(m.NoP,1);
PathLength=zeros(m.NoP,1);
UtilityLength=zeros(m.NoP,1);
PenaltyLinks=[3 5]; *% which combination of links receives a penalty*
x2=[0 5000/NoI 0 5000/NoI 0 5000/NoI 2200/NoI 0]; % background traffic
fraction=[0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1];
bias=[1 1 1];
PenValue=0;
NoI=length(fraction);*% fraction of the traffic that is loaded in one iteration*

% Determine length of the paths and penalize path

```
for j=1:m.NoP
    for i=1:m.NoL
        PathLength(j)=PathLength(j)+m.LinkLength(i)*m.LinksPath(j,i);
        UtilityLength(j)=UtilityLength(j)+m.LinkLength(i)*m.LinksPath(j,i)*bias(m.RoadCat(i));
    end
    if m.LinksPath(j,PenaltyLinks(1))==1
      if m.LinksPath(j, PenaltyLinks(2))==1
        Penalty(j)=PenValue;
      end
    end
end


    % Start of the assignment

for k=1:NoI
    PathTime=zeros(m.NoP,1);% Determine travel times of the paths
    for i=1:m.NoL
        Traveltime(i)=m.LinkLength(i)/m.speed(i)*60 *(1+0.6*(x(i)/m.Capacity(i))^4);
    end

    % Variant specific part: Determine route choice probabilities.
    for i=1:m.NoP
        P(i)=....; % Route choice probabilities
    end

    % Load traffic according to the route choice probabilities.
    for i=1:m.NoL
        for j=1:m.NoP
            x(i)=x(i)+fraction(k)*m.Demand*P(j)*m.LinksPath(j,i);
        end
    end
    x=x+x2;
end


Results.apad=[x(7) x(4) x(3)]; % The flows on the paths are specified by links 7, 4 and 3.
```

A.1.3 Dynamic assignment

```
 % Read initial path and vehicle file for initial values
A=dlmread('output_path.dat');
B=xlsread('output_vehicle.xls');

% Initialization of variables
NoP=3;
NoL=37; % Number of Links
n=10;
a=43;
NoV=B(1,1);
LinksPath=zeros(NoP,NoL);
LinksPath(1,8)=1; LinksPath(1,9)=1; LinksPath(1,12)=1; LinksPath(1,16)=1;
LinksPath(2,2)=1; LinksPath(2,3)=1; LinksPath(2,4)=1; LinksPath(2,6)=1; LinksPath(2,7)=1; LinksPath(2,9)=1;
LinksPath(2,14)=1; LinksPath(2,16)=1; LinksPath(2,36)=1; LinksPath(2,37)=1;
LinksPath(3,1)=1; LinksPath(3,3)=1; LinksPath(3,4)=1; LinksPath(3,6)=1; LinksPath(3,7)=1; LinksPath(3,9)=1;
LinksPath(3,10)=1; LinksPath(3,16)=1; LinksPath(3,36)=1; LinksPath(3,37)=1;

% Read travel times of previous runs (in this example 10 runs) C1=dlmread('output_td_linktraveltime1.dat');
C2=dlmread('output_td_linktraveltime2.dat');
C3=dlmread('output_td_linktraveltime3.dat');
C4=dlmread('output_td_linktraveltime4.dat');
C5=dlmread('output_td_linktraveltime5.dat');
```

```
C6=dlmread('output_td_linktraveltime6.dat');
C7=dlmread('output_td_linktraveltime7.dat');
C8=dlmread('output_td_linktraveltime8.dat');
C9=dlmread('output_td_linktraveltime9.dat');
C10=dlmread('output_td_linktraveltime10.dat');

% Determine the route choice probabilities

for i=1:31 %Because we have 31 time intervals
    for j=1:NoP
        PathTime1(j,i)=0;
        PathTime2(j,i)=0;
        PathTime3(j,i)=0;
        PathTime4(j,i)=0;
        PathTime5(j,i)=0;
        PathTime6(j,i)=0;
        PathTime7(j,i)=0;
        PathTime8(j,i)=0;
        PathTime9(j,i)=0;
        PathTime10(j,i)=0;
        for h=1:NoL
            PathTime1(j,i)=C1(a*(i-1)+6+h,5)*LinksPath(j,h)+PathTime1(j,i);
            PathTime2(j,i)=C2(a*(i-1)+6+h,5)*LinksPath(j,h)+PathTime2(j,i);
            PathTime3(j,i)=C3(a*(i-1)+6+h,5)*LinksPath(j,h)+PathTime3(j,i);
            PathTime4(j,i)=C4(a*(i-1)+6+h,5)*LinksPath(j,h)+PathTime4(j,i);
            PathTime5(j,i)=C5(a*(i-1)+6+h,5)*LinksPath(j,h)+PathTime5(j,i);
            PathTime6(j,i)=C6(a*(i-1)+6+h,5)*LinksPath(j,h)+PathTime6(j,i);
            PathTime7(j,i)=C7(a*(i-1)+6+h,5)*LinksPath(j,h)+PathTime7(j,i);
            PathTime8(j,i)=C8(a*(i-1)+6+h,5)*LinksPath(j,h)+PathTime8(j,i)
            PathTime9(j,i)=C9(a*(i-1)+6+h,5)*LinksPath(j,h)+PathTime9(j,i);
            PathTime10(j,i)=C10(a*(i-1)+6+h,5)*LinksPath(j,h)+PathTime10(j,i);
        end
    end

    PathTime(1,i)=1/n*(PathTime1(1,i)+PathTime2(1,i)+...+PathTime9(1,i)+PathTime10(1,i));
    PathTime(2,i)=1/n*(PathTime1(2,i)+PathTime2(2,i)+...+PathTime9(2,i)+PathTime10(2,i));
    PathTime(3,i)=1/n*(PathTime1(3,i)+PathTime2(3,i)+...+PathTime9(3,i)+PathTime10(3,i));

     % Determine route choice probabilities, depended of route choice model
    P(1,i)=...;
    P(2,i)=...;
    P(3,i)=...;
end


    % Write background traffic in path and vehicle file

for i=1:NoV
    if B(2*i+1,13)==6
      if B(2*i+2,1)==4
        A(i,:)=[15 4 1 9 2 25 26 3 14 0 0 0 0 0];
        B(2*i+1,8)=9;
      end
    else
      if B(2*i+1,13)==1
        if B(2*i+2,1)==6
          A(i,:)=[10 5 8 6 13 18 19 15 0 0 0 0 0 0];
          B(2*i+1,8)=8;
        end
      end
    end
end
```

```
%Write results in path and vehicle file
k=0;
l=0;
m=0;


for i=0:30
    for j=1:NoV
        if B(2*j+1,13)==1
           if B(2+2*j,1)==2
              if B(2*j+1,4)<=2*(i+1)
                 if B(2*j+1,4)>2*i
                    randomnumber=rand;
                    if rand<P(1,i+1)
                       A(j,:)=[10 5 8 6 13 0 0 0 0 0 0 0 0 0];
                       B(2*j+1,8)=5;
                       k=k+1;
                    else
                       if randomnumber <P(1,i+1)+P(2,i+1)
                          A(j,:)=[10 5 4 1 9 2 25 26 3 6 13 0 0 0];
                          B(2+2*j-1,8)=11;
                          l=l+1;
                       else
                          A(j,:)=[10 5 4 1 7 2 25 26 3 6 13 0 0 0];
                          B(2+2*j-1,8)=11;
                          m=m+1;
                       end
                    end
                 end
              end
           end
        end
    end
end


aa.B=B;
aa.A=A;
regelveh1=' %d %d # of vehicles in the file, Max # of stops\n';
tekstveh1=[NoV 1];
regelveh2=' # usec dsec stime usrcls vehtype ioc #ONode #IntDe info ribf comp OZ\n';

fid=fopen('vehicle.dat','at');
fprintf(fid, regelveh1, tekstveh1);
fprintf(fid, regelveh2);
fclose(fid);
regelveh20=' %d %6.2f\n';



for j=1:NoV %write file for all vehicles
    regel='een twee drie vier vijf zes zeven acht negen tien elf twaalf derten virten \n';
    regelveh='een twee drie vier vijf zes zeven acht negen tien %1.4f %1.4f %d\n';
    if A(j,1)>9
       regel1=regexprep(regel,'een', '   %d');
    else
       if A(j,1)==0
          regel1=regexprep(regel,'een', '');
       else
          regel1=regexprep(regel,'een', '    %d');
```

```
    end
  end
  if A(j,2)>9
    regel2=regexprep(regel1,'twee', '   %d');
  else
    if A(j,2)==0
      regel2=regexprep(regel1,'twee','');
    else
      regel2=regexprep(regel1,'twee', '    %d');
    end
  end
  if A(j,3)>9
    regel3=regexprep(regel2,'drie', '   %d');
  else
    if A(j,3)==0
      regel3=regexprep(regel2,'drie','');
    else
      regel3=regexprep(regel2,'drie', '    %d');
    end
  end
..
..
  if A(j,14)>9
    regel14=regexprep(regel13,'virten', '   %d');
  else
    if A(j,14)==0
      regel14=regexprep(regel13,'virten','');
    else
      regel14=regexprep(regel13,'virten', '    %d');
    end
  end
  if B(j*2+1,1)>9999
    regelveh1=regexprep(regelveh, 'een', ' %d');
  else
    if B(j*2+1,1)>999
      regelveh1=regexprep(regelveh, 'een', '  %d');
    else
      if B(j*2+1,1)>99
        regelveh1=regexprep(regelveh, 'een', '   %d');
      else
        ifB(j*2+1,1)>9
          regelveh1=regexprep(regelveh, 'een', '    %d');
        else
          regelveh1=regexprep(regelveh, 'een', '     %d');
        end
      end
    end
  end
  if B(j*2+1,2)>9
    regelveh2=regexprep(regelveh1, 'twee', '    %d');
  else
    regelveh2=regexprep(regelveh1, 'twee', '     %d');
  end
  if B(j*2+1,3)>9
    regelveh3=regexprep(regelveh2, 'drie', '    %d');
  else
    regelveh3=regexprep(regelveh2, 'drie', '     %d');
  end
  if B(j*2+1,4)>99
    regelveh4=regexprep(regelveh3, 'vier', ' %3.2f');
  else
```

```
    if B(j*2+1,4)>9
      regelveh4=regexprep(regelveh3, 'vier', ' %3.2f');
    else
      regelveh4=regexprep(regelveh3, 'vier', '  %3.2f');
    end
  end
  if B(j*2+1,5)>9
    regelveh5=regexprep(regelveh4, 'vijf', '    %d');
  else
    regelveh5=regexprep(regelveh4, 'vijf', '     %d');
  end
  if B(j*2+1,6)>9
    regelveh6=regexprep(regelveh5, 'zes', '    %d');
  else
    regelveh6=regexprep(regelveh5, 'zes', '     %d');
  end
  if B(j*2+1,7)>9
    regelveh7=regexprep(regelveh6, 'zeven', '    %d');
  else
    regelveh7=regexprep(regelveh6, 'zeven', '     %d');
  end
  if B(j*2+1,8)>9
    regelveh8=regexprep(regelveh7, 'acht', '    %d');
  else
    regelveh8=regexprep(regelveh7, 'acht', '     %d');
  end
  if B(j*2+1,9)>9
    regelveh9=regexprep(regelveh8, 'negen', '    %d');
  else
    regelveh9=regexprep(regelveh8, 'negen', '     %d');
  end
  if B(j*2+1,10)>9
    regelveh10=regexprep(regelveh9, 'tien', '    %d');
  else
    regelveh10=regexprep(regelveh9, 'tien', '     %d');
  end

  tekst=[];
  for i=1:14
    if A(j,i) =0
      tekst=[tekst A(j,i)];
    else
      break
    end
  end
  tekst2=[];
  for i=1:2
    tekst2=[tekst2 B(j*2+2,i)];
  end
  fid=fopen('path.dat','at');
  fprintf(fid, regel14, tekst);
  fclose(fid);
  fid=fopen('vehicle.dat', 'at');
  fprintf(fid, regelveh10, B(j*2+1,:));
  fprintf(fid, regelveh20, tekst2);
  fclose(fid);
end
```

