"A generic approach to a route guidance strategy"

Msc Thesis F.S. Zuurbier

May, 2005

TU Delft
Technische Universiteit Delft

OVV

"A generic approach to a route guidance strategy"

Msc Thesis F.S. Zuurbier
May, 2005

TU Delft
Technische Universiteit Delft

DHV

Extract

A generic approach to a route guidance strategy,
F.S. Zuurbier

The thesis presents a generic methodology able to generate optimal controlled dynamic prescriptive route guidance to be disseminated by means of variable message signs (VMS). The methodology is generic in the sense it can be used on any network topology, with any number of VMS's, for different scenario's (e.g. recurrent congestion or accidents), based on a flexible user defined objective function and will work as long as feasible route alternatives exist.

The methodology uses a new and for this purpose developed macroscopic model called DSMART (dynamic, 1st order, macroscopic, single user class, probit route choice, split vectors) and a customized parallel implemented evolutionary algorithm (EA). By using simulation data from the DSMART model in combination with customized mutation operators in the EA, VMS settings are generated in a smart way to increase convergence speed. A prototype of the methodology has been developed in Matlab and applied in a case study to the city of Rotterdam in the Netherlands by using a network representation (approx. 500 links) of the motorway with six different VMS's and the connected urban network. The case study illustrated the generic nature by successfully optimizing all three different scenarios ranging from the: everyday recurrent morning congestion, a simulated accident and an extreme event (football final match) by generating a dynamic set of VMS settings for all six instances. As a measure for optimization, the objective function was utilized to express *the generalized gross travel time* which led to a system optimal assignment. To implement the methodology, a rolling horizon control framework is suggested including a *crude scenario manager.* This suggested type of scenario management could be able to reduce the usually astronomical high number of scenarios needed in other systems, to a workable amount. Based upon expected efficiency increase when professionally implemented in the suggested rolling horizon implementation, it is expected that optimal controlled VMS settings could be calculated online for the case study network.

Keywords: DSMART, DTA model, evolutionary algorithm, prescriptive, route guidance, road side, optimal control, rolling horizon, scenario management, generic

A thesis submitted in part fulfillment of the regulations for the degree of Master of Science in Transportation and planning, TU Delft, June 8th 2005.

Commission:
Bliemer, Dr. MCJ
Chen, Dr. Y
Hoogendoorn, Dr. ir. SP
Horstmeier, Ir. THW
Wiggenraad, Ir. PBL
Zuylen, Prof. dr. HJ van

Table of contents

List of figures

List of tables

Preface

I would like to thank my parents for their everlasting and ongoing support during my college years, and for never having questioned[i] my study ways, which right about now, add up to nine years. I would like to thank Prof. van Zuylen for providing me with the challenging thesis subject I was looking for, and the interesting story[ii] which led me to read a great book. I would like to thank my commission members, Michiel and Serge in particular, for taking part in this project, and their input during the many brainstorm, sparring sessions and unscheduled meetings. I would like to thank DHV and Dr. Chen for offering me an internship at the Amersfoort office and all the freedom to work on this project, which proved a very enriching experience. And I would like to thank the many people who put up with me for the past 12 months and had to listen to me mumbling and rambling on about the ins and outs of route guidance, matlab, evolution, 3D animations, etc over and over again.

Although the project took longer than expected, I enjoyed the time I spent on it and learned a great deal about DTM and am looking forward to what is to come.

Frank Stendert Zuurbier
Delft 27 May, 2005

Abstract

This thesis describes *the development of a methodology, aimed at generically generating optimal dynamic route guidance, which can be disseminated by means of roadside DRIPS on any traffic network in the form of prescriptive advices.*

With *generic* we mean that the prescriptive guidance in the form of explicit route recommendations can be generated for *any given traffic network*, as long as it has some feasible re-routing alternatives and the size of the network is limited (e.g. 500 links). The generic nature of the methodology also implies that the DRIPS can be placed *anywhere* within the network, and the *optimal advices* are generated for *any given time*. Another feature of the generic nature is the ability of the methodology to generate optimal RG for different *scenarios* or *network conditions* without intricate formulated exceptions. As long as the desired real life network state can be modeled, it can be optimized.

With *optimal* we mean, by default, that the generated route guidance advices will result in *the closest to system optimal which can be achieved by the limited rerouting possibilities the roadside placed DRIPS offer*. The generation scheme is based on an *optimal control loop* and advices generated will *qualitatively* exceed *any other form of control* such as predictive, feedback or closed loop.

The apposition *by default* is added deliberately to stress the fact that the *system optimal assignment* is not the only possibility. The methodology allows a *custom objective function* to be formulated based on *ex-ante, running or ex post* variables which describe the traffic network in the simulation in *any given combination*. This feature allows *intricate objectives* to be formulated, which can satisfy the growing number of demands coming from different *societal* stakeholders or viewpoints such as policymakers, environmental concern, lobbyists, safety and security, network robustness etc.

How does it work?
For the methodology to work properly, a *calibrated traffic assignment* of the observed recurrent real life traffic phenomena in a DTA model must be available. This usually means the morning and evening peak are split into two different scenarios. In this network we first define *the position* and the number of desired DRIPS. Followed by defining for each DRIP, at maximum three, *groups of destination nodes* for which the DRIP could *possibly* provide useful *route guidance information* and estimate the *likeliness* for this to happen in order to speed up the process. When this is done, the *alternative routes* which *can be chosen* when passing a DRIP are explicitly stored in the route choice process.

This base situation, as sketched above, is now optimized by maximizing the value of the chosen objective function. This is done by dynamically *routing* those travelers, who pass a DRIP and head for a specific destination in one of the groups, via one of the explicit stated *alternative routes*.
This implies that all travelers heading for those destinations who are a member of the same group get the same routing advice from which they can benefit. Based on the *compliance* a certain percentage of those travelers will follow the advice or travel their original calibrated path.

The *simulation horizon* is divided into a *variable* number of periods (e.g. 12 periods of 15 minutes) and the algorithm calculates the *optimal distribution,* of travelers heading to each one of the three *destination groups,* over the *alternative routes* per *time period.* The optimization process is aimed at finding the *set of dynamic DRIP settings* which will result in a maximal value for the objective function at the end of the horizon, hence the name *optimal route guidance advices.*

The type of optimization heuristic chosen is a highly customized parallel implemented *evolutionary strategy* and chosen for a number of reasons. For one it is likely to find a *global optimum* due to its stochastic nature of the search process. In addition very dedicated *operators*[1] have been developed

[1] An operator is a function which changes part of a DRIP setting based on various variables, aimed at improving the objective function.

which use *network knowledge* and *simulation data* to effectively generate (mutate) DRIP settings. (An operator is a function which changes a partial DRIP setting according based on some specific variables) And third, this type of heuristic works very well in combination with offline scenario management.

The actual calculated desired distribution, when passing a DRIP, over the alternative routes, during *a time period*, is *not binary*, but will frequently become real valued. This can be easily translated into *prescriptive route guidance advices* based on the *length of the period* and the calculated *fraction*.

The evolutionary algorithm is based on a *cycle* where the quality of the proposed DRIP settings, expressed in the score value for the objective function, increases over time until at a certain point an optimal value is achieved and we can say that when disseminating those DRIP settings the resulting assignment will maximize the chosen objective function and lead to a (system) optimum.

Why is it useful?
This type of *dynamic traffic management* is a *soft management*[1] measure and can be implemented without any expected *societal resistance*. In current practice, DRIPS are *already installed* and are used to display *descriptive route information* based on the current network state. It is generally recognized that providing qualitatively better information, for example based on prediction, would lead to a better distribution of traffic. One of the difficulties in generating *qualitatively better* route *information* is in dealing with *consistency*. When providing users with *predicted information* about the *expected network* this information must be *consistent* with the actual *experienced network conditions*. If not, travelers will experience an inconsistency, which, if this is repeatedly the case, in the end leads to a loss in confidence of the system. Unlike predicting the weather, a prediction of the network state has influence on the future network state and this problem is considered very difficult to solve.

In this proposed methodology we do *not provide information* about the network, but provide users with an *explicit stated prescriptive route guidance advice*. An example of such is "heading for A take B" which is very different from a *descriptive route information* like "queue A via B 3 km, queue A via C 5 km".

Which exact type of guidance to provide in the real life situation can be viewed upon as a matter of ergonomics. When difficult complicated situations arise which differ from the recurrent conditions, *prescriptive* information is considered better. Yet, under *normal*, expected conditions, providing descriptive information is regarded best. A combination between providing *information* first, followed by an *explicitly stated route recommendation* (or vice versa) could very well be the most user friendly solution. This way the prescribed advice is justified to the traveler by means of (network) information, which makes him/her more likely to become compliant to the advice.

In this thesis, the choice was made to generate *prescriptive* information *only* and can largely be attributed to the desired *generic nature* of the methodology, and therefore its necessary possibility to deal with varying extreme situations. There are however additional reasons for providing *prescriptive information as well.* On the one hand, a traveler needs no more knowledge about the network to interpret the information given and decide which alternative he/she will take. The choice is simple and based on faith in the system, either *follow* the advice or *not*. On the other hand *prescriptive information* has less of a problem in dealing with consistency since there is no quantitative base for comparison, *for how could the traveler have known if the other alternative would have been shorter?* A third possible benefit could be the fact that traveler compliance to prescriptive information will be easier to estimate.

Why does it work?
The methodology has been applied in a *case study* performed on the Rotterdam motorway network in the Netherlands. In this study, a calibrated situation for a three hour morning peak period was created which formed the base situation to be optimized. The methodology has been applied to three

---

[1] There is no need to follow up on the information

scenarios: *normal morning peak congestion, an extreme event and an accident* and was successful at optimizing all three scenarios. This case study illustrated the methodology to be applicable to real life traffic networks and deal with different scenarios and still be successful at optimizing the traffic assignment.

In this thesis an implementation of the methodology in a *rolling horizon* framework, where *optimal prescriptive route guidance* can be calculated in an online environment, is also provided in 8.1.

Although the *technical realization* as described in this thesis would not be usable in such an online environment, due to long computation times. Based on the expected decrease in this time by the suggested alternative realization and improvements to the methodology it could be implemented far more efficient and indeed become applicable online. (Optimal DRIP settings for the next 45 minutes could be calculated for the Rotterdam situation in less than five minutes)

1 Introduction

1.1 A generic approach to a route guidance strategy

The original problem presented as the subject of this thesis was to: *develop a generic approach to a route guidance strategy*. Now, route guidance comes in many forms and can be divided into three main *forms of dissemination*: *in car, en–route (roadside) and pre trip.* In addition the *type* of the route guidance is usually split into *descriptive* guidance, providing information on travel times or queue lengths, and *prescriptive guidance* which provides explicit route recommendations to the traveler, and as a last *distinguishing* feature, the *nature* of the guidance in *time* can be static or dynamic.

This thesis focuses upon the *roadside dissemination* of route guidance by means of *dynamic route information panels (*DRIPS) because some opportunities lie ahead for improvement there. In the current practice in the Netherlands we see that DRIPS are being used to provide *dynamic descriptive information*, based on the *current network state* (feedback control).

In this thesis it is suggested to provide *prescriptive* guidance using DRIPS for a number of reasons. Prescriptive advices:

- can be used as a *dynamic traffic management instrument* and be put to use to steer toward a system optimal assignment of the network
- are easier to interpret since they require no forehand network knowledge.
- are *ergonomically* better suited to deal with complex situations which makes this type of advice more generic
- do not provide a *quantitative base* for comparison (which is the case between expected network conditions, resulting from providing descriptive guidance, and experienced network conditions when traveling further downstream) which makes consistency less of an issue
- could make the estimation of user compliance easier, since the only choice one has to make is follow or discard the provided route recommendation.

Modeling prescriptive guidance
Prescriptive guidance can be easily modeled in a dynamic traffic assignment (DTA) model when the *dynamic route choice process* is translated into *dynamic destination specific split fractions*. By adopting this approach, the problem of finding a *good route guidance strategy* is reduced to finding a set of optimal destination specific split fractions at the DRIP locations, which maximizes a certain objective function. The relation between destination specific split fraction and prescriptive route guidance is illustrated below.

Lets say we have a destination specific split fraction located at link *i*, for destination *d* during a period *p* (e.g. 10 minutes) which *splits* the traffic heading for a certain destination over two outgoing links in the following way $\psi_{i,d,p}\begin{bmatrix}0.4\\0.6\end{bmatrix}$. The network is modeled in such a way, the link is located on the position where a DRIP is and the two downstream connected links (over which the traffic is spread) both lead to an alternative route. We could translate this advice into a *prescriptive route recommendation* which is displayed for a total of four minutes pointing to the first connected link and a route recommendation which is displayed for a total of 6 minutes prescribing travelers to take the other link.

Objective function
As mentioned before, one of the benefits of *prescriptive guidance* is that it can be used to steer the assignment toward a system optimum. If we translate the *desired system optimum* into an objective function all we need to do is find a set of DRIP settings which maximizes this objective function.

The main theoretic problem however, is how to translate a *system optimum* into an objective function when considering a *dynamic network assignment*? The apposition *theoretic* was added deliberately because in all likeliness, when implementing a system for a road authority to generate route guidance other objectives could very well prevail over a pure system objective.
For example preventing the assignment of certain inner city links at full capacity, minimizing emissions, guarantying the availability of emergency services etc. With this in mind a very versatile objective function is created which expresses the assignment into a score value. The objective function is generic in the sense that it can use variables from the simulation, network and assignment process as well as post calculated variables. The used objective function in the case study is *the generalized gross travel times* which can deal with both congestion and hidden congestion (in the form of prevented/ diminished network access).

Custom build DTA model: DSMART
In order to work with the *destination specific split factions* and make use of certain running variables during simulation (cumulative number of arrivals and departures) to calculate the objective function, a custom dynamic traffic assignment (DTA) model was created, called DSMART. The model is macroscopic 1st order based on the triangular fundamental diagram and approximated by means of the Godunov approach and implemented in Matlab. The routes are calculated periodically by means of a *probit model*, based on the link travel times and translated into periodically changing destination specific split fractions. The model can very adequately deal with queue spillback, buildup and dissipation. Although only one *user class* is supported, traveler behavior in terms of compliance to the prescriptive route guidance is implemented and should provide enough assignment detail.

Optimal control & Rolling horizon framework
By adopting the above approach, where the route guidance generation problem is solved by maximizing a certain objective function, an optimal control approach is used. This is desirable over the current practice where feedback control is used since it allows the steering toward a system optimum. In addition, optimal control fits perfectly within the more generic *rolling horizon* control framework, which works by periodically calibrating the model to the current measured network state. By adopting this approach, the model needs only predict a short time in the future (horizon) and calculate optimal DRIP settings for this period.

Optimizing the DRIP settings using an evolutionary algorithm
The number of possible DRIP settings is based upon the number of DRIPS, the resolution of the split fraction, the number of periods used in the simulation and the number of connected alternative links for which a DRIP can provide advice. In the Rotterdam case study performed in this thesis, the number of possible alternative settings is in the order of 10e+280 (whereas there are estimated to be 10e+70 atoms in the universe). When optimizing DRIP settings in the rolling horizon approach, this number drops significantly but it serves to illustrate the complexity of the problem. To search this vast search space, it was chosen to work with an *evolutionary strategy*. This particular heuristic has some great features, namely: it supports a parallel implementation and the use of *strategic information* from the DTA model. In addition, it would support user intervention by means of *best guessed DRIP settings by an experienced traffic operator* which could greatly enhance the computation speed.

Case study for the Rotterdam network
The methodology is applied to the Rotterdam traffic network which has a typical ring way structure and enables alternative routing using six DRIPS. This model was derived from the *zuidvleugel model* (DHV) and gives a description of the motorway network and the underlying urban network (divided evenly in 50% motorway links and 50% city links). The network has about fifty origin and destination nodes and is calibrated to resemble the morning peak congestion.

Optimal DRIP settings calculated for three different scenarios
The methodology is applied to the Rotterdam situation for three different scenarios, namely: everyday morning congestion, an extreme event in the form of the UEFA finals at the football dome (soccer) and an accident on the ring way. The methodology was able to successfully optimize all three

scenarios by calculating a set of DRIP settings which increased the network performance significantly. These scenarios where optimized using the assumption that 100% of all users would comply to the explicit route recommendations merely to illustrate the methodology. If more realistic user compliance values are used, other DRIP settings would be evolved and the quantitative assignment results would perhaps become more shaded. It does not however change the methodology in any way.

Based on the case study some general conclusions regarding the developed methodology we could state that the approach chosen is generic in the sense it can be applied to any network under any conditions and calculate optimal controlled prescriptive route guidance settings. Although the needed convergence times where very long for this prototype, if implemented within the rolling horizon context, supported by a crude scenario database and technically implemented in a computational efficient way it is estimated that optimal DRIP scenario's could be calculated in less than five minutes, for the Rotterdam network that is. In addition the evolutionary algorithm, as it was implemented, has some great opportunities for improvement which are identified and listed in the recommendations section.

1.2 Reading map

This thesis deals with developing a methodology for generically generating route guidance for real life traffic networks, in this chapter we will discuss the different chapters and provide a *map* for the reader.

In chapter two we will discuss *route guidance and information* in general and put it in a *dynamic traffic management* context. This will serve to illustrate some of the benefits and challenges facing the development of this kind of strategy and illustrate the situation regarding route guidance in the Netherlands.

In chapter three the problem is formally stated in the *problem analysis* and the objective for this thesis is presented together with the most important research questions.

In chapter four, the result of the literature study is presented. This chapter discusses and reviews the *main building blocks* needed to develop a system for generating route guidance such as the control loop, dynamic traffic assignment models and various optimization techniques.

In chapter five the resulting methodology used to generate the route guidance is formalized. We will demonstrate how DRIP settings can be translated / simulated in the route choice process by means of *destination specific split fractions*. Furthermore this chapter will detail the exact specifications of the *dynamic traffic assignment model* and the type of *optimization technique* needed. It will become clear that a fast analytic macroscopic dynamic traffic assignment model is needed in conjunction with a genetic algorithm to search the enormous solution space. We will also discuss how the resulting *dynamic network assignment* can be evaluated by means of a scalar score using the *generalized total travel time*.

In chapter six we will discuss the foundation of the custom built analytic macroscopic dynamic traffic assignment model. We will explain the basic formula of the kinematic wave model, its ability to model traffic for multiple destinations and the route choice process.

In chapter seven the optimization process is detailed. We will show how the problem of finding a set of optimal DRIP settings can be solved by means of an *evolutionary algorithm*. The different steps within the generation process and its different customized operators will be thoroughly discussed together with the parallel implementation in a client-server structure.

In chapter eight the methodology is implemented within a *rolling horizon context* and the technical realization in Matlab is discussed together with the *standard settings* and a *step by step plan* describing how the optimization should be carried out.

In chapter nine the methodology is applied in a *case study* based on the (motorway) network for the city of Rotterdam (the Netherlands), by calculating optimal route guidance advices for the six DRIPS located there. We will show how the calculated dynamic route guidance advices are able to optimize the network assignment for *three different scenarios*, being: *the recurrent morning congestion*, *rerouting due to an incident* and *rerouting due to extreme demand*. We will discuss the results by comparing the assignment of the *calibrated scenario* with the *optimized scenario* by means of a graphical assignment plot and some assignment statistics.

In chapter ten the conclusions regarding the proposed methodology are presented. Especially the open nature of the proposed methodology, the fast calculation process as a result of the parallel implementation, the possibility to work with very complex (political, environmental, economic, or combined) objectives, the ability to generate guidance for different scenarios and the opportunities which are possible when combined with a DSS system are considered valuable.

2 Route guidance and Information systems in general

This chapter discusses *route guidance* and *information systems* (RGIS) in terms of the expected benefit of such systems and in what shapes and sizes they can be found. The next paragraph discusses the systems and their role in a *dynamic traffic management* context. Next we will use the situation in the Netherlands to detail the current application and identify some opportunities for improvement. The remainder of this chapter will be used to present the main challenges when developing a *generic approach to a route guidance strategy.*

Route guidance and information is the practice of informing a traveler about the route or *traffic network* conditions he or she can take or expect. Route guidance and information can be as simple as the information panels above the motorway stating the remaining distance to a certain destination or as complex as the *in car navigation kit* based on global positioning systems (GPS) which specifically prescribe the best route to take. Route guidance and information is aimed at *informing* a traveler about the network conditions or *prescribing* explicit route recommendations with the sole purpose to *aid in the route choice process* every traveler has to make during a trip.

2.1 Why can route guidance and information systems be beneficial?

In this paragraph we will briefly discuss why route guidance and information systems can be beneficial to the route choice process.

Factors that influence route choice typically relate to one of the four categories:

a. perceived available routes (cognitive map)
b. character of the traveler
c. the trip to be made
d. other circumstances

For simplicity we assume the choice to travel, destination, car modality and departure time have already been made. Remaining factors to the route choice process that fall mainly within the first category could be:

i. existence and awareness of alternative routes
ii. availability and conditions on alternative routes

In this case, RGIS systems are considered to be beneficial for a number of reasons:

- RGIS systems can be used to compensate for *inefficiency* in the route choice process which may be caused by differences between the user perceived network and the actual network conditions (category a).
    - Informing about the existence of alternative routes
    - Informing about the availability and conditions on alternative routes
- RGIS systems could be put to use to steer the network assignment toward a system optimum, which is desirable over a user equilibrium assignment by explicitly recommending routes

2.2 A common subdivision of RGIS

Route guidance and information systems (RGIS) is the general term for a wide variety of systems which somehow aid the traveler in his/her decision making regarding route choice in the broad sense. The collection of different systems can roughly be divided by means of the following criteria: *nature and topic of the guidance / information, behavior in time and the type of dissemination*.

The nature and topic of the information provided by RGIS is can sometimes be interpreted differently. In this thesis, a division is made into:

i. traffic information; general purpose descriptive information regarding the traffic or network in general
ii. route information; specific descriptive information regarding the condition of a route or alternative routes
iii. route guidance; prescriptive explicit route recommendations to a traveler to assist him/her in her route choice.

The *nature in time* of RGIS is frequently divided into:

i. static; Static RGIS provide unchangeable guidance or information regarding route or network. For example static detour routes for trucks carrying dangerous materials or the direction signs above the road indicating off ramps and/or remaining distance to a certain destination.
ii. dynamic; Dynamic RGIS provide time dependant information regarding the network or routes. For example a temporary detour when road maintenance is carried out or additional roadside information during large events or city parking tours displaying the number of vacant parking spots.
iii. real time; Real time RGIS provide information or guidance based on the measured network. E.g. dynamic route information panels (DRIPS)[1] placed over the road, carrying information about current queue length or travel times.

2.2.1 Three dominant systems

The *type of dissemination* is usually divided into three categories and this subdivision is frequently used when talking about RGIS in general.

i. pre trip; Pre trip information is consulted by the traveler before he/she starts a journey, or sometimes even before making a choice about the modality to use. E.g. internet, ceefax or radio/TV bulletins
ii. road side / en route; when information or guidance is provided road side or en route this is general information or guidance not specifically aimed at one single user but at groups of travelers
iii. in car; in car systems provide guidance or information specifically aimed at one traveler

> *Pre-trip* communication possibilities include the internet, phone services, mobile devices, television, and radio. If the road user has decided to complete the trip by car, he/she may continue to receive information or advice via appropriate *en-route* devices such as radio services (RDS-TMC), road-side variable message sings (VMS, or DRIP), or special in-car equipment, in order to make sensible routing decisions at bifurcation nodes of the network. While radio broadcasting services and VMS have been in use for more than 25 years (and their number is steadily increasing), individual route guidance systems employing *in-car devices* and two-way communication with control centers are in their infancy. At this point, it is appropriate to distinguish among two alternative policies (which in some cases may be combined) of providing en-route information versus explicit route recommendation. [3.pg 2058]

Especially the remark about distinguishing between two different policies is important to note. In this thesis when talking about *route guidance*, we mean providing *explicit route recommendations or advices.*

[1] DRIPS are also referred to as variable message signs (VMS), dynamic message signs (DMS), or sometimes even changeable message signs (CMS).

The effectiveness of the three different categories has been researched as well in a study [4] where the three systems where evaluated in a static simulation. In this research cut off levels are identified to the amount people equipped with an in car system or using pre trip; data beyond these levels the effect of providing information or guidance actually worsened the overall network conditions compared to the benchmark case where no drivers where equipped.

> "The results suggest that there could be an optimal level of subscription to driver information or market penetration[1]. Subscription to real-time driver information services is recommended, but a high level of subscription may result in extra delay. VMS's are not as effective as either *pretrip* information or *in-car* route guidance at their respective optimal levels of market penetration[2]," [4. pg 1]
>
> 1: Cut off levels of 60% for in car guidance and 45% for pre trip. Beyond these levels traffic conditions deteriorate in comparison to the benchmark case where no drivers where equipped.
> 2: Possible improvements of 14-15% for pre-trip systems, 16-25% for in-vehicle systems and 1-4% for VMS (the latter largely dependant upon expected rate of compliance) Improvement or deterioration was based upon average network speed.

Based on this research, the DRIPS as they are used nowadays are considered *not* to be very effective. Perhaps the idea of using the DRIPS in disseminating prescriptive route guidance advices such as explicit route recommendations might increase their effectiveness.

2.3 Traffic and demand management and RGIS

Traffic and transport are components of society, the economy and our culture. They have to be considered as part of a multi-layered system [1, pg 9]. The *transport market* is where transport demand and supply meet and the *traffic market* is where infrastructure and vehicles meet. Dynamic Traffic Management is the collection of all traffic control measures that allocate temporal and spatial utilization of infrastructures and vehicle fleets by means of dynamic signals. [2, pg 2]

RGIS are applied to both *demand management* (transport market) and *traffic management* as is illustrated in the diagram below:

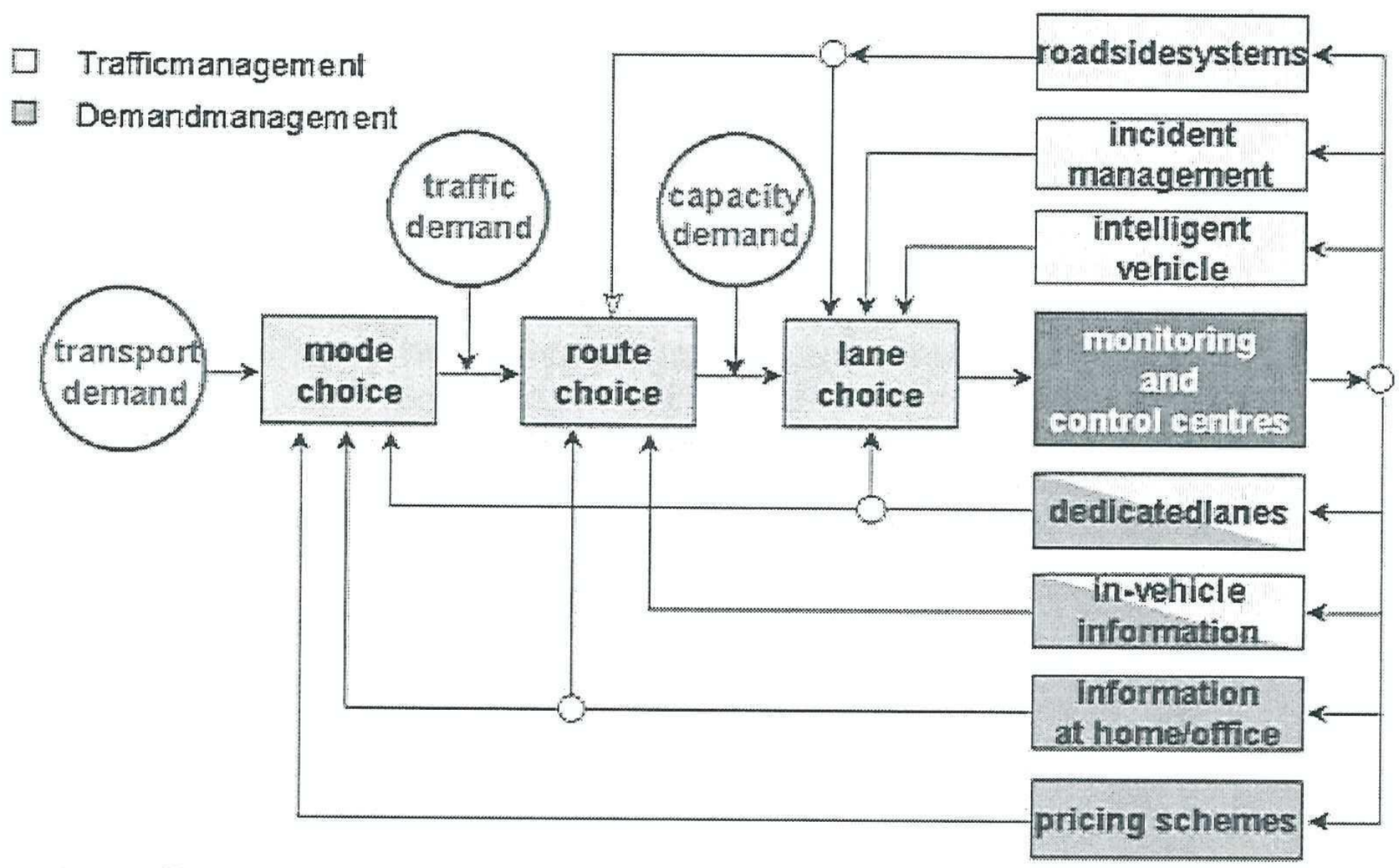


Figure 2-1 Traffic and demand management in relation to different RGIS

In the diagram we see how the different types of RGIS systems are applied to the transport and traffic market. The focus of this thesis is on *roadside systems*, falling within the dynamic *traffic management* context. Often dynamic traffic management is seen as a subset of ITS, *Intelligent Transport Systems*. In ITS a wider range of computer applications can be involved, for instance logistic systems, tracking and tracing of cargo, P&R management etc, we will discuss the ITS field briefly in the following paragraph.

2.3.1.1 ATIS and ATMS

Within the ITS framework, six major areas of application can be mentioned

1. Advanced Traffic Management systems (ATMS)
2. Advanced Traveler Information Systems (ATIS)
3. Advanced Vehicle control and safety systems (AVCSS)
4. ADAS, Advanced Driver Assistance Systems
5. Advanced public transportation systems (APTS)
6. Commercial vehicle operations (CVO)

When talking about RGIS, these fall within the first two areas of application: ATMS and ATIS in a similar way as they are applied to traffic demand and transport demand.

If we look at ATIS and ATMS from a management point of view, we can place them within the following relation to a traffic management center [21.].

Where maintaining and optimizing traffic conditions as a result of demand and supply is the goal and ATIS and ATMS are the means to this end.

Concerning RGIS, the *pre trip* systems in this context fall within the ATIS range of application, whereas the *roadside systems* can be placed within the ATMS range.

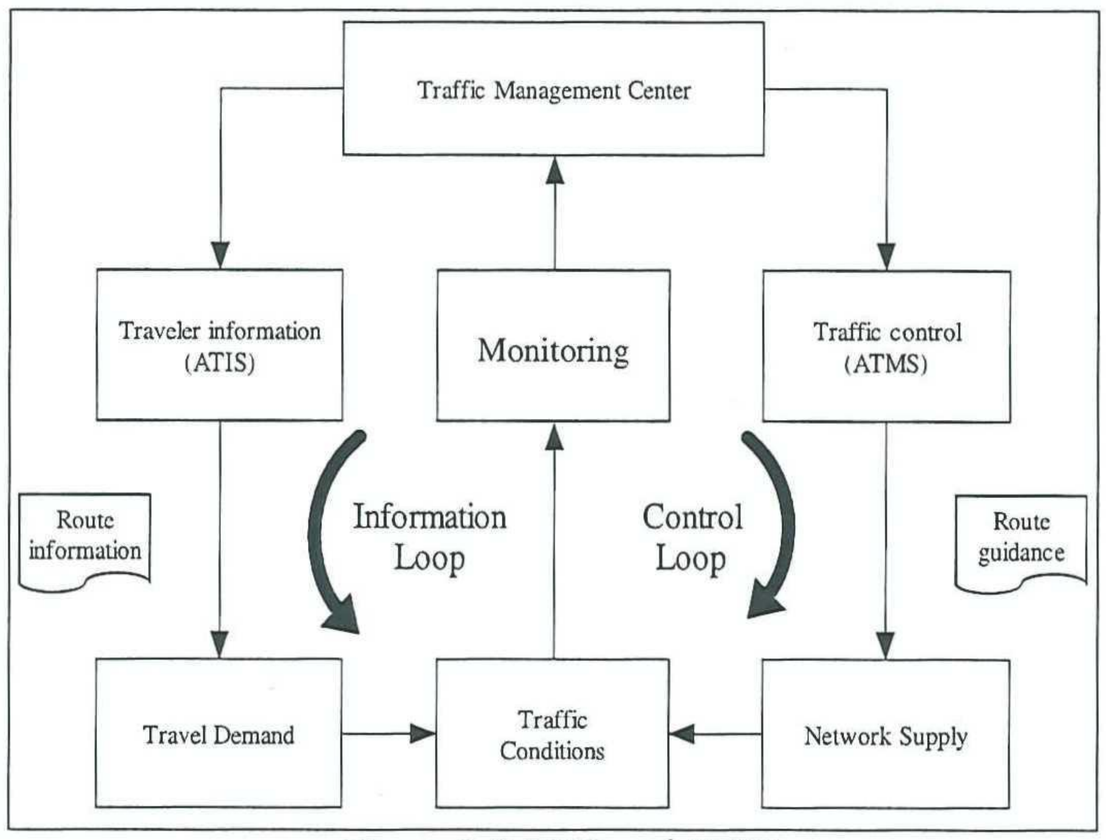


Figure 2-2 ATIS and ATMS

In a concluding remark we can characterize the RGIS the following questions:

i. Is the system providing *descriptive information* or *prescriptive guidance*?
ii. Where is the information or guidance provided; *pre trip* or *en-route*?
iii. Is it aimed at a *specific user* (in car) or targeting *a range of users* (en route)?
iv. Is the information or guidance *free* or *fee* based?
v. Is the information or guidance aimed at attaining an objective; is there any purpose served? And if so is what is desired; a user or a system optimum?

2.4 Current situation of RGIS in the Netherlands

In this paragraph the situation regarding the state of the art concerning RGIS in the Netherlands is investigated, followed by the current practice of providing route information and traffic information by means of DRIPS in more detail.

2.4.1 State of the art application of RGIS

In current practice initiatives on *dynamic route guidance* have been developed, for instance *the city parking guidance* where travelers follow a static route and are confronted with dynamic signs displaying the space left in a parking garage. But also mobile DRIPS which are placed alongside the road when maintenance is carried out or during extreme events / conditions.

Another type of roadside dissemination in the Netherlands is the *real time route or traffic information on DRIPS*. These DRIPS carry information regarding *current queue length* or *current travel time* toward a certain point.

The *in car route guidance* is becoming increasingly popular; these systems consists of a computer in the car which guides the users carefully through a network based on GPS signals.
Most systems nowadays do not incorporate current network conditions, they are excellent for finding you're way in a network to which one is unfamiliar, but they do not guide you through alternate routes to avoid queues. A new generation of route planners is finding its way to the market which does incorporate network information using TMC (traffic message channel) to update to current network conditions.

Currently initiatives are undertaken toward deployment of *graphical route information panels* (GRIPS) displaying traffic and route information. The major benefit of these types of panels is that travelers can determine whether or not their exit is still free from queue.

The pre trip RGIS is represented by an increasing number of route planners, public transport planners and detailed network information concerning current queues.

This thesis focuses on dynamic road side prescriptive route guidance by means of explicit route recommendations disseminated by means of DRIPS; as illustrated below.