A.2 Shortest route algorithm

A.2.1 Dijkstra algorithm

```
function [P]=DijkstraOD(origin, destination)

% Read network files
A=dlmread('Links.txt');
B=dlmread('Nodes.txt');

% Initializing variables
NoL=length(A(:,1));
NoN=length(B(:,1));
MaxTranslated=max(B(:,1));
IDTranslated=zeros(MaxTranslated,1);
NodesConnected=zeros(NoN,NoN);
Predecessor=zeros(1,NoN);
Infinity=10000;


for j=1:NoN
    NodeID(j)=B(j,1);
    NodeX(j)=B(j,3);
    NodeY(j)=B(j,4);
    IDTranslated(NodeID(j))=j;
end


for i=1:NoL
    LinkStart(i)=A(i,2);
    LinkEnd(i)=A(i,3);
    LinkSpeed(i)=A(i,6);
    LinkCap(i)=A(i,10);
    LinkLength(i)=A(i,4);
    NodesConnected(IDTranslated(LinkStart(i)), IDTranslated(LinkEnd(i)))=LinkLength(i);
end


for i=1:NoN
    for j=1:NoN
        if NodesConnected(i,j)==0
          DistanceMatrix(i,j)=Infinity;
        else
          DistanceMatrix(i,j)=NodesConnected(i,j);
        end
    end
end


P.DistanceMatrix=DistanceMatrix;
P.Infinity=Infinity;
P.NoN=NoN;

AA=origin; %start node
distance=1000000*ones(1,NoN);
distance(origin)=0;
X=[];


%Start of the algorithm
for j=1:NoN
```

```
    if NodesConnected(origin,j)>0
      distance(j)=NodesConnected(origin,j);
      X=[X; j];
    end
end
for i=1:length(X)
    Predecessor(X(i))=origin;
end

%Select the point with the shortest distance from X
for n=1:NoN
    HulpArray=[];
    for i=1:length(X)
      HulpArray = [HulpArray distance(X(i))];
    end
    [C I]= min(HulpArray);
    PointSelect=X(I);
    AA=[AA PointSelect];
    for j=1:NoN
      hulp1=0;
      hulp2=0;
      if NodesConnected(PointSelect,j)>0
        if distance(j)> distance(PointSelect)+ NodesConnected(PointSelect,j);
          distance(j)=distance(PointSelect)+ NodesConnected(PointSelect,j);
          Predecessor(j)=PointSelect;
        end
        for i=1:length(AA) %check if point is not already contained in AA
          if AA(i)==j
            hulp1=hulp1+1;
          end
        end
        for i=1:length(X) %check if point is not already contained in X
          if X(i)==j
            hulp2=hulp2+1;
          end
        end
        if hulp1 ==0
          if hulp2==0
            X=[X; j];
          end
        end
      end
    end
    X(I)=[];
    if length(X)==0
      n;
      break
    end
end


    %find path of the origin destination pair
Path=destination;
a=destination;


for i=1:100
    if Predecessor(a)==0
      Path=[0];
    else
      if Predecessor(a)==origin
        Path=[origin Path];
```

```
        break
      else
        a=Predecessor(a);
        Path=[a Path];
      end
    end
end


for i=1:length(Path)
    PathNodes(i)=Path(i);
end


    P.Path=PathNodes;
P.LengthShortestPath=distance(destination);
```

A.2.2 k-shortest route algorithm

```
function List = KshortestPaths(origin,destination,k)

% Initializing variables
Infinity=10000;
a = DijkstraOD(origin,destination); %Determine first shortest and add this path to List
List(1).shortestPath= a.Path;
List(1).lengthShortestPath=a.LengthShortestPath;
List(1).partitionLinks = [];
partitionList(1).shortestPath = a.Path;
partitionList(1).lengthShortestPath=a.LengthShortestPath;
partitionList(1).partitionLinks = [];
distanceMatrix = a.DistanceMatrix; %Determine distanceMatrix within this M-file
templist(1).path = [];
templist(1).length = a.Infinity;
templist(1).link = [];



while length(List) < k
        tempDistanceMatrix = distanceMatrix; %Remembers original distanceMatrix
        lastShortestPath = partitionList(end).shortestPath; %Takes last shortest path from List
        partitionedLinks = partitionList(end).partitionLinks; %The removed links
        if ĩsempty(partitionedLinks)
          if size(partitionedLinks)== [2,1]
            temp1 = partitionedLinks(1,1);
            temp2 = partitionedLinks(2,1);
            distanceMatrix(temp1,temp2)=a.Infinity;
          else
            for i=1:length(partitionedLinks)
                temp1 = partitionedLinks(1,i);
                temp2 = partitionedLinks(2,i);
                distanceMatrix(temp1,temp2)=a.Infinity; %Block this link in distanceMatrix
            end
          end
        end
        % Add new path to templist
        for i=1:length(lastShortestPath)-1
          excl1 = lastShortestPath(i);
          excl2 = lastShortestPath(i+1);
          temp = distanceMatrix(excl1,excl2); %Remembers temporarily link distance of [excl1 excl2]
          distanceMatrix(excl1,excl2)= a.Infinity;
          c = DijkstraODkshortest(origin,destination,distanceMatrix); %Calculates shortest path without blocked link
```

```
            templist(end+1).path = c.path;
            templist(end).length = c.LengthShortestPath;
            if isempty(partitionedLinks)
               templist(end).link = [excl1;excl2];
            else
               templist(end).link =[partitionedLinks(1,:) excl1;partitionedLinks(2,:) excl2];
            end
            distanceMatrix(excl1,excl2)= temp; %Repairs link distance in distanceMatrix
         end
         temp = a.Infinity;
         index = -1;
         %Determines shortest route in temp list, which is not contained in List
         for i=1:length(templist)
            if templist(i).length<temp
               temp = templist(i).length;
               index = i;
               link = templist(i).link;
            end
         end
         if index == -1
            break;
         end

         partitionList(end+1).shortestPath = templist(index).path;
         partitionList(end).lengthShortestPath = templist(index).length;
         partitionList(end).partitionLinks = link;

         newpath=1; % Determine of chosen path is new
         for i=1:length(List)
            if length(templist(index).path)==length(List(i).shortestPath)
               if templist(index).path == List(i).shortestPath
                    newpath = 0; % Do not add path if it is not new
               end
            end
         end
         if newpath == 1 % Add path if it is new
            List(end+1).shortestPath = templist(index).path;
            List(end).lengthShortestPath = templist(index).length;
            List(end).partitionLinks = link;
         end
         templist(index) = []; % Remove chosen path from templist
         distanceMatrix = tempDistanceMatrix; % Repair distanceMatrix
end
```

A.3 Enschede Test on 1 OD with experienced travel times

A.3.1 The network

```
%Read Network file
A=dlmread('Links.txt');
B=dlmread('Nodes.txt');
D=xlsread('TranslateNodeZone.xls');

% Initializing variables
NoL=length(A(:,1));
NoN=length(B(:,1));
MaxTranslated=max(B(:,1));
IDTranslated=zeros(MaxTranslated,1);
NodesConnected=zeros(NoN,NoN);
Predecessor=zeros(NoN,NoN);
```

```
for j=1:NoN
    NodeID(j)=B(j,1);
    NodeX(j)=B(j,3);
    NodeY(j)=B(j,4);
    IDTranslated(NodeID(j))=j;
end


for i=1:NoL
    LinkStart(i)=A(i,2);
    LinkEnd(i)=A(i,3);
    LinkSpeed(i)=A(i,7);
    LinkCap(i)=A(i,10);
    LinkLength(i)=A(i,4);
    NodesConnected(IDTranslated(LinkStart(i)), IDTranslated(LinkEnd(i)))=LinkLength(i)/LinkSpeed(i)*0.06;
end

% Determine shortest route between every pair of register points (Dijkstra algorithm)
for m=1:NoN
    %Initialization of variables
    AA=m;
    distance(m,:)=1000000*ones(1,NoN);
    distance(m,m)=0;
    X=[];

    forj=1:NoN
        if NodesConnected(m,j)>0
            distance(m,j)=NodesConnected(m,j);
            X=[X; j];
        end
    end
    for i=1:length(X)
        Predecessor(m,X(i))=m;
    end

    for n=1:NoN
        %Select the point with the shortest distance from X
        HulpArray=[];
        for i=1:length(X)
            HulpArray = [HulpArray distance(m,X(i))];
        end
        [C I]= min(HulpArray);
        PointSelect=X(I);
        AA=[AA PointSelect];

        for j=1:NoN
            hulp1=0;
            hulp2=0;
            if NodesConnected(PointSelect,j)>0
                if distance(m,j)> distance(m,PointSelect)+ NodesConnected(PointSelect,j);
                    distance(m,j)=distance(m,PointSelect)+ NodesConnected(PointSelect,j);
                    Predecessor(m,j)=PointSelect;
                end
                for i=1:length(AA) %check if point is not already contained in AA
                    if AA(i)==j
                        hulp1=hulp1+1;
                    end
                end
                for i=1:length(X) %check if point is not already contained in X
                    if X(i)==j
```

```
                hulp2=hulp2+1;
              end
            end
            if hulp1 ==0
              if hulp2==0
                X=[X; j];
              end
            end
          end
        end
        X(I)=[];
        if length(X)==0
          n;
          break
        end
    end end

%Determine all possible node combinations origins=[];
destination=[];
PosNodeID=D(:,2);
for i=1:length(PosNodeID)
    PosNode(i)=IDTranslated(PosNodeID(i));
end
for i=1:length(PosNodeID)
    destination=[destination PosNode];
    HelpNode=[];
    for j=1:length(PosNodeID)
        HelpNode=[HelpNode PosNode(i)];
    end
    origins=[origins HelpNode];
end

% Write path between each regiter point pair
for k=1:length(origins)
    Path=destination(k);
    a=destination(k);
    origin=origins(k);
    for i=1:100
        if Predecessor(origin,a)==0
          Path=[0];
        else
          if Predecessor(origin,a)==origin
            Path=[origin Path];
            break
          else
            a=Predecessor(origin,a);
            Path=[a Path];
          end
        end
    end
    PathNodes(k,:)=zeros(1,34);
    for i=1:length(Path)
        PathNodes(k,i)=Path(i);
    end
end


dlmwrite('PathBetweenNodes.dat', PathNodes);
dlmwrite('Distances.dat', distance);
```