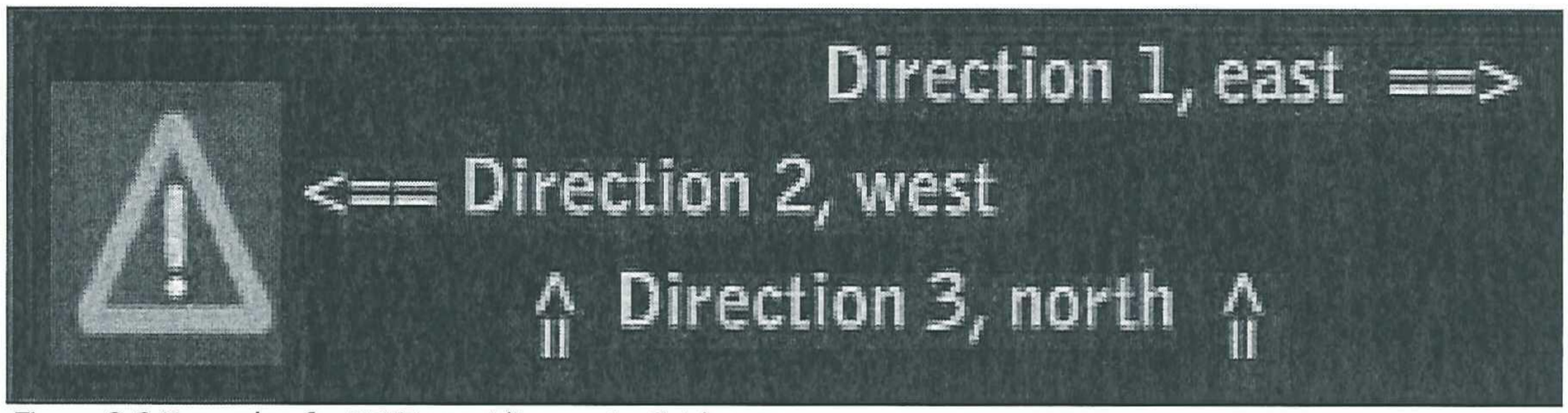


Figure 2-3 Example of a DRIP providing route Guidance

2.4.2 National Traffic Management in the Netherlands

There are five regional *Traffic Management Centers* (TMC)[1] in the Netherlands, coordinated by one national TMC (VMC-NL)[2] and flanked by the *Traffic information center* (TIC) and the *project bureau incident management* (PIM).

Typical activities are feeding the DRIPS with the correct route or network information to display, matrix panels, metering, incident management and *peak-strip* management.

The objective is to effectively spread correct information to road users. For this purpose KLPD (royal national police services) and RWS (national road authority) joined forces and formed the TIC (Traffic Information Center).

The TIC, operational in this way since 1998, provides network wide 24H basic information to various other information service providers (ISP), who again provide this information in a wide range of services / devices to the broad public. The PIM is another flanking organization which deals with national incident management.

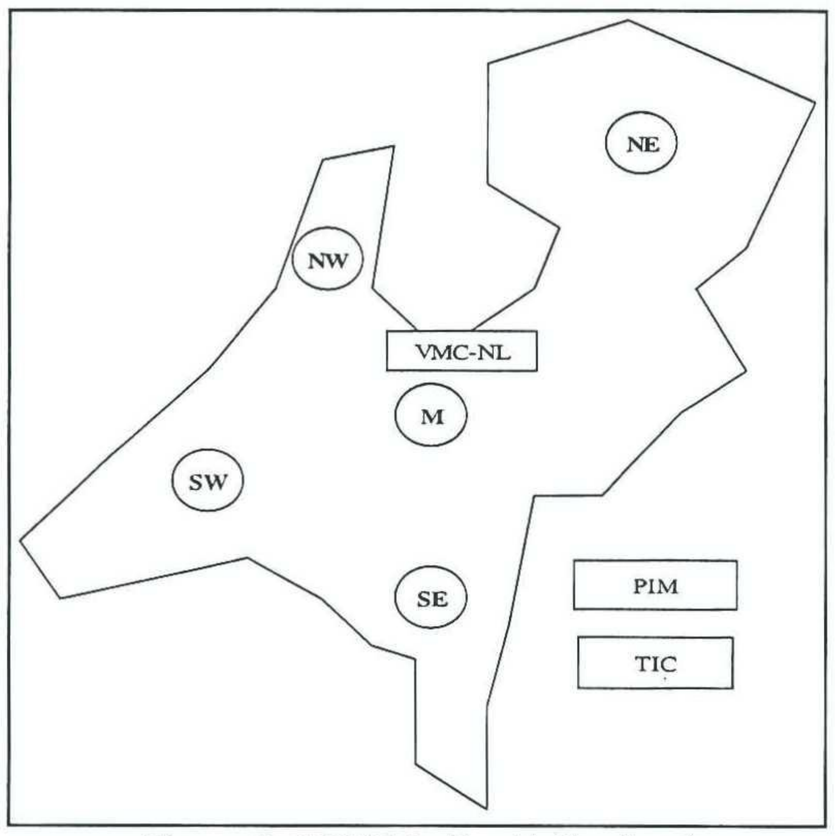


Figure 2-4 TMC in the Netherlands

The map to the right shows the current deployment of DRIPS in the motorway network. Currently about 80 DRIPS are installed in Holland on the motorways and another 23 are installed on the urban network, a number which is expected to grow to approximately 120 in 2006.

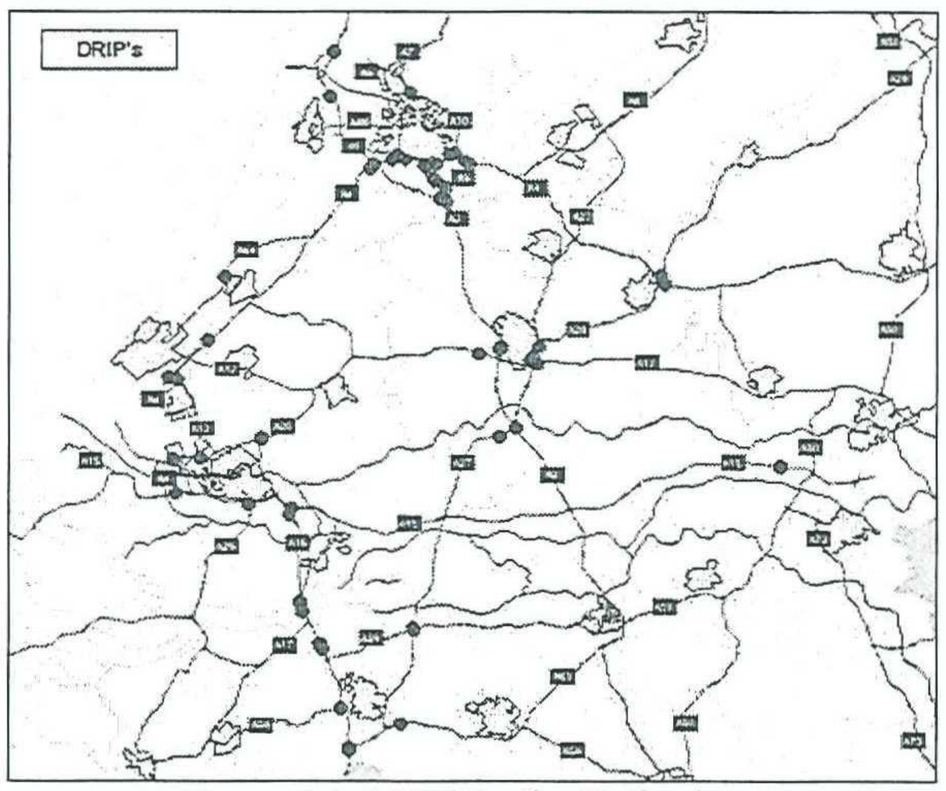


Figure 2-5 DRIPS in the Netherlands

Every 4 seconds actual traffic information is refreshed within one of the regional TMC (this is called MTM2[3] data). This information is used in a simple algorithm that calculates the actual queue length, which in turn is refreshed every minute on the different DRIPS. The tendency is to present delays instead of queue lengths[4].

Recently the installed DRIP systems formed a family of systems without uniformity or standard. There has been a successful standardization attempt in the type of messaging resulted in the *Central DRIP management System* (CDMS).

---

[1] NE: Wolheze; NW: Velsen; SE: Geldrop; SW: Rotterdam; M: Utrecht
[2] VMC-NL Verkeers management centrale nederland.
[3] Most traffic information is derived from the MONICA system
[4] Source: http://www.rws-avv.nl/avv/nl/benutten/maatregelenapplicatie/

2.4.2.1 Traffic management practice in the Netherlands

The current practice in national dynamic traffic management revolves around three different scenarios: regular use, incidents and extreme events.

Regular use
The regional traffic control center (RTCC) is fed information on its network by use of the MTM2 system. This information is used to calculate the information used in the DRIP message. There is a one minute delay on actual traffic situation and drip message change. There are two areas where the drips are being used to display travel time (A13 toward Rotterdam and the triangular area to the north east of Utrecht[1] , A27 – A28 – A1). In all other situations where queue length is displayed these message are shown when queue length exceeds 1 km. The shown messages are reactive, displaying information, not prediction.

Incidents
In current practice, the most involved instances like – traffic police, RWS (road authority), ambulance services, fire dept, ANWB, local environmental services and salvage companies- cooperate to deal with incidents as quickly as possible, this way some vital areas of the highway are under 24h surveillance. The coordination is done by PIM (project bureau incident management). In the current situation incidents can be detected or reported as follows:

Incidents are reported to different dispatch rooms, for instance: the national or regional police, fire dept, CPA or ANWB. All dispatch rooms exchange their information with the police dispatcher who warns the CMI/CMC (central dispatch point for incident management/central dispatch point for truck salvaging). Police dispatch rooms decide who sends a surveillance unit.

Incidents can also be detected in various way. Travelers or police surveillance call their detection in at the dispatch room of the national police (KLPD). Or whenever an employee of RWS detects an incident (for instance TMC operators, using video surveillance) they call it in to the KLPD as well.

When an incident has reached the KLPD control room it is further determined which action to take. This can include traffic management; in this case the regional or national traffic management center must determine what measures to apply (Ranging from rerouting to sending a cleaning crew). [5.]

If the incident is small and the effects are on the regional level, the DRIP settings are adjusted by the regional TMC, using a DSS which gives some directions for the messaging.
In the case, when the effects of the incident influence other regions the national VMC-NL coordinates the DRIP settings of different regions using a similar DSS. Dealing with incidents has been standardized in PIM (project bureau incident management)

Large events
In current practice some large events are reoccurring and a predefined script is used to deal with the situation. Some examples are the funeral of our late Queen Juliana, festivals like *Dance valley*, *Lowlands* or the annual "*Nijmeegse vierdaagse*". In addition to the existing DRIPS *mobile DRIPS*[2] are used for more information.

---

[1] A survey in this area indicated that here is no clearly stated preference regarding the type of information. Queue length or Travel time
[2] A mobile DRIP can display 5 lines of 10 characters and be deployed alongside the road, these settings are also controlled from the RTCC mostly based on human expertise.

In current practice, about 60 persons work with road side messaging in the regional TMC's and the VMC-NL round the clock 7 days a week. The actual DRIP portals cost about €60.000 and connecting them to the system about €140.000 whereas maintenance costs are estimated to be about 5-10% of the foundation costs a year[1]

2.5 Challenges in developing roadside RGIS

This paragraph discusses the most prominent challenges facing the successful development of a roadside route guidance and information system.

2.5.1 Actors involved

To every societal problem there are different actors and in the following text we will discuss the main actors involved in our problem and their relation to a roadside RGIS.

The system involves different actors with different objectives and consequently behavior; we can define three main classes of actors, the *road authority, the (transport) service provider* and the *road users*.

The authority has different attributes, such as: its level of governance be it national, regional or municipal. We can attribute different levels of power in terms of manpower, financial resources or knowledge of the system (measurement and access to data).

The (transport) service provider for example: ANWB (union), or public transport bus companies, can sometimes have a strong relation with the road authority in the case of public transport or when providing additional services (ANWB provides signs and markings on motorways). There role is usually ambiguous since they provide service to both traveler and authority.

In the case of road users we can define attributes such as origin and destination, users who are already guided or unguided and perhaps most important the traveler behavior in relation to roadside route guidance or information.

Objectives
The classes of actors can have different objectives such as the well being of their inhabitants in the case of the road authority or municipality. Most likely road travelers/users will want to follow some kind least cost path depending on personal priorities but in the case of the road authority, objectives are far more complex. They can be environmental, political or intricately formulated in terms of specific link requirements on a network (desired speed, intensity etc). Transport service providers will have different objectives depending on the nature of their service.

2.5.1.1 Actor behavior

Traveler
The main challenge regarding the traveler behavior is its reaction to the provided roadside RGI. This is a very difficult aspect to model and could have a great influence on actual calculation of the roadside RGI. In the example below there is no meaningful effect found attributed to the DMS:

[1] as explained by, M. van der Vusse, VMC-NL

> In an American investigative report attempting to determine the effect a DMS[1] system has on driver behavior at a site in the Hampton Roads area of Virginia. They find a negligible effect on diversion due to the installed DMS. In their primary conclusions they state among others that: *"weak messages where displayed, possibly the secondary route was to far away and drivers where unwilling to divert."* [6.]

The *unwillingness* of a driver to divert is related to the *cognitive map* of a traveler. *"A Cognitive maps enables us to construct a mental representation of an environment which cannot be seen from one vantage point alone."* [7.]

This *cognitive map* is also mentioned in [8.] where the value of the actual information presented to the traveler is stressed:

> "The challenge for dynamic route guidance (DRG) is not simply to provide drivers with knowledge of existing alternative routes, but to provide them with sufficient detail about the current state of those alternatives with regard to the driver's present route. It is erroneous to assume that drivers are unaware of alternatives. From an information processing approach one could say that a driver who has habitually traveled the same route for a number of years will have abstracted from existing knowledge, generalized from prior experience of route structure and therefore will certainly have developed a cognitive representation of the likely spatial arrangement of connecting and adjacent roads. It is perhaps more accurate to say that drivers lack accurate knowledge about the current conditions of these alternatives, and are therefore unwilling to change routes."

In the same paper it is stated that:

> Driver's willingness to accept and use DRG will in part depend upon their perception of DRG as a useful driving aid. This assessment will be quickly formed, probably as a result of a fairly limited number of trials. This corresponds to the popular notion that users may loose confidence in the system when provided with bad information.

Road authority
The behavior of the road authority should be included here as well. By this we not so much mean the personalities of the traffic managers but more the different *systems* used by the road authority and *how they behave*. The main difficulty in finding a generic approach to a route guidance strategy is integrating the many different systems used. These difference can be in the data they used, the (geographic) network they operate on or their isolated nature in which they where once created.

2.5.2 Specific roadside RGIS issues

By focusing on road side systems we have to deal with different network aspects which will be discussed here.

The network level
When deploying DRIPS in a network they can be deployed at different *network levels*, based on the possible effectiveness or societal relevance of the DRIP. In the situation in the Netherlands this is mainly between the *motorway* and the *urban network*. In [9.] some suggestions about the location are made:

[1] DMS; Dynamic message sign

> Variable message signs (VMS) could become a key element in efforts to introduce integrated control in a corridor under near-term scenarios of ITS deployment. The strategy here is to view routing as a control method, and the distribution of flows in the network as a mechanism available to traffic managers seeking to optimize the performance of the system. Thus, by integrating the routing and the traffic control, it would be possible to reduce congestion and improve capacity. With the ATMS strategies available today, VMS could be placed at strategic decision points on freeway access roads that is, where drivers must decide whether to get onto the freeway. More signs could be placed along the arterial streets where additional indications and/or messages could be presented to the drivers. (...) Initially, VMS will serve as a primary tool for displaying traffic and route information, and for sending messages jointly at different points to yield the desired spatial distribution of flows over the network.

In Rotterdam several urban DRIPS have been deployed on so called feeder links but their effectiveness was difficult to measure. The messaging used on these DRIPS is generated by the regional TMC, and not the municipality itself. Therefore a hybrid system that somehow coordinates the DRIPS on different network levels will be likely to lead to better results.

Locations within a road network

When more clarity is known about what network level or combination of levels to consider, the exact DRIP positions on the network *links* must be determined. In general a DRIP can be placed when: there is a reason to provide actual traffic data, it is considered useful for user or road authority and the information can be displayed in a meaningful, understandable (graphic) fashion.

The following criteria are used to positively rank different locations for usefulness:

i. congestion probability;
ii. intensity
iii. priority of the road within the network
iv. detour factor
v. presence of detection systems

Usually these criteria apply to certain specific points within a network, such as b*ifurcation points,* or *freeway access points.* [10.]

Coordinated vs. isolated

A lot of DTM control measures currently in use are small scale isolated systems.

> The current traffic control schemes often only act on an isolated scale, while seldom taking the varying objectives of different user classes into account. Current traffic control measures often solve congestion on a highway section or a part of the motorway by moving it to another part of the motorway network or to the local street network. To further improve traffic conditions, we need to simultaneously consider both the motorways and the urban roads. Hence, we require a coordinated network-wide traffic control set-up for hybrid networks that also takes multi-class characteristic of traffic flows into account. www.amici.tudelft.nl

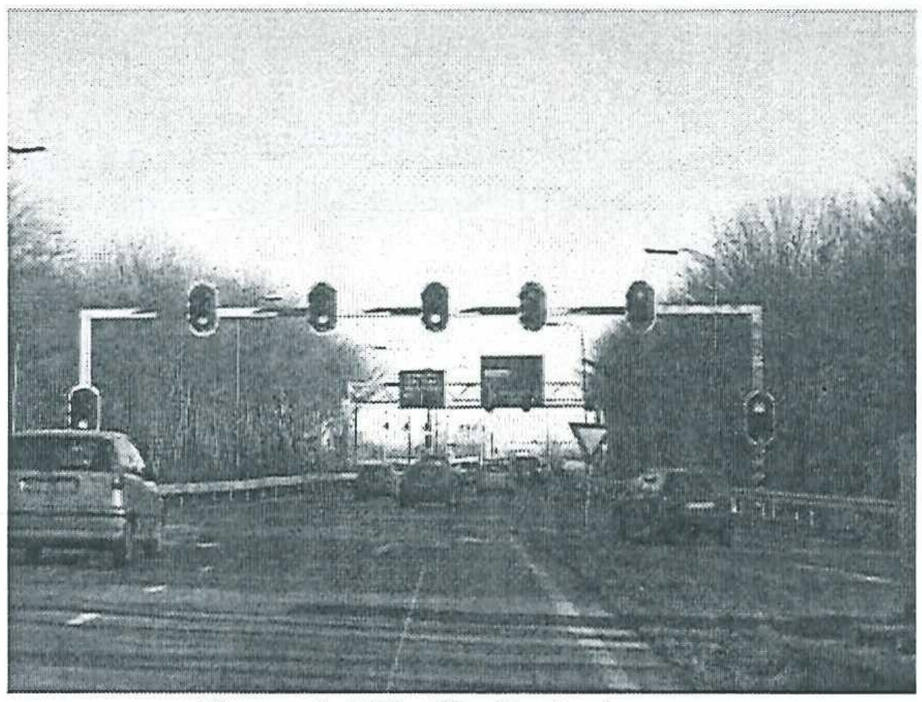


Figure 2-6 Traffic Control

The statement above is partially applicable to the current *hierarchic build* of the road side RGI system by means of DRIPS. There are some opportunities to further integrate control, for example between DRIPS on the motorway network and those located on the urban network.

2.5.3 Dealing with different network conditions

Another main challenge facing RGIS is for what types of network *conditions* they are designed, there are three main scenarios possible:

- expected network conditions such as morning and evening queue
- Incidents which disturb the network
- Events which features extreme demand

Expected network conditions
Different network situations require different traffic management measures. Within the DACCORD program [11.] an alarm phase model has been defined which differentiates between the following recurrent traffic conditions and accompanying measures:

Table 2-1 DACCORD network conditions and consequent measures;

Condition	Name	Service	Measure
D < Q	Free flow	no traffic control services	Inactivate traffic control tools Inform about travel time
0.8 Q < D < Q	Unstable traffic	Prevent onset congestion	Activation of: · traffic speed smoothing · ramp metering · mainline metering · rerouting · travel time information
D >= Q	Congestion	Optimize throughput. Manage queues.	Harmonized activation of: · co-coordinated ramp metering · presentation of dynamic route information Traffic queue management by: · activation of buffer zones · dissemination of route information
D >> Q	Gridlock	Resolve gridlock	Centralized activation of: · mainline metering · dissemination of route information

D= demand, Q = Capacity,

Strangely enough rerouting is only suggested when preventing the onset of congestion.

In the table below, the main causes for congestion on the traffic network in the Netherlands [12.] are given:

Table 2-2 Motorway traffic jam causes and their respective chances in the Netherlands

No	Description	[%]		Description	[%]
1	Evening peak	35	5	Maintenance during day	3
2	Morning peak	31	6	Maintenance during evening	2
3	Outside peak	13	7	*Accident voyeurism*	1
4	Accidents	13	8	Other	1

Given this table it would be very beneficial if network conditions could be improved by means of *road side route guidance* in the case of (the onset) of peak congestion and accidents.

The other two network conditions have already been discussed in 2.4, but it should be clear that some *opportunities* lie ahead for effective route guidance using DRIPS in different situations.

2.5.4 Dimensions of the traffic control problem

Controlling a *traffic management system* is a type of control problem with different levels of control, in which we make the following division:

Macro: Automatic vs. manual
When talking about controlling a *traffic system* we can differentiate between *automatic* or *manual* control. In general, the simpler the problem the more likely a fully automated system (isolated) is able to handle the task. In the case of route guidance however, simple automated examples do exist (parking routes, bridge message signs) but the majority of the more complex systems are still supervised by a traffic manager.

Meso: General traffic control system
When dealing with a partially automated traffic control system it generally consists of the following components: the real traffic system, the monitoring system, the controller and the actuators.

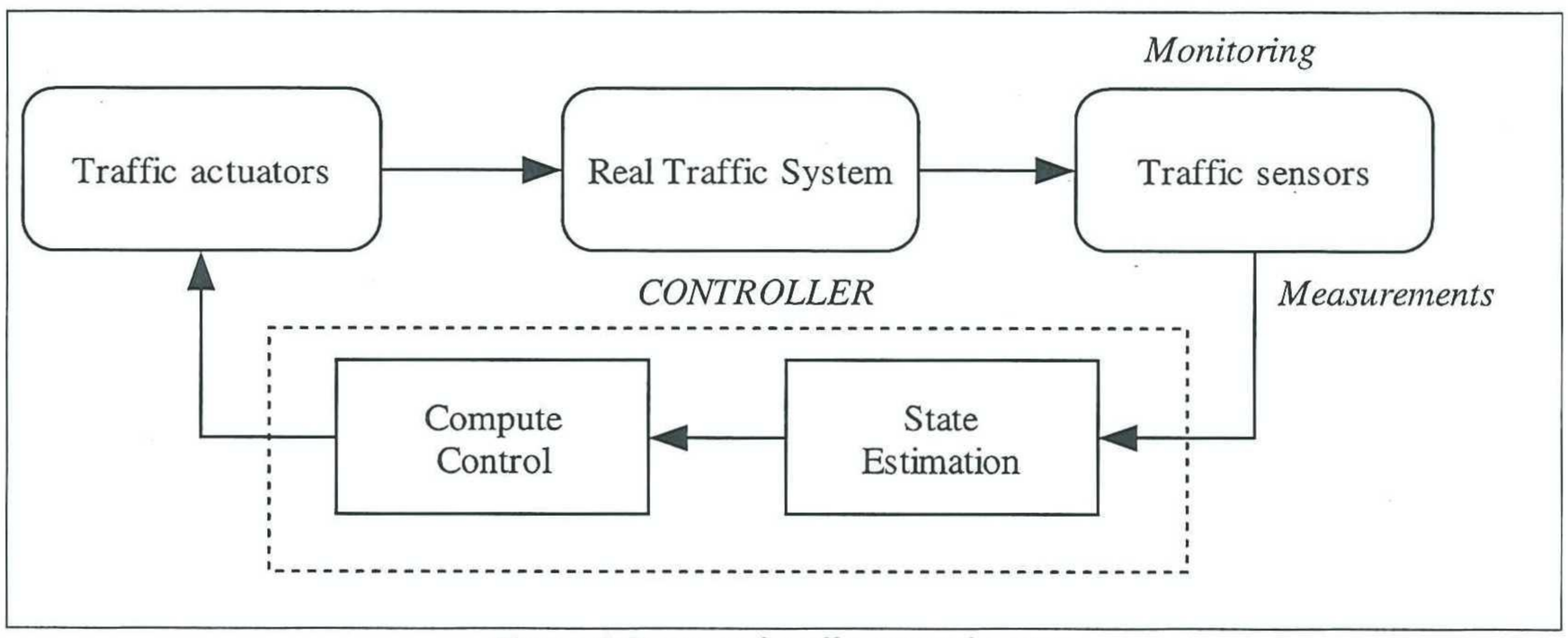


Figure 2-7 general traffic control system

In this case the *actuators* are the DRIPS in the broad sense. (If we where to use internet information pages to disseminate pre-trip information it would become an ATIS service). The traffic *sensors* can be various type of detectors (pneumatic, infrared, induction, radar, video, probed vehicles). The *state estimation* is a step needed to derive the state of the system from the incoming data, which is in general apt to noise (different type of data filtering can be used here such as *Kalman filtering*[1] suggested in [11.])

Micro: Computation control methodologies
When computing the actual *control* (guidance or information) one can choose between four typical approaches: *open loop, feedback, predictive and optimal* (this is further explained in 4.5.)

[1] Although diagnosing a traffic situation is more then Kalman filtering alone. The general idea is about accepting or rejecting a hypothesis describing possible problems based on the measurement data. This has more to do with a Bayesian estimation: what is the chance on status X given measurements Y.

It is interesting to take a look at the current type of control for calculating the DRIP information in the Netherlands. There are two types of information displayed *queue length* and *travel time*. For most motorways, induction loop data is available divided into segments of about 500 m. This data is used to calculate the average speed per segment. Now this average speed is the base for the information currently displayed. All segments with a speed below 35 km/hr are considered to be queued, whereas in calculating segment travel time, its length is divided by the average speed.

When calculating the actual DRIP information the different characteristics are considered for a number of segments (being queued or average speed) and averaged somewhat before disseminated by means of DRIP. In general we can say a simple robust form of feedback control is used to calculate the information on the DRIPS.

Based on an interview with a TMC Utrecht spokesman

The route or network information thus currently displayed on the DRIPS is based on a *feedback control* loop. What is important to note is the possibility to use a *predictive* or even an *optimal control* approach to the computation which could yield better results.

Coordinating control
When dealing with different systems and levels of control, the issue of coordination comes into play. Network wide coordination of road side messaging is necessary for RGIS to deal with extreme situations. In the case of a centralized generation of route information or guidance, coordination is guaranteed by traffic managers and operators. Yet when local, isolated information is used uncoordinated control could take place.

2.5.5 Consistency between predicted and experienced travel time

When dealing with calculated road side RGI, in the form of *descriptive information* based on *prediction* (or optimal control for that matter) one of the major issues is dealing with *consistency*.

For the traveler, *predictions about future network conditions*, which one w*ould* encounter downstream, are more valuable then a description about the *current state* of the network further downstream. This is off course due to the fact that current network states downstream might have changed during the time it took the traveler to get there.

Consistency comes into play when providing users with *predictions* about *their expected traffic conditions they will encounter* during their upcoming journey. If these predictions *do not coincide* with the *actual experienced* traffic conditions the prediction was *not consistent*.
When users are provided several times with *inconsistent* and *bad* information they may *loose confidence* in the system.
The difficulty in consistency is twofold; on the one hand you have to predict future conditions for route guidance or information based on demand and supply, or in other words the dynamic OD matrix and the current network condition. On the other hand you have to anticipate how many users will follow the actual provided route guidance or information and thus change future network conditions.

Unlike predicting the weather which has no influence on the actual weather conditions, predicting future traffic situations and providing users with this information has considerable impact on the actual predicted future traffic conditions. To cope with consistency robust travel time estimators are needed. Different ways of dealing with consistency are discussed in [13.] and [14.].

2.5.6 Road side dissemination issues

The displaying of road side information has two important features: the *syntax* (what to display, the format) and the *semantics* (what does it mean)

There is a reasonable amount of literature available on what and how to display using variable message signs. In most cases this is either a local or national document based upon experience with one or a few road side VMS systems. The general conclusions always state:

- The data needs to be easily to interpret
- A message on a single line is most effective
- The data must be reliable, consistent and robust

Sometimes more in depth recommendations are made[1]. In practice, most DRIPs are controlled by one of the five national traffic centers and messaging has been standardized in the central DRIP management system (CDMS), the systems can display 3 lines of about 27 characters.

The system used in the Netherlands distinguishes the following categories of messages:

i. route and route choice information
ii. blockade information
iii. incidents information
iv. weather information
v. road condition
vi. non traffic related messages[2]

Also in other literature additional categories are used, such as:

vii. Regulatory (for example high occupancy vehicles)
viii. Special events

The different categories all use different *semantics*. Depending on the type of the information (actual or predicted) the following types can be described:

i. travel time [min]
ii. queue length [km]
iii. alternative routes [-]
iv. Text information pertaining to specific situations [-]

Compliance & Faith
When providing prescriptive route guidance by means of explicit route recommendations the compliance determines the amount of user which will follow the routing advice. When providing descriptive route or network information, the user faith in the system determines to what extent the provided information is of influence to the choice process regarding route choice of the traveler passing the DRIP.

---

[1] For more information on syntax: [16.] and [17. keyword HARDIE]
[2] After traveler complaints about useless VMS messages, as occurred in the Netherlands, stating that the *display was out of order* ("route informatie-paneel buiten bedrijf") the government standardized these kind of messages into "file vrij" *congestion free*. Also some experiments where carried out with the display of *motto's (slogans)*, such as "Bob jij of Bob ik" (a slogan belonging to a don't drink and drive campaign) of "Bel handsfree". A survey carried out under Dutch motorists revealed that congestion information deserves priority, when there is no real information traffic related *motto's* can be displayed. [18.]

3 Problem analysis

In the previous chapter we have discussed route guidance and information in general and its current practice in the Netherlands and we have also seen some of the (current) difficulties linked to this type of management measures. Given these findings we will now formally state the problem and resulting objective for this thesis together with the main focus and some limitations.

3.1 Problem statement

The problems and opportunities concerning route guidance and information in its current application, which have been discussed in the previous chapter, are summarized in the text below.

Current situation
The *road side route guidance and information systems* in the Netherlands are an important instrument within the *dynamic traffic management context.* We have seen that the traffic management is distributed hierarchically over national, regional and municipal *traffic management centers*. If focus on the *road side route guidance and information systems,* we see that here are *three* scenarios under which these systems operate. Under *normal recurrent congestion conditions* we see that a simple *feedback control loop* is used to generate en route messaging in terms of route information displaying *queue length* and *travel time.* When dealing with extreme situations (incidents or extreme demand), a DSS is used to aid traffic manager in displaying appropriate messages for these specific circumstances.

Problems
The *route information* currently used, queue length and travel time, is *descriptive* of nature and requires *forehand knowledge* of the network by the traveler to be interpreted correctly. The provided information is based on the *current state of the system* by means of a *feedback control* loop which makes it likely that under certain *near congestive circumstances* travelers will experience *different* network conditions in comparison to the *descriptive route information* they saw on the DRIPS. The very essence of *descriptive information* (queue length in [km] and travel time in [minutes]) provides travelers with a *quantitative base* for comparison. In the case where a traveler encountered a longer queue or travel time than was displayed on the DRIPs it is remembered as a *bad experience* with this kind of route information and reoccurring bad experiences could diminish the faith of a user in the system.

Opportunities
Some great possibilities lie ahead in improving the existing situation:
- Generating *route guidance* in terms of *prescriptive information* disseminated by (existing) DRIPS. This way the base for quantitative comparison is eliminated.
- Calculating this prescriptive information by means of a *predictive* or even *optimal control* loop should provide qualitative better information.
- Developing a robust approach which could deal with different scenarios and flexibility in the number / location of DRIP's.

3.2 Thesis objective

We know that the current practice provides some opportunities for improvement, which leads us to the following objective for this thesis.

"Develop a methodology which is able to generate appropriate *en-route route guidance* in terms of *prescriptive information*. This methodology should preferably be *independent* of various *network conditions* such as traffic density, topology, link types etc and be able to generate route guidance for *different scenario* as they occur in practice. To further improve the quality of the route guidance, it should be investigated if a better alternative for *feedback control* could be applied."

Research questions
Given this objective, during the development of the desired methodology it will be likely that the following questions must be answered:

- What kind of control loop would be best suited for generating guidance?
- How does one translate *prescriptive information* into actual DRIP settings?
- What is *good* route guidance, what objective or function should be accomplished/optimized?
- How do you deal with *user behavior/compliance* when generating prescriptive information?
- In what form should such a methodology be implemented in real life?

3.3 Requirements and limitations

This thesis deals with generating *guidance* in the form of *prescriptive information* instead of *descriptive information*. We have seen that by providing network information, users might have difficulty in translating the information to a correct choice. We have also seen that this information must be *consistent* because travelers are able to compare the *experienced* network conditions with the *predicted* ones, if the routing information was followed.
When *route guidance,* in the form of *prescriptive information* is given, travelers will have no *quantitative base* for comparing their experienced network conditions to, and comparison will thus become more difficult. In addition, when dealing with *prescriptive information,* the choice is simple: one either follows the advice or not and no *forehand network knowledge* is required to first interpret the information and choose an alternative. Prescriptive route recommendations are also ergonomically better suited to deal with extreme network situation and therefore desirable when for the development of a generic approach.

Guidance can be disseminated by different means at different times. This thesis deals with the en-route way of disseminating guidance by means of DRIPS. As for the *pre trip* dissemination a lot of instances are deployed and being developed. Whereas the in-car way of providing route guidance is dominated by commercial packages. It is the belief of the author that the en-route or roadside way of providing route guidance by means of DRIPS is an opportunity which can be used to improve network conditions under various circumstances.

As shown, the current practices regarding the generation of DRIP settings is split into on the one hand *automated operation* during normal recurrent congestive conditions and *manual operation* ,supported by a DSS, when dealing with *different scenarios*. In this thesis we hope to develop a methodology which could be used in all scenarios and therefore further automating the control.

In the ensuing of this paragraph the boundaries and limitations are listed.

3.3.1 Requirements

The most important requirements to the methodology are listed below:

- The methodology will generate *prescriptive route guidance*
- The guidance will be disseminated by means of *DRIPS*
- The route guidance advices are dynamic in time
- The methodology must be independent upon network topology
- The methodology must be independent upon number or position of the DRIPS
- User behavior in terms of compliance must somehow be integrated into the methodology
- The aim of the prescriptive guidance must be in improving the overall network conditions
- The methodology must include an implementation framework
- The methodology must be able to be applied to a realistic traffic network

3.3.2 Limitations

- The methodology does not generate descriptive route information
- The methodology does not need to generate a true global optimum
- The methodology does not formulate the exact to be disseminated messages
- This thesis deals mainly with the generation / optimization methodology not the exact implementation
- The case study is to illustrate the generic nature and the ability to be applied to a real life situation, and does not necessarily have to generate *ready to be disseminated* advices
- For this thesis the focus will be on disseminating guidance toward achieving a system optimal assignment

4 Literature review concerning roadside route guidance

In Chapter two the *background* and *context* of route guidance and information where given. In this chapter we will focus on the aspects that are *most likely needed* for developing a generic approach to a route guidance strategy.