A.3.2 Determine route choices

```
function [m]= RouteShares
%This file calculates the route shares for different route choice model between tm7 and ck6 if we follow the travel times from the data

%load travel times of the ten shortest paths
A=dlmread('TravelTimes1400_1500.txt');
B=dlmread('TravelTimes1500_1600.txt');
C=dlmread('TravelTimes1600_1700.txt');
D=dlmread('TravelTimes1700_1800.txt');

% Initializing parameter settings
theta=5;
cf=4;
beta=1;
gamma=1;
penalty=0;
PenRoute=[];
power=1;
Periods=4; % Number of periods in a hour
NoP=10; % Number of Paths

%Calculate the commonality factor functions
gamma=1;
beta=1;

E=dlmread('Links.txt');
F=dlmread('Nodes.txt');

%The choice set with the 10 shortest paths
TenPath(1,:)=[5 4 6 7 12 13 15 16 17 64 63 111 58 107 59 60 96 56 50 51 52 47 46 0 0 0 0 0 0 0];
TenPath(2,:)=[5 4 6 7 12 13 15 16 17 98 26 25 27 106 28 42 104 92 41 108 44 105 93 49 48 116 47 46 0 0];
TenPath(3,:)=[5 4 6 7 8 9 14 13 15 16 17 64 63 111 58 107 59 60 96 56 50 51 52 47 46 0 0 0 0 0];
TenPath(4,:)=[5 4 6 7 12 13 67 66 70 69 68 71 61 95 59 60 96 56 50 51 52 47 46 0 0 0 0 0 0 0];
TenPath(5,:)=[5 4 6 7 8 9 14 13 15 16 17 98 26 25 27 106 28 42 104 92 41 108 44 105 93 49 48 116 47 46];
TenPath(6,:)=[5 4 6 7 8 9 14 13 67 66 70 69 68 71 61 95 59 60 96 56 50 51 52 47 46 0 0 0 0 0];
TenPath(7,:)=[5 4 6 7 12 13 15 16 17 64 63 111 58 97 62 61 95 59 60 96 56 50 51 52 47 46 0 0 0 0];
TenPath(8,:)=[5 4 6 7 12 13 67 66 70 69 68 71 61 62 97 58 107 59 60 96 56 50 51 52 47 46 0 0 0 0];
TenPath(9,:)=[5 4 6 7 8 9 14 13 15 16 17 64 63 111 58 97 62 61 95 59 60 96 56 50 51 52 47 46 0 0];
TenPath(10,:)=[5 4 6 7 8 9 14 13 67 66 70 69 68 71 61 62 97 58 107 59 60 96 56 50 51 52 47 46 0 0];

NoL=length(E(:,1));
NoP=10;
NoN=length(F(:,1));
MaxTranslated=max(F(:,1));
IDTranslated=zeros(MaxTranslated,1);


for j=1:NoN
    NodeID(j)=F(j,1);
    IDTranslated(NodeID(j))=j;
end


%Determine Linklength
for i=1:NoL
    LinkSpeed(i)=E(i,7);
    LinkLength(i)=E(i,4);
    LinkStart(i)=E(i,2);
    LinkEnd(i)=E(i,3);
end
```

```
for h=1:NoL
    LinkStartMatlab(h)=IDTranslated(LinkStart(h));
    LinkEndMatlab(h)=IDTranslated(LinkEnd(h));
end


%Determine the links of the paths
LinksPath=zeros(NoP,NoL);
for j=1:length(Path1)-1
    for i=1:NoL
        for h=1:NoP
            if LinkStartMatlab(i)==TenPath(h,j)
              if LinkEndMatlab(i)==TenPath(h,j+1)
                LinksPath(h,i)=1;
              end
            end
        end
        end
    end
end


%Determine Lij and Nan
Lij=zeros(NoP);
PathLength=zeros(NoP,1);
Nan=zeros(NoL,1);
for i=1:NoP
    for j=1:NoP
        for k=1:NoL
            if LinksPath(i,k)==LinksPath(j,k)
              if LinksPath(i,k) ==1
                Lij(i,j)=Lij(i,j)+LinkLength(k);
              end
            end
        end
    end
end
for j=1:NoP
    for i=1:NoL
        PathLength(j)=PathLength(j)+LinkLength(i)*LinksPath(j,i);
    end
end
for j=1:NoP
    for i=1:NoL
        Nan(i)=Nan(i)+LinksPath(j,i);
    end
end


% Calculate the CF
for i=1:NoP
    hulp(1,i)=0;
    hulp(2,i)=0;
    hulp(3,i)=0;
    hulp(4,i)=0;
    hulp(5,i)=0;
    for j=1:NoP
        hulp(1,i)=hulp(1,i)+(Lij(i,j)/(sqrt(PathLength(i))*sqrt(PathLength(j))))^gamma; % sum lij/sqrt(li lj)
    end
    for j=1:NoL
```

```
        if LinksPath(i,j)==1
          hulp(2,i) = hulp(2,i)+ LinkLength(j)/PathLength(i)*Nan(j);
          hulp(3,i)= hulp(3,i)+LinkLength(j)/PathLength(i)*log(Nan(j));
          hulp(5,i)= hulp(5,i)+ LinkLength(j)/PathLength(i)*1/Nan(j);
        end
    end
    CF(1,i)=-beta *log(hulp(1,i));
    CF(2,i)=-beta*log(hulp(2,i));
    CF(3,i)=-beta*hulp(3,i);
    for j=1:NoP
        if j ≅i
          hulp(4,i)=hulp(4,i)+(Lij(i,j)/(sqrt(PathLength(i)*PathLength(j))))*((PathLength(i)-Lij(i,j))/
                    (PathLength(j)-Lij(i,j)));
        end
    end
    CF(4,i)=-beta*log(1+hulp(4,i));
    CF(5,i)=beta*log (hulp(5,i));
end


fori=1:Periods
   for j=1:NoP
       P1a(j,i)= exp(theta*(-A(j,i)+CF(cf,j)))/(exp(theta*(-A(1,i)+CF(cf,1)))+exp(theta*(-A(2,i)+CF(cf,2)))+
                 exp(theta*(-A(3,i)+CF(cf,3)))+exp(theta*(-A(4,i)+CF(cf,4)))+exp(theta*(-A(5,i)+CF(cf,5)))+
                 exp(theta*(-A(6,i)+CF(cf,6)))+exp(theta*(-A(7,i)+CF(cf,7)))+exp(theta*(-A(8,i)+CF(cf,8)))+
                 exp(theta*(-A(9,i)+CF(cf,9)))+exp(theta*(-A(10,i)+CF(cf,10))));
       P1b(j,i)=exp(theta*(-B(j,i)+CF(cf,j)))/(exp(theta*(-B(1,i)+CF(cf,1)))+exp(theta*(-B(2,i)+CF(cf,2)))+
                 exp(theta*(-B(3,i)+CF(cf,3)))+exp(theta*(-B(4,i)+CF(cf,4)))+exp(theta*(-B(5,i)+CF(cf,5)))+
                 exp(theta*(-B(6,i)+CF(cf,6)))+exp(theta*(-B(7,i)+CF(cf,7)))+exp(theta*(-B(8,i)+CF(cf,8)))+
                 exp(theta*(-B(9,i)+CF(cf,9)))+exp(theta*(-B(10,i)+CF(cf,10))));
       P1c(j,i)=exp(theta*(-C(j,i)+CF(cf,j)))/(exp(theta*(-C(1,i)+CF(cf,1)))+exp(theta*(-C(2,i)+CF(cf,2)))+
                 exp(theta*(-C(3,i)+CF(cf,3)))+exp(theta*(-C(4,i)+CF(cf,4)))+exp(theta*(-C(5,i)+CF(cf,5)))+
                 exp(theta*(-C(6,i)+CF(cf,6)))+exp(theta*(-C(7,i)+CF(cf,7)))+exp(theta*(-C(8,i)+CF(cf,8)))+
                 exp(theta*(-C(9,i)+CF(cf,9)))+exp(theta*(-C(10,i)+CF(cf,10))));
       P1d(j,i)=exp(theta*(-D(j,i)+CF(cf,j)))/(exp(theta*(-D(1,i)+CF(cf,1)))+exp(theta*(-D(2,i)+CF(cf,2)))+
                 exp(theta*(-D(3,i)+CF(cf,3)))+exp(theta*(-D(4,i)+CF(cf,4)))+exp(theta*(-D(5,i)+CF(cf,5)))+
                 exp(theta*(-D(6,i)+CF(cf,6)))+exp(theta*(-D(7,i)+CF(cf,7)))+exp(theta*(-D(8,i)+CF(cf,8)))+
                 exp(theta*(-D(9,i)+beta(g)*E(CF,9)))+exp(theta*(-D(10,i)+beta(g)*E(CF,10))));
   end
end


P1=[P1a;zeros(2,NoP);P1b; zeros(2,NoP); P1c; zeros(2,NoP);P1d];
xlswrite('Result.xls',P1);
```

A.4 Enschede test on 1 OD with Dynasmart travel times

A.4.1 Determine route choices

%This file calculate the route choice probability for every quarter of the first two hours

```
    %Initializing variables
cf=4;
beta=8;
theta=1;
penvalue=0;
PenRoute=[];
FavourRoute=[];
power=1;
```

```
%Determine commonality factor
gamma=1;
beta=1;

E=dlmread('Links.txt');
F=dlmread('Nodes.txt');

%The choice set with the 10 shortest paths
TenPath(1,:)=[5 4 6 7 12 13 15 16 17 64 63 111 58 107 59 60 96 56 50 51 52 47 46 0 0 0 0 0 0 0];
TenPath(2,:)=[5 4 6 7 12 13 15 16 17 98 26 25 27 106 28 42 104 92 41 108 44 105 93 49 48 116 47 46 0 0];
TenPath(3,:)=[5 4 6 7 8 9 14 13 15 16 17 64 63 111 58 107 59 60 96 56 50 51 52 47 46 0 0 0 0 0];
TenPath(4,:)=[5 4 6 7 12 13 67 66 70 69 68 71 61 95 59 60 96 56 50 51 52 47 46 0 0 0 0 0 0 0];
TenPath(5,:)=[5 4 6 7 8 9 14 13 15 16 17 98 26 25 27 106 28 42 104 92 41 108 44 105 93 49 48 116 47 46];
TenPath(6,:)=[5 4 6 7 8 9 14 13 67 66 70 69 68 71 61 95 59 60 96 56 50 51 52 47 46 0 0 0 0 0];
TenPath(7,:)=[5 4 6 7 12 13 15 16 17 64 63 111 58 97 62 61 95 59 60 96 56 50 51 52 47 46 0 0 0 0];
TenPath(8,:)=[5 4 6 7 12 13 67 66 70 69 68 71 61 62 97 58 107 59 60 96 56 50 51 52 47 46 0 0 0 0];
TenPath(9,:)=[5 4 6 7 8 9 14 13 15 16 17 64 63 111 58 97 62 61 95 59 60 96 56 50 51 52 47 46 0 0];
TenPath(10,:)=[5 4 6 7 8 9 14 13 67 66 70 69 68 71 61 62 97 58 107 59 60 96 56 50 51 52 47 46 0 0];

NoL=length(E(:,1));
NoP=10;
NoN=length(F(:,1));
MaxTranslated=max(F(:,1));
IDTranslated=zeros(MaxTranslated,1);


for j=1:NoN
    NodeID(j)=F(j,1);
    IDTranslated(NodeID(j))=j;
end


%Determine Linklength
for i=1:NoL
    LinkSpeed(i)=E(i,7);
    LinkLength(i)=E(i,4);
    LinkStart(i)=E(i,2);
    LinkEnd(i)=E(i,3);
end


for h=1:NoL
    LinkStartMatlab(h)=IDTranslated(LinkStart(h));
    LinkEndMatlab(h)=IDTranslated(LinkEnd(h));
end


%Determine the links of the paths
LinksPath=zeros(NoP,NoL);
for j=1:length(Path1)-1
    for i=1:NoL
        for h=1:NoP
            if LinkStartMatlab(i)==TenPath(h,j)
              if LinkEndMatlab(i)==TenPath(h,j+1)
                LinksPath(h,i)=1;
              end
            end
        end
        end
    end
end
```

```
%Determine Lij and Nan
Lij=zeros(NoP);
PathLength=zeros(NoP,1);
Nan=zeros(NoL,1);
for i=1:NoP
    for j=1:NoP
        for k=1:NoL
            if LinksPath(i,k)==LinksPath(j,k)
              if LinksPath(i,k) ==1
                 Lij(i,j)=Lij(i,j)+LinkLength(k);
              end
            end
        end
    end
end
for j=1:NoP
    for i=1:NoL
        PathLength(j)=PathLength(j)+LinkLength(i)*LinksPath(j,i);
    end
end
for j=1:NoP
    for i=1:NoL
        Nan(i)=Nan(i)+LinksPath(j,i);
    end
end


% Calculate the CF
for i=1:NoP
    hulp(1,i)=0;
    hulp(2,i)=0;
    hulp(3,i)=0;
    hulp(4,i)=0;
    hulp(5,i)=0;
    for j=1:NoP
        hulp(1,i)=hulp(1,i)+(Lij(i,j)/(sqrt(PathLength(i))*sqrt(PathLength(j))))^gamma; % sum lij/sqrt(li lj)
    end
    for j=1:NoL
        if LinksPath(i,j)==1
          hulp(2,i) = hulp(2,i)+ LinkLength(j)/PathLength(i)*Nan(j);
          hulp(3,i)= hulp(3,i)+LinkLength(j)/PathLength(i)*log(Nan(j));
          hulp(5,i)= hulp(5,i)+ LinkLength(j)/PathLength(i)*1/Nan(j);
        end
    end
    CF(1,i)=-beta *log(hulp(1,i));
    CF(2,i)=-beta*log(hulp(2,i));
    CF(3,i)=-beta*hulp(3,i);
    for j=1:NoP
        if j ~=i
          hulp(4,i)=hulp(4,i)+(Lij(i,j)/(sqrt(PathLength(i)*PathLength(j))))*((PathLength(i)-Lij(i,j))/
                    (PathLength(j)-Lij(i,j)));
        end
    end
    CF(4,i)=-beta*log(1+hulp(4,i));
    CF(5,i)=beta*log (hulp(5,i));
end


A=dlmread('Nodes.txt');
NoN=length(A(:,1));
for j=1:NoN
```

```
    NodeID(j)=A(j,1);
end
B=dlmread('DynaNodeParams.txt');


FFTT=[3.9673 4.9529 4.3160 4.0075 5.3016 4.3562 4.4427 4.2889 4.7914 4.5376];


for i=1:30
    for j=1:10
        PathMatlab(j,i)=TenPath(j,i);
    end
end


PathChoice=zeros(10,60);
for i=1:30
    for j=1:10
        if PathMatlab(j,i)>0
           PathQuestor(j,i)=NodeID(PathMatlab(j,i));
        else
           PathQuestor(j,i)=0;
        end
        for h=1:NoN
            if PathQuestor(j,i)==B(h,1)
               PathChoice(j,i)=B(h,2);
            end
        end
    end
end


    %Determine average linktime for every quarter
D=dlmread('output_td_linktraveltime1.dat');
NoL=243;


period=[1 8; 8 15; 16 23; 23 30; 31 38; 38 45; 46 53; 53 60];
AverageLinkTime=zeros(8,NoL);
for h=1:8
    for i=period(h,1):period(h,2)
        for j=1:NoL
            AverageLinkTime(h,j)=AverageLinkTime(h,j)+D(6+j+(i-1)*249,5)/8;
        end
    end
end


%Determine travel time for the ten paths
for i=1:NoL
    LinkEnd(i)=D(6+i,3);
    LinkStart(i)=D(6+i,2);
end
TT=zeros(8,10);
for h=1:8
    for i=1:NoP
        aa=find(PathChoice(i,:),1,'last');
        for j=1:aa-1
            for g=1:NoL
                if PathChoice(i,j)==LinkStart(g)
                   if PathChoice(i,j+1)==LinkEnd(g)
                      TT(h,i)=TT(h,i)+AverageLinkTime(h,g);
```

```
                end
            end
        end
    end
end
end

%Determine routeshare for the first quarter
sumexponent(1)=0;
for i=1:NoP
    exponent(1,i)=exp(theta*(-FFTT(i)^power+CF(cf,i)));
    for j=1:length(PenRoute)
        if i==PenRoute(j)
            exponent(1,i)=exp(theta*(-FFTT(i)^power+CF(cf,i))-penvalue);
        end
    end
    for j=1:length(FavourRoute)
        if i==FavourRoute(j)
            exponent(1,i)=exp(theta*(-FFTT(i)^power+CF(cf,i))+penvalue);
        end
    end
    sumexponent(1)=sumexponent(1)+exponent(1,i);
end
%And for the other quarters
for h=1:7
    sumexponent(h+1)=0;
    for i=1:NoP
        exponent(h+1,i)=exp(theta*(-TT(h,i)^power+CF(cf,i)));
        for j=1:length(PenRoute)
            if i==PenRoute(j)
                exponent(h+1,i)=exp(theta*(-TT(h,i)^power+CF(cf,i))-penvalue);
            end
        end
        for j=1:length(FavourRoute)
            if i==FavourRoute(j)
                exponent(h+1,i)=exp(theta*(-TT(h,i)^power+CF(cf,i))+penvalue);
            end
        end
        sumexponent(h+1)=sumexponent(h+1)+exponent(h+1,i);
    end
end
for h=1:8
    for i=1:NoP
        P(h,i)=exponent(h,i)/sumexponent(h);
    end
end
```

A.5 Enschede test on 10 OD-pairs

In the C-logit variants we average the travel times over the iterations, in the congestion game variants we average the travel times and the route choice probabilities and in the smooth fictitious play variants we average the perceptions of the travel times and the route choice probabilities. Therefore we use different m-files for the C-logit, congestion game, and smooth fictitious play variants. This section describes the total m-file of the C-logit variants, the congestion game specific part of the congestion game m-file, and the smooth fictitious play specific part of the smooth fictitious play m-file.