To accomplish this, we will start by discussing other algorithms developed for generating route guidance. With this knowledge and the ideas so far, we define a *crude layout* of the methodology we will be developing. Based on the different *elements* within this layout *the subjects of* we have researched *traffic modeling, the objective function, optimization and different control methodologies* and summarized the results.

At the end of this chapter, we will have sufficient knowledge to formalize the methodology in the next chapter by making weighted decisions regarding the elements used.

4.1 Other instances of generating route guidance

This paragraph discusses some other examples of generating route guidance. We start with a real life application of a RG system in Scotland. After this we focus on a theoretic approach aimed at generating consistent route guidance.

4.1.1 Scottish interurban network

A real life application for generating route guidance on VMS's has been developed in the Scottish interurban network and described in [19.]. The paper presents the design, implementation, and evaluation work performed by a European consortium for the development of a Variable Message Sign (VMS) information and guidance system in the interurban Scottish highway network. The control strategy employed is based on simple automatic control concepts with both feedback and feed forward terms subject to user-optimum constraints. Feed forward terms are employed for the prediction of travel times and delays along long-distance interurban highway links. Simulation studies demonstrate the potential improvements achievable with these kinds of control measures and control strategies. The implementation concepts and evaluation results are outlined.

In the diagram on the next page, their adopted *control strategy* is shown with the following accompanying text [19. pg 178]:

> A route flow and queuing model is applied in order to predict the situation at incident locations in terms of delays due to queuing. The model is fed with the flow measurements which are taken at the beginning of the routes. An important additional source of information is the incident reporting through the FEDICS system. Simple queue detection (occupancy measurements) at network locations, where sufficient detectors are available, supplements the incident reporting.
>
> The regulator does not produce a continuous output signal but has to select a certain plan out of a number of predefined plans. If no predefined plan matches the current needs, it is allowed to overwrite some legends with appropriate texts. The aim of the automatic control approach is to use the VMS in order to achieve a certain diversion impact towards alternative routes for relieving certain critical (overloaded or blocked) stretches, thus improving the overall network performance. The probable diverting impact of specific VMS legends is roughly known through route choice simulator surveys.

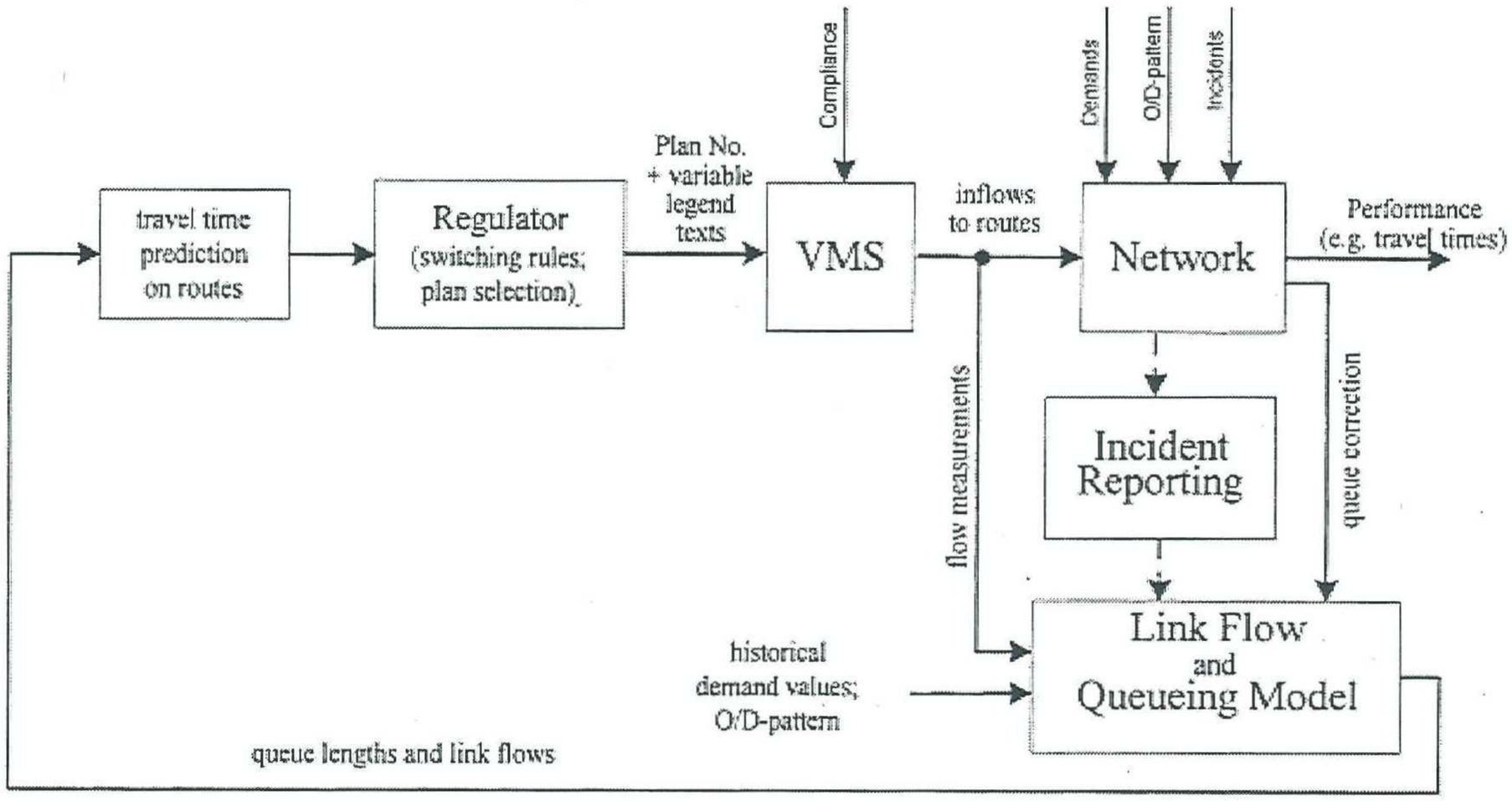


Figure 4-1 Adopted control strategy in the Scottish interurban network

In the final conclusion [19. pg 186] the authors state that:

> An important feature of the control strategy is the automatic selection of appropriate VMS display combinations for any traffic situation. This provides a potential advantage over other control approaches that rely on **predefined traffic scenarios whose number may increase to astronomically high levels** for networks with all but very simple topology. It should be noted, however, that in the present application it was an operational requirement for the control strategy to select a VMS plan out of a pre specified plan directory, hence the potential flexibility of the automatic control approach was only used up to a certain extent.

The *scale problem* with predefined scenarios is still valid today, however as we will show in 8.1 a suggested *crude scenario manager* could be useful since it only contains a realistic limited number of scenarios which need only *very roughly* match the measured network state.

4.1.2 Route guidance and generation in DYNAMIT

In this paragraph we review the route guidance component of DYNAMIT as discussed in [13.].

> This paper describes the route guidance generation component of DYNAMIT, a dynamic traffic assignment (DTA) system intended for deployment in a traffic center and capable of generating real-time prediction-based guidance information.(…)
>
> In order to guarantee the credibility of the information system, the guidance provided by DYNAMIT is consistent, meaning that it corresponds to traffic conditions that will be experienced by drivers. DYNAMIT provides user-optimal guidance, having the property that users cannot find a path that they would prefer to the one they chose based on the provided guidance. DYNAMIT is designed to operate in real time, using traffic volume and control system state data to estimate and predict time-dependent origin-destination (OD) flows and network conditions, and generating route guidance that is consistent with the predicted traffic conditions. It is designed to make departure time, mode and route recommendations for a variety of information system technologies and information dissemination strategies.

In the diagrams below the DYNAMIT system and the *route guidance generation control strategy* is illustrated.

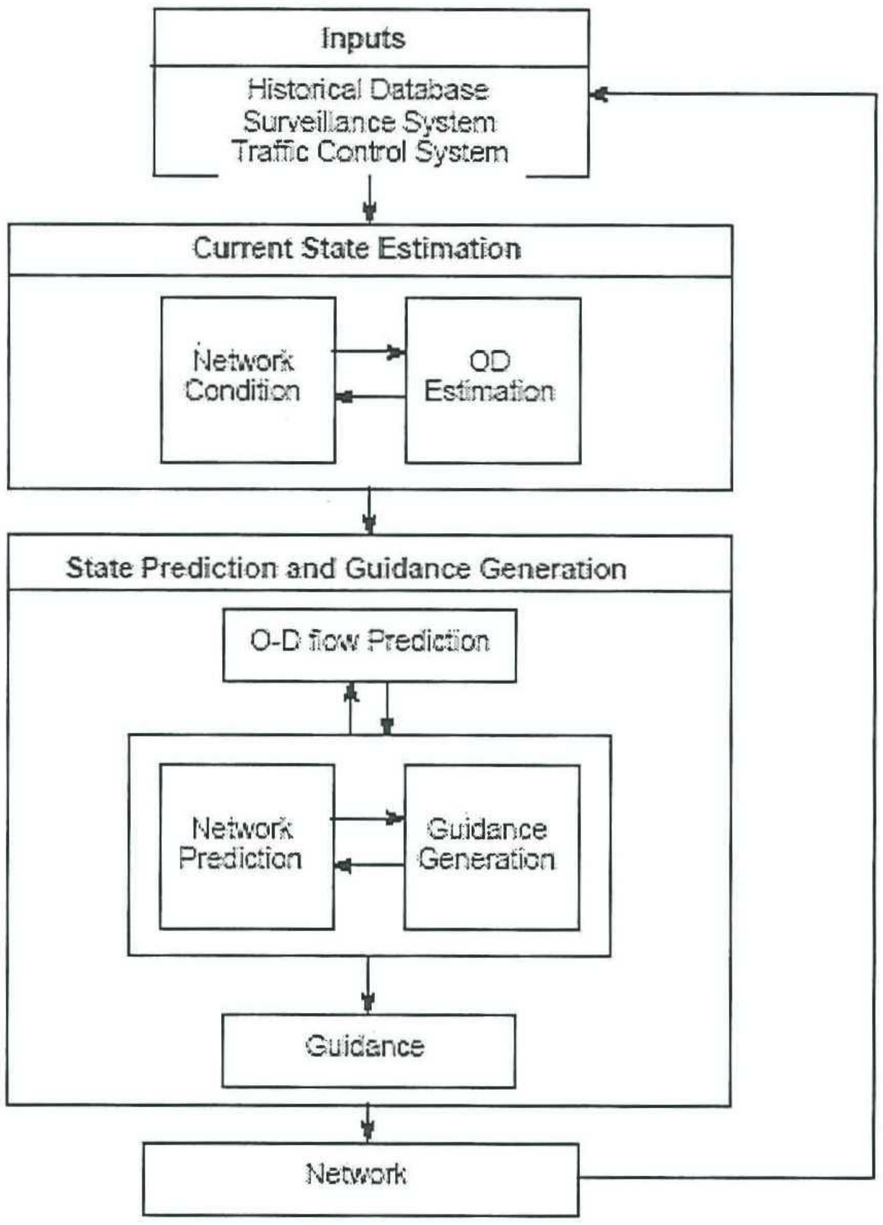


Figure 4-2 Overall DYNAMIT system structure

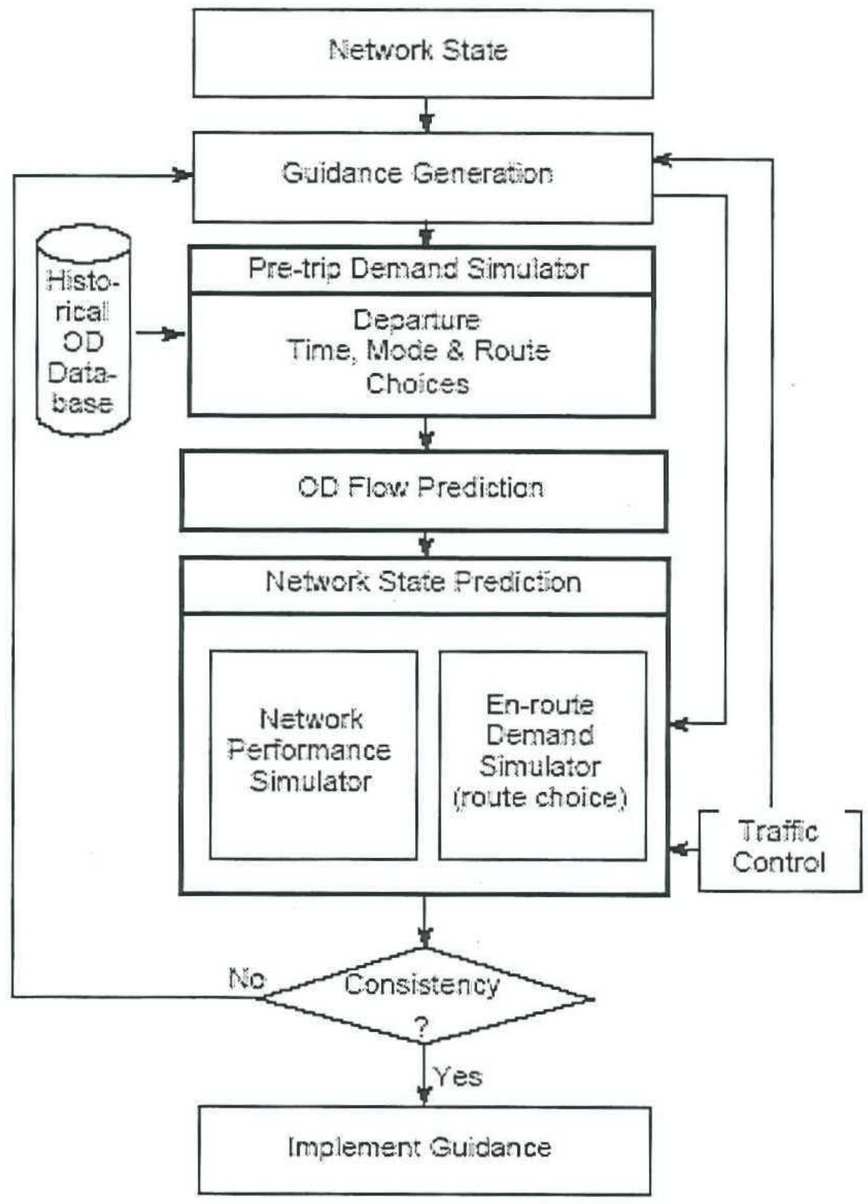


Figure 4-3 Prediction based guidance generation in DYNAMIT

DYNAMIT can generate both anticipatory descriptive and prescriptive guidance. Anticipatory descriptive guidance informs travelers about the traffic conditions they are likely to encounter on different feasible paths from their current position to their destination. Anticipatory prescriptive guidance recommends a path to travelers based on expected traffic conditions along alternative feasible paths. (…)

Generation of consistent unbiased guidance is not a trivial task. The complication arises from the fact that drivers who receive guidance information may change their paths as a result of that information. This may lead in turn to a change in the time-dependent link travel times on which the guidance information was based, and so the original guidance may be invalidated. Guidance generation is therefore a fixed point problem which may be stated as follows: find x such that $x = f(x)$ where x is the vector of the values of a variable that is used as the basis for guidance generation, and $f(x)$ is the vector of values which results from travelers' reactions to the guidance they receive. (…)

The objective of the guidance generation function implemented in DYNAMIT is to provide guidance which is consistent and unbiased. Consistent and unbiased guidance implies that the travel times which are used for guidance generation are unbiased estimates of the conditions which users actually experience on the paths they are predicted to take (based on the travel behavior models.) This objective eventually leads to user-optimality.

Given the structure of this solution an iterative algorithm referred to as *the time smoothing algorithm* was needed to solve the fixed point problem. In [20.] an additional investigation in to the route guidance generation capabilities of *Dynamit* is presented.

4.1.3 General remarks on generating route guidance

The two examples discussed in the previous paragraph, both strive to attain a user optimum by means of descriptive route information based on a prediction of travel times on the network.
To actually achieve this, the predicted information would have to be perfect in the sense that it is consistent and the confidence in the system would have to be a 100% as well, and the route information would have to be presented at all decision nodes in the network.
The first example is interesting since it has been installed and is running for quite some time now and successful due to the scale and controllability of the problem. Also the idea of using a set of pre-defined legends that can be overwritten is keen and keeps the number of needed scenarios limited.

In the second example, the problem of generating route guidance is formulated as a fixed point problem which needs to be solved in order to provide consistent route information. DYNAMIT is pushing the envelope as it comes to route guidance generation and can operate real time.

Given the approaches mentioned in the previous paragraph and the experience with the current approach in the Netherlands we now formulate some general remarks about generating route guidance and use them as a guideline for our literature review. These two examples both uses the following a control loop:

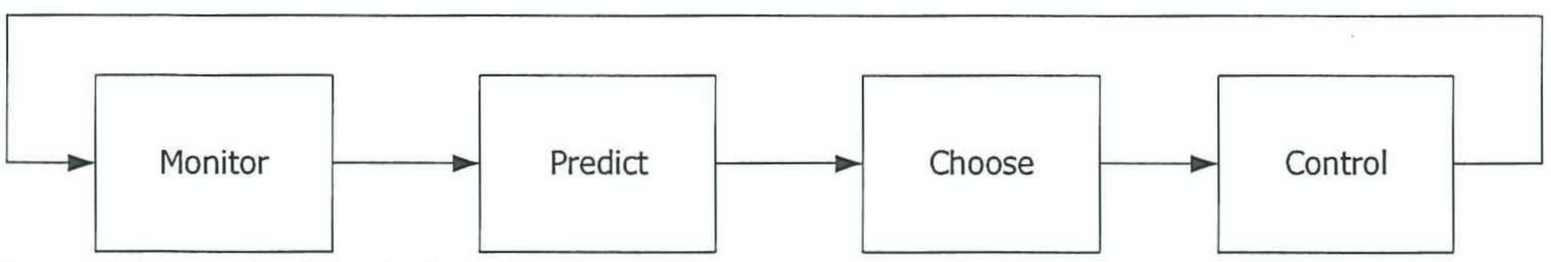


Figure 4-4 Standard control cycle

Crude layout
Independent of the type of control we will somehow need to model the network. With this model we can either make a simple *state estimation* or in the more sophisticated approach make a *prediction* about future network conditions. Given the network we would like to generate route guidance to satisfy a certain *objective* (system optimal assignment, applicable under different scenarios), preferably optimized (*consistent* or most effective guidance). This route guidance could then be disseminated to the DRIPS whereas its effects could be fed back into the system by adopting a *rolling horizon* approach. With all this in mind we can subtract the following layout:

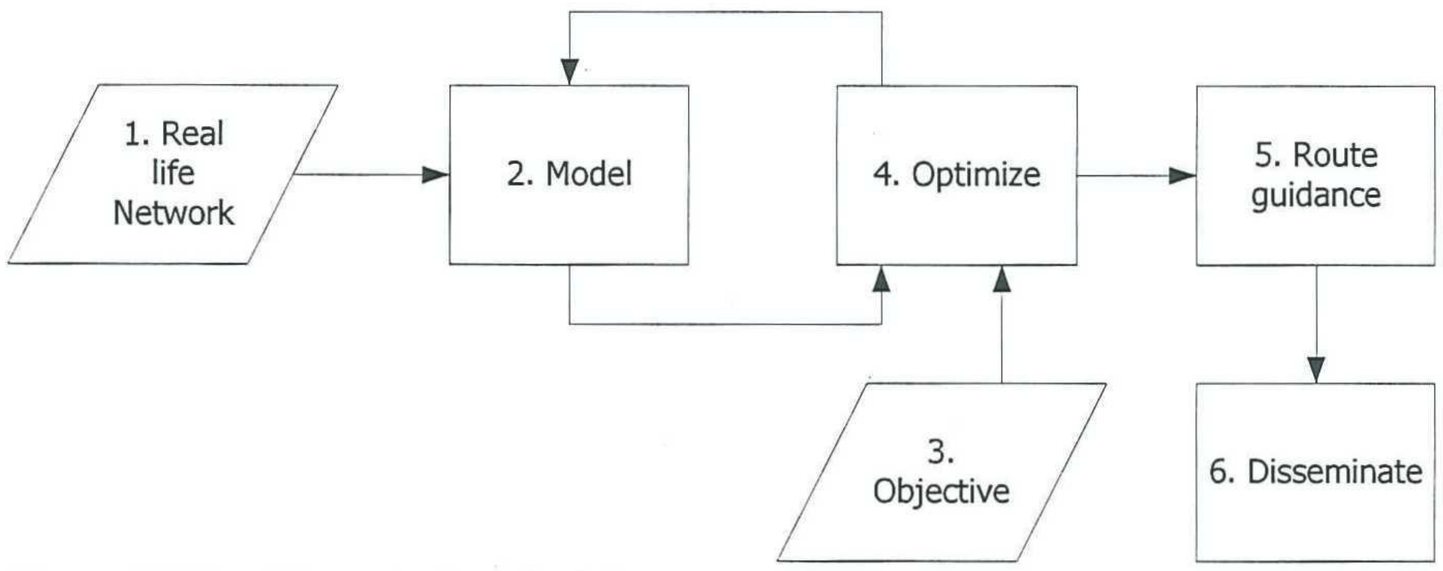


Figure 4-5 Possible control methodology

In the following paragraphs elements 3,4 and 5 are discussed. *The real life network*, the measurement and filtering required to create input to the model (to use for either state estimation or prediction) are important but not the main focus of this thesis. This subject is dealt with in numerous articles and is more important in the later stages when implementing the algorithm, not developing it. This also holds for the *route guidance* and actual *dissemination* where important aspects concerning these two have already been discussed in chapter two.

4.2 Modeling general traffic operation

The model plays an important role in the whole process. It can be used to simply represent the current measured state of the real life traffic network (as is done with DRIPS info in the Netherlands nowadays) to making a sophisticated simulation of traffic, and even be able to make predictions about future network conditions. This paragraph will be used to describe this last type of *dynamic traffic assignment models* (DTA) since it is likely we will be using this type of model.

The general idea of a *DTA model* is shown in the following diagram:

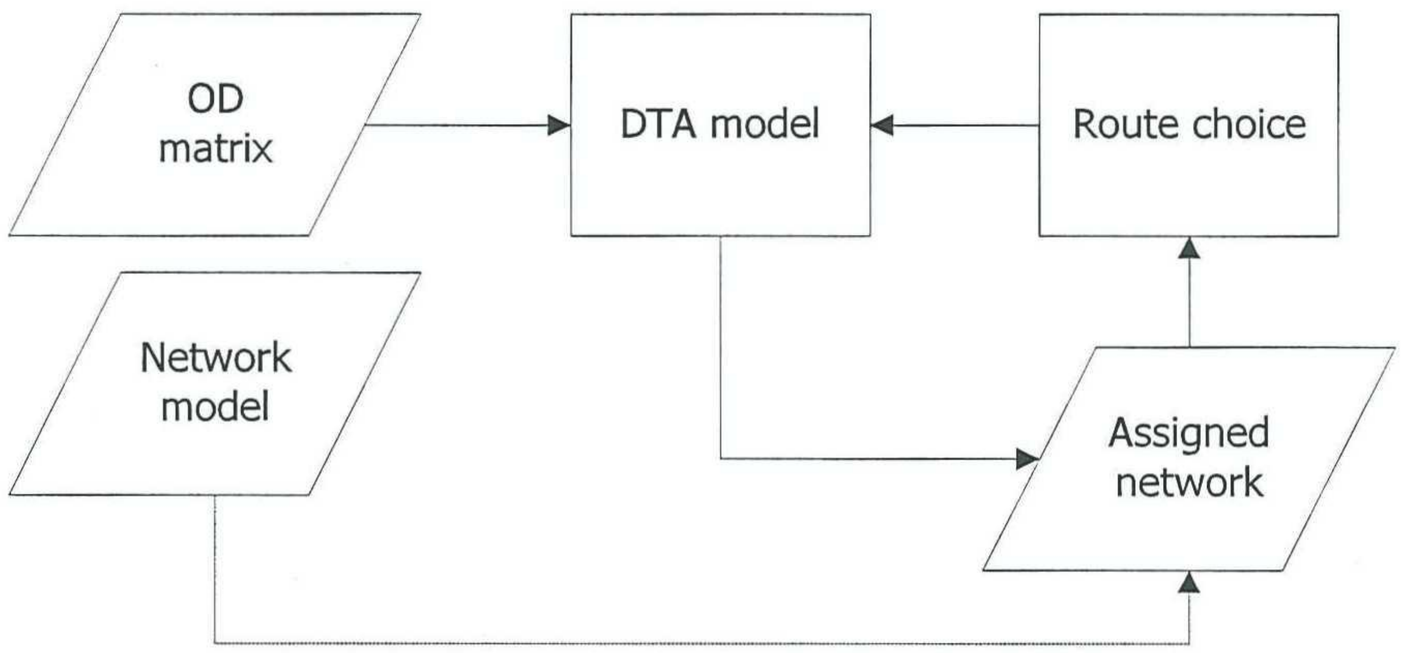


Figure 4-6 General working of a DTA model

A road network is translated into a so called *network model or representation*. This representation has all important road characteristics in it like: how they are connected to each other, their maximum speed, their capacity and the maximum number of vehicles which could be queued per kilometer. This network has several entry and exit points, we call origins and destinations. The demand for this network is described in an OD matrix. This matrix has data about how many people want to travel between origin O and destination D per a certain amount of time (This matrix could also vary within a day). For instance, in the morning peak people will want to travel from their home to their work, In the evening peak however they will want to travel the opposite direction. The model simply uses the demand per entry point and assigns them to the network using some most likely routes. These routes can be calculated before the assignment is made, or during the assignment based upon the changes in the network which occur during assignment and the nature of the model.

In normal traffic modeling practice some consideration would also be given to *pre trip* modeling; the choice of *departure time* and *transport mode*. These kind of pre-trip assumptions are not important for generating road side route guidance since this only affects people who have effectively chosen to travel the network by car.

In the following subparagraphs we will be discussing the various aspects of traffic modeling according to the diagram above.

4.2.1 Modeling traffic operation

This paragraph will briefly discuss some differences between *dynamic traffic assignment models* which are of importance to our methodology by means of the following criteria:

- Macro, meso or micro modeling

The different models that exist can roughly be divided by detail into three categories.

Macroscopic; is a representation of traffic by means of flows and densities. We don't know anything about individual users, but we know what their combined effect is on the network. Macroscopic modeling is the fastest way to model a network because it requires the less computation in comparison to other models.

Microscopic;, Modeling every user or traveler within the network individually. Microscopic modeling is best used when user behavior is an important factor in the modeling objective and there is a high demand for a qualitative simulation of real life.

The third type is *meso modeling.* In general it uses groups of users to assign to the network. It takes less time than micro modeling but ensures some user differentiation which is reflected in the *quality* of the assignment.

- User classes

Another distinguishing feature is the ability to handle multiple user classes. The benefit of assigning different user classes is in the ability to handle different types of traffic such personal cars, trucks, RG equipped drivers, elder drivers yuppies etc which is reflected in a more realistic assignment of traffic.

In general the higher the detail of a model, the more demanding it is in computational effort. For the generation of route guidance based on predictive modeling it is important to capture all dynamic effects concerning capacity (queues, traffic flows, speed etc) but also the user behavior regarding *route guidance* in terms of compliance.

4.2.2 Typical assignments of traffic demand to a network by a DTA model

A model is used to *assign* the traffic demand described by the OD matrix to the network in a *particular fashion.* Some misconception might occur due to a perhaps deviant definition of assignment used in this thesis; by assignment we mean: *"The state of a network as the result of assigning the demand for transport described in an OD matrix to the network by means of a certain set of dynamic routes between these origins and destinations and the interaction between vehicles and between vehicles and the network.* "

There are three main types of assignment of assignment used. The first is a *system optimal assignment*; in this case, the assignment is done in such a way the *network as a whole* is optimized. There would be no other way of assigning the traffic to the network which would lead to a better performance of the network (what is considered *optimal performance* depends upon the criteria used for evaluating the network, for example average speed). The second type of assignment is the *user optimal assignment*. This is achieved when all routes actually used between a source and a destination have the same cost and this cost is no larger than the delay along any other route between that source-destination pair. This assignment is also called the *Wardrop equilibrium assignment*. The third type of assignment is the *calibrated assignment*, an assignment that resembles the real life traffic network best.

Assignment detail
The detail of an assignment is largely dependant upon the nature of the model chosen. Microscopic models are better in simulating traveler behavior whereas macroscopic modeling is better suited to handle congestion and related effects (*spillback, queue buildup and dissipation* etc). Another important aspect of the assignment detail is the *predictive capability* (or quality of the *forecasted assignment)*. If an *optimal control methodology* is used for generating guidance the predictive capability of a model is used to evaluate the future effects of disseminated route guidance. If this prediction is qualitatively poor, so will the resulting route guidance advices be

As a concluding remark we could say that for the generation of route guidance we need first be able to handle congestion and related phenomena in an adequate manner. On the one hand they influence the network performance, which we try to optimize, and on the other hand, these conditions determine the availability and nature of alternate routes. Second the model used must be easily calibrated if we where to adopt a rolling horizon approach. And third, it's likely we will use an iterative approach to the actual generation of guidance so a *fast* model in terms of computational speed is desired. In all this leads to the preliminary conclusion that a *dynamic macroscopic traffic assignment model* is preferred for the development phase.

4.2.3 Route choice

In *traffic modeling terms*, a *route* is a chosen path which connects an origin with a destination. In the morning peak this could be translated into the question how one drives from home to work. When considering *dynamic traffic assignment models* we already know that the routes can vary over time since the type of assignment leads to different network conditions at different times. Usually this leads to an iterative structure, described by the diagram in 4-6, where routes are recalculated during the assignment. The iterative structure is needed to allow travelers to change routes, or use different routes at different times which is a result of the assumed user behavior and model (explained below). Due to this iterative nature, the *route choice process* has a significant impact on the computational speed at which a *dynamic traffic assignment* is made.

In the following we will briefly discuss the underlying of common practiced route choice calculation methods.

Starting with the notion of *rational user behavior*, where a traveler is supposed to behave in a rational way, we continue by assuming that travelers try to *optimize their (subjective or perceived) utility* of a translocation (in this case a trip by car).

In the classical psychological study of perception, *Thurstone's Law of comparative judgment;* the perceived level of a stimulus equals its objective level plus a random error. The probability that one object is judged higher than a second is the probability that this alternative has the higher perceived stimulus. When the perceived stimuli are interpreted as levels of satisfaction, or utility, this can be interpreted as a model for economic choice in which utility levels are random, and observed choices pick out the alternative that has the highest realized utility level. This connection was made in the 1950's by the economist Jacob Marschak, who called this the *random utility maximization hypothesis*, abbreviated to *RUM*.

When *maximizing individual utility* in terms of translocation or travel, this implies that a traveler chooses a route which maximizes his or her perceived individual utility. This utility can be expressed in (a combination of) different ways such as: travel time, travel distance, cost, comfort, scenery, dislike for road types or areas, event chaining etc. In this thesis we will mainly focus on travel time.

Using the RUM model to simulate route choices is plausible when:
- travelers learn the network conditions since a large portion of them will repeatedly make the same journey

- people can have access to information on the current (possibly different, unexpected) state of the network
- the day to day fluctuations are limited

However RUM is less or non plausible when:
- conditions differ greatly from the recurrent expected conditions (extreme event, incident)
- travelers are unfamiliar with the network
- the information they base their decisions on is of poor quality
- choices are no longer made deliberately (based on utility), but out of habit
- a traveler isn't that keen on changing routes
- choices are made based full network knowledge at all time

In the next paragraph we will discuss how the RUM model can be used to calculate routes.

4.2.3.1 Calculating routes

The routes used in traffic *assignment model* are usually based on calculated routes by means of a *shortest path algorithm*. When paths are expressed in time, a shortest path would mean the shortest travel time path. Because this path is the shortest it would maximize the utility of that trip.

There exist some variety of these algorithms but the most fundamental ones are *Dijkstra* (easy, fast) and *Floyd-Warshall (all pair SP)* algorithms. In general a *matrix* or *adjacency list* is filled with information regarding the *cost* to travel that link. The algorithm then uses these input structures to compute a shortest path between a given origin and destination.

Calculating routes can also become computationally demanding when dealing with large networks or dynamically changing network conditions. Consider a graph G consisting of *n* nodes and *m* edges, to find the *all pair shortest paths* using Floyd Warshall algorithms would run in $O(n^3)$, whereas finding the single source shortest path using Dijkstra would be done in $O(m \log n)$. The point is, calculating paths is computationally expensive.

RUM routes
If a random component would be added to the *link costs,* the *shortest paths algorithms* would in fact calculate routes with the highest perceived utility. The objective value of the utility would be the time it takes to travel that route under normal conditions. The random term would enable a subjective interpretation of this travel time; travelers may choose a route which is based on *their idea, or mental map* of the road network they intend to travel, which could be incomplete, distorted or heavily influenced by previous journeys. This route could very well not be the objective shortest time path at all.

Translating routes in a model
When *route choice* in a network is calculated by a shortest paths algorithm, this is represented as a set of consecutive *nodes* or *links* which should be followed for the desired OD pairs. This route can then be used in the assignment in two distinct ways, namely as:

i. Absolute routes, or
ii. Destination specific *split fractions*

Storing route choice as absolute routes is most common and enables a lot of flexibility and detail and used frequently in micro simulation.

Storing route choice as *destination specific split fractions* is more common in macroscopic models since they allow a fast calculation of flow distribution over nodes. A destination specific split fraction is illustrated best by the picture to the right.

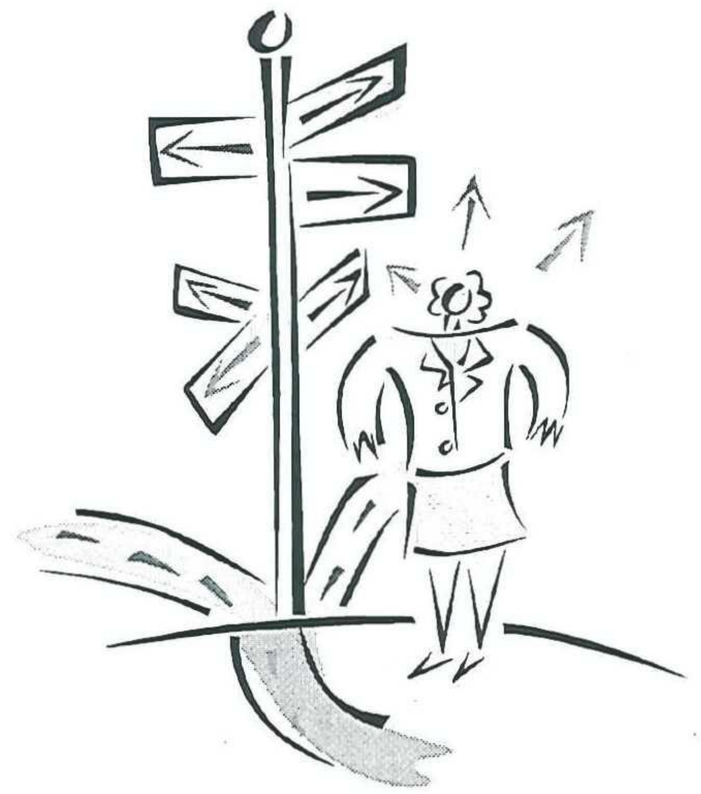


Figure 4-7 Destination specific Split fractions

Imagine *direction poles* situated at the end of *every* link in a network with an equal number of *signs* per pole to the total number of destinations in the network. For every destination, one distinct sign on the pole tells the traveler which link to take next.

The great additional benefit of destination specific split fractions is their 1-on-1 translation of *route guidance disseminated by means of DRIPS*.