A.5.1 C-logit variants

%This file calculate the route choice probability for every quarter of the first two hours

```
    %Initializing variables
CFChoice=4;
beta=5;
theta=1;
penvalue=0;
PenRoute=[];
FavourRoute=[];
power=1;


CF=xlsread('CommonalityFactor.xls'); % See the previous section for calculation of the commonality factors
NoP=5; %Number of paths
E=dlmread('Nodes.txt');
NoN=length(E(:,1));
for j=1:NoN
    NodeID(j)=E(j,1);
end
B=dlmread('DynaNodeParams.txt');


    FFTT=[3.4249 2.9841 3.4595 4.3277 3.4985 2.7317 3.0861 3.0663 3.7375 3.9189 3.8286 3.5196 4.1830 4.1632
4.0103 1.7232 2.6260 2.2284 3.1000 2.8026 2.0358 2.2172 2.6316 3.4126 3.5344 1.9966 2.9822 2.4720 3.3734 3.4940
2.5093 2.5605 3.6797 3.0359 4.1245 2.7409 2.3001 2.7755 3.9723 4.1537 4.5218 4.0810 4.2627 4.5564 5.4246 3.9673
4.9529 4.3160 4.0075 5.3016];


%The paths of the choiceset
C=xlsread('ChoiceSet.xls');
for i=1:10
    for j=1:5
        PathsChoice((i-1)*5+j,:)=C((i-1)*6+j,:);
    end
end


for i=1:34
    for j=1:50
        PathMatlab(j,i)=PathsChoice(j,i);
    end
end


PathChoice=zeros(10,60);
for i=1:34
    for j=1:50
        if PathMatlab(j,i)>0
           PathQuestor(j,i)=NodeID(PathMatlab(j,i));
        else
           PathQuestor(j,i)=0;
        end
        for h=1:NoN
           if PathQuestor(j,i)==B(h,1)
              PathChoice(j,i)=B(h,2);
           end
        end
    end
end


    %Determine average linktime for every quarter
D1=dlmread('output_td_linktraveltime1.dat');
D2=dlmread('output_td_linktraveltime2.dat');
D3=dlmread('output_td_linktraveltime3.dat');
```

```
D4=dlmread('output_td_linktraveltime4.dat');
D5=dlmread('output_td_linktraveltime5.dat');
D6=dlmread('output_td_linktraveltime6.dat');
D7=dlmread('output_td_linktraveltime7.dat');
D8=dlmread('output_td_linktraveltime8.dat');
D9=dlmread('output_td_linktraveltime9.dat');
D10=dlmread('output_td_linktraveltime10.dat');
D=(D1+D2+D3+D4+D5+D6+D7+D8+D9+D10)/3;
NoL=243;


period=[1 8; 8 15; 16 23; 23 30; 31 38; 38 45; 46 53; 53 60];
AverageLinkTime=zeros(8,NoL);
for h=1:8
    for i=period(h,1):period(h,2)
        for j=1:NoL
            AverageLinkTime(h,j)=AverageLinkTime(h,j)+D(6+j+(i-1)*249,5)/8;
        end
    end
end


%Determine travel time for the paths of the choice set
for i=1:NoL
    LinkEnd(i)=D(6+i,3);
    LinkStart(i)=D(6+i,2);
end

TT=zeros(8,50);
for h=1:8
    for i=1:NoP*10
        aa=find(PathChoice(i,:),1,'last');
        for j=1:aa-1
            for g=1:NoL
                if PathChoice(i,j)==LinkStart(g)
                  if PathChoice(i,j+1)==LinkEnd(g)
                    TT(h,i)=TT(h,i)+AverageLinkTime(h,g);
                  end
                end
            end
        end
    end
end


%Determine routeshare for the first quarter
for k=1:10
    sumexponent(1,k)=0;
    for i=1:NoP
        exponent(1,(k-1)*5+i)=exp(-theta*(FFTT((k-1)*5+i)^power-beta*CF(CFChoice,(k-1)*5+i)));
        for j=1:length(PenRoute)
            if i==PenRoute(j)
              exponent(1,(k-1)*5+i)=exp(-theta*(FFTT((k-1)*5+i)^power-beta*CF(CFChoice,(k-1)*5+i))-penvalue);
            end
        end
        for j=1:length(FavourRoute)
            if i==FavourRoute(j)
              exponent(1,(k-1)*5+i)=exp(-theta*(FFTT((k-1)*5+i)^power-beta*CF(CFChoice,(k-1)*5+i))+penvalue);
            end
        end
        sumexponent(1,k)=sumexponent(1,k)+exponent(1,(k-1)*5+i);
    end
```

```
end


% Determine routeshare for the other quarters
for h=1:7
    for k=1:10
        sumexponent(h+1,k)=0;
        for i=1:NoP
            exponent(h+1,(k-1)*5+i)=exp(-theta*(TT(h,(k-1)*5+i)^power-beta*CF(CFChoice,(k-1)*5+i)));
            for j=1:length(PenRoute)
                if i==PenRoute(j)
                   exponent(h+1,(k-1)*5+i)=exp(-theta*(TT(h,(k-1)*5+i)^power-beta*CF(CFChoice,(k-1)*5+i))-penvalue);
                end
            end
            for j=1:length(FavourRoute)
                if i==FavourRoute(j)
                   exponent(h+1,(k-1)*5+i)=exp(-theta*(TT(h,(k-1)*5+i)^power-beta*CF(CFChoice,(k-1)*5+i))+penvalue);
                end
            end
        sumexponent(h+1,k)=sumexponent(h+1,k)+exponent(h+1,(k-1)*5+i);
        end
    end
end
for h=1:8
    for k=1:10
        for i=1:NoP
            P(h,(k-1)*5+i)=exponent(h,(k-1)*5+i)/sumexponent(h,k);
        end
    end
end


    %Determine all chosen paths
A1=dlmread('Path_hour12.dat'); A2=dlmread('Time_hour12.dat');
NoN=length(E(:,1));
MaxTranslated=max(E(:,1));
IDTranslated=zeros(MaxTranslated,1);

a=14465; %Amount of vehicles of hour 1 and 2

ZoneNode=zeros(24,1);
ZoneNode(1,1)=93; ZoneNode(2,1)=97; ZoneNode(3,1)=122; ZoneNode(4,1)=48; ZoneNode(5,1)=44;
ZoneNode(6,1)=109; ZoneNode(7,1)=112; ZoneNode(8,1)=34; ZoneNode(9,1)=117; ZoneNode(10,1)=16;
ZoneNode(11,1)=66; ZoneNode(12,1)=103; ZoneNode(13,1)=101; ZoneNode(14,1)=81; ZoneNode(15,1)=57;
ZoneNode(16,1)=54; ZoneNode(17,1)=45; ZoneNode(18,1)=8; ZoneNode(19,1)=12; ZoneNode(20,1)=83;
ZoneNode(21,1)=5; ZoneNode(22,1)=85; ZoneNode(23,1)=68; ZoneNode(24,1)=123;


TimeType=A2;
Paths1=A1;
forj=1:length(remember)
   Paths1(remember(j)-j+1,:)=[];
   TimeType(remember(j)-j+1,:)=[];
end


%Sort paths and vehicles on departure time
TimeType id=sortrows(TimeType,1);
for i=1:length(Paths1(:,1))
    Paths(i,:)=Paths1(id(i),:);
end
```

```
ChoiceCount=zeros(8,50);
%Assign vehicles of the ten OD pairs to the route according to P
for i=1:length(Paths(:,1))
    if Paths(i,1)==85
      last=find(Paths(i,:),1 ,'last');
      if Paths(i,last)==109
        for time=1:8
          if TimeType(i,1)<15*time
            if TimeType(i,1)>=15*(time-1)
              x=rand;
              if x<P(time,1)
                Paths(i,:)=PathChoice(1,:);
                ChoiceCount(time,1)=ChoiceCount(time,1)+1;
              else
                if x<P(time,1)+P(time,2)
                  Paths(i,:)=PathChoice(2,:);
                  ChoiceCount(time,2)=ChoiceCount(time,2)+1;
                else
                  if x<P(time,1)+P(time,2)+P(time,3)
                    Paths(i,:)=PathChoice(3,:);
                    ChoiceCount(time,3)=ChoiceCount(time,3)+1;
                  else
                    if x<P(time,1)+P(time,2)+P(time,3)+P(time,4)
                      Paths(i,:)=PathChoice(4,:);
                      ChoiceCount(time,4)=ChoiceCount(time,4)+1;
                    else
                      Paths(i,:)=PathChoice(5,:);
                      ChoiceCount(time,5)=ChoiceCount(time,5)+1;
                    end
                  end
                end
              end
            end
          end
        end
      ...
      ...
      ...
      if Paths(i,1)==5
        last=find(Paths(i,:),1 ,'last');
        if Paths(i,last)==48
          for time=1:8
            if TimeType(i,1)<15*time
              if TimeType(i,1)>=15*(time-1)
                x=rand;
                if x<P(time,46)
                  Paths(i,:)=PathChoice(46,:);
                  ChoiceCount(time,46)=ChoiceCount(time,46)+1;
                else
                  if x<P(time,46)+P(time,47)
                    Paths(i,:)=PathChoice(47,:);
                    ChoiceCount(time,47)=ChoiceCount(time,47)+1;
                  else
                    ifx<P(time,46)+P(time,47)+P(time,48)
                      Paths(i,:)=PathChoice(48,:);
                      ChoiceCount(time,48)=ChoiceCount(time,48)+1;
                    else
                      if x<P(time,46)+P(time,47)+P(time,48)+P(time,49)
                        Paths(i,:)=PathChoice(49,:);
                        ChoiceCount(time,49)=ChoiceCount(time,49)+1;
                      else
```

```
                        (i,:)=PathChoice(50,:);
                        ChoiceCount(time,50)=ChoiceCount(time,50)+1;
                    end
                end
            end
        end
    end
end
end
end
end
end
end
end

% Write paths and vehicles in file
count=1;
for i=1:length(Paths(:,1))
    Vehicle(2*i-1,1)=count;
    Vehicle(2*i-1,2)=Paths(i,1);
    Vehicle(2*i-1,3)=Paths(i,2);
    Vehicle(2*i-1,4)=TimeType(i,1);
    Vehicle(2*i-1,5)=1;
    if TimeType(i,2)==1;
      Vehicle(2*i-1,6)=1;
    else
      Vehicle(2*i-1,6)=2;
    end
    Vehicle(2*i-1,7)=1;
    NoN(i)=find(Paths(i,:),1 ,'last');
    Vehicle(2*i-1,8)=NoN(i);
    Vehicle(2*i-1,9)=1;
    Vehicle(2*i-1,10)=0;
    Vehicle(2*i-1,11)=0.0000;
    Vehicle(2*i-1,12)=0.0000;
    for j=1:24
      if Paths(i,1)==ZoneNode(j)
        origin(i)=j;
      end
      if Paths(i,NoN(i))==ZoneNode(j)
        destination(i)=j;
      end
    end
    Vehicle(2*i-1,13)=origin(i);
    Vehicle(2*i,1)=destination(i);
    Vehicle(2*i,2)=0;
    count=count+1;
end

    regelveh1=' %d %d # of vehicles in the file, Max # of stops\n';
tekstveh1=[NoV 1];
regelveh2=' # usec dsec stime usrcls vehtype ioc #ONode #IntDe info ribf comp OZ\n';

fid=fopen('vehicle.dat','at');
fprintf(fid, regelveh1, tekstveh1);
fprintf(fid, regelveh2);
fclose(fid);
regelveh20=' %d %6.2f\n';
```

```
for j=1:NoV %write file for all vehicles
    regel='een twee drie vier vijf zes zeven acht negen tien elf twaalf derten virten \n';
    regelveh='een twee drie vier vijf zes zeven acht negen tien %1.4f %1.4f %d\n';
    if Paths(j,1)>9
      regel1=regexprep(regel,'een', '  %d');
    else
      if Paths(j,1)==0
        regel1=regexprep(regel,'een', '');
      else
        regel1=regexprep(regel,'een', '   %d');
      end
    end
    if Paths(j,2)>9
      regel2=regexprep(regel1,'twee', '  %d');
    else
      if Paths(j,2)==0
        regel2=regexprep(regel1,'twee','');
      else
        regel2=regexprep(regel1,'twee', '   %d');
      end
    end
    if Paths(j,3)>9
      regel3=regexprep(regel2,'drie', '  %d');
    else
..
..
    if Paths(j,14)>9
      regel14=regexprep(regel60,'virten', '  %d');
    else
      if Paths(j,14)==0
        regel14=regexprep(regel13,'virten','');
      else
        regel14=regexprep(regel13,'virten', '   %d');
      end
    end
    if Vehicle(j*2+1,1)>9999
      regelveh1=regexprep(regelveh, 'een', ' %d');
    else
      if Vehicle(j*2+1,1)>999
        regelveh1=regexprep(regelveh, 'een', '  %d');
      else
        if Vehicle(j*2+1,1)>99
          regelveh1=regexprep(regelveh, 'een', '  %d');
        else
          ifVehicle(j*2+1,1)>9
            regelveh1=regexprep(regelveh, 'een', '   %d');
          else
            regelveh1=regexprep(regelveh, 'een', '    %d');
          end
        end
      end
    end
    if Vehicle(j*2+1,2)>9
      regelveh2=regexprep(regelveh1, 'twee', '   %d');
    else
      regelveh2=regexprep(regelveh1, 'twee', '    %d');
    end
    ...
    ...
    if Vehicle(j*2+1,9)>9
      regelveh9=regexprep(regelveh8, 'negen', '   %d');
```

```
    else
      regelveh9=regexprep(regelveh8, 'negen', '    %d');
    end
    if Vehicle(j*2+1,10)>9
      regelveh10=regexprep(regelveh9, 'tien', '   %d');
    else
      regelveh10=regexprep(regelveh9, 'tien', '    %d');
    end

    tekst=[];
    for i=1:14
      if Paths(j,i) =0
        tekst=[tekst Paths(j,i)];
      else
        break
      end
    end
    tekst2=[];
    for i=1:2
      tekst2=[tekst2 Vehicle(j*2+2,i)];
    end
    fid=fopen('path.dat','at');
    fprintf(fid, regel60, tekst);
    fclose(fid);
    fid=fopen('vehicle.dat', 'at');
    fprintf(fid, regelveh10, Vehicle(j*2+1,:));
    fprintf(fid, regelveh20, tekst2);
    fclose(fid);
end
```