An in depth discussion of *The foundations of Dynamic Traffic Assignment, the past, the present and the future* is provided in the excellent paper by S. Peeta and A.K. Ziliaskopoulos in [22.]

4.3 Objective function

Finding the right set of DRIP settings, means finding those setting which lead to an optimal assignment. But what is an optimal assignment?

From the different assignments already discussed previously the *system optimal assignment* would be the most likely type of assignment we want to achieve with the *route guidance*. This means we will be searching for a set of *route guidance DRIP settings*, which when used / followed by the travelers in the *assignment*, lead to a system optimum. The system optimum can sometimes become unfair because it allows a *few* to suffer for the *many*. In some cases a *constrained* system optimum is used, which will guarantee some minimal requirements, thus preventing exploitation.

To find this type of assignment we have to formulate some type of *network evaluation function* as the translation of what we consider being a good or bad system optimal assignment into a numerical value. Given this network evaluation, the basic idea of optimization is minimizing or maximizing this function under a given set of variables, something like: $\min f(x), x \in \Re^n$. From now on this function may be referred to as *objective function* or network evaluation function.

Finding exact formulations of objective functions in the literature is difficult. In most analytic models the system performance is expressed by means of the *link performance function* (the *cost* to traverse a link), where the objective function is the sum of all link performance functions in the network and should be minimized. This minimal cost state is found by means of an algorithm proposed by *Frank and Wolfe* which basically uses an iterative approach where an AON (all or nothing) assignment is repeated and used as the route choice until a certain stop criterion is met and the assignment is

minimal. Because link costs can be interpreted in many different ways it is able to handle a great deal of objectives such as, safety (speed), pollution / noise emission (density) etc.
The above information was taken from [23.]

Other measures have been proposed as well, for example the total travel time spent in the network (TTS) or average network speed. However, in the Netherlands new visions on traffic management and urban planning are devised with every new election and lead to much more intricately formulated demands which, if satisfied, are considered system optimal. To facilitate this kind of objective the network evaluation / objective function must become more versatile than the *link performance function* alone.

4.3.1 Calculating the objective function

In calculating a network evaluation score there are basically two possibilities:

i. The model and the objective function can be separated and the objective function is calculated after the assignment is made. (blue, continuous)
ii. Exploit the time dynamics within the model and calculate the objective function during the assignment process. (red, dotted)

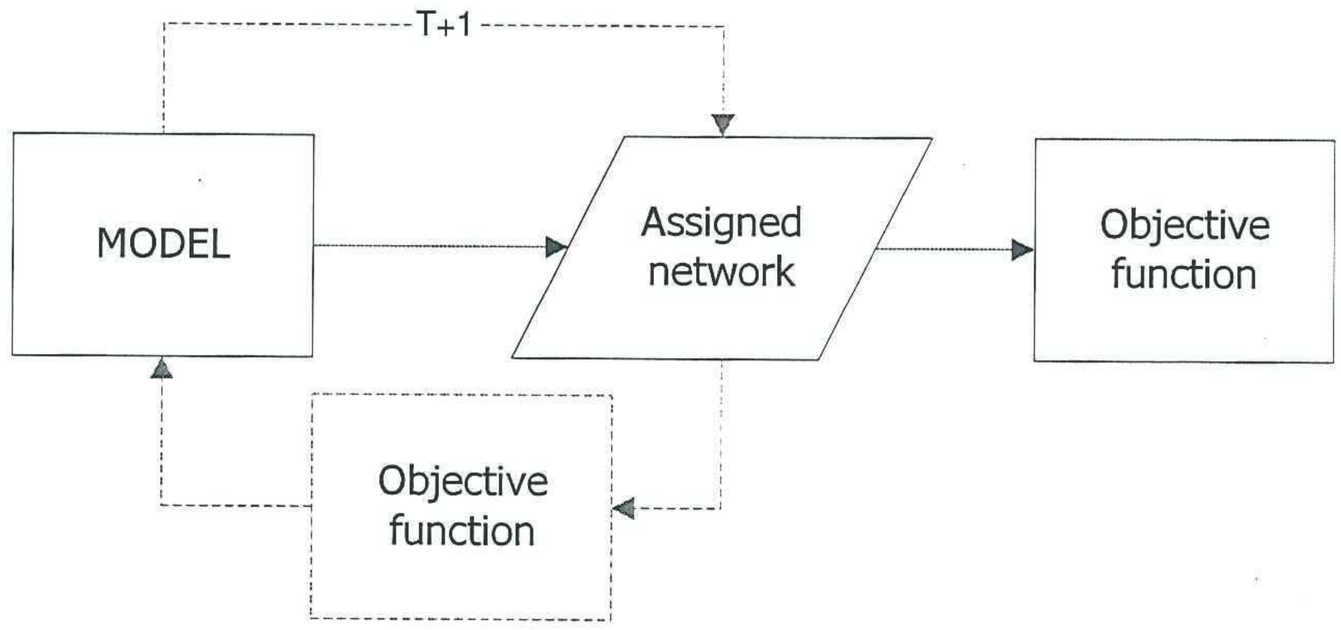


Figure 4-8 Different approaches in calculating a network evaluation score

4.4 Optimization

We will discuss the various approaches to optimization which could be applied in the case of generating optimal route guidance advice. We begin with the classification in the figure below:

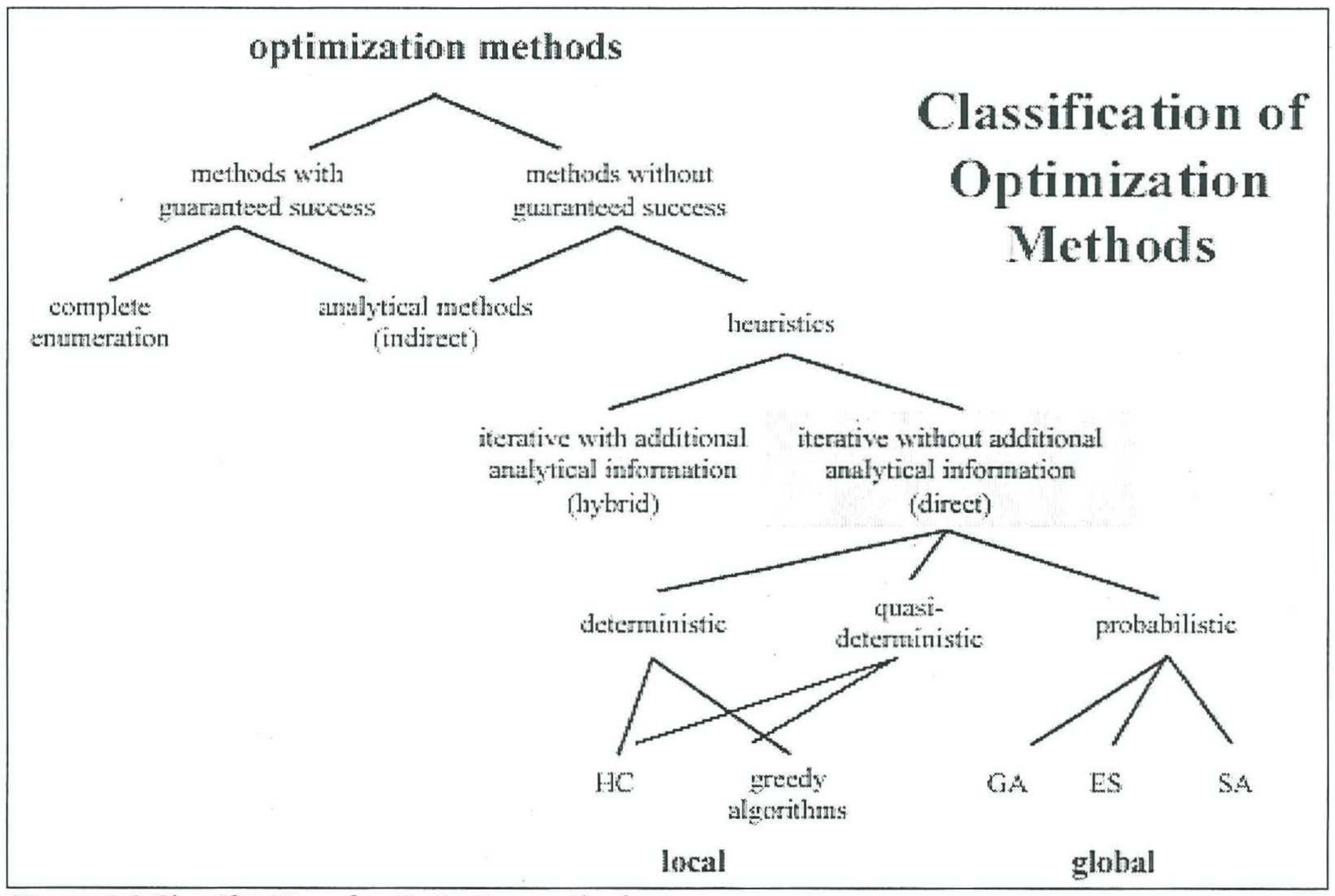


Figure 4-9 Classification of optimization methods

We see the methods being branched into several categories. We will discuss the division on the 2nd level in more detail.

Complete enumeration
By using complete enumeration to optimize a given set of DRIPS, it would mean that systematically all possible feasible combinations of DRIPS must be calculated in advance and evaluated in a traffic assignment model to compare them to each other. When adopting a *grid search,* the solution space is divided into a finite number of intervals which are evaluated. This is comparable to some instances of generating route guidance (Scottish interurban network) where a selection is made from a pre defined set of possible RG advices which can be displayed.

Analytical methods
Applying an (indirect) analytical method into finding optimal route guidance settings would start with modeling the dynamic traffic assignment in an analytical manner. This itself would prove difficult, but would even become more difficult if we where to incorporate the DRIPS as well. And finally, the system of equations would have to be solved by the various methods available.

4.4.1 Heuristics
When optimization is analytically difficult and enumeration impossible one resolves to heuristic. This in fact is a group of techniques aimed at finding optimal solutions by *searching* for them in a smart way.

Iterative with additional analytic information
If the translation of the problem is possible in an analytic manner, this fact can be exploited within iterative search heuristics. For example the existence of derivative/gradient information can be used in guiding the search process, like *Euler's method* or *Runge Kutta*.

Deterministic and quasi deterministic
These types of heuristics, for example hill climbing (HC) and *greedy algorithms* use a (quasi) deterministic approach into finding an optimum. This means that searching is guided in a deterministic way. A good example is the *grid search* which discretizes the search space into a set of points and calculates all objective values for these points to find an optimum. There is no guarantee however that the selected points will lead to a global optimum. Another example of deterministic searching are algorithms which are given an initial solution and a searching direction. The great benefit is that it is a quick way of finding an optimal solution. Its inevitable drawback however is the likeliness to find local optimum which is inherent to the fact that a search direction was added. In a figurative way of speech it means we tell the heuristic where to look and it looks there and finds an optimal solution, but it will not look in other directions. We can say with certainty that the found solution is optimal for the *local region* we told the algorithm to look, but we don't know if it's a global optimum as well.

Probabilistic
Probabilistic heuristic methods are driven by a chance or at least randomized search process. Examples are simulated annealing, genetic algorithms and evolutionary strategies. All these methodologies share the fact that they span their search in different directions within the solution space. These directions are the result of the chance process and make these kind of heuristics better equipped into finding a global optimum. Their drawback however is, due to the fact that their magnitude of search is far greater, they use a lot more computation time.

4.5 Network control methodology

In this chapter we discuss some of the different control approaches which could be applied in order to succeed in a generic approach to a route guidance strategy. In general there are three categories of *mathematical control* in use nowadays [24.]:

- Classical (and state-space) control is a mature field that utilizes rigorous mathematics and exact modeling to exercise precise control over the dynamic response of a system.
- Neural control overlaps much with classical control. It is less precise in its formulation yet may yield better performance for certain applications when accurate modeling of the relevant phenomena is prohibited.
- Fuzzy control is less rigorous, but is a simple approach which generates adequate results for many problems.

In the next paragraphs we take a closer look at the most important *classic* control approaches and discuss those most dominant in traffic management systems.

4.5.1 Classic control methodologies in traffic management

In the category of *classical and state space control*, the following approaches are used for traffic control in general. [15. CH4]

Open loop control
Open loop methodologies do not incorporate information regarding the current state of the system, nor do they use predictions to determine the resulting control law. That is, the open-loop control laws are predetermined off-line, frequently based upon historic data. (signalized intersections)

Feedback control
Feedback control only uses estimations of the current state to determine the control laws. Neither predictions from historic data, nor predictions from traffic models are employed. The estimation of the current state is used directly to compute the control law, for instance, the display of instantaneous travel times or delays.

Predictive control
Within predictive control, current state information is used, frequently in conjunction with historic data, to predict future network states. Control laws are then based upon future network conditions to improve their usability. For instance a VMS displaying experienced travel time.

Optimal control
Optimal control is a form of predictive control where an objective function is optimized explicitly. This usually is an iterative process where a candidate control law is formulated and then judged using the prediction model. When the control law is considered to maximize the objective function, the control law is passed to the actuators and the system is optimized.

4.5.2 Coordinated and integrated traffic control systems

In the literature basically three approaches exist for coordinating traffic control measures: model-based optimal control methods, knowledge based methods and methods that use simple feedback or switching logic.

- Model based optimal control methods

Model-based optimal control techniques use a model for predicting the future behavior of the traffic system based on:

i. Current traffic state
ii. Expected traffic demand on the network level, possibly including origin – destination relationships, and other possible external influence, such as weather conditions
iii. The planned traffic control measures

Since the first two items cannot be influenced on the short term, the future performance of the traffic system is optimized by selecting an appropriate scenario for the traffic control measure. Methods that use optimal control or model predictive control take the complex nonlinear nature of traffic explicitly into account.

This is approach is also referred to as the *rolling* or *receding horizon* approach. The following citation explaining the rolling horizon approach is taken from [26. pg 53.]

> The rolling horizon approach is a practical method for real-time (or quasi-real time) demand-responsive control. It is especially suited for problems requiring future demand information for the entire planning horizon, which is difficult to obtain reliably. The basic idea of the rolling horizon approach is to use currently available information, and near-term forecasts with some degree of reliability, to solve a problem on-line while preserving the effectiveness of the computational procedure in determining "good" control strategies.

- Knowledge based methods

Knowledge based traffic control methods typically describe the knowledge of the traffic control system in terms that are comprehensible for humans. Via reasoning mechanisms the knowledge based system generates a solution (control measure) given the current traffic situation. A typical motivation for these kind of systems is to help TMC operators to find a good (not necessary the best) combination of control measures.

- Control parameter optimization

In this approach a relatively simple control law is used for a certain traffic control measure. The control parameters are found by simulating a large number of scenarios and optimizing the performance.

The contents of this paragraph are described in more detail in [25. §2.5]

4.5.3 Characteristics of calculated control

The control methodology chosen determine some of the characteristics of the total control process, such as:

i. Quality or effectiveness; *Rubbish in is Rubbish out*
ii. Speed of the system; Control *must be calculated very fast to use online*
iii. Characteristics of the solution;
    a. Existence; *Is there an optimal solution?*
    b. Uniqueness; *is the found solution unique or do others exist?. (global / local)*
    c. Sensitivity; *How sensitive is the solution to its input variables?*
    d. Robustness; *How vulnerable is control to measurement inaccuracies or disturbances?*
    e. Reliability*; is there a graceful degradation when the system breaks down?*
iv. Generality; *Is the control system applicable to different networks?*
v. Complexity vs. simplicity;
vi. Distributed or central calculation?; *parallel processing.*

As to these characteristics one could state at forehand that an effective *route guidance strategy* must be able to calculate the guidance in a quick way to make it applicable online. In addition, the methodology must not be highly sensitive to some of its input parameters rendering it a nightmare to calibrate for a practical application. The resulting methodology should be robust in the sense that the components which are used can be improved later on without the necessity to redesign or alter the methodology.

5 Specification of the route guidance generation methodology

In this chapter a series of choices will be motivated, which will result in the final *control methodology* and the elements used. Elements such as: the exact type of model, optimization method, and the objective/evaluation function. Their different possibilities and most important aspects have already been discussed in the previous chapters, which leaves this chapter to decide upon the exact nature of the methodology. In the next chapters, the different elements will be discussed in more detail.

5.1 Optimal control

We have already discussed different control approaches in 4.5 and we must now make a choice as to what kind of control loop we will adopt. Basically we can choose between *predictive control* and *optimal control (*in a rolling horizon context, figure 4-4).

Control objective
This decision also depends on the objective we want to achieve with our route guidance. The objective for the route guidance generated will most likely be either a (constrained) system optimum or a user optimum. In 4.3 we have already stated a *system optimum* would be the better objective. This sounds fair since the calculation and dissemination is basically made possible by public funding and should therefore be put to use for a public cause. But perhaps more convincing is the idea that policy makers should be able to define the route guidance objective as a weighted sum of the interest of different stakeholders.

Control methodology
If we strive for user optimum, we could already approximate this roughly by means of feedback control (as is done now), or make a qualitative better approximation by means of predictive control. If we where to adopt optimal control we would get best results. Approximating a system optimum by means of feedback or predictive control is much more difficult in comparison to a user optimum and if political issues would be included in the objective functions this would even become more difficult. From this point of view adopting an optimal control methodology seems logical.

For real life implementation, the *model predictive control* (MPC) or *rolling horizon* seems like the best type of control. This is due to the fact that *normal optimal control* has an *open-loop* structure which means that disturbances must be known on forehand and the traffic model has to be very accurate to ensure sufficient precision in assignment for the whole simulation. With the MPC approach, the current network state and demand are fed back into the system which makes it possible to incorporate disturbances. Another benefit is the fact that due to this periodic feed back with the current state, the period, used for near future prediction, for generating DRIP settings could remain short which makes computation faster and better suited for online application. For this thesis we will focus on the *route guidance generation model* part of this approach which can be seen in the picture below.

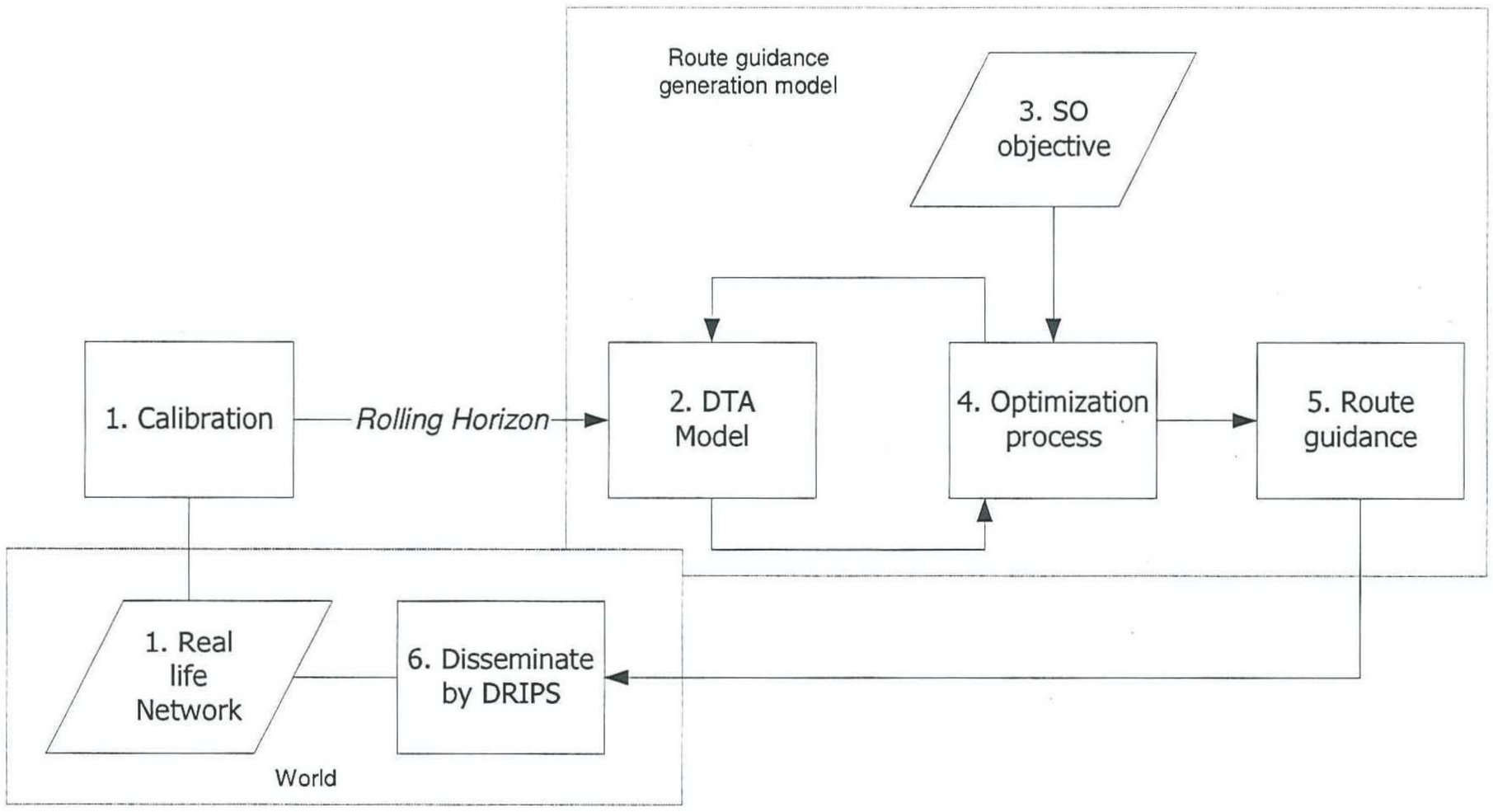


Figure 5-1 Model predictive / Rolling Horizon control methodology

In the following paragraphs we will discuss the exact nature of the elements 2 – 6 in the diagram above.

5.2 Macroscopic dynamic traffic assignment model

The general operation of traffic assignment models was described in 4.2, we will now have to decide upon the type of model that will be used in the methodology. To do so we will take a look at the necessary features that must be present in such a model.

5.2.1 Necessary features

Dynamic
The nature of the *route guidance advices* we would like to generate is dynamic in time. To compute such advices in an *optimal control methodology* relies on a certain level of realism in the assignment, a level which can only be achieved by dynamic modeling. However, dynamic modeling also requires dynamic input in the form of an OD, which is difficult to establish, and a dynamic model is more difficult to calibrate in time. But still, the benefits of modeling the dynamic effects of route guidance outweigh by far its implementation problems.

Macroscopic
For the purpose of generating route guidance the DTA model must be able to evaluate the effect of disseminated guidance on the one hand and be able to somehow support the generation process on the other hand. Because of the iterative nature of the control methodology chosen the model must be also be fast in terms of computation time. In addition, because the route guidance we want to generate will have to be coordinated on a network scale, instead of a local traffic control measure, the model must be able to deal with medium sized networks. It is not so much the detail of the assignment that is important, but more the macroscopic effects of the assignment. Which links are blocked, where do *queues build up*, *spill back*, or *dissipate* and as a result what is the effect on the network (spare) *capacity* and *alternative routes*. These last features are particularly well represented in macroscopic modeling and less in meso or micro modeling. Given all the considerations above the model will be using a macroscopic approach.

Route choice
The route choice process and DRIP modeling are closely related to each other. We have shown in 4.2.3 that the implementation of route choice by means of *destination specific split fractions* is a great opportunity for modeling DRIPS and very well implemented within macroscopic modeling.

User behavior
User behavior is a very important factor in the success of using route guidance to improve network conditions as we have seen in 2.5.1.1 the effect of route guidance on users can be described on the short term by means of *user compliance* and on the long term by *user faith* in the system. For our approach we consider the first to be most important and should be implemented. Being able to model this correctly would be of great value to the quality of the calculated route guidance. It would be interesting to know how effective the route guidance would be under the condition *that a certain percentage of users would comply* with the guidance. The long term effects of *user faith* in the system can be incorporated into the *control methodology* by using different values of *compliance* over time.

5.2.2 Build a custom model

At this point we are left with the decision to use an existing model or build a custom one. The available model considered to use is METANET. METANET is a very fast macroscopic 2nd order model with a long range of applications. We will now discuss some of the drawbacks and advantages in using METANET.

Drawbacks
The maximum cardinality of a node in METANET is two, and therefore only able to handle bifurcation nodes. Off course this can be compensated for by using dummy links, but this would require elaborate network modeling. The main problem however is, if one tries to model a DRIP located at a node with a cardinality of three or higher (three different directions) this would have to be treated different from a DRIP at a bifurcation node and complicate the universality of the generation process.

Another important drawback is the fact that the simulation process itself could provide useful information to use in the iteration process. Using an existing model means that this *process information* must be reconstructed afterward. As we have seen in 4.3 the objective function within the transportation context can be calculated afterward or during the simulation process. This latter option could prove very difficult to implement in an existing model.

Advantage
METANET is an established model with well documented behavior and various applied case studies. Developing a custom model would consume a lot of time and effort to validate which is unnecessary and which could qualitatively generate worse results.

Despite the extra time needed to develop a custom model and the uncertainty about the quality of its assignment, the benefits gained (from the process of building) outweigh the drawbacks. Building such a model is easier than it sounds, a relative simple but effective macroscopic model can be created if based on the *kinematic wave model*. This type of model is a combination of the *conservation of vehicles equation* and the *fundamental diagram* solved by means of numerical approximation called the *Godunov scheme*.

5.3 Network evaluation; the objective function

In this paragraph we decide upon what kind of objective function should be used as an evaluation of the resulting traffic assignment given a set of DRIP settings. As discussed earlier the formulation of what is considered optimal cannot always be expressed by the *link performance function*, due to complex political/policy motivated demands. For example, if an alternative exists which would allow

people to travel through the city instead of traveling the motorway more complex restrictions are likely to be added. Restrictions like: the usage or avoiding of specific links, nodes or routes, a minimum amount of available spare capacity after assignment, the availability of emergency services etc.

The objective function we are looking for should therefore have an *open structure* to accommodate such complex demands, but for now we focus on the *development* of the *methodology*, and not so much on the exact specification of a realistic objective function. We need a simple basic objective function which allows the methodology to adequately *evolve* usable route guidance.

The main problem with finding this basic network evaluation function is the need to formulate *constraints* as well to avoid exploitation. Because the generation process of *route guidance* is steered by the *scalar value* of the objective function (network evaluation), all assignments with higher values are considered better. In almost all cases when an objective function was formulated without constraints the genetic algorithm *evolved* route guidance settings which led to some form of *exploitation* to increase the score.

Some experimenting had to be done in finding a workable objective function. During this search the *methodology* has been applied to *all links* in the network, not only the DRIP links. By doing so, drawbacks of the proposed objective function could be identified more easily (earlier). For example:

Average speed
When using average speed as an indicator, behavior was observed where links close to the origin nodes where blocked. This effectively had the result that fewer cars where able to enter the network and as a result most links could be traversed without congestion. So the average speed increased at the cost of few congested links which effectively prohibited cars accessing the network.

TTS
Frequently the total ravel time spent (TTS) in the network is put forth as an indicator. When we applied this type, the same behavior was observed as with the average speed. The total travel time spent in the network was minimal as a result of fewer cars being able to access the network.

In both cases a constraint should have been added which would guarantee travelers *access* to the network by penalizing the evaluation score if this was prevented. Formulating such constraints is not easy, for one what kind of penalty should be added? And, must there be some type of proportionality when at a certain point the marginal benefit of allowing another vehicle to enter the network could become negative?

Taxi tip
Another type of objective function was used where each *arrival* at the destination added 50 points to the evaluation score and each *experienced minute of delay* per vehicles reduced the evaluation score with 2.5. This was a reasonably good evaluation but led to very *safe* types of assignments. (the slightest form of congestion plummeted the evaluation score, so the evolution process followed a *safe path* which sort of optimized the calibrated assignment instead of actually re-routing)

5.3.1 Generalized total travel time spent

The resulting objective function used in the case study is simple and based on two variables calculated during the assignment process. (This last feature was one of the reasons to build a custom model)

Given the following simulation variables:

dt Time step [hr].

$q_{i,d,t}^{in/out}$ in/out flow of link i at time t heading for direction d in [veh/hr].

$Q_{o,d,t}$ Dynamic OD demand from origin o heading for destination d at time t in [veh/hr].

Let

$$L_{i,d,t}^{Travellers} = L_{i,d,t-1}^{Travellers} + q_{i,d,t}^{in} dt, \tag{1}$$

Be the *total number of travelers* which have accessed link i heading for direction d at time t and can be calculated during simulation of the traffic assignment by a summation.

Cumulative sum of desired departures
The cumulative sum of desired departures, is equal to the cumulative sum of travelers that *want* to access the network, expressed by the dynamic OD demand. So vehicles that are *denied access* to the network, because connecting links to the origin are *saturated,* are still counted in the cumulative sum of desired departures.

$$q_{i,d,t}^{in} = Q_{o,d,t} \quad \forall i, o \in Origins \tag{2}$$

In (2) we define i to be a member of the collection of origin links and o to be a member of the collection of origin nodes. i and o are uniquely linked, in a way that every origin node has one origin link, the same holds for the destination nodes and links). By substituting (2) in (1) we can calculate the desired cumulative sum of departures.

$$D_d = \int_0^T L_{i,d,t}^{Travellers} dt, \forall i \in Origin\,Links \tag{3}$$

Cumulative sum of arrivals
In a similar manner we define the cumulative sum of arrivals as the cumulative summed number of vehicles which have reached their destination d the following formula:

$$A_d = \int_0^T L_{i,d,t}^{Travellers} dt, \forall i \in Destination\,Links \tag{4}$$

This is basically the same formula, but except for the *desired demand* we cumulatively sum the *actual input* into the destination link.

On the next page, we will illustrate the relation between the $q_{i,d,t}^{in/out}$ in the network, the total number of travelers at any time t $L_{i,d,t}^{Travellers}$ and the cumulative summed property D_d and A_d

In the top figure we have shown the $q_{i,d,t}^{in/out}$ at any time *t* for a fictive situation:

The *demand* corresponds to the number of vehicles which want to *enter* the network ($q_{i,d,t}^{in}$) in red and the *arrivals* are the number of vehicles which have arrived at their destination and leave the network ($q_{i,d,t}^{out}$) in green.

The second figure displays $L_{i,d,t}^{Travellers}$, the summed property of inflow and outflow $q_{i,d,t}^{out}$ at any time t during the simulation. (summed demand and arrival)

The third figure is the summation of $L_{i,d,t}^{Travellers}$ at any time t during the simulation and represents the *cumulative summed demand* D_d *and arrival* A_d at any time t.

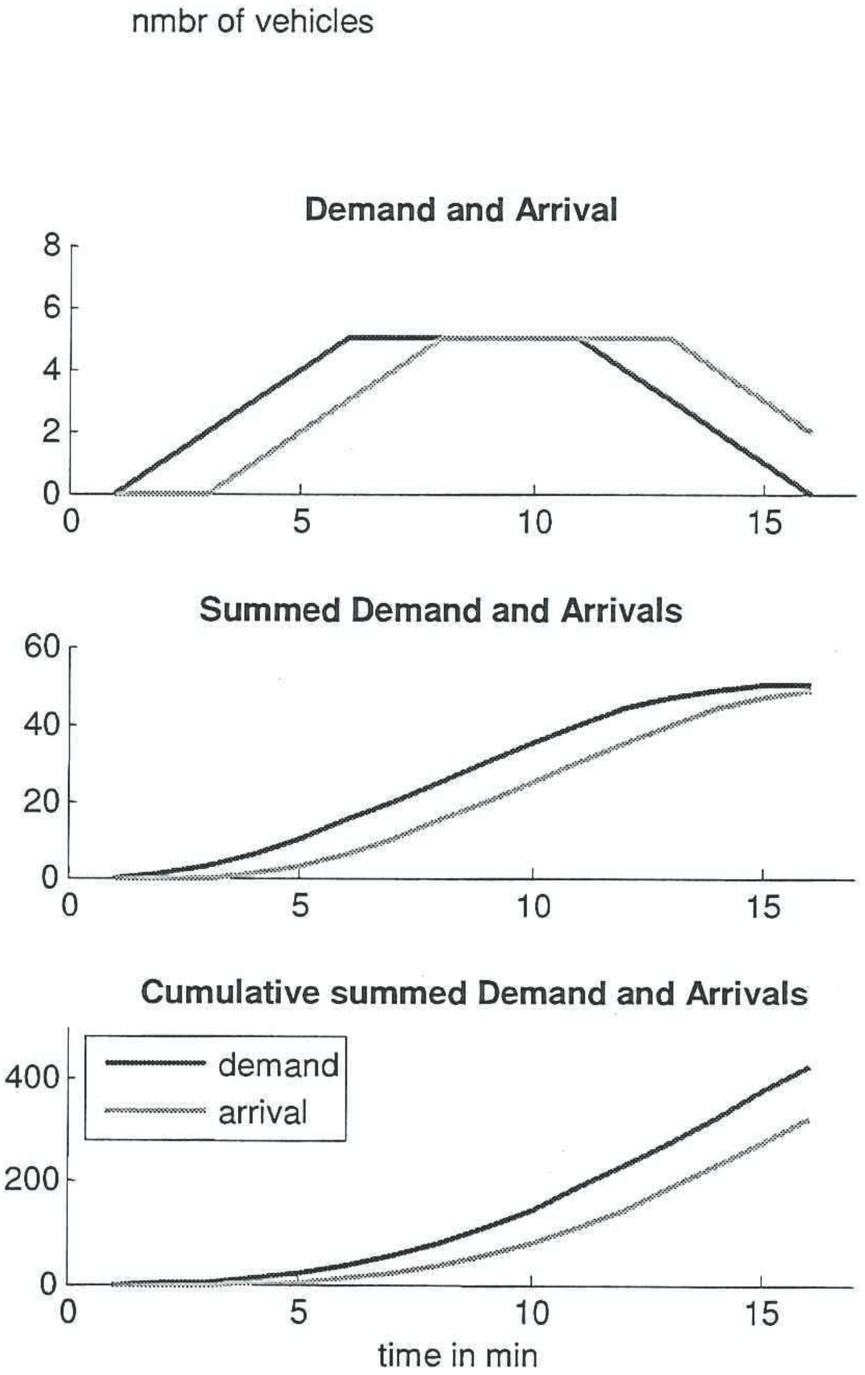


Figure 5-2 Generalized total travel time spent

Since the *cumulative sum of desired departures* is a value solely determined by the dynamic OD matrix, it is the *cumulative sum of arrivals* which influences the score.

If we take the difference between D_d and A_d, at the end of the simulation time T, we strive in minimizing this value. The sooner the more people *arrive* at their destination, the *higher* the value of the cumulative sum and the smaller the difference.

In the picture above[1], the enclosed area in subplot 2, contains the *generalized total travel time spent* in the network at different time T. This value is the same as the difference between the cumulative summed demand and arrival value at any time t (subplot 3).

Depending on the simulation duration we calculate the difference between both curves at time *t=T* and consider the difference the *generalized gross travel time* which we try to minimize. In general the philosophy behind the objective function is:

"The sooner the more people would have arrived at their destination, the better network conditions they would have to have encountered"

Note that the *generalized gross travel time* at time t=T formulated as $D_d - A_d$ is still a vector, and contains the specific values *per destination*! It should be understood that this leaves room for

[1] The picture depicts the development of the generalized gross travel time in a fictive network over time if one was to start with an empty network.

tremendous possibilities in terms of specifying weight factors per destination, where high prioritized destinations could be weighted heavier.

During the development of our methodology we keep it *small and simple* (KISS) and do so by summing the total value of the *generalized gross travel time* for al destinations. However this still leaves one more problem to be solved which is related to the fact we try to *maximize* our objective function instead of minimizing.

By calculating the quotient between a *constant absolute value* and the *summed generalized gross travel time*, we have found our objective function. For this constant value it was found that making a *sum* of the *cumulative sum* of the *total OD demand* proved pragmatic.

Let (5) resemble the total OD demand,

$$Q_0^{Total} = \sum_{O,D} OD \tag{5}$$

With this we define our objective function as:

$$F_{OBJ} = \frac{\int_0^T t Q_o^{Total}\, dt}{\sum_d D_d - \sum_d A_d} \tag{6}$$

Hidden congestion
By taking the *travel desire* into account instead of the actual travelers in the network, *hidden* congestion is expressed in the objective function. Would this have not been done, network conditions where *origin demand access* to the network is limited e.g. (and thus effectively improving network conditions since fewer vehicles result in higher speeds) could lead to higher scores. (this behavior was observed in test networks, where the split fractions related to the origin links where optimized as well).

Exploitation
The objective function as formulated, *where the summed generalized gross travel time* can be interpreted as the *total utility of all travelers*, does however allow *exploitation* since it is a classical formulation of total utility. If a few where to suffer for the many, but this would increase the objective function, it would be considered beneficial.

The problem with exploitation can however be partially addressed by formulating threshold values per destination which should be met or otherwise would result in a penalty. By doing so we use a modified[2] version of the definition of *pareto's utility optimization*[1] instead of the classic utility optimization. [27. pg 231]. A little exploitation is allowed for the benefit of the system but too much exploitation is not allowed and will be penalized.

---

[1] A Pareto optimization is a change from situation *a* to *b* in such a way, no one in *b* is worse of than in *a* while at least someone in *b* is better off than in *a*.
[2] If one was to formulate threshold values for being worse off, for example 5 minutes more delay in situation *b* compared to *a* is the maximum acceptable and would not be considered worse off. Every delay above this would be penalized in such a way, the objective function value would plummet.

5.4 Direct heuristic optimization using an evolutionary strategy

At this point we have selected the control methodology and decided upon a macroscopic analytic model, what remains now is the specification of the exact type of optimization technique. To do so, we first take a look at the *complexity of the problem* we are dealing with.

Complexity of the problem
If we are to consider the Rotterdam network we could identify 6 different points where DRIPS can be placed. To keep things simple, say we provide route guidance for one hour where every advice must be constant for at least one minute and we use all three lines on a DRIP to give advice for three different directions. This is off course a theoretical case but it serves to illustrate the complexity of the problem. If we are to calculate the number of possibilities we could say that one line of text can have 61 possible settings within one hour. So one DRIP could have 61^3 possibilities and six DRIPS will have something like 61^{18} or 136753052840548005895349735207 88 number of possibilities. Off course they are not all valid and a lot could be discarded at forehand yet it serves to say that finding one solution within these possibilities is like finding a needle in a haystack. Given this solution space, it is wise to say that efforts in developing a methodology will not be aimed at an enumerative method.

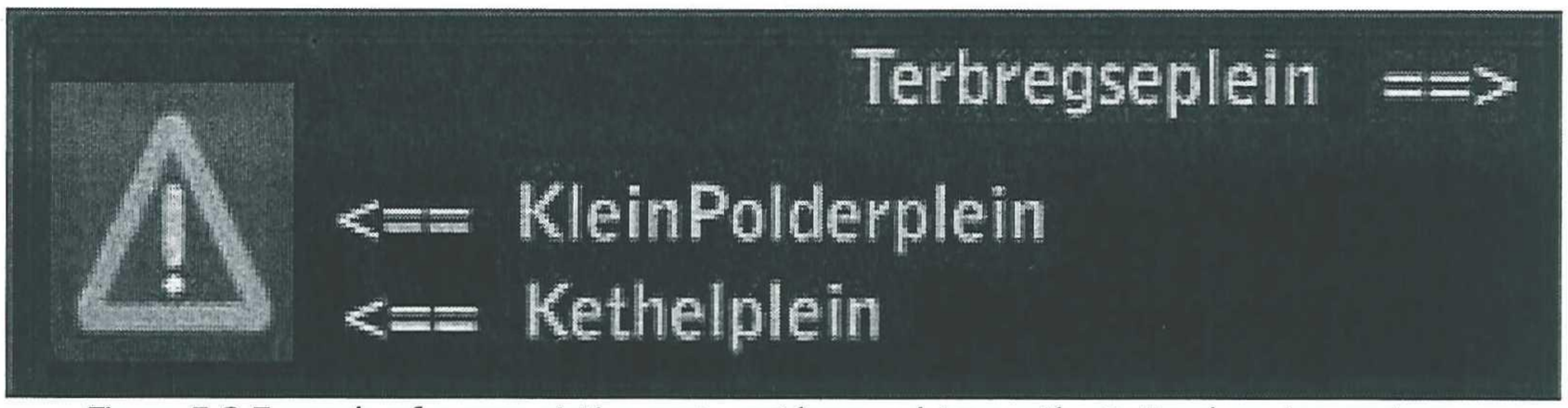


Figure 5-3 Example of a prescriptive route guidance advice on the Rotterdam ring motorway

Iterative vs analytic approach
If we consider the various approaches depicted in *Figure 4-9 Classification of optimization methods* we are left with the choice to decide between an *analytic* or *heuristic* approach to the optimization problem. Finding the DRIP settings based on a *one shot analytic strategy* means that the problem must be translated into an analytic formulation which would then be solved, this is most unlikely. And if this was realized, the assumptions needed to solve the analytic formulation would very likely limit the usability of the solved assignment. In addition, the model that is build (chapter 6) uses a numerical approximation and provides no real derivative information that could be used. We can therefore conclude in using an *iterative approach* with a *direct heuristic* to search for optimal route guidance advices.

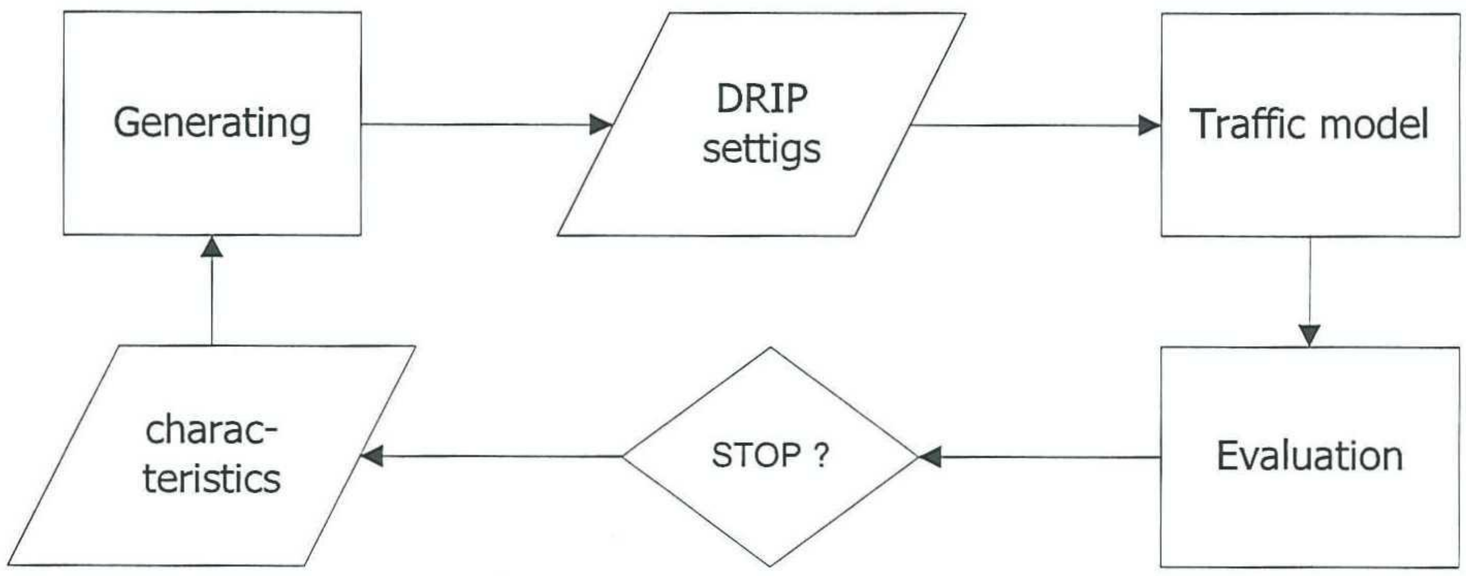


Figure 5-4 Iterative optimization

5.4.1 Evolutionary algorithms

Several *direct heuristic* methods could be applied; namely *simulated annealing* or *evolutionary algorithm.* In this paragraph we will show why the *evolutionary algorithms* are the most suitable for this kind of optimization.

The different DRIP settings we look for should be optimized globally. This might seem trivial, but in all the possible combinations of DRIP settings, for example the six DRIPS in Rotterdam, it is very likely a lot of *local optima* will exist. Our heuristic must be able to look for the *global* or *near global* optimum in this huge search space and therefore preferably be chance driven. This leaves us with the choice between an *evolutionary algorithm* and *simulated annealing*.

Evolutionary Algorithm
An evolutionary algorithm (also EA, evolutionary computation, artificial evolution) is a generic term used to indicate any population-based metaheuristic optimization algorithm that uses mechanisms inspired by biological evolution, such as reproduction, mutation and recombination (see genetic operators).

Candidate solutions to the optimization problem play the role of individuals in a population, and the cost function determines the environment within which the solutions "live" (see also fitness function). Evolution of the population then takes place after the repeated application of the above operators.

Specific examples of EAs are given below. Most of these techniques are similar in spirit, but differ in the details of their implementation and the nature of the particular problem to which they have been applied.
Genetic algorithms - This is the most popular type of EA. One seeks the solution of a problem in the form of strings of numbers (traditionally binary), virtually always applying recombination operators in addition to selection and mutation;
Genetic programming - Here the solutions are in the form of computer programs, and their fitness is determined by their ability to solve a computational problem;
Evolution strategy - Works with vectors of real numbers as representations of solutions, and typically uses self-adaptive mutation rates;
Evolutionary programming - Like genetic programming, only the structure of the program is fixed and its numerical parameters are allowed to evolve.

Because they do not make any assumption about the underlying fitness landscape, it is generally believed that evolutionary algorithms perform consistently well across all types of problems (see, however, the no-free-lunch theorem). This is evidenced by their success in fields as diverse as engineering, art, biology, economics, genetics, operations research, robotics, social sciences, physics, chemistry, and others.

Simulated Annealing
Simulated annealing (SA) is a generic probabilistic meta-algorithm for the global optimization problem, namely locating a good approximation to the global optimum of a given function in a large search space.

The name and inspiration come from annealing in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. The heat causes the atoms to become unstuck from their initial positions (a local minimum of the internal energy) and wander randomly through states of higher energy; the slow cooling gives them more chances of finding configurations with lower internal energy than the initial one.

In the simulated annealing (SA) method, each point s of the search space is compared to a state of some physical system, and the function E(s) to be minimized is interpreted as the internal energy of the system in that state. Therefore the goal is to bring the system, from an arbitrary initial state, to a state with the minimum possible energy.

At each step, the SA heuristic considers some neighbours of the current state s, and probabilistically decides between moving the system to state s' or staying put in state s. The probabilities are chosen so that the system ultimately tends to move to states of lower energy. Typically this step is repeated until the system reaches a state which is good enough for the application, or until a given computation budget has been exhausted.

The neighbours of each state are specified by the user, usually in an application-specific way. For example, in the traveling salesman problem, each state is typically defined as a particular tour (a permutation of the cities to be visited); then one could define two tours to be neighbours if and only if one can be converted to the other by interchanging a pair of adjacent cities.

(Source: http://wikipedia.org/)

The nature of *evolutionary algorithms* is better equipped, or at least judged so by the author, for generating DRIP settings compared to *simulated annealing* for a number of reasons. E.g.

- the *mutation operator* can be used directly to input possible and smart *DRIP settings* into a new solution.
- the ability to use different *DRIP scenario's* as parents to form a new DRIP scenario could yield great profits in computation time.

- the possibility to implement *evolutionary strategies* in a parallel computing scheme which greatly increase overall calculation speeds as demonstrated in [28.]. (Although SA can be implemented parallel as well),
- the flexibility offered by the different types of operators (selection ,crossover, mutation) allow a lot of opportunity for tuning and tweaking to the particularities of the problem at hand.

Genetic algorithms and evolutionary strategies do not differ that much. In essence, the usage of real parameters and characteristics of the solution to steer the generation process are considered to be an evolutionary strategy.

5.5 Modeling prescriptive route guidance

One of the last remaining elements in the chosen methodology is how to *model* the *prescriptive route guidance* itself.

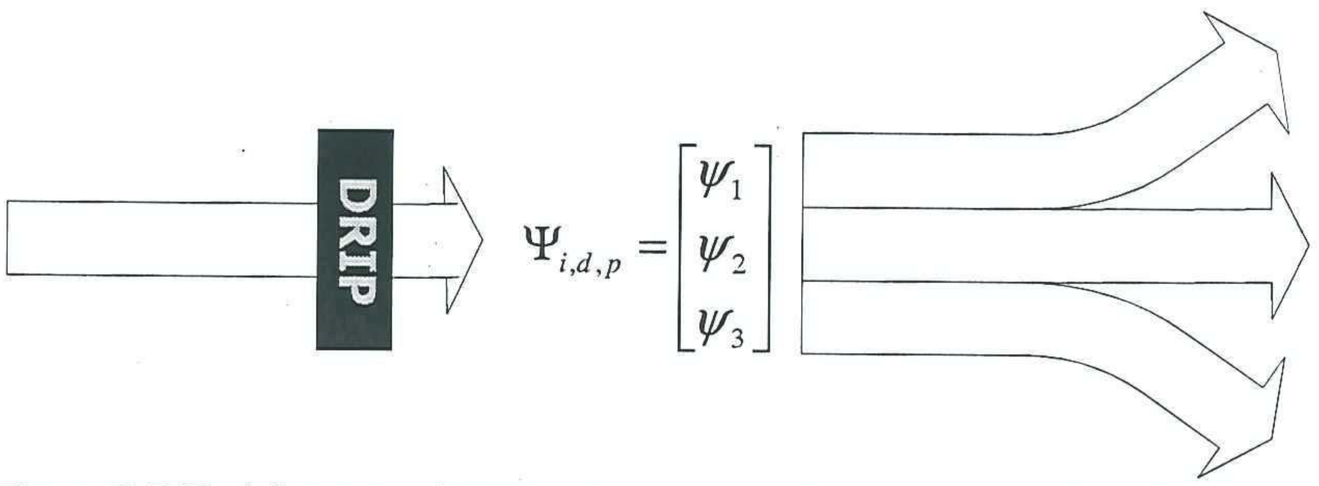


Figure 5-5 Modeling actual DRIP settings using destination specific split fractions

If we look at the picture above we see a theoretic intersection which has 1 incoming and 3 outgoing links. The vector $\Psi_{i,d,p}$ is a vector containing the *split fractions* for the considered input link *i* for traffic heading to direction *d.* in simulation period *p*. These *split fraction vectors* are present at ALL links in the network, for ALL destinations for ALL periods.

Now if this particular split fraction would be $[1\ 0\ 0]^T$ the traffic would turn left, or in the case of $[0\ 1\ 0]^T$ it would go straight through. But why stop here, split fractions can become real values as well, as long as the sum equals 1. E.g. the split vector $[0.2\ 0\ 0.8]^T$ would send 20% of all the traffic left and 80% of the passing traffic right. By simply changing the split vector $\Psi_{i,d,p}$ at the (pre) specified DRIP locations for destination(s) to which the DRIP has impact we can model its effect. The *genetic algorithm* is used to *generate* the split vectors $\Psi_{i,d,p}$ for all DRIP locations for all periods and evolve the network assignment into an assignment with a higher score. As we have stated earlier on, the modeling of user behavior is very important. In the next chapter we will show how to deal with user compliance to the prescriptive information.

5.5.1 Complexity of the problem

The adopted approach did increase our solution space a bit in regards to finding a set of dynamic DRIP settings. It is now based on the *scalar resolution r* of the split fraction, the *length* of the split vector *l*, the number of periods *P* for which to calculate, the number of DRIPS in the network with two alternative routes L_2 and the number of DRIPS with three alternative routes L_3. Given the fact we still provide guidance for three destinations per DRIP we get the following formula for our solution space.

$$n = \left\{ \left(\frac{1}{r} + 1 \right)^{3L_2 P} * \left(\sum_{i=1}^{\frac{1}{r}+1} i \right)^{3L_3 P} \right\} \tag{7}$$

Let's say we use a resolution *r=0.1*, we calculate guidance for a total of *P=12* periods (three hours divided into 15 minutes), we have two DRIPS with three alternatives $L_3=2$ and four drips with two alternatives, $L_2=4$. We'd find a solution space with size: $n = 11^{3*4*12} * 66^{3*2*12}$; This would be something to power e+280, for comparison there are estimated to be about 1.0E78 atoms in the universe.

5.6 Disseminating calculated route guidance

The last problem remaining is how to disseminate a real valued split fraction by means of DRIPS. If we find an optimal effective DRIP setting on link *i* for a given period *p* like $\Psi_{i,d,p}^{eff} = [0.5\ 0.2\ 0.3]$ we can translate this in the following way. Let's say we have a period *p* of 10 minutes for which the DRIP settings have been calculated. We disseminate the following information in time:

For the first five minutes we could say:

For the following two minutes we would say:

And for the last three minutes we would say:

To attain a better *spread* of flow over the directions we instead of assigning the traffic to the different directions in consecutive order, we could also vary this to something like [3x1 1x2 1x3 2x1 2x3 1x2] (three minutes advice 1, 1 minute advice 2....) Note that these DRIP settings all apply to *one line* of text only. The *route guidance* for two other destinations could be provided simultaneously.

5.7 Summary of the methodology

Consider the diagram above where the resulting optimization scheme is displayed. Note that this is the *model* part in the rolling horizon approach as discussed in 5.1.

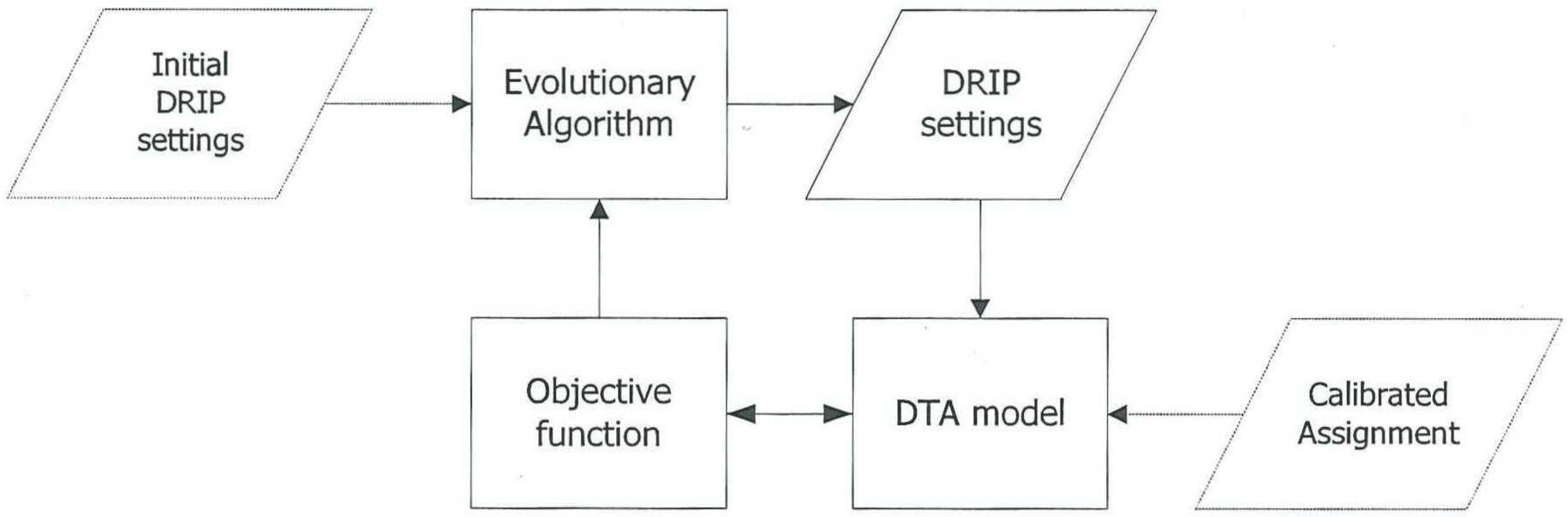


Figure 5-6 Route Guidance generation methodology

We create a macroscopic dynamic traffic assignment model where the *normal* route choice is simulated by means of *destination specific split fractions*. We then create a *network model* and dynamic OD matrix of a motorway network for which we want to calculate *prescriptive route guidance* for a number of *pre defined DRIPS.* We then calibrate this network to match the real life situation. (this is in fact a *one time* state estimation in the rolling horizon context) and give this assignment a quantitative score using the *generalized travel time objective function*. Now we have a base for the generation process using the evolutionary algorithm.

The algorithm changes the split fractions at the DRIP locations for a number of destinations, or collection of destinations, for which we want to provide the prescriptive route guidance. Under the assumption that: a *certain percentage* of users will follow the advice and the *routes they follow are correctly modeled*, we let the algorithm evolve DRIP settings, which lead to higher evaluation scores in terms of the generalized travel time.

In the initial state, we use DRIP settings based on the normal route choice people would make without guidance. These are already calculated in the calibrated assignment and provide a good starting point.

6 DSMART traffic model

In this chapter we will discuss the DSMART model, *dynamic stochastic macroscopic assignment of road traffic model,* which has been developed for this thesis. To do so, we will discuss the overall operation of the model in the next paragraph and use this as a guideline for a more in depth discussion of the various aspects. In the previous chapter we have formalized the methodology we will be using and shown the need and necessary features for a *dynamic traffic assignment* model. The exact nature of the model is detailed in this chapter. We start by providing a general description of the model and work our way down to the different components such as OD handling, assignment and route choice and discuss the theoretical assumptions and mathematical solutions for each. At the end of this chapter we provide a summary and the link to the optimization algorithm in the next chapter.

6.1 General description

The general build of a dynamic traffic assignment model was already shown in 4.2 and this description remains valid. In the figure below we use the same elements and place them in a functional context:

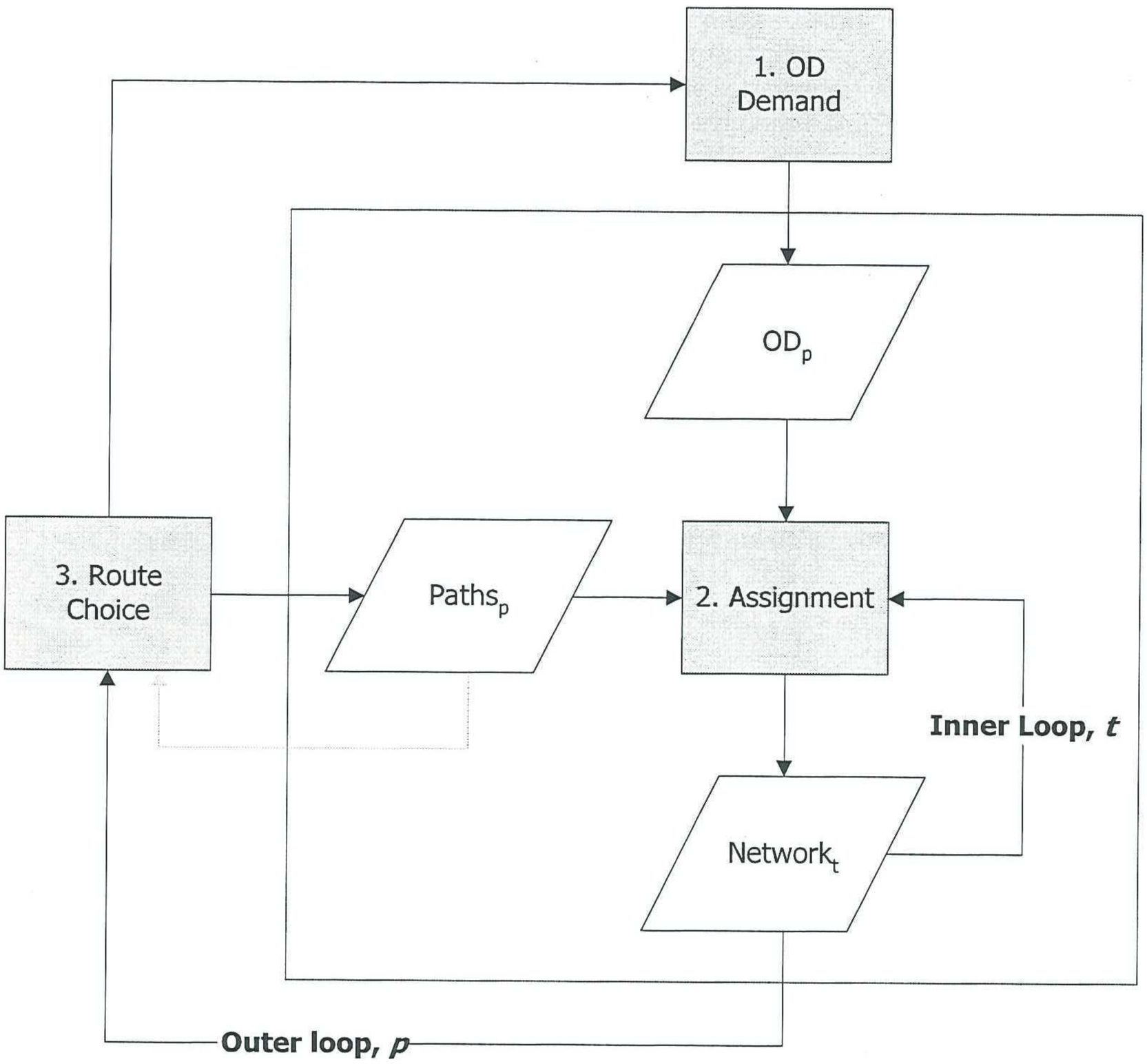


Figure 6-1 Functional outline of the dynamic macroscopic traffic model

In this diagram, there are two *loops* shown: the *inner* and the *outer* loop. Both loops are dynamic loops in time but with different step size. The inner loop has a step size *dt*, the smallest step size in the model. Every step *dt,* a new *network state* is calculated, based on: the *traffic demand* expressed

by the *OD*, the *paths* between the origins and destinations, and the travelers already in the network heading for their destinations. This smallest time step is by default 30 seconds and is able to model congestion effects with a high level of detail in time.

The outer loop has a step size *p*, which is called a *period*. The number of periods one uses in the simulation is a user defined variable, by default 12 periods for a simulation of 3 hours. Every *period* a new *OD* is calculated that is used by the inner loop for that period. This way, dynamic effects such as *differences in travel demand* due to peak hours can be modeled. Every period, the paths that are used to travel between the origins and destinations are recalculated as well, enabling people to take different routes at different times according to the network state (queues).

The OD data is based on a static matrix which represents the average demand in vehicles between origin and destination in [veh/hr]. A technique called *time slicing* is used to capture some more dynamic effects by varying the demand each period. This varying is done in such a way, the demand profile is similar to the measured intensity on the road network.

The assignment module enables the modeling of traffic represented by a *single user class* with *multiple origins and destinations*. The traffic is modeled based on the *kinematic wave model*, which is a combination of the *conservation of vehicle equation* and the (linear) *fundamental diagram*. To approximate this kinematic wave model, a numerical representation is made by means of the Godunov scheme. This approach is also known as a 1st order approach which is adequate in terms of the demands we formulated in 5.2.1.

In the following paragraphs, we will use discuss the three main steps in the model in detail as they are described in the diagram: OD demand, Assignment and Route choice. But first we discuss some other considerations which have to betaken into account before developing the model:

6.1.1 Considerations

Computation speed

The *need for speed* in simulation terms is very high due to the iterative cycle(s) in the control methodology. As we have seen above, the modeling process itself comes with two *nested loops* which are demanding in terms of computation time themselves. We therefore have to take into account the various aspects which are of influence the computation speed. We can define these as follows:

Table 6-1 Computation speed considerations in the DSMART Model

Inner loop	Outer loop	Simulation variables
Serving OD demand Link progression Node progression	Time slicing OD Updating network characteristics Calculating routes	Simulation duration Step size dt Number of periods Network complexity Route choice trials

6.2 OD matrix

The OD matrix is a representation of estimated travel demand between different points in the network. Calculating an OD matrix is a difficult task especially the calculation of a time dependant dynamic OD matrix. In this thesis we will be using a third party static OD matrix which has been calibrated for the morning peak of the Rotterdam network, created by RWS and DHV. This base matrix was created using a *regression model* on a *zonal division* of the study area. This way, zone *generation* and *attraction* could be calculated which where used in a *gravity model* to estimate the actual demand per origin and destination. The matrix has been calibrated to the morning peak by means of *traffic counts* and *cordon data*.

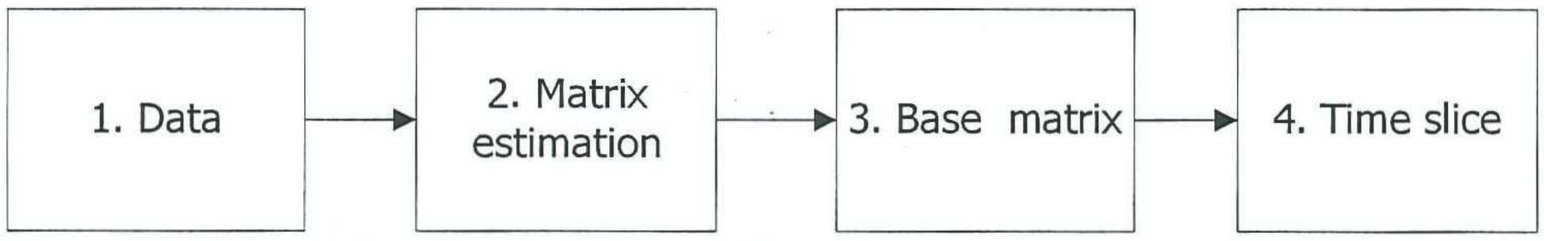


Figure 6-2 Different steps in calculating an OD matrix

These steps are outlined above in blocks one to three. Because this OD data is *static*, the actual demand is the *average demand* in vehicles per hour for a two hour period between 07:00 – 09:00 in the morning, we will try to make it more dynamic. From experience one knows, that in most networks, there is a great difference between the amount of traffic at 07:00 and at 09:00 hours, which is partially the result of a change in the *total size* of the demand over time. We try to incorporate this type of effect in our model by *time-slicing* the OD matrix.

Assumptions
- Changes in departure *demand size and time* as the result of *changing network conditions* is not taken into account, and therefore considered *inelastic*.

6.2.1 Time slicing

By time slicing, one changes the demand values expressed in the static OD_s matrix during different time periods (5). The value of the time slices τ_p are based on measurements on the ring way network and considered representative for the whole demand pattern expressed in the OD_s. This is off course an unrealistic approximation and could very well be enhanced by using different time slice values for different OD pairs but still captures the dynamic demand nature.

$$OD_p = OD_s \tau_p \ ,[\text{veh/hr}] \tag{8}$$

$$\sum \tau_p = p \ ,[-] \tag{9}$$

The sum of all time slices must equal the number of all periods used, so no extra demand can be generated, or demand could disappear.

The resulting OD_p demand is the *dynamic* OD demand during period *p* and always expressed in [veh/hr].

6.3 Assignment

In this paragraph the actual assignment of traffic to the road network is discussed. We start by discussing the kinematic wave model (or LWR model, or first order model) and show why this is applicable for our problem. Afterward we will discuss the mathematical model and the numerical approximation. Most theory in this paragraph has been derived from [29. CH9]

6.3.1 Theory

The underlying assumption of a macroscopic traffic flow model is that traffic can be described as a *continuum, similar to a fluid or a gas*. The most important equation in any macroscopic traffic flow model is the *conservation of vehicle equation*. If for a specific roadway sections, more vehicles are entering than leaving, then the number of vehicles in the section must increase.

$$\frac{\partial k}{\partial t}+\frac{\partial q}{\partial x}=r(x,t)-s(x,t)\text{, and} \qquad (10)$$
$$q=Q(k)\text{, [veh/hr]}$$

Where $k=k(x,t)$ denotes the density at a certain point x at time t, and $q=q(x,t)$ describing the intensity of vehicles passing the cross section x at time t, while r and s respectively denote the *inflow* and *outflow* from and to on- and off ramps. The *kinematic wave model* assumes that the traffic volume can be determined from the *fundamental diagram* relation between the traffic density and the volume, the $q=Q(k)$ relationship. Using this partial differential equation, we can describe the dynamics of the traffic flow under the assumption that the speed of the traffic reacts instantaneously and locally to the traffic density.

Fundamental diagram

The fundamental diagram describes the relation between the *capacity* and the *density* and is depicted in the upper part of figure 6-3 in its *simplest triangular form* (Daganzo), which is also computationally the fastest and will therefore be used in our model.

By calculating the quotient between *capacity* and *density* we find the corresponding *speeds*, which are shown in the lower part of figure 6-3.

With this relation and the *conservation of vehicle equation* we can model traffic in a very simple but yet straight forward way called *the kinematic wave model*.

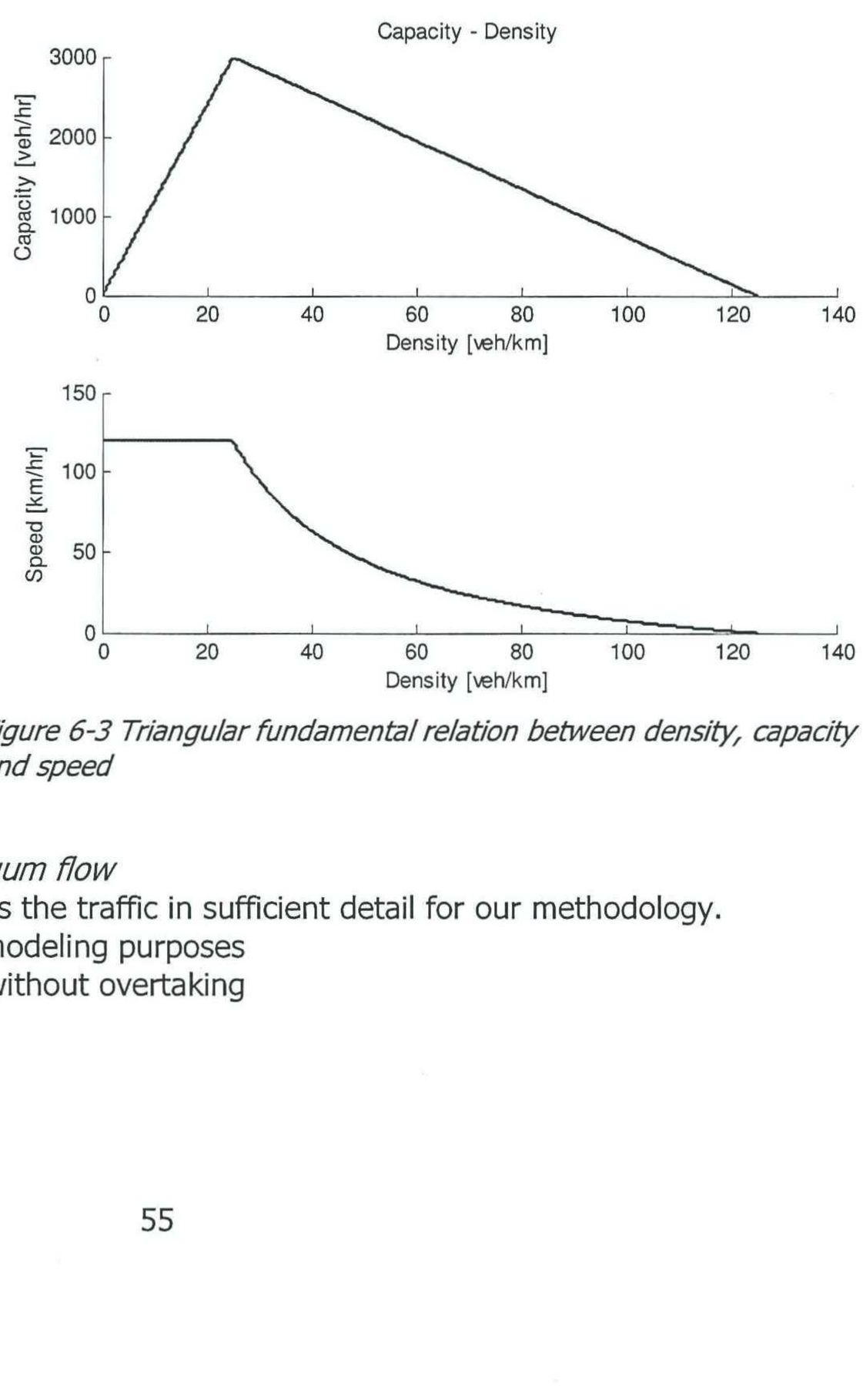


Figure 6-3 Triangular fundamental relation between density, capacity and speed

Assumptions

- Traffic can be modeled as a *continuum flow*
- The *kinematic wave model* describes the traffic in sufficient detail for our methodology.
- A single user class is sufficient for modeling purposes
- Traffic is modeled first in first out, without overtaking

In order to justify this assumption, we will consider the main advantages and drawbacks of a 1st order macroscopic traffic modeling approach.