A.5.2 Congestion game variants

```
%Initializing variables
combi=0.25; % distance fraction
NoP=5;
n=2; %The number of the iteration

% Determine utilities for every OD pair
for i=1:5
    OD1(1,i)=(1-combi)*FFTT(i)+PathLength(i)*combi;
    OD2(1,i)=(1-combi)*FFTT(i+5)+PathLength(i+5)*combi;
    OD3(1,i)=(1-combi)*FFTT(i+10)+PathLength(i+10)*combi;
    OD4(1,i)=(1-combi)*FFTT(i+15)+PathLength(i+15)*combi;
    OD5(1,i)=(1-combi)*FFTT(i+20)+PathLength(i+20)*combi;
    OD6(1,i)=(1-combi)*FFTT(i+25)+PathLength(i+25)*combi;
    OD7(1,i)=(1-combi)*FFTT(i+30)+PathLength(i+30)*combi;
    OD8(1,i)=(1-combi)*FFTT(i+35)+PathLength(i+35)*combi;
    OD9(1,i)=(1-combi)*FFTT(i+40)+PathLength(i+40)*combi;
    OD10(1,i)=(1-combi)*FFTT(i+45)+PathLength(i+45)*combi;
end
for j=2:8
    for i=1:5
        OD1(j,i)=(1-combi)*TT(j-1,i)+PathLength(i)*combi;
        OD2(j,i)=(1-combi)*TT(j-1,i+5)+PathLength(i+5)*combi;
        OD3(j,i)=(1-combi)*TT(j-1,i+10)+PathLength(i+10)*combi;
        OD4(j,i)=(1-combi)*TT(j-1,i+15)+PathLength(i+15)*combi;
        OD5(j,i)=(1-combi)*TT(j-1,i+20)+PathLength(i+20)*combi;
        OD6(j,i)=(1-combi)*TT(j-1,i+25)+PathLength(i+25)*combi;
        OD7(j,i)=(1-combi)*TT(j-1,i+30)+PathLength(i+30)*combi;
        OD8(j,i)=(1-combi)*TT(j-1,i+35)+PathLength(i+35)*combi;
        OD9(j,i)=(1-combi)*TT(j-1,i+40)+PathLength(i+40)*combi;
```

```
        OD10(j,i)=(1-combi)*TT(j-1,i+45)+PathLength(i+45)*combi;
    end
end

% Determine route choice probabilities
Phulp=zeros(8,50);
for h=1:8
    [c1 i1]=min(OD1(h,:));
    [c2 i2]=min(OD2(h,:));
    [c3 i3]=min(OD3(h,:));
    [c4 i4]=min(OD4(h,:));
    [c5 i5]=min(OD5(h,:));
    [c6 i6]=min(OD6(h,:));
    [c7 i7]=min(OD7(h,:));
    [c8 i8]=min(OD8(h,:));
    [c9 i9]=min(OD9(h,:));
    [c10 i10]=min(OD10(h,:));
    Phulp(h,i1)=1;
    Phulp(h,i2+5)=1;
    Phulp(h,i3+10)=1;
    Phulp(h,i4+15)=1;
    Phulp(h,i5+20)=1;
    Phulp(h,i6+25)=1;
    Phulp(h,i7+30)=1;
    Phulp(h,i8+35)=1;
    Phulp(h,i9+40)=1;
    Phulp(h,i10+45)=1;
end
POld=dlmread('Probabilities.dat');
for j=1:50
    for i=1:8
        P(i,j)=Phulp(i,j)/n+POld(i,j)*(n-1)/n;
    end
end
dlmwrite('Probabilities.dat',P);
```