Advantages
- The most important properties of traffic flow are addressed by macroscopic modeling, such as: the formation and dissipation of queues, shock waves etc.
- Macroscopic modeling enables an easy calculation average travel times, speeds and other quantities related to traffic density
- The models can be written in closed form and are generally deterministic and less sensitive to small disturbances in the input.

Drawbacks
- Traffic speed is described by a *stationary speed-density relationship* and reacts instantaneously to changing network conditions. (there is no hysteresis, or acceleration / deceleration curves which are observed in real life)
- The kinematic wave theory shows (speed) shock formation by steeping speed jumps to infinite sharp discontinuities in the concentration. It produces discontinuous solutions irrespective of the smoothness of the initial conditions. These are in contradiction with the smooth shocks observed in real-life traffic
- The kinematic wave theory is not able to describe regular start-stop waves with amplitude dependant oscillation times which have been observed in real life traffic.
- Kinematic wave models do not address traffic instability, which are often observed in real life that in certain traffic concentration ranges small perturbations can induce violent transitions.

To meet these shortcomings, more complicated models have been developed. Mostly based upon an expansion of the conservation of vehicles equation, incorporating anticipative terms to model vehicle speed more correctly (2nd order model like Payne etc). Results may prove to be qualitatively better but also less straightforward to predict or counter intuitive.

Given the considerations made above, we may conclude that a *1st order macroscopic model* is able to model effects such as *congestion and spillback* in a crude but very effective way. Though the exact traffic behavior may lack somewhat realism we have to keep in mind that this is not the primary goal of our model. If we recall In 5.2.1 we stated that its primary features would be to correctly model queues (buildup, dissipation and spillback) and the effect on network (spare) capacity which is done quite well in this type of model.

6.3.2 Numerical approximation by mean of Godunov Scheme

To solve this 1st order model in an analytic manner is quite painstaking and would yield poor results due to the strict boundary conditions needed (besides being impossible for the author). An alternative is using a n*umerical approximation,* which means that *time and space* are divided into *discrete portions*. In this model we use the time step *dt* as a discretisation of time (inner loop) and the division of links into smaller units called *cells* as a discretisation of space.

Cells
Cells can have a *fixed length* or be made dependant upon *maximum link speeds*, this last approach is chosen, effectively making the model better suited for use in a network where links can have different *speed restrictions*. In this model, the cell length is based on the *maximum speed* of the link and the *chosen time* step.

Example

timestep dt=60 [sec], Link length L= 12 [km], Travel speed V=120 [km/h],
Cell length Lc= 120*60/3600 =2 [km], number of cells n= 12/2 =6 [-]

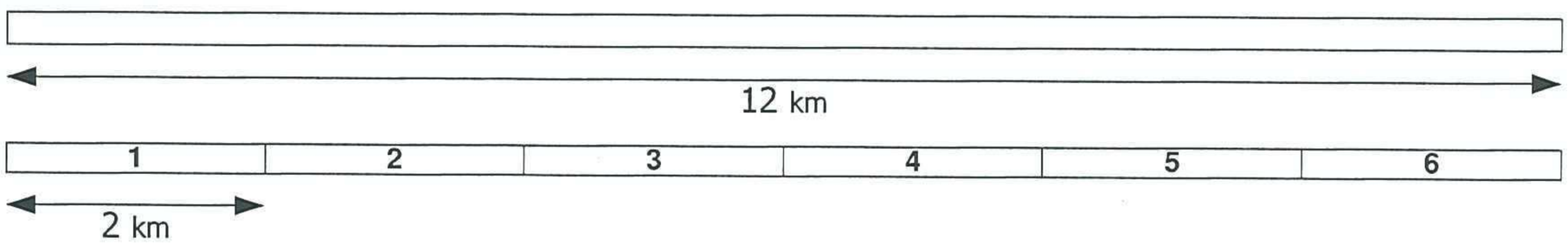


Figure 6-4 Dividing a link into cells

Godunov scheme

The *Godunov* scheme allows a simple type of numerical computation to model the progression of flow between cells. Behold the example where three cells are shown:

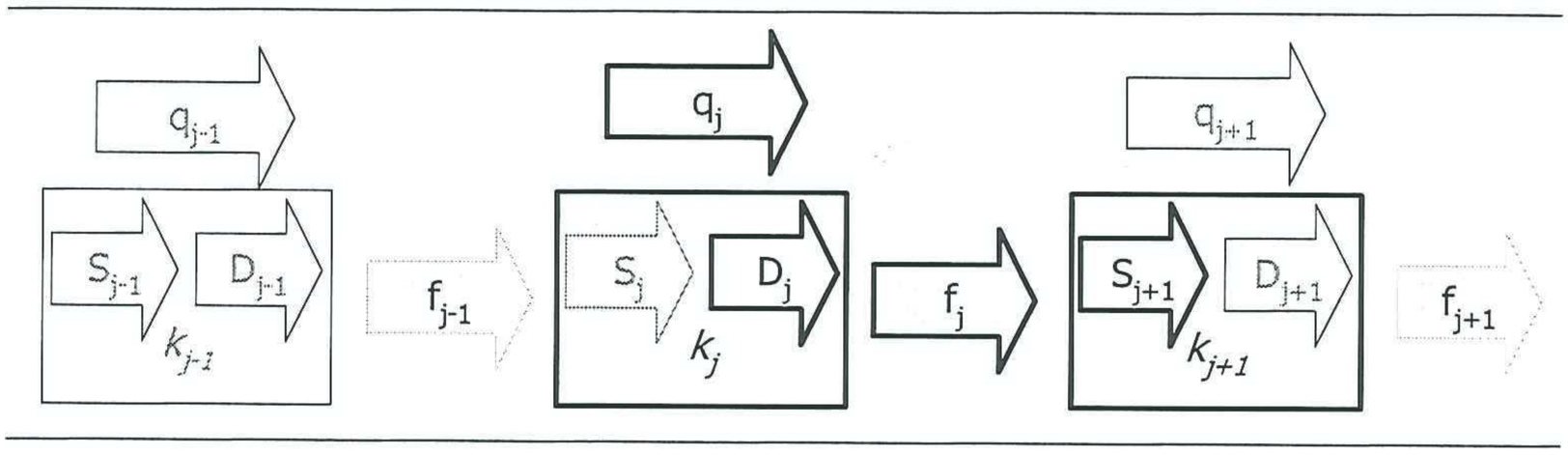


Figure 6-5 Consecutive cells on a roadway with attributes: density, demand, supply, flow and flux

The *Godunov* scheme works as follows: consider two subsequent cells j and $j+1$. To determine the number of vehicles flowing from cell j to cell $j+1$, we first determine the flow in the *cell* depending on the fundamental diagram

$$q_j = \begin{cases} V_j k_j & k_j \le k_c \\ Q\dfrac{k_{jam} - k_j}{k_{jam} - k_c} & k_j > k_c \end{cases}, \text{[veh/hr]} \qquad (11)$$

Now, depending on this flow, we calculate:

- The demand D_j (maximum number of vehicles that can flow out of cell j) of cell j and
- The supply S_j of cell $j+1$ (maximum number of vehicles that can flow into cell j+1)

By:

$$D_j = \begin{cases} q_j & k_j \le k_c \\ c & k_j > k_c \end{cases}, \; S_{j+1} = \begin{cases} c & k_{j+1} \le k_c \\ q_{j+1} & k_{j+1} > k_c \end{cases}, \text{[veh/hr]} \qquad (12)$$

The Demand D_j equals the flow q_j in a cell as long as the density k_j is less than the critical density k_c, otherwise the capacity c of the cell restricts the outflow. The supply S_{j+1} equals the capacity c of the cell as long as the density k_{j+1} is less than the critical density k_c, otherwise the maximum cell inflow is dictated by the number of vehicles that flow out of the cell, i.e. q_{j+1}.

The actual flow from cell *j* to cell *j+1* then becomes the minimum of demand and supply i.e.

$$f_j = \min(D_j, S_{j+1}) \text{ ,[veh/hr]} \tag{13}$$

Where f_j denotes the flow out of cell *j*.

Clearly the flow f_{j-1} out of cell *j-1* is the flow F_j into cell *j*. The new density in cell *j* can be calculated by determining the net inflow and multiplying it by *dt*.

$$k_j = k_j + dt(F_j - f_j) \text{ ,[veh/km]} \tag{14}$$

It turns out that the Godunov scheme has excellent properties, such as accurate representation of shocks, ability to incorporate flow-density relations that are a function of x, etc.
So far, we have a model and a numerical approximation which enables us to describe the traffic behavior on a single link from the beginning to the end of this link. This is not enough, we need to be able to model traffic on a single link heading for *different directions* and somehow be able to let the traffic be *progressed by nodes* to allow different routes to be used. In the following paragraph we will show that by *generalizing* the scheme we are able to model traffic heading for different directions using the same links. After that, we will show how *destination specific split fractions* can be used to progress traffic heading for different destinations via nodes.

6.3.3 Generalization to multiple destinations

The Godunov scheme described is able to calculate the progression of traffic from the beginning to the end of a link quite nicely. This however is not enough for modeling traffic in a network. To do so, the scheme needs to be expanded to correctly model the flow of traffic heading for *different destinations* on the same link.

This is done by adding another dimension to the division of links into *cells*, namely for each *destination* in the network an extra row of cells is added.

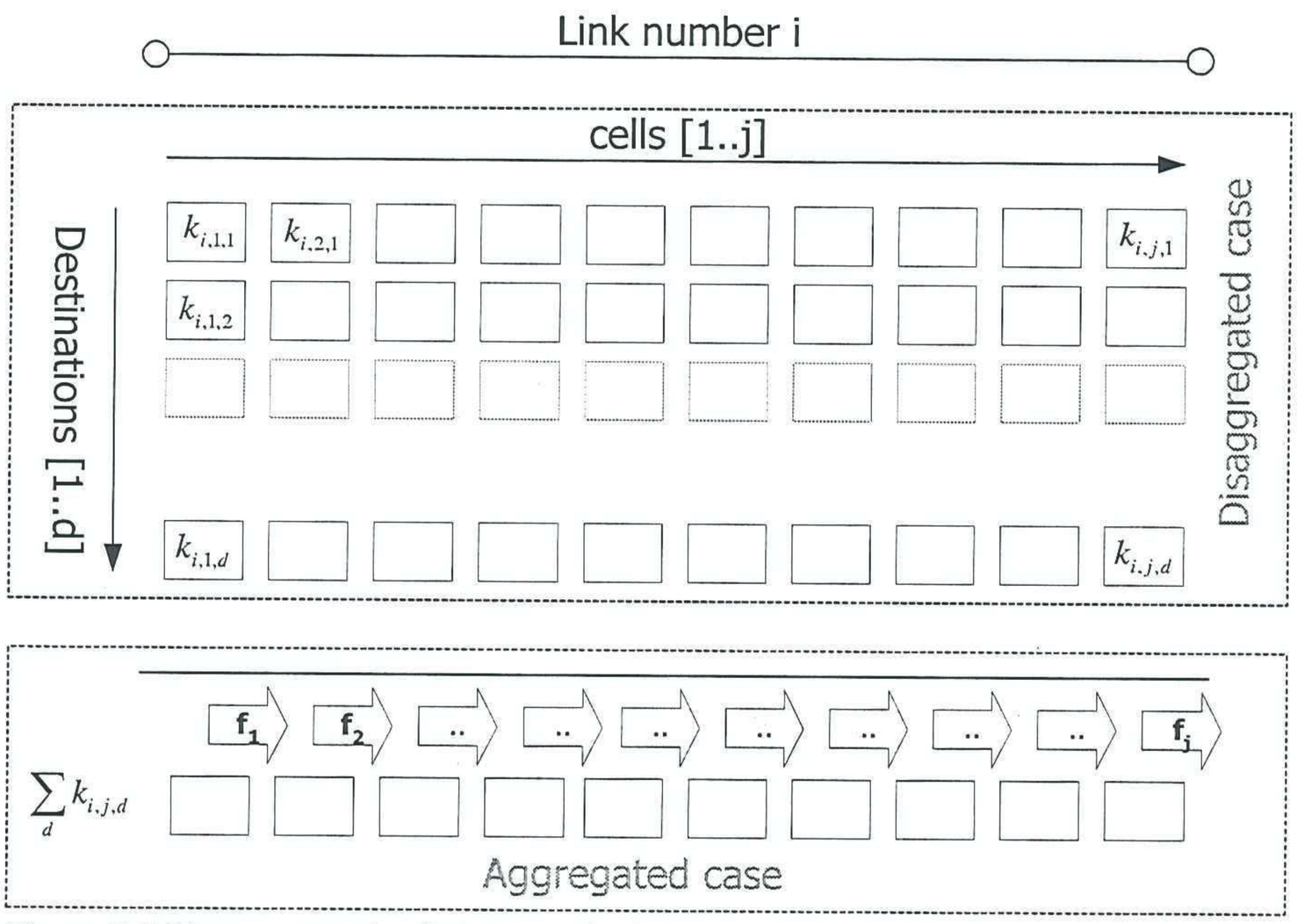


Figure 6-6 Disaggregated cell division of a link

Now as we can see in the picture above, we split the cells into two types: the *aggregated* and *disaggregated* cells. The aggregated cells are the column wise sum of all disaggregated cells and contain the *actual density* in a particular cell *j* on link *i*. The disaggregated cells contain information regarding the density $k_{i,j,d}$ of link *i*, for cell number *j* and heading for a specific direction *d*.

We use the *Godunov* approach on the *aggregated case*, to calculate the resulting flows f_j flowing out of aggregated cell *j*. We then simply divide this flow by the total density in the cell to use it as a *multiplier* for the *disaggregated densities* which is called the *flow speed.*

$$V_j^{flow} = \frac{f_j}{\sum_d k_{j,d}}, [\text{km/hr}] \tag{15}$$

With this flow speed (multiplier), we can calculate the *disaggregated flows* out of the *disaggregated* (destination specific) *cells*.

$$f_{j,d} = k_{j,d} V_j^{flow}, [\text{veh/hr}] \tag{16}$$

This way we can progress the traffic heading for different destinations individually, based on their total summed density using the Godunov numerical approximation.

6.3.4 Progressing traffic past nodes by means of split fractions

For our model to work correctly we only need to be able to guide traffic heading for different directions via the different nodes in the network using *destination specific split fractions* (from now on referred to as *split vector*). This split vector is a translation of the routes chosen to travel to a certain destination. The split vector is used to determine where the density, stored in the last *disaggregated cells* of a link, is distributed to next (in terms of outgoing links).

In the example to the right we see a node A with two incoming links and three outgoing links. At the end of the links the disaggregated cells are drawn. If we look closely we see that there are nine destinations in this example.

We define the following:

B_n^{in}, B_n^{out} — A collection of links for which node n functions as input/output.

K_n^{in}, K_n^{out} — The number of links for which node n functions as input / output.

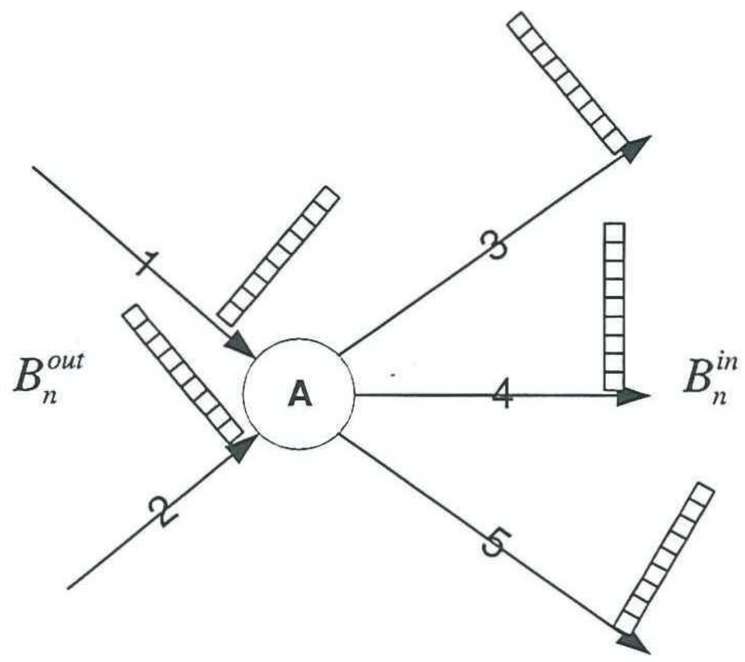


Figure 6-7 Node progression

The node progression determines the amount of travel that is possible between the *links 1* and *2* who output at *node A* and *links 3, 4 and 5* for which *node A* functions as input, based on the split vector. In 4.2.3.1 it was stated that storing routes in split vectors would prove very useful when using macroscopic modeling, now we will show why.

Split matrix multiplication

Route choice is stored by means of *split vectors* $\Psi_{i,d,p}$, which are used to distribute the d*isaggregated demand* of a link *i* heading for destination *d* at period *p* (discussed in 5.5). A *split*

matrix $\vec{\Psi}_{i,p}$ is a column wise listing of the single *split fraction vector* and can be used to distribute *all disaggregated* demand (all destinations) for one link over all output links in *just one multiplication.*

In the figure to the right, we graphically depict a multiplication using a split matrix for traffic coming from link 1, heading for nine different directions and thus being distributed over three different outgoing links.

The *disaggregated demand* for *link 1* is shown in the left column. We see a demand of 8,7 and 6 [veh/hr] heading for destinations 4,5 and 6 respectively. Next to the demand we also see the *split matrix* as, collection of split vectors, for link 1 and the resulting distribution over the outgoing links 3,4 and 5.

This distribution of the disaggregated demand of link 1 over the links 3,4 and 5 is done by means of one simple matrix multiplication called *fprod*.

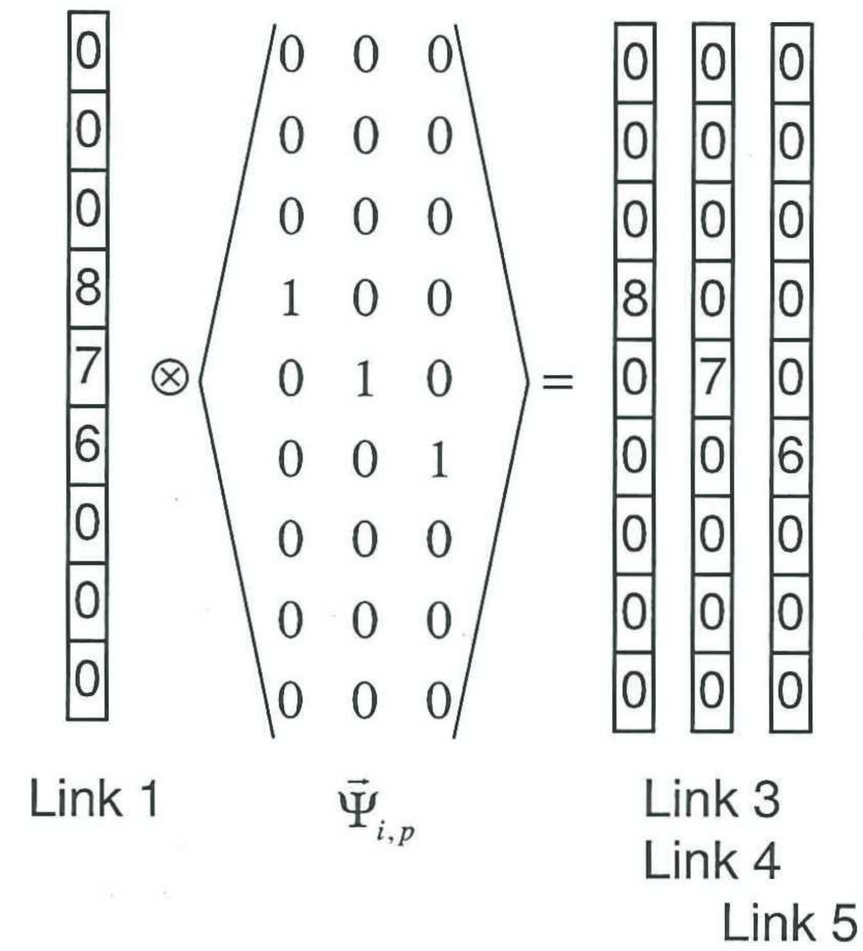


Figure 6-8 Multiplication using a split matrix

The formal definition of the *fprod* multiplication is given below:

$$fprod(A,B) = \begin{Bmatrix} a_{11} \\ a_{21} \\ a_{31} \end{Bmatrix} \otimes \begin{Bmatrix} b_{11}\, b_{12}\, b_{13} \\ b_{21}\, b_{22}\, b_{23} \\ b_{31}\, b_{32}\, b_{33} \end{Bmatrix} = \begin{Bmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{11}b_{13} \\ a_{21}b_{21} & a_{21}b_{22} & a_{21}b_{23} \\ a_{31}b_{31} & a_{31}b_{32} & a_{31}b_{33} \end{Bmatrix} \quad ,[-] \qquad (17)$$

By using this kind of multiplication the disaggregated demand of links 1 and 2 can be distributed over the links 3,4 and 5 simply by repeating the multiplication for links 1 and 2 with the appropriate *disaggregated demand* and *split matrix* while summing the resulting distribution over links 3,4 and 5.

This approach is favored to a dynamic programming solution because it does not require an extra loop and makes use of an *elegant* vector multiplication. (The model is implemented in Matlab which is poor at loops, but very efficient in terms of vector multiplication)

6.4 Route choice

The last element from the functional diagram in 6.1 we need to address is the *route choice* process. We will start by discussing the theory surrounding this process followed by the mathematical model. Then we will address the implementation of some expansions needed to correctly model the effect disseminated route guidance.

6.4.1 Utility maximization

As discussed earlier in 4.2.3 route choice can be modeled as a *utility maximization* problem, based on *rational user behavior*. Random utility models simulate the choice made by the generic individual from a set of alternatives under the assumption that the analyst is able to correctly specify this set.

Let the perceived utility of an alternative be given by:

$$U_a = V_a + \varepsilon_a \text{ , [-]} \tag{18}$$

Where U_a is the *perceived utility* of alternative a, V_a the *systematic utility* and a *random residual* ε_a which is the (unknown) deviation of the utility perceived by the user from the systematic utility.

Assumption

- Link travel times, based on the cell densities and the fundamental diagram, used as the systematic utility component can be considered as a *good enough* representation of systematic disutility.
  Other, more intricate, models also include travel distance as a distinct type of systematic utility component (or criteria's like road type preference, no-go areas etc)

Random utility theory provides us with a tool to calculate the *probability* an alternative will be chosen, given the total set of alternatives, based on the *perceived utility*. Perhaps the most famous and widely used model is the *logit model* with its inevitable drawback of considering *largely overlapping routes* as *independent alternatives,* and the requirement to be able to define the total set of alternatives.

Another approach to the random utility theory in calculating routes is the probit model, where the perceived utility of an alternative *route* is calculated by means of the perceived *link* utilities. Shortest routes are being determined based on a random sample of travel times from the link probability distribution. The probit path choice model, results from the assumption that the random residual elements of the vector ε, are normally distributed with null mean.

6.4.1.1 Probit route choice

The probit model works by selecting a route, based on the sum of the user *perceived link utilities* which make up for the route, not the route as a whole. To do so, we first calculate the perceived link utility of all the links in the network.

$$t_a = L_a + \Theta z \text{ , [hr]} \tag{19}$$

$$\Theta = \alpha L_a \text{ , [hr]} \tag{20}$$

Let t_a be the perceived link travel time, for a link *a* in the network, drawn from a *normal* distributed set of travel times with a mean value $\mu = L_a$ and a standard deviation $\delta = \alpha L_a$ (Where α is set by default to 0.1 and *z* is a normal distributed variable wit null mean).

With the perceived link utility t_a, we can construct a least cost *route* with a *perceived* value *r* by finding a set of links *a* which are in concordance to (18):

$$r = \min \sum_a t_a, \forall a \in A \tag{21}$$

And obey the following conditions:

- The set forms a continuous connection between origin and destination
- No loops are present

Finding this set of links is solved by means of a shortest path algorithm described in the next paragraph.

6.4.1.2 Floyd Warshall Shortest path algorithm

At this point the remaining problem is finding a set of links *a* which satisfies condition (18). There exist a wide variety of these algorithms. The oldest and most elegant is the Dijkstra algorithm which solves the *single source* shortest path (SP) problem. The Floyd-Warshall algorithm, which we will be using, solves the *all pair* shortest path problem by means of dynamic programming. And at last, a new breed[1] of SP algorithms is finding its way in DTA modeling called the *k^{th} shortest paths algorithms*. These last are not only able to determine the actual shortest, but also the k^{th} near optimal path which can be a very useful in calculating routes. However, for this DSMART Model we need to know people will intend to travel between all origins and destinations, we therefore decide upon using the FW algorithm.

Adjacency matrix

The algorithm works by searching the *adjacency matrix* for an *intermediate* node between two points which would make the distance to travel shorter. The adjacency matrix is a matrix which specifies whether or not two nodes are connected by a link, and if so to which cost. The adjacency matrix therefore has a size NxN with N being the number of nodes in the network and a value of 0 if there is no connection. If there is a connection (e.g. there is a link which connects two nodes), its value is given by the perceived link utility t_a.

FW Algorithm

The algorithm runs in $O(N)^3$ (N being the number of nodes) and the main recursively definition of the algorithm is given by:

$$d_{ij}^{k} = \begin{cases} W_{ij}; & if\ k = 0 \\ \min(d_{ij}^{k-1}, d_{ik}^{k-1} + d_{kj}^{k-1} & if\ k > 0 \end{cases} \tag{22}$$

[1] Other new breeds of SP algorithms use a heuristic component E.g. A^*, guessing which direction to take.

The algorithm works by checking whether the distance between nodes *i* and *j* could become shorter if the intermediate node *k* would be used. If this is the case, the distance is adjusted and the intermediate node *k* is stored in another matrix.

Initially when constructing the adjacency matrix, the travel times are used to display connectivity and unconnected nodes are given the value of infinity. When the algorithm starts the initial distance matrix (indicated with the D) is the same as the adjacency matrix. Then in *k* time, each node is checked to see if it could function as an intermediate shorter node.

```
for k=1:N
    for I=1:N
        for j=1:N
            if D(ij) > D(ik) + D(kj)
                P(ij)=k
                D(ij)=D(ik) + D(kj)
            end
        end
    end
end
```

Figure 6-9 FW Pseudo code

Path reconstruction

The end result of the FW algorithm is a predecessor matrix *P*, indicating which intermediate node should be used if one travels from *o* to *d* and a matrix containing the value of the shortest distances The predecessor matrix can be used to find the actual traveled node path, which in term van be used to determine the actual set of links *a* which satisfies condition (18).

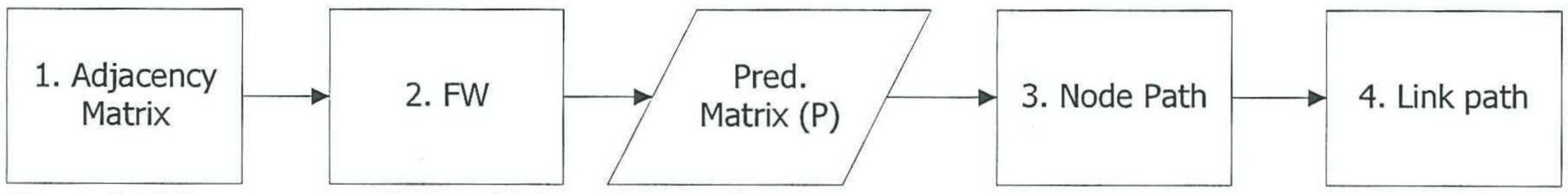


Figure 6-10 Reconstructing link paths from the FW algorithm

6.4.2 Periodic stochastic route choice

Based on the *probit model* and the *Floyd Warshall algorithm* we are able to calculate the *user perceived least cost paths* between any given pair of origin *o* or destination *d* in a network. If we where to assign all travel demand between the OD pair to this *user perceived shortest path*, we would be using a so called All Or Nothing (AON) assignment using just one path.

To circumvent this type of assignment, instead of calculating just *one* path we repeat the calculation process for a number of times and hope that the *random residual* term (17) enables *other* paths to be *perceived as shortest path* as well, rendering it a *stochastic* process and enabling a more *spread* assignment.

The *periodic* approach implies that we will be recalculating routes every period. This means that every time a new period starts (outer loop, *dp*) new routes are calculated, based on the user perceived network conditions at that time. This is a very disputable assumption since it is unlikely users will have full network knowledge at the beginning of every period, however:

Assumption

- paths based on a *periodically calculated user perceived shortest paths* by means of full network knowledge are better that static unchangeable pre defined paths

Another difficulty arising with the periodic approach is the fact that all travelers in the network use the same paths during each period, even those who have set of on their journey during an earlier period and perhaps even with a different initial path. To compensate for this problem, we introduce A path *gamma* factor which adds the routes of the pervious period to the newly calculated routes.

The calculation process is schematized in the picture to the right. At the beginning of every period the *probit path choice process* is repeated for a number of trials (default 5). During these trials, we first calculate the adjacency matrix based on the *user perceived link travel times* and some penalties[1], the next step is determining the actual paths using the FW algorithm and storing the paths in an intermediate structure.
When the number off trials has passed, the intermediate calculated paths are converted into *routes* by means of *destination specific split fractions*. We then add the routes of the previous period with a weighted factor gamma to find the resulting routes.

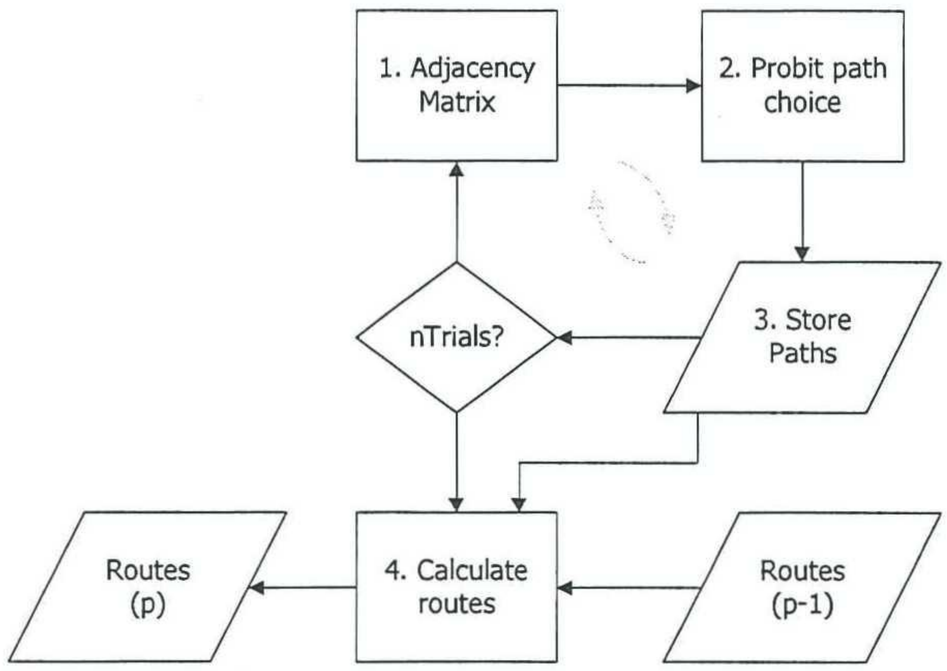


Figure 6-11 Route Choice calculation Process

1. Beside the residual error term, an extra node traversal penalty is added to the link. Links with lower speeds and smaller capacities are considered less attractive and therefore receive a higher penalty

Example
In the example below, we see a simple network consisting of 4 nodes and 4 links with different lengths. We will be computing the resulting routes in terms of split fractions between A and D with 10 trials.

We first determine the number of times a specific path has been chosen as the user perceived least cost path during the trials. (table below)

Path	f()
ABD	2
AD	7
ACD	1

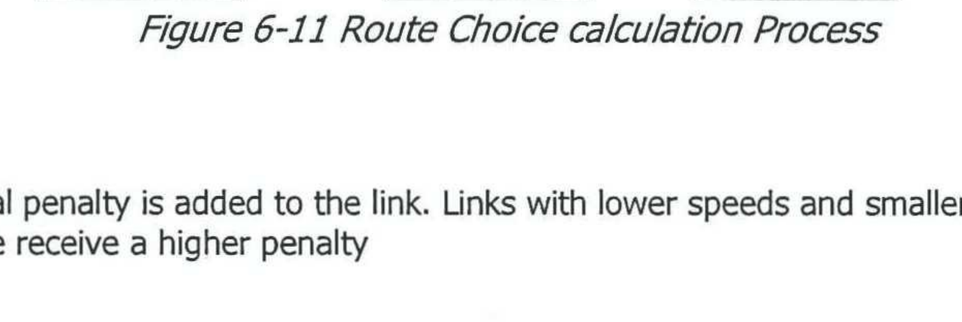


Figure 6-12 Probit path choice illustration

This table represents the stored paths in step 3 from the previous figure and is used for calculating the destination specific split fractions using the formula below.

$$\psi_{i,d,p} = \frac{[[f(L_c)]]}{ntrials}, \forall i \in B_n^{out} \wedge c \in B_n^{in} \text{ , [-]} \qquad (23)$$

The destination specific split fraction $\psi_{i,d,p}$ for link *i*, destination *d*, period *p* is based on the frequency with which a link *c*, being a member of the links B_n^{in} for which node *n* functions as an input, is chosen divided by the number of trials. (Note figure 28 where the definition of the input and output links is given).

In our example this would yield: $\psi_{1,D,p} = \frac{[2 \;\; 7 \;\; 1]}{ntrials} = [0.2 \;\; 0.7 \;\; 0.1]$

Path Gamma
Calculating the resulting *destination specific split fractions* by using the *calculated probit paths* and (20) result in a set of *route choices* which could be spread over a number of links. However, no

matter how complex the resulting routes are, they can be handled mathematically using standard operators as a result of the translation into split fractions.

As stated earlier, the *path gamma* factor is implemented to incorporate the route choices from the previous period into the current one. This is done using the following formula

$$\Psi_{i,d,p+1} = (1-\chi)\Psi_{i,d,p} + \chi\Psi_{i,d}^{temp}\text{, [-]} \qquad (24)$$

By default the value of χ is 1.

6.5 Modeling DRIP settings

To model the DRIP settings correctly, some custom enhancements *must* be added to the *probit route choice process* described in the previous paragraph. Resulting in a five step process described below:

Step 1: Free flow routes
The first level of route choice consists of calculating the *free flow routes,* the actual shortest paths for every *link* to every destination in the network. The result is stored in binary split fractions (this is a deterministic process). The free flow routes are needed in networks where DRIP settings might route travelers on links which have never been considered before.

Step 2: Probit route choice
As described in 6.4

Step 3: Highway routes[1]
The highway routes consist of a set of split fractions which make sure that people who travel the highway in a certain direction, will follow this direction on the highway as long as possible until they find the nearest exit to their destination. If this step is left out we can find ourselves in the following situation: If an accident occurs, and a DRIP guides people in a direction which isn't the actual shortest path, we will observe the following behavior. Based on the *user compliance* a certain fraction of people will follow the advice and take the next link as suggested by the DRIP. If this route has not *always* been calculated as a shortest path in step2, or it has *never* been calculated as such we will experience *leakage*. A certain percentage of travelers will take the off ramp, turn around and take the on ramp back heading in the opposite direction of the DRIP advice using a previous defined shortest path.

Step 4: Custom routes
The next level of route choice is adding some pre defined alternative routes which can be used by the DRIP, but are not on the highway. For example in the Rotterdam network a feasible alternative could be driving straight through the city. Now when travelers pass a DRIP and are confronted with a split fraction that guides them through the city,

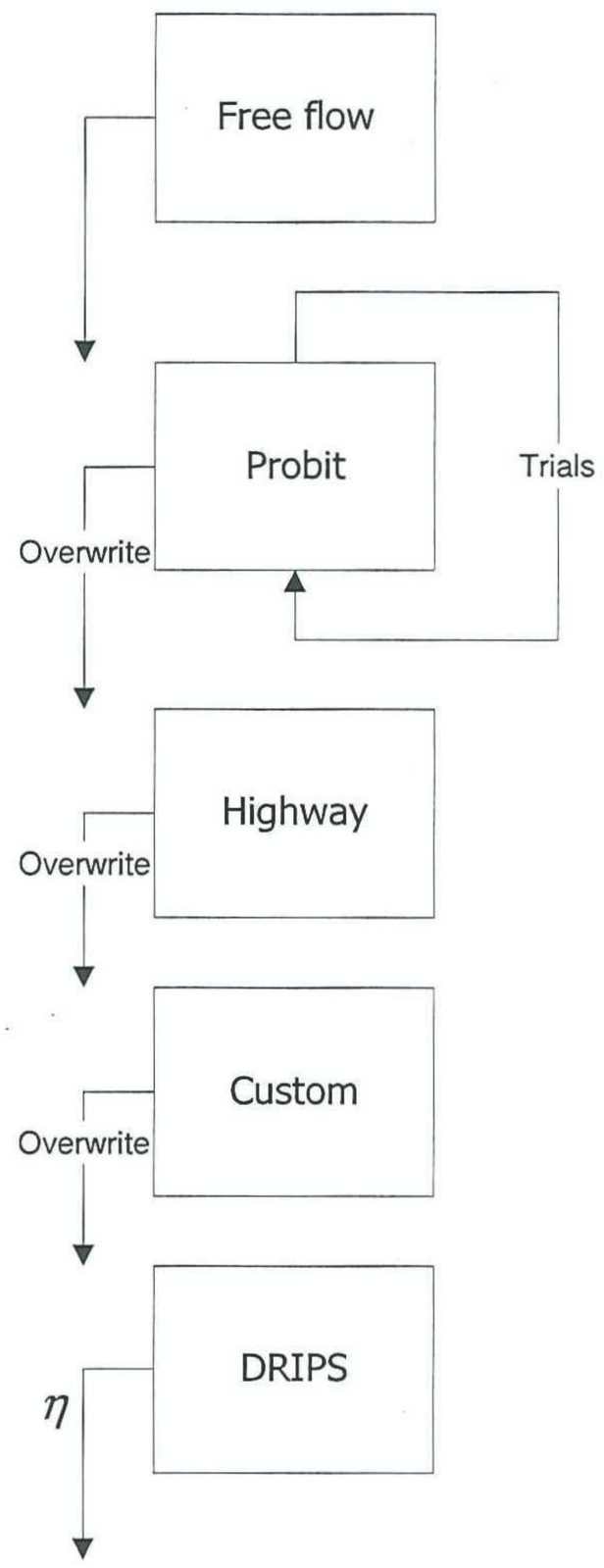


Figure 6-13 Custom five step route choice

they could very well be confronted on the next link with a split fraction calculated by the *base routes* from step 1 sending them to their destination using the city network in an unintended way. However the alternative route which is considered when the *guidance was provided* was to take a simple route straight through the city heading for the highway network. The custom routes are used to overwrite

these previous *split fractions* making sure that the destinations for which the DRIP information send the through the network, will be reached by the intended (simple, straight-line, custom) route.

Step 5: DRIP Links
DRIP settings are generated by the *genetic algorithm*, and overwrite the split fractions at the a certain point. To model *user disobedience*, these new split fractions can be a weighted sum of the *old split fractions* and the newly calculated split fraction by the *genetic algorithm*, hence the word *add* in the diagram.

1. Fortunately, calculating the highway routes to undo the unwanted *leakage* effect is easy. We simply make a network representation where *off ramps* are panelized heavily by considering their travel time a 1000 hours. We then proceed by calculating for all *highway links* the path from the *end node* of this highway link to *all the destination nodes* belonging to one of the *DRIP directions* for which guidance is calculated. We only use the first *consecutive link* in this path to calculate the *destination specific split fraction* at the end of the considered link. The effect of this measure will be that the *paths calculated* will make use of the highway as long as possible until they finally must leave and take an off ramp heading for their destination. This way when a DRIP tells people to use the highway in a different direction then normally, they still would follow the highway all the way until they reach their destination from opposite direction.

6.5.1 Mathematical formulation of DRIP route choice enhancements

In the figure on the previous page, it is also shown how the different steps deal with the custom added *split fractions*. When the free flow (step 1) routes are calculated and translated into the split fractions, we overwrite them with the route choices from the probit process (step 2). The route choices which are calculated in the highway routes step (step 3) overwrite the choices made in the probit process to make sure the remaining journey is traveled as was intended by the DRIP. The Custom routes (step 4) also overwrite the probit route choices for the same reason. And finally, the DRIP settings (step 5) do *not* overwrite the old split fractions but are added based on the presumed user compliance at that point, as is explained below.

Step 5: DRIP links; modeling 1st order user compliance
As we have stated earlier on, modeling DRIP settings using *destination specific split fractions* provides a great opportunity for modeling *human behavior* in terms of *compliances* (4.2.3)

On the one hand we have the *calibrated split fraction* $\psi_{i,d,p}^{calibrated}$ which is used to resemble the real life situation, the split fractions that will likely be used if no DRIPS are active. On the other hand we have the calculated DRIP setting $\psi_{i,d,p}^{DRIP}$ which is aimed at guiding travelers in a certain direction. The user compliance factor η can be used to mimic user obedience / disobedience using the formula below:

$$\psi_{i,d,p}^{eff} = (1-\eta)\psi_{i,d,p}^{calibrated} + \eta\psi_{i,d,p}^{DRIP}, [-] \qquad (25)$$

Example
If one was used to taking a left turn (in the example from 5.5), but the DRIP would now advise a right turn we are able to model a *user compliance* of $\eta^{1st} = 40\%$ in the following way.

$$\Psi_{i,d,p}^{eff} = (1-\eta)\begin{bmatrix}1\\0\\0\end{bmatrix} + \eta\begin{bmatrix}0\\0\\1\end{bmatrix} = \begin{bmatrix}0.6\\0\\0.4\end{bmatrix}$$

So the resulting flow at period *p* for destination *d*, will take a 40% right and 60% left turn. We call this user compliance of the 1st order.

6.6 Summary

In this chapter we have discussed the traffic assignment model. The basic theory is a kinematic wave model consisting of a 1st order conservation of vehicle equation and a simple triangular fundamental diagram. Queues are modeled quite well in terms of spillback, build up and dissipation. Some drawbacks have been discussed, namely that traffic conditions react instantaneously to changing conditions. We have shown how we are able to model traffic heading for different directions and how we progress the traffic past nodes using the destination specific split fractions. We have also discussed the dynamic nature of the OD matrix which is attained by using a technique called time slicing. And finally we have discussed our route choice process, being a probit model which is able to select paths based on perceived user utility and translate them into a set of routes by adopting a stochastic approach. These routes are updated periodically which enable more realistic routing and makes it possible to calculate dynamic DRIP settings. Storing the routes as split fractions gives them mathematical properties which come in handy when customizing our route choice process to enable the modeling of DRIP settings.

The model described in this chapter is a *tool* which is used in evaluating generated DRIP settings using the *objective function* we already discussed in 5.3.1l; *the generalized gross travel times*. We have already shown how we can incorporate the calculated DRIP settings in the route choice based on an estimated parameter describing the *user compliance*. With this parameter we are able to make a better simulation of the effects of proposed DRIP settings.

In the next chapter we will discuss how the actual DRIP settings are generated and optimized using the genetic algorithm and complete the optimal control methodology. The verification of the DSMART Model presented in this chapter and the optimization algorithm is given in appendix I and should be read after finishing the next chapter.

7 Optimization of the DRIP settings using an evolutionary algorithm

In this chapter we will discusses the optimization algorithm, and start by explaining the basic operation of an *evolutionary algorithm* in order to provide some insight into this interesting type of heuristic. Following, we will discuss the different customizations and adaptations which have been made to the *general* theory in order to make it *efficiently* applicable to our problem. With these considerations in mind we will provide an *in depth* discussion of the resulting overall process and mathematical formulation of the various operators used. The verification of the optimization algorithm presented in this chapter is given in appendix I, together with the verification of the DSMART model.

7.1 About Evolutionary Algorithms

What we have named earlier on as a *genetic algorithm* is in fact a member of a more generic set of optimization techniques called *evolutionary algorithms*, which we will discuss here.

> An evolutionary algorithm (also EA, Evolutionary Computation, and Artificial Evolution) is an algorithm using evolutionary techniques inspired by mechanisms from biological evolution such as natural selection, mutation and recombination to find an optimal configuration for a specific system within specific constraints.
>
> Evolutionary algorithms include *genetic programming* and *genetic algorithms* which use the binary gene transmission and mutation mechanism as an optimization technique. Another technique is *evolutionary programming*, which allows one to parameterize computer programs to find optimal solutions according to a goal function. And the last branch is the *evolution strategies*, which work with vectors of real numbers as representations of optimization problems; mutation and adaptation of mutation rates are important working mechanisms there.
>
> Most of these techniques are similar in spirit, but differ largely in the details of their implementation and the nature of the particular problem domains they have been applied to. *Evolutionary algorithms* are often used to design engineering systems in the place of manual design where the complexity of the optimization problem is beyond human comprehension. (www.brainyencyclopedia.com)

In this thesis we call our optimization technique a Genetic Algorithm, though strictly speaking it has more to do with an Evolutionary Strategy. In addition, a lot of customization has been done to the basic structure of the genes, and the operators of the evolutionary strategy making it particular suitable for generating DRIP settings in an efficient way.

7.1.1 How do they work

Evolutionary algorithms (EA) are techniques which perform very well in finding a global or near global optimum in complex or extremely large solution spaces. An EA is a way of solving problems where the search is steered by *previous solutions* and a *process of chance*.
An EA generates solutions to a problem by taking *elements of previous solutions* and *(re)combining* them into a *new* solution. These new solutions are *then altered by a random mutation* to a certain desired extent. The result is a *new solution* based on previous work but with some altered elements (elements are called *genes*) which could prove very useful. In EA terms, a solution is called a *chromosome* and it's specific values or parameters to the problem is called the *gene data*.

When a new solution has been created, it is evaluated as a solution to our initial problem and *graded* afterward by means of a (scalar) score value we call a *fitness score.* This fitness score translates how

well or *worse* the solution performed and is used to place the proposed solution in a *ranking* together with the previously defined solutions (*objective function*). The collection of ranked solutions is called the *pool* and can be viewed upon as the *top##* of best performing solutions. Whenever a new solution is created, it *inherits* properties based on its parent solution(s) by a process which is called *crossover*. In addition some of the properties of the solution are *randomly* altered by a process called mutation rendering this particular solution unique.

The parents to a new solution are chosen from the pool by a process which is called *selection*. In general the higher the *fitness score* a chromosome (potential parent) has within the pool, the bigger its chance of being selected as a parent becomes. By adopting this technique of *selecting* parents and *inheriting* their characteristics based on their *fitness score* the overall value of the fitness score becomes higher as the number of created *generations* increase. The whole process of creating new solutions and evaluating them can be viewed upon as a *batch* process, where every cycle a *batch* of solutions is created called a *generation*. Based on the individual scores of the solutions from these generations they are added, or discarded to/from the pool if they perform better or worse than the worst solution in the pool. Given time, bad solutions with low fitness score will be replaced by newer and better solutions, thus increasing the overall pool quality. And given the quality of the parents, it is the idea that the quality of the offspring will increase as well.

This process is illustrated in the diagram below:

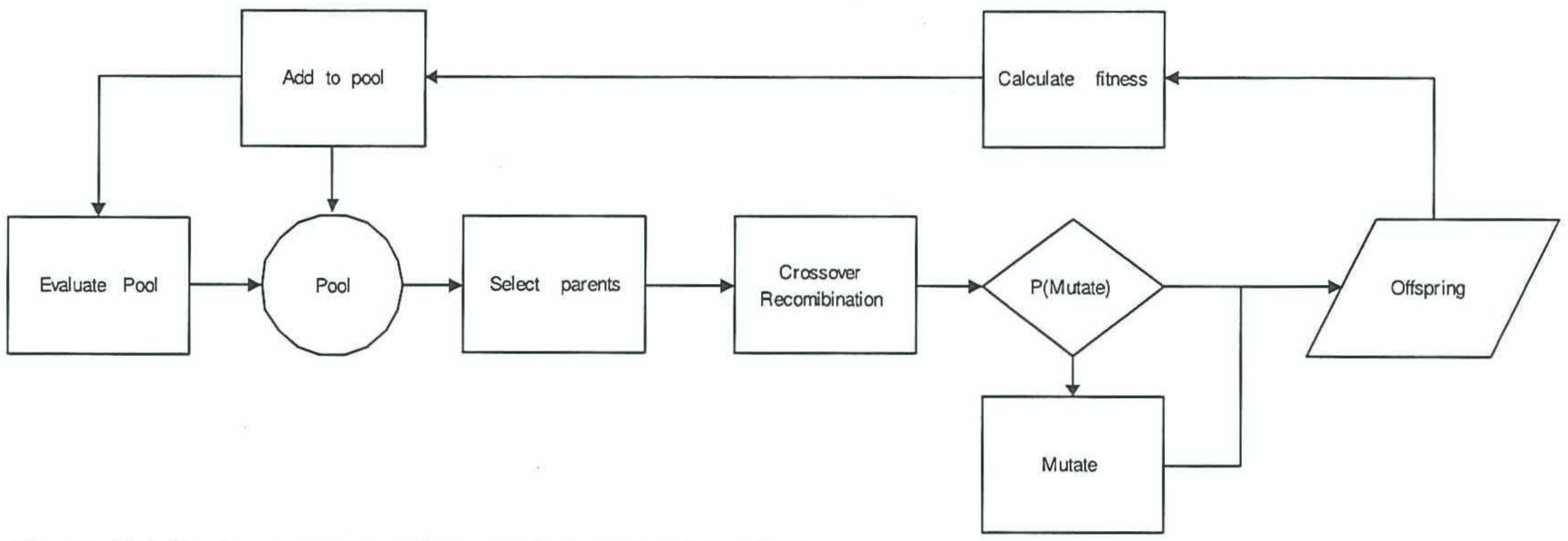


Figure 7-1 Basic operation of an evolutionary algorithm

i. The *pool* is the central theme within all EA, it is a collection off solutions, called *chromosomes*, to a given problem combined with an indication of their performance as a solution to this problem. This indication is called a *fitness value* and is used to make a distinction between the solutions. In general, the higher the score, the better the solution.
ii. *Select parents*. Selecting parents is the process where the best solutions from the pool are selected to become parents to new offspring. There are several ways of selecting parents but they are all based on the *fitness score*.
iii. *Crossover & Recombination*. The first operator used to construct new offspring is the *crossover* operator. This operator swaps elements of the solution between parents. Like in human reproduction, a child's DNA is a combination of the DNA of its parents. This is what the crossover operator does, the structure and size of the solution is the same, yet some values are from the father and some values are from the mother, rendering the child unique. The *recombination* operator is a special feature for evolutionary strategies; it enables new values for the offspring to be constructed as a function of the parental values.
iv. *Mutation.* Mutation is based on chance, which is represented by the choice block in the diagram. Both the *genes which are affected* and the *extent* of the alterations all depend on chance. The mutation operator is very important into finding new solutions, taking it

one step higher. Mutation enables large, fundamental changes which may prove very beneficial to the solution.

v. *Calculate fitness score.* When new offspring has been generated, there performance to the problem is evaluated and translated into a *fitness value* by means of an objective function

vi. *Add to pool & evaluate pool.* When the offspring is given a *fitness value*, it is added to the pool if its score is high enough. In this case the worst solution from the pool is replaced by the fresh, higher rated offspring.

About chromosomes and genes

Within EA, solutions to a problem are translated into a structure called chromosomes. The chromosomes contain all the necessary parameters which can be used in order to solve or optimize the problem. We call these parameters the *genes* or *gene data* and discuss the way in which they can be stored.

- *Binary* string; In classic *genetic algorithms* the gene data is stored as one large string of binary data. All *parameter values* are transformed into their binary representation and placed after each other in *one* large *string* of ones and zeroes.
- *Real valued data;* The binary string representation does have some drawbacks, namely being the fact that they are *bad* in dealing with real valued problems and show some *characteristic crossover behavior*. (Due to the classic binary representation, a differences in the position of one bit can change a value from 64 to 128, which could very well be a *too big a difference*. This is somewhat *eased* by using a translation scheme called *gray codes*, but still remains problematic). The *real valued* representation does not have these drawbacks and uses one large string of real values, but needs a special type of operator called *recombination* to be able to change the values during the crossover.
- *Strategy variables;* In some EA, gene data is accompanied by *strategy variables* which facilitate the mutation (or crossover) process of these genes into a desired direction.

The branch of *evolutionary computation* is not so much an exact science with strict formulations and rules of conduct as other types of heuristics optimization techniques might be. In the following pages we will discuss the chromosomes and the different operators used to create new chromosomes by means of a few examples. It is important to note that this is not so much a formal definition, but more an interpretation of the process in order to make things understandable.

7.1.2 Selection, crossover and mutation

Selection

Selection is the process used to determine the parents from the pool who will function as parents to the offspring which is generated. The most frequently used types of selection are:

a. *Proportionate reproduction*; The fitness value of a chromosome is proportionate to its chance of being chosen. This is the most common form of selection. A possible drawback of this method can occur when fitness values do not differ a lot from each other, and so the chances of getting selected become almost the same for all solutions.

b. *Ranking reproduction*; The fitness value of a solution defines the rank within the pool. These ranks are used to determine the chance of being selected.
One could also argue that in the case of a pool with chromosomes where the fitness values are very close to each other, ranking could also be an unfair solution. Because the lowest in rank might differ from fitness value in absolute sense very little but its chance of being selected is very low. This could very fast reduce the *diversity* of the genes.

c. *Tournament selection*; In tournament selection a random number of chromosomes is drawn from the solution and the best is chosen to be a parent. This process repeats until the mating pool is full.

Crossover
The crossover is the operator that swaps *gene* information between solutions.

The example to the right shows how the crossover operator works. We have two parent solutions at the outer right and outer left. They both function as a *blueprint* to the two offspring solution.

The crossover operator starts by selecting the beginning and end of the *genes* which will be crossed. In the example this region is indicated between the arrows. The middle two strings represent the offspring and the new gene data.

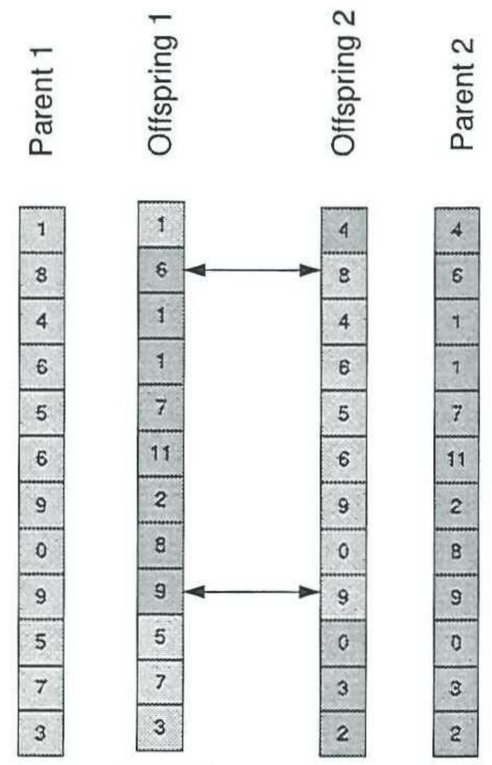


Figure 7-2 Crossover example

Recombination
Recombination works in a similar way as the crossover yet instead of copying the values, it creates a *sort of average* value between the original parent genes and the selected genes from the other chromosome.

In the example, two types of recombination are shown. Offspring 1 is created by the normal average values of the genes of both parents.

Offspring 2 is created by calculating an average and adding a distortion term ε. This is a normal distributed term with an average μ=1.

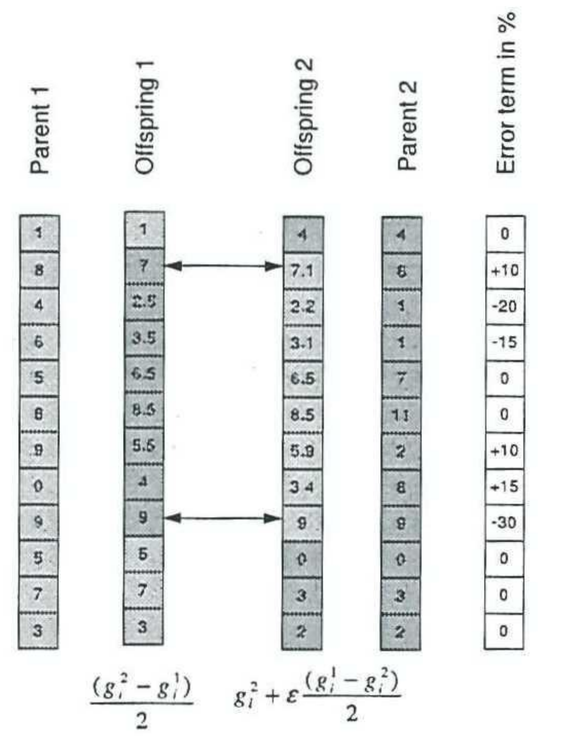


Figure 7-3 Recombination example

Mutation
Mutation is an operator which changes values randomly. Mutation starts by first selecting the genes to alter, the position in the genome, and continues by changing the actual values for this gene.

Mutation is very important to evolution, since it is able to extrapolate gene data and perhaps take solutions to a *new level*

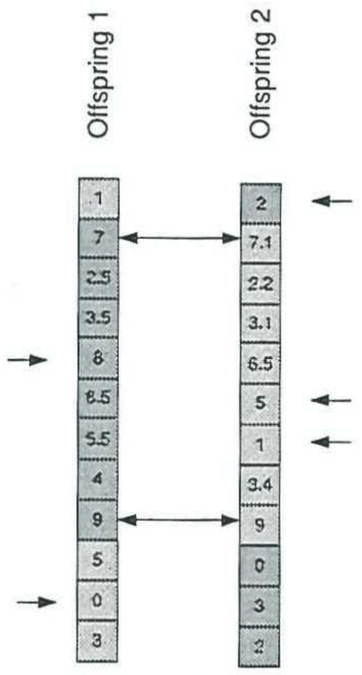


Figure 7-4 Mutation example

7.2 Customizations to the evolutionary algorithm

The general theory of evolutionary algorithm as presented, lends itself quite nicely to be applied in a wide variety of problems. Yet it proved very difficult to actually optimize using the *normal* real valued *string* representation of a chromosome. So in order to be able to search the huge search space efficiently we had to change the structure of our chromosomes / gene data and the way they are handled by the different operators. We will use this paragraph to summarize the most profound deviations from the general theory.

Chromosome and gene data
In this thesis we have created a somewhat customized chromosome. Our gene data consist of *real valued split vectors* $\psi_{i,d,p}$ per *link* per *destination* per *period*. This means that the actual *gene* data is not a binary or real number, but has the form of a real valued *vector of variable length* (depending upon the cardinality of the node to which the link is connected). In addition, the mapping of the gene data is not done in the form of one long string, but is stored in a three dimensional matrix SPF indexed by the *link, destination and time period.*

Strategy variables
Beside the SPF matrix we also define *strategy variables* to enhance the search process. The strategy variables are divided into two groups, aimed at efficiently:

- selecting a gene, and
- creating a new split vectors.

Selecting a gene: Active gene Selection
When working with either crossover or mutation, one first has to select the *genes* which will either be crossed or mutated. Since our gene data was mapped in a three dimensional it proved very beneficial to actively select genes instead of determining random locations. Some examples of the active gene selection used, are:

- only those genes (read: destination specific split fractions) at *links* which where actually used in the resulting assignment of the DSMART Model. (When a chromosome is added to the pool, it has been evaluated in the DSMART Model. One of the reasons of building a custom model was the feature of generating data which could be useful for the generation process).
- *destination* selection based on chance. Genes belonging to different groups of *destinations* have a different *chance* on being selected, depending upon their usefulness to the link position of the DRIP.
- Constructing *super periods*, to simultaneously select genes representing split vectors at the same link, for the same direction but during a different period.

Creating new split vectors: Mutation
The mutation process was altered thoroughly because of the nature of a mutation. Different mutation operators where created with a different chance on being used. The effect of the mutation could be entirely random, or based on strategy variables like network knowledge, spare capacity of links during the assignment etc.

Recombination
The recombination process had to be adjusted in order to work with split vectors. During this process it became clear that extrapolating a vector had some unwanted effects (impossible routing) which led to the fact that only averaging recombination is used.

7.3 The process in general

Before we begin describing the process in detail it might be wise to summarize the process so far:

We have built a macroscopic DTA model called DSMART which is able to evaluate the effect of a proposed DRIP setting in terms of the generalized gross travel time. We have been able to calibrate this model for the morning peak congestion of the Rotterdam network by means of an OD matrix and a set of dynamic route choice (translated into destination specific split fractions), which match the morning queue. At six points in this network, actual DRIPs are located and we can change their effect on the route choice of travelers by changing the destination specific split fractions at that location. What remains, is using the optimization process to find an optimal set of DRIP settings per destination per time.

In the following paragraphs the whole process of optimization will be discussed using the diagram below as a roadmap.

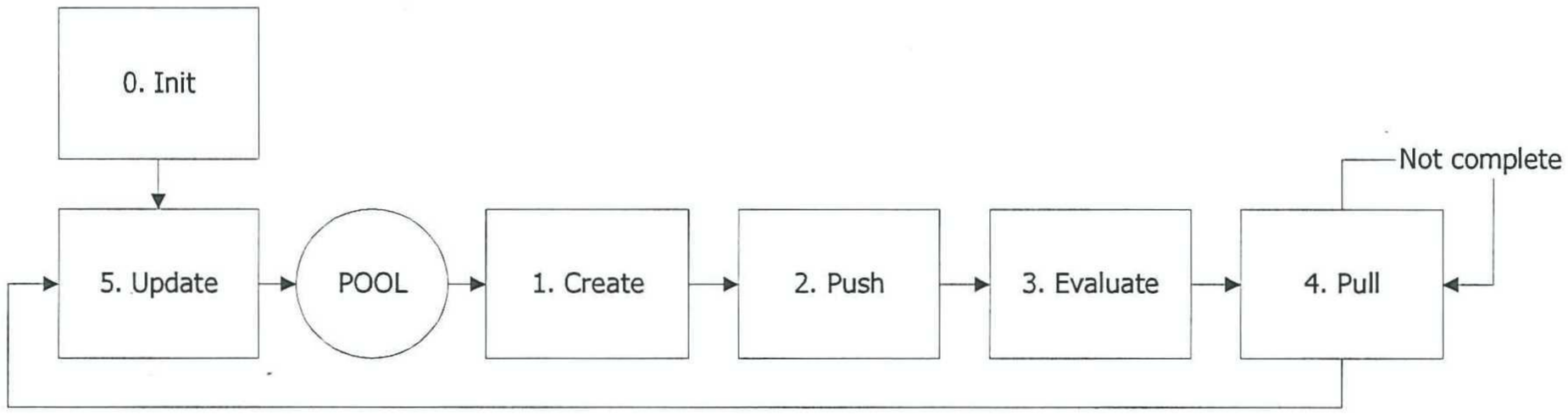


Figure 7-5 The optimization process using a custom evolutionary algorithm outlined

0: Init
The first step in the optimization process is loading the *calibrated situation* as our start point into the POOL and setting the different variables which influence our optimization process.

1: Create
Based on the number of chromosomes already in the pool we select a number of parents and create a new *generation* using the crossover and mutation process. By default a generation consists of eight different solutions.

2: Push
The new generation of solutions are evaluated in the DSMART Model. This step is called *push* because we push them to a *file server* where they are distributed amongst different computers who are all simultaneously running the DSMART Model.

3: Evaluate
Each model makes an assignment using the new DRIP settings and evaluates them by means of the *generalized gross travel time* and stores this result as the fitness value.

4: Pull
Whenever a chromosome has been assigned in the DSMART Model and given a fitness value, it is *pulled* back into the optimization process to be updated. This process continues until *all* chromosomes of one generation are pulled, if so the process moves on to step 5.

5: Update
Based on the fitness score the newly evaluated chromosomes are added to the POOL while older obsolete chromosomes with lower values are removed.

In the next paragraph the variables used in the optimization will be discussed, followed by an in depth explanation of the exact functional nature of the different steps. To explain these processes in an understandable way, schematics and formula's are provided. One has to keep in mind, that these type of algorithms have more to do with computer implementation than exact mathematics. Some of the provided formulas should therefore be interpreted as *pseudo code*.

7.3.1 Variables

This paragraph lists the variables which are used in the optimization process, grouped by their area of application. In the following paragraph we will show how the variables are used in the different steps of the optimization process described in the previous picture.

Overall variables
The overall variables are used to *steer* the optimization process and provide settings for the different steps within. These are the variables which are set at the *init phase* and can be *changed* during the optimization process.

g	Generation number
C	The collection of chromosomes
Θ	The Pool
Φ	Generation
Φ^{ID}	ID counter for unique chromosome ID's.
Φ_g^{size}	Generation size at generation g. {8}
$\Phi_g^{resolution}$	Split vector resolution at generation g {0.1}
$\Omega^{type}, \in \mathrm{N}$	Absolute size of variables of certain *type*
$\upsilon^{type} \in [0..1]$	Relative size of variables of certain *type*
P^{CC}, P^{mut}	Single Chance on crossover, mutation
λ_m^{type}	The chance on event *m* from the chance vector *type*
$\alpha \in [0..1]$	Uniform distributed random variable

DRIP variables
The DRIP variables describe the location of the DRIPS, the grouped destinations and the chance vectors. Before we continue with formalizing the variables we graphically represent below:

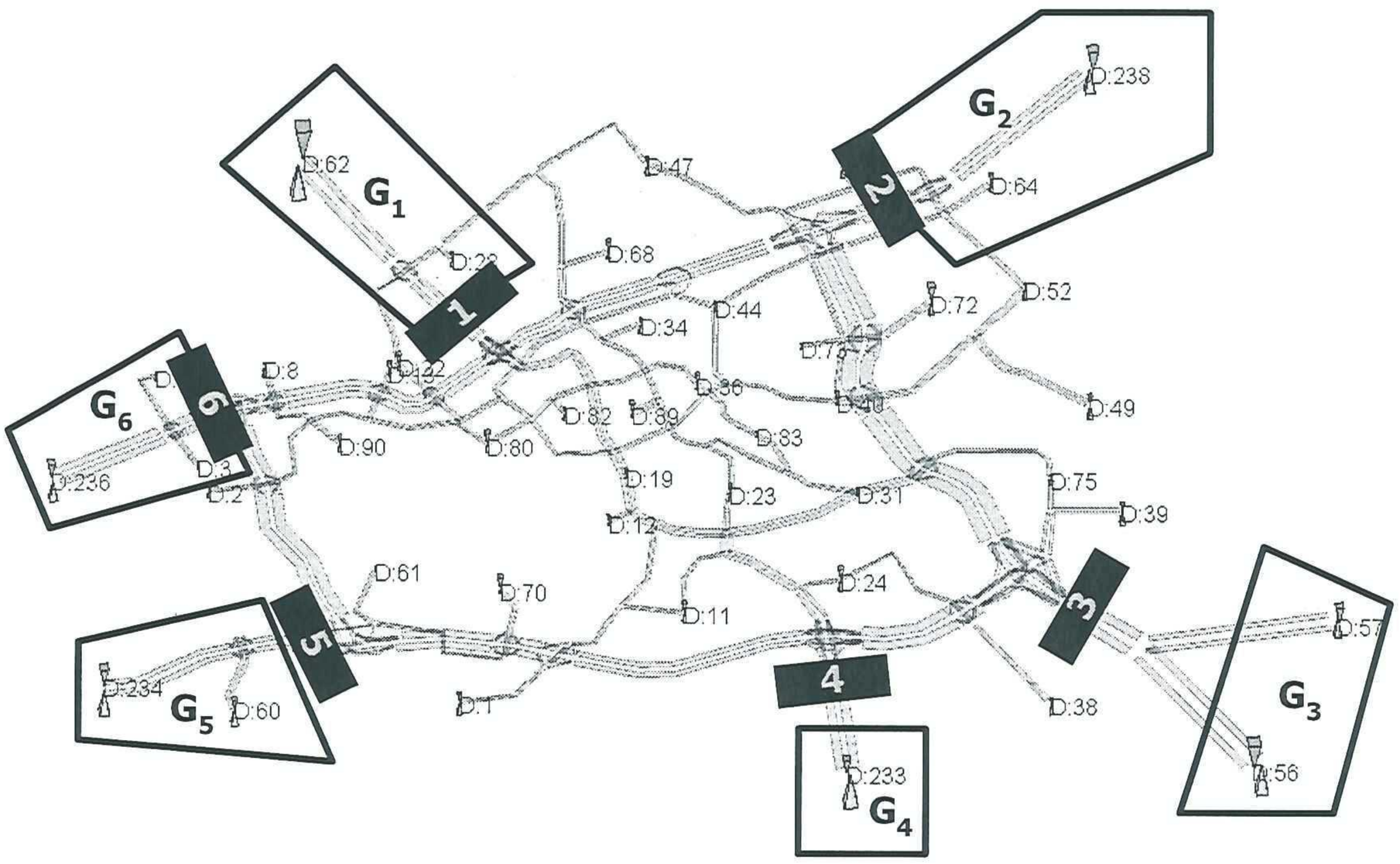


Figure 7-6 Location of the DRIPS and destination groups for which they can provide information in Rotterdam

In the network above, we see six DRIPs indicated by [redacted]. We also see six groups of destinations, numbered $G_1..G_6$ captured within a black polygon. (A destination is represented as a normal pyramid pointing upward)

The idea is that all destinations *d* within a group *G* can use the information from a DRIP and each DRIP can carry information for three *groups*, (the three lines of space on a DRIP). In this case, the DRIPS on the *north side* (6,1,2) can provide information for the groups on the *south side.* (G_3,G_4,G_5), and the DRIPS on the south side (5,4,3) can provide information for the groups on the north side (G_6,G_1,G_2)

Example
If DRIP 3 was to provide information for group 6, it would mean that the split vector $\Psi_{i,d,p}$ would be the same for all destinations d within group G_6. Simultaneously DRIP 3 could also provide information for groups G_1 and G_2 by using a different split vector $\Psi_{i,d,p}$ per group for all destinations within these groups.