A.5.3 Smooth fictitious play variants

```
%Initializing variables
NoV=317; % Number of vehicles between the 10 OD in the first 2 hours


% Determine the perception of the vehicles
PerceptionOld=dlmread('Perception.dat');
ChosenRoute=dlmread('ChosenRoute.dat');
DepartInterval=dlmread('DepartInterval.dat');
Perception=PerceptionOld;
for j=1:NoV
    for i=1:8
        if DepartInterval(j)==i
          Perception(j,ChosenRoute(j))=PerceptionOld(j,ChosenRoute(j))*(n-1)/n+TT(i,ChosenRoute(j))*1/n;
        end
    end
end

dlmwrite('Perception.dat', Perception);


%Determine route choices
for h=1:NoV
    sumexponent(h)=0;
```

```
    for k=1:10
        if ChosenRoute(h)<6
          forі=1:NoP
              exponent(h,i)=exp(-theta*(Perception(h,i)^power-beta*CF(CFChoice,i)));
              forj=1:length(PenRoute)
                 ifі==PenRoute(j)
                   exponent(h,i)=exp(-theta*(Perception(h,i)^power-beta*CF(CFChoice,i))-penvalue);
                 end
              end
              for j=1:length(FavourRoute)
                 if i==FavourRoute(j)
                   exponent(h,i)=exp(-theta*(Perception(h,i)^power-beta*CF(CFChoice,i))+penvalue);
                 end
              end
          end
          ...
          ...
          ...
        else
          if ChosenRoute(h)<46
              for i=1:NoP
                 exponent(h,i)=exp(-theta*(Perception(h,i+40)^power-beta*CF(CFChoice,i+40)));
                 for j=1:length(PenRoute)
                   if i+40==PenRoute(j)
                      exponent(h,i)=exp(-theta*(Perception(h,i+40)^power-beta*CF(CFChoice,i+40))-penvalue
                   end
                 end
                 for j=1:length(FavourRoute)
                   if i+40==FavourRoute(j)
                      exponent(h,i)=exp(-theta*(Perception(h,i+40)^power-beta*CF(CFChoice,i+40))+penvalue);
                   end
                 end
              end
          else
              for i=1:NoP
                 exponent(h,i)=exp(-theta*(Perception(h,i+45)^power-beta*CF(CFChoice,i+45)));
                 for j=1:length(PenRoute)
                   if i+45==PenRoute(j)
                      exponent(h,i)=exp(-theta*(Perception(h,i+45)^power-beta*CF(CFChoice,i+45))-penvalue);
                   end
                 end
                 for j=1:length(FavourRoute)
                   if i+45==FavourRoute(j)
                      exponent(h,i)=exp(-theta*(Perception(h,i+45)^power-beta*CF(CFChoice,i+45))+penvalue);
                   end
                 end
              end
          end
        end
    end
    for j=1:5
        sumexponent(h)=sumexponent(h)+exponent(h,j);
    end
end

for i=1:NoV
    for j=1:5
        P(i,j)=exponent(i,j)/sumexponent(i);
    end
end
```

Appendix B

Dynamic assignment demon player game

In the demon player game every link failure probability is set one for one on one, while the other link failure probabilities are set on 0. The travel times are considered for every link failure scenario and the demon selects the link that causes the highest social cost when his link failure probability is put on 1. To simulate this process in a dynamic assignment, we have to build for a network with n links $n + 1$ networks. In one network the network is build with the normal link capacities, in the other n networks exactly 1 link is damaged and has the damaged capacity. Remember that all links have to have the same lengths in the demon player game (see Section 5.2.4). If we would divide the links in smaller links, sucht that we have links of equal length we would need 42 links in our network, which would result in 43 different networks!
However dividing of links in smaller links is not without consequences for the dynamic assignment. Dynasmart determines every 5 seconds the intensity on a link. The determined intensity is constant on the total length of a link within the 5 seconds interval. If the inflow on a link reaches capacity or if the inflow is higher than the capacity, than the a delay is suffered on this link. Suppose that route 1 is just one long link and the demand on link 1 is higher than the capacity, than the traffic suffers a delay on the total length of this link. Now suppose that route 1 consists of several smaller links, the length of the route, the road capacity and the traffic demand remains the same. Now the traffic receives a delay on the first link, this link forms a bottleneck such that the outflow equals the capacity of this link. The next links will cause a much smaller delay because they get an inflow which does not exceeds their capacity. The delay on the first link is not very high, because this link is not that long. Eventually the total delay is much higher on the network with one link than on the network with several smaller links. Therefore dividing the links in smaller links will influence our results in a dynamic assignment. To have a more realistic image of the delays on the link we do not divide the links in smaller links in the network for Dynasmart, but we do divide the links in smaller links for the network in Matlab. The travel times are calculated in Dynasmart for the longer links, we translate this to travel times of the smaller links by the following formula:

$$TT_{Matlab}(i) = \sum_{j} TT_{Dynasmart}(j) * D(j,i) * \frac{LL_{Matlab}(i)}{LL_{Dynasmart}(j)}$$

Where $TT_{Matlab}(i)$ is the travel time of link i in the network of Matlab, $TT_{Dynasmart}(j)$ the travel time of link j in the network of Dynasmart, $LL_{Matlab}(i)$ the length of link i in the network of Matlab, $LL_{Dynasmart}(j)$ the length of link j in the network of Dynasmart, and D a matrix with zeros and ones. $D(j,i) = 1$, if link i is a part of link j. A link has to have the same capacity and speed over the total length, and there should not be any detractions on the link. We have 9 links in the network of the illogical routing test, see Figure B.1. Which would result in 10 networks for the demon player game.
Even if we could adjust the demon player game in Dynasmart such that we did not need to do 40 iteration, we still need to run 10 networks for this simple network. Imaging how many networks we have to run for a network of realistic size! In our research goal we have stated that we want to develop a route choice model that results in a more realistic traffic model, and that in combination with the assignment model calculates the routes in little computational time. With the current technical abilities it is not possible to use the demon player game in little computational time for the dynamic assignment. Therefore we do not test the demon player game in the dynamic assignment.

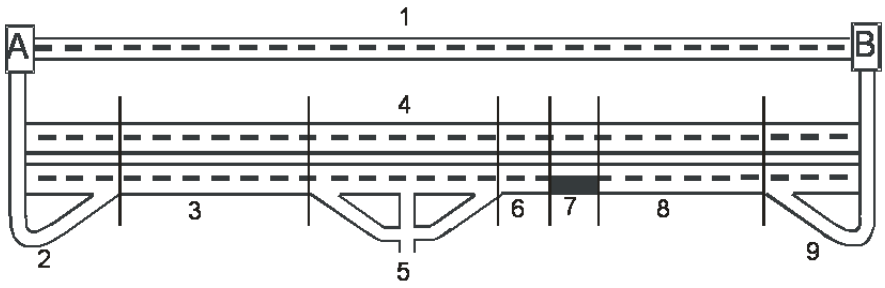


Figure B.1: The network of the logical routing test in 9 links