$z \in Z$	DRIP number *z* from the collection of DRIPS Z
$i_{z,G,p}^{DRIP} \in L$	Link number *i* from the collection of all links L, which functions as DRIP, based on the DRIP number z from the collection of DRIPS *Z* that is providing information for group *G* at period *p.*
$G_s \in d^{DRIPS}$	The collection of destinations *d* belonging to group *s.*
$\kappa_z \in G$	A vector referencing the groups *G* for which DRIP link *z* can provide information
λ_z	A chance vector, describing the chance that DRIP *z* provides information for group *G* from the vector K_z

Chromosome variables
Another group of variables is those dealing with the actual chromosomes and are listed below. A *chromosome* is represented by the icon below.

$C_n.ID$;	A chromosome is identified by its ID, which is unique.
$C_n.age$;	Each new *generation* a chromosome is allowed to stay in the POOL, it ages +1
$C_n.SPLIT$;	The SPLIT matrix used for route choice
$C_n.SPF_{i,d,p}$;	The SPF matrix used for route choice
$C_n.LA_{i,p}$;	*Link activity table* describing the usage of links per period during assignment. (*gene mapping* strategy variable)
$C_n.Score$	The fitness value for this chromosome

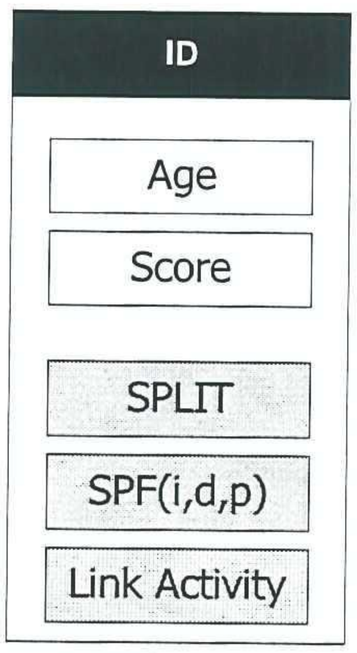


Figure 7-7 Chromosome Icon

Every chromosome has the complete set of route choices in it, stored in the SPLIT and SPF matrices, used to make a *dynamic traffic assignment*. The genetic algorithm only overwrites the split fractions for the DRIP locations and directions.

POOL variables

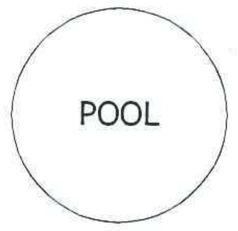
The final group of variables listed are those dealing with the POOL. The pool is the actual collection of chromosomes and has the following integer variables:

Θ^{max} ; Maximum Pool size; The maximum number of *mature* chromosomes that may exist in the POOL at any given time. {30}

Θ_g^{size} ; current size of the pool during generation *g*

Θ^{copies} ; Maximum number of copies that may exist in the POOL, {3}

$\Theta^{maturity}$; Maturity age. The age at which chromosomes enter the *survival of the fittest struggle*, {2}



$\Theta_g(n) = C$ Indicates chromosome number *n* from the POOL at generation time *g*

7.4 Creating chromosomes

So far we have provided an overview of the whole optimization process by means of the roadmap in figure 7-5. The *init phase* has already been addresses briefly since it holds nothing more than the initial setting of the various parameters and the loading of the calibrated route choices into the POOL as the first chromosome with score 0. We now continue with describing the first step, circled in red.

In this paragraph we discuss the 1st phase in the roadmap known as *create*. Creating chromosomes *efficiently* is the core of the whole optimization process and is therefore described in detail. This step holds almost all novelties which have been developed.

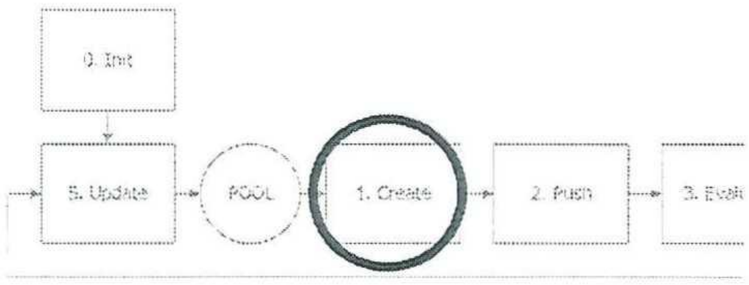


Figure 7-8 On the road: step1

Different steps in creating chromosomes
The exact nature of the *creation phase* is a process of five consecutive steps and described in the diagram below with corresponding paragraph numbers:

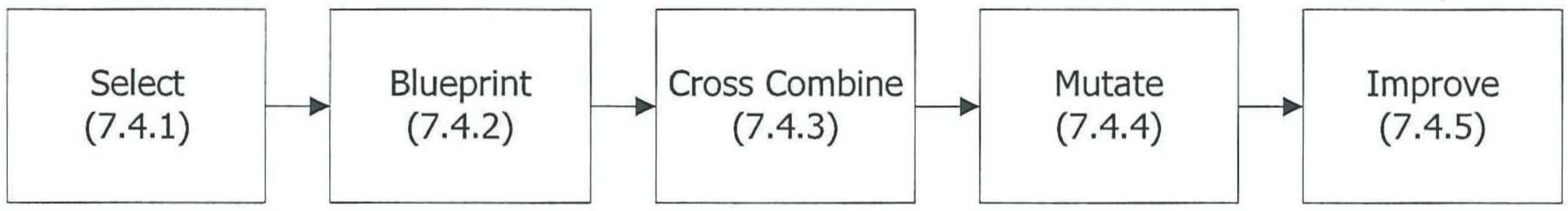


Figure 7-9 The different steps in creating chromosomes

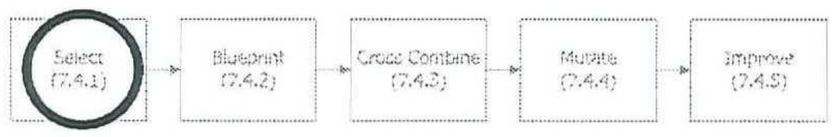
First, a selection of *parents* is made from the pool based on the *fitness value*. The parents are used as blueprints to a new *intermediate generation* and all their *gene data* is copied. After this, the gene data related of the intermediate generation is *crossed* and *recombined* in the *cross combine* step. In the next step the different chromosomes of this intermediate generation are *mutated* as efficiently as possible. In the final step of the creation process we try to improve the generated *split vectors* to make them better suitable and finish the creation process.

In the following paragraphs we will discuss the *selection, blueprint, cross combine, mutate* and *improve* steps one by one. However, the very first step in the process is the updating of the current generation number, as represented below:

$$g = g + 1, [-] \tag{26}$$

With this new generation number, we start the process of creating a new generation Φ_g. During the steps of figure 7-9 this generation is considered *intermediate* until the moment it is *pushed*.

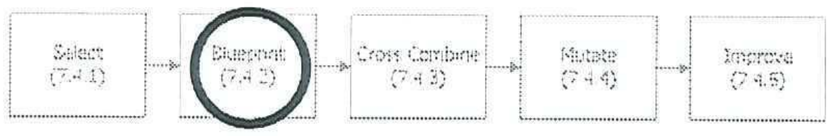
7.4.1 Selection



The selection of parents is done by means of a simple *tournament selection* scheme. Depending on the size Θ_g^{size} of the POOL during generation *g*, a number of *m* chromosomes are drawn from the POOL depending on the user defined size $\upsilon^{contes\tan ts}$, by default 0.6, in a α random manner (Uniform). Each time a parent is selected, its number is stored and the chromosome is placed back in the POOL for it to be selected again, a common practice in selection schemes. The contestant with the largest score from the selected group becomes parent and this process is repeated *n* times, until a number of Φ^{size} parents have been selected.

$$m = \upsilon^{contes\tan ts}\Theta_g^{size}, n = [1..\Phi^{size}],\ h = [1..m]$$
$$P_{n,h}^* = \Theta(\alpha\Theta_g^{size}).score \qquad (27)$$
$$P_n = P_{n,h}^*\ \forall n \in \max(P_{n,h}^*)$$

With $m, n, \alpha\Theta_g^{size}, P \in \mathrm{N}$. In the case where more than one *n* is found to be the maximum, the first value is used.



7.4.2 Blueprinting

The *blueprinting* step is where all characteristics belonging to the selected parents from the POOL to the intermediate generation Φ_g *P* are copied. We do this by copying the gene data, updating the description of the chromosome, set the age to 0, create a new ID and set its score to 0. This process is repeated for all *n* selected chromosomes.

$$P_n = \Theta(P_n)$$
$$\Phi^{id} = \Phi^{id} + 1$$
$$P_n.ID = \Theta^{ID}$$
$$P_n.score = 0 \qquad (28)$$
$$P_n.age = 0$$
$$P_n.description = 'parent'\Theta(P_n).ID$$

7.4.3 Cross Combine

The cross combine operator is used to *swap* gene information belonging to the DRIPS by making a *crossover* and a *recombination* of the gene data belonging to the intermediate generation. This data is stored per chromosome in the SPF matrix. Now remember that the mapping of the *gene data* was done in three dimensions: *link number i, destination d and period p*. By selecting a set of DRIPs *z* at random, we are able to determine the corresponding position *i* , then continue by selecting a *destination group* for which the DRIP displays information we find our indices *d* leaving only the period *p* to be selected. To simplify the process, we automatically consider all periods at once. So we don't just cross one single advice for one period, but all advices for all periods.

Assumption:
During the various trials, working with all the periods at once during crosscombine yielded quite good results. Yet the process has since then changed considerably so it should be investigated if this is still the best approach. For now, we stick to the assumption that working with all periods at once leads to good results.

The process works as follows: we first make a selection of the actual DRIPs $z \in Z$ which will be *crosscombined,* if we know which DRIP, we also know the exact link location i^* in the SPF matrix. Then, for every selected DRIP z we determine for which group G_z the gene information will be selected, this way we know the actual destinations d^*. When this is done, the chromosomes are paired up and the cross combine process takes place.

When all selected DRIPS *z* have successfully been crosscombined, the process is finished and moves on the next step: *mutation.*

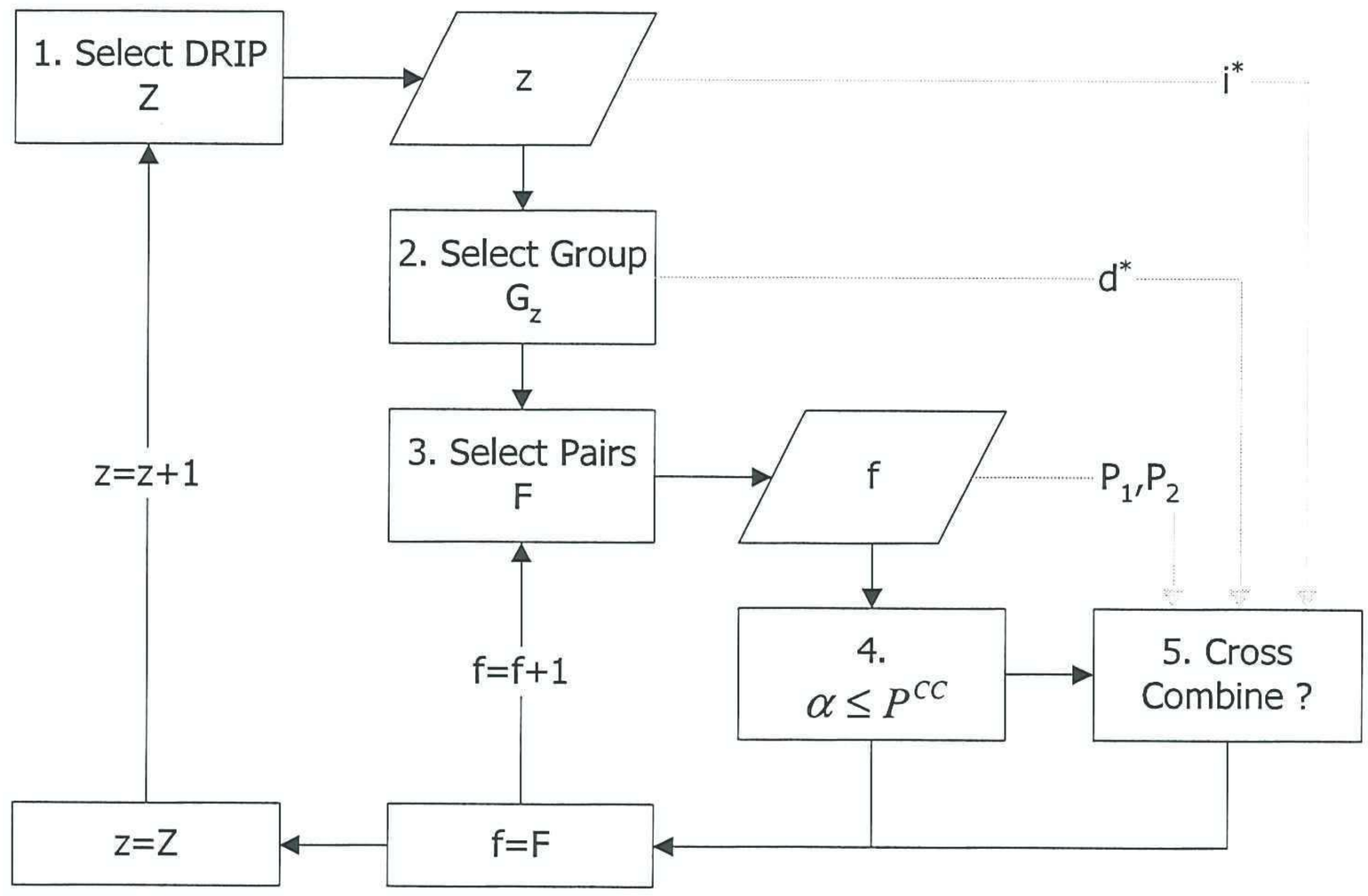


Figure 7-10 The Cross Combine process in detail

In the following paragraphs we will discuss the mathematics concerning the numbered items in the diagram above. The paragraph numbers start with the *block numbers* used above.

7.4.3.1 [Step 1]: Selecting the DRIP links to cross combine

We start by randomly selecting how many DRIPs will be used in the *crosscombine process.*

$$\delta \in [\upsilon^{\min .selection}, \upsilon^{\max .selection}], \in [0..1]$$
$$S = \delta\Omega^{DRIPS}, \in \mathrm{N} \tag{29}$$

Where Ω^{DRIPS} is the number of DRIP links and δ is a random number uniformly chosen from the user defined range. Then we randomly pick the DRIP numbers *z*, from the collection of DRIPS, to form the collection Z and make sure every z is unique.

$$z_s \in Z, where\ s = [1..S] \tag{30}$$

The actual link reference in the SPF matrix becomes $i^* = i^{DRIP}_{z.G,p} \in L$

7.4.3.2 [Step 2] Picking a group from a chance vector and selecting the destination nodes

Second, we determine for which destination group G the gene information will be swapped. We know the DRIP number *z*, and we pick the position in the group vector K_z by using the chance vector λ_z .

We make a new vector h with the first element 0, and the remaining elements equal to the cumulative sum of the chance vector λ_z .

$$h = [0 \quad \lambda_{z,1} \quad \sum_{n=1}^{2}\lambda_{z,1} \quad .. \quad \sum_{n=1}^{N}\lambda_{z,n}] \tag{31}$$

Now we use the uniform distributed chance α to determine the position for which it holds that:

$$h_n \le \alpha < h_{n+1} \tag{32}$$

The resulting group for which the gene data will be swapped is now equal to $\mathrm{K}_{z,n}$ With destination nodes $d^* = G_{\mathrm{K}_{z,n}} \in D$.

7.4.3.3 [Steps 3&4] Pairing up and deciding to crosscombine

Third, we pair up the intermediate generation Φ_g into F pairs of two parents $P_{f,1}$ and $P_{f,2}$ and decide whether or not they will swap information. This is based on the P^{CC} chance which is default set to 0.9. If the condition $\alpha \leq P^{CC}$ is met, we start the *crosscombine process* by providing it with the following information:

- the parents $P_{f,1}$ and $P_{f,2}$ belonging to pair f,
- the indices for the $SPF_{i,d,p}$ i^*, d^* and the all period vector $p^* = [1..P]$.

7.4.3.4 [Step 5] Crossover and Recombination

Now all data which is needed to perform the actual *crossover* or *recombination* process is known, and the operation can actually be performed; a process which is detailed below:

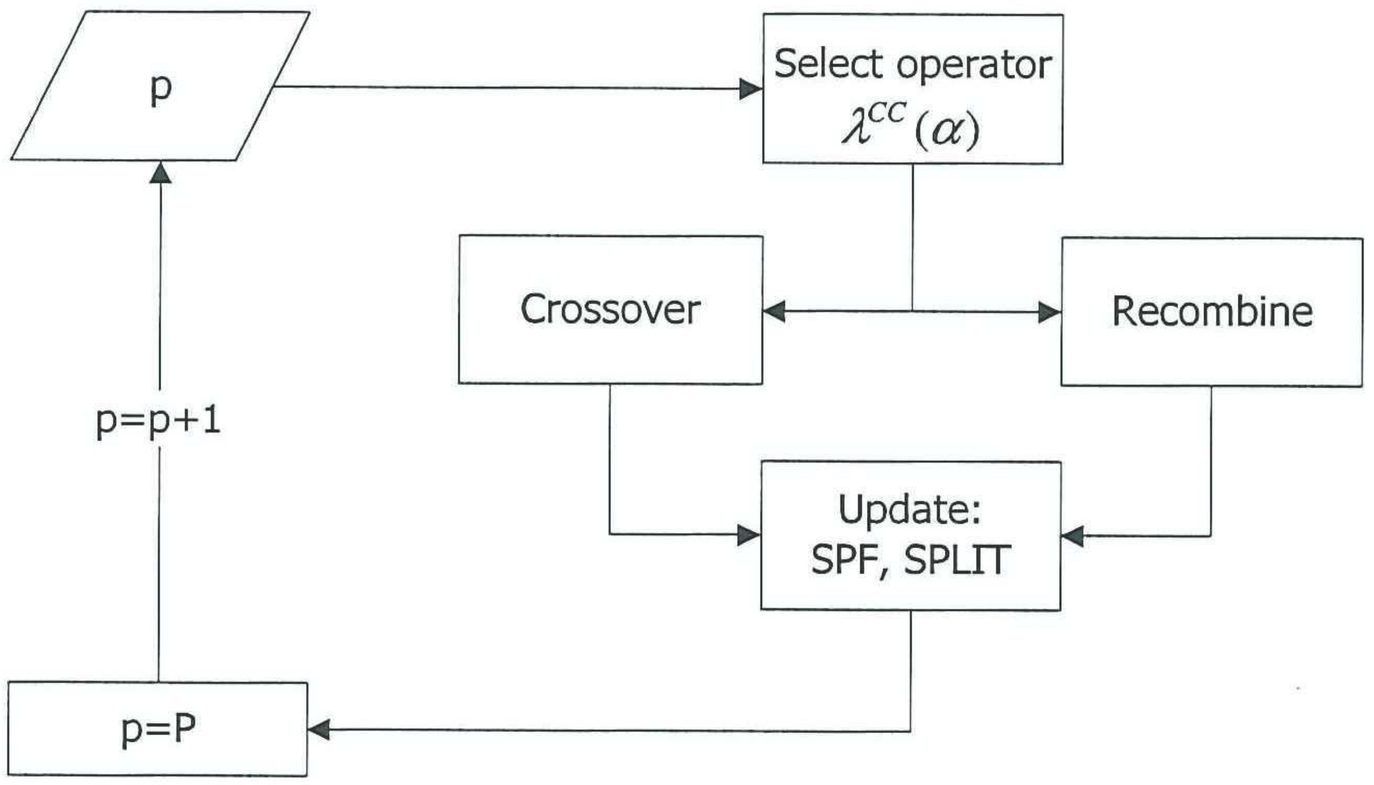


Figure 7-11 Performing the actual crossover or recombination

Individual selection of crossover or recombination per period
Though we have decided upon using all periods, it still has to be selected whether or not we will use a crossover or a recombination to swap the actual gene data. The cycle above shows how for every period p it is chosen to either perform a crossover or a recombination, based on the chance vector $\lambda^{CC}(\alpha)$ and a random number α. (by default the chance vector is [5 3]). When this choice has been made the actual operation can be performed.

Crossover
The crossover simply exchanges the split vectors $\psi_{i^*,d^*,p}$ for parent $P_{f,1}$ with parent $P_{f,2}$ and vise versa.

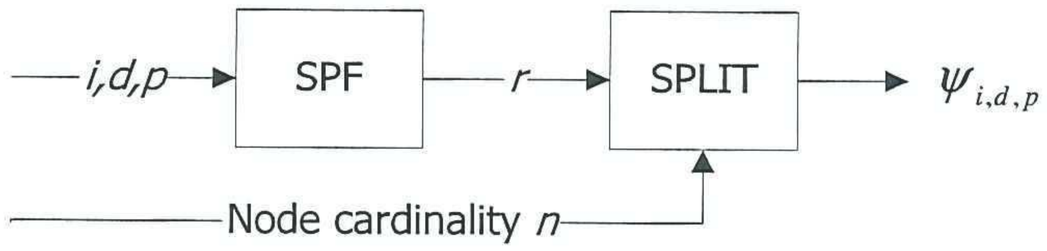
$$\psi^{P_{f,1}}_{i^*,d^*,p} = \psi^{P_{f,2}}_{i^*,d^*,p}, \qquad \psi^{P_{f,2}}_{i^*,d^*,p} = \psi^{P_{f,1}}_{i^*,d^*,p} \tag{33}$$

At this point we should address the way split vectors are stored for further comprehension:

Storing the actual split vectors

So far we have only mentioned the $SPF_{i,d,p}$ matrix which maps the different genes in three dimensions and shown that the actual gene is a *split vector* $\psi_{i,d,p}$ of variable length. To map this efficiently in terms of memory, redundancy and usability we have created a custom *reference structure* to work with.. When the $SPF_{i,d,p}$ matrix is indexed for a requested $\psi_{i,d,p}$ the retrieved value by the matrix is not the actual split vector but a *scalar integer reference* to a row in a second structure, called the $SPLIT_{r,m}$ table. This table has a variable amount of rows r which are used for the indexing, but a fixed number of columns. (default $m=10$). This means that the maximum *length* of a split vector can be 10 or in other words, a node can have at maximum 10 links to which it outputs.

Now whenever a split vector is needed, we 1) determine the number of outward connected links to this node, being the length n of the split vector 2) index the $SPF_{i,d,p}$ matrix and find the row reference r, and 3) use the row reference r to find the correct row in the and the $SPLIT_{r,m}$ table and copy the columns [1..n] to find the split vector $\psi_{i,d,p}$ with the correct length n.

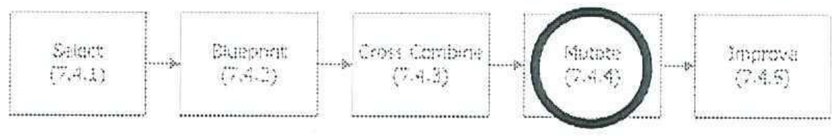


Whenever a split vector $\psi_{i*,d*,p}^{P_{f,1}}$ is calculated by (3) we check the table $SPLIT_{r,m}^{p_{f,1}}$ of the corresponding parent to see if this split vector already exists. If this is the case, we simply update the reference in the $SPF_{i,d,p}^{P_{f,1}}$ matrix to the row number r with the correct vector. Otherwise we add a new row and update the reference.

Recombination

The recombination simply averages both vectors into a new one. In normal recombination, an extrapolation factor δ could be added as well, however this did not have the desired effect and was left out. (it would lead to unwanted routing)

$$\psi_{i*,d*,p}^{P_{f,1}} = \frac{\psi_{i*,d*,p}^{P_{f,2}} + \psi_{i*,d*,p}^{P_{f,2}}}{2} \tag{34}$$



7.4.4 Mutating

Following the *crosscombine* process, the mutation operator is applied (see figure). During this process, *new* split vectors are created for the DRIPs and stored in the *route choice data*. The creation process is *guided* by means of strategy variables which result in a much more efficient mutation.

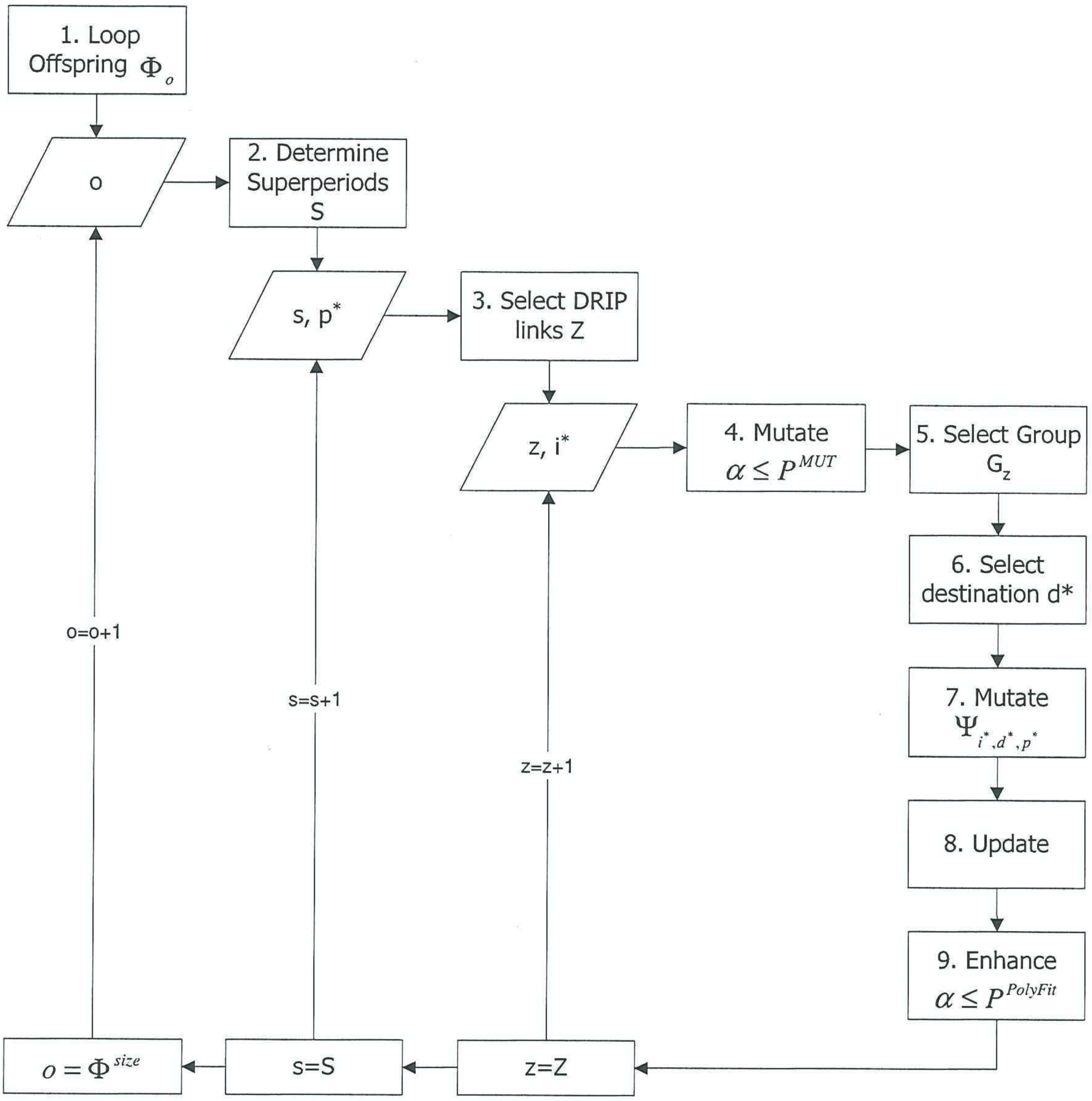


Figure 7-12 Functional diagram of the mutation operator

The mutation process is outlined in the diagram above, and discussed below:

1. All members of the intermediate generation Φ_g are mutated individually. Therefore a loop is set up so all chromosomes Φ_o are addressed individually.

2. A set of superperiods S is made and each superperiod s is mutated individually. The first period of the current superperiod s is period number p^* and considered representative. The superperiod is an aggregated division of the normal periods; discussed in the next paragraph.

3. Based on the collection of DRIPS Z, each individual DRIP z is considered for the current chromosome Φ_o during the current superperiod s. The link location, also index in the SPF matrix, belonging to this DRIP z is known as i^*

4. It is determined if a mutation takes place for the offspring Φ_o for the superperiod s on the selected DRIP z, based on the random variable and the chance on mutation $\alpha \leq P^{mut}$

5. If a mutation takes place, the algorithm continues by selecting for the current DRIP z a corresponding group G_z, for which the mutation will take place, based on the chance vector λ_z^{DRIP}. (See 7.4.3.2)

6. From this selected group G_z, a representative destination d^* is randomly selected and we consider the splitvector ψ_{i^*,d^*,p^*} as the representative *old split vector* ψ_{i^*,d^*,p^*}^{old}

7. The old split vector ψ_{i^*,d^*,p^*}^{old} is mutated into a new vector, which is discussed in detail in 7.4.4.2

8. The new split vector ψ_{i^*,d^*,p^*}^{new} is updated in the *gene* data (read: SPF and SPLIT) of the current offspring Φ_o as the new setting for DRIP z, during super period s, based on the selected *typical addition method*, by the chance vector $\lambda^{Addition}(\alpha)$. This process is explained in 7.4.4.3

9. Based on the chance vector $\alpha \leq P^{polyfit}$, all the settings (split vectors) for DRIP z toward destination group G_z are fitted to a 2nd order polynomial for all periods this process is explained in 7.4.4.4

10. And finally, all the *new generated split fractions* are discretized into split values with a resolution of $\Phi_g^{resolution}$ (default 0.1)

In the ensuing of this paragraph, we discuss some of the steps from figure 7-12 in more detail. This is indicated by the paragraph name beginning with the corresponding *block number#* and the prefix [Step #].

7.4.4.1 [Step 2] Super periods

A super period is a *collection* of periods for which the DRIP settings are the same. By introducing the *superperiod* it becomes easier to mutate beneficial DRIP settings in multiple periods. In the picture to the right, an example is given for three different types of *period divisions.*

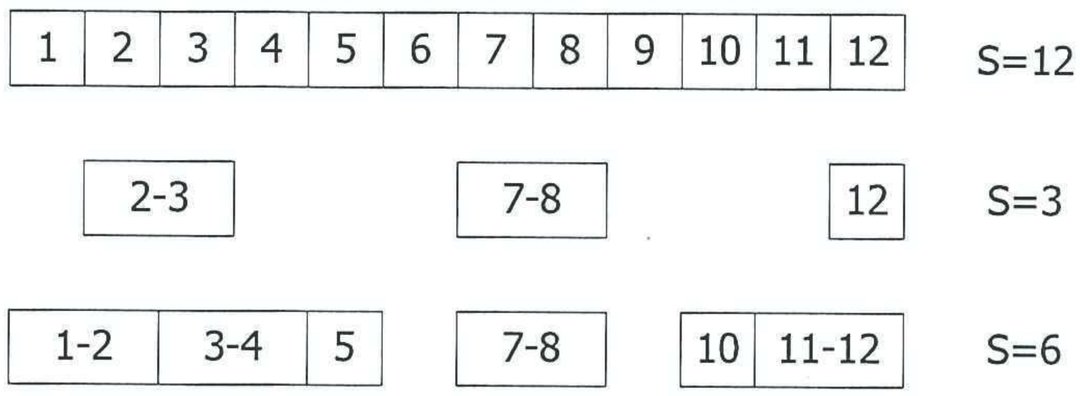


Figure 7-13 Graphical representation of a superperiod

The top row is the normal distribution of periods, by default *12* consecutive periods. The other rows show two different *super period* divisions. A super period is constructed in the following way:

i. The number of *super periods S* is determined from the user settings.

$$\text{Where } S = \Omega^{perios} \nu^{SP}, \quad \nu^{SP}(\alpha) \in [\nu^{SP}_{\min} .. \nu^{SP}_{\max}] \leq 1 \tag{35}$$

$\Omega^{periods}$ is the number of periods used in the simulation, and $\nu^{SP}_{\min}, \nu^{SP}_{\max}$ are user defined boundaries which describe the interval from which the random variable α is selected

ii. When the number of super periods is determined, they are filled with actual values
This process is based on the trick of defining two vectors with random selected unique values from the interval $[1..\Omega^{periods}]$ with length *S*. We call the start and stop vector, now we sort them in ascending order. And the *nth* element in the start and stop vector determine the start and stop period for *super period n*.

iii. When the *super periods* are created, we call the first period within a super period the representative period p^*.

During the development of the optimization process, it became clear that the actual *optimization* would have to be *instigated* by the *mutation* operator. Where it would later be carried on into new generations by the *crossover* and slightly averaged / marginally increased by the *recombination* operator. The problem arising was how to mutate consecutive periods in a beneficial way? The chances for this to happen by pure random mutation are very slim but could be increased tremendously by developing a variable *superperiod*.

7.4.4.2 [Step 7] Mutation operators

When it has been decided for which chromosome Φ_o, for which DRIP z, for which super period s and destination group G_z a mutation will take place we use the old split vector ψ_{i^*,d^*,p^*} as a starting point for our mut(il)ation process.

As mentioned earlier, we try to generate *useful split vectors* in the hope of achieving a high level of beneficial mutations and thus increase the overall optimization time. In order to do so, five distinct types of mutation have been developed who each take a different approach to the process and use different strategy variables.

[2]: New vector
The new vector mutation generates random *new real values* to use for the split vector.

i. First the number of new values is determined based on the random variable α and the user defined *maximum new values length* setting $v_{\max}^{NV}$ (default =1)

$$n = \Omega^{outlink} v^{NV}, \qquad v^{NV}(\alpha) = \alpha v_{\max}^{NV} \tag{36}$$

Note that $v_{\max}^{NV}$ is the *relative maximum new value length*, so for a node with $\Omega^{outlink} = 2$, at best 2 new values can be constructed.

ii. Second, the location and values are randomly drawn to construct the new split vector, which is afterward normalized by (34), so it holds again that: $\sum \psi = 1$.

$$\psi = \frac{\psi}{\sum \psi} \tag{37}$$

[3]: Scale
The scale operator uses the old vector information and proceeds by scaling this. In this process the individual difference between the actual split proportions (the elements of the split vector) is increased or decreased. The process is done in four steps:

i. the *average value* of the split portions in the vector is calculated
ii. it is randomly determined to either scale the values *up* or *down* (increase or decrease the difference with average value)
iii. the *size* of the scale is determined based on a random value α and the *user defined maximum scale size*: $v^{ScaleSize}(\alpha) = \alpha v_{\max}^{ScaleSize}$
iv. Afterward the vector is normalized by (34) so $\sum \psi = 1$

[4]: Hustle
The hustle operator just hustles the existing *split portion values* within the old split vector in a random manner, resulting in the same values at different positions in the same set of *used* output links. (no other *links* are selected because the position of non zero values within the split vector remains the same)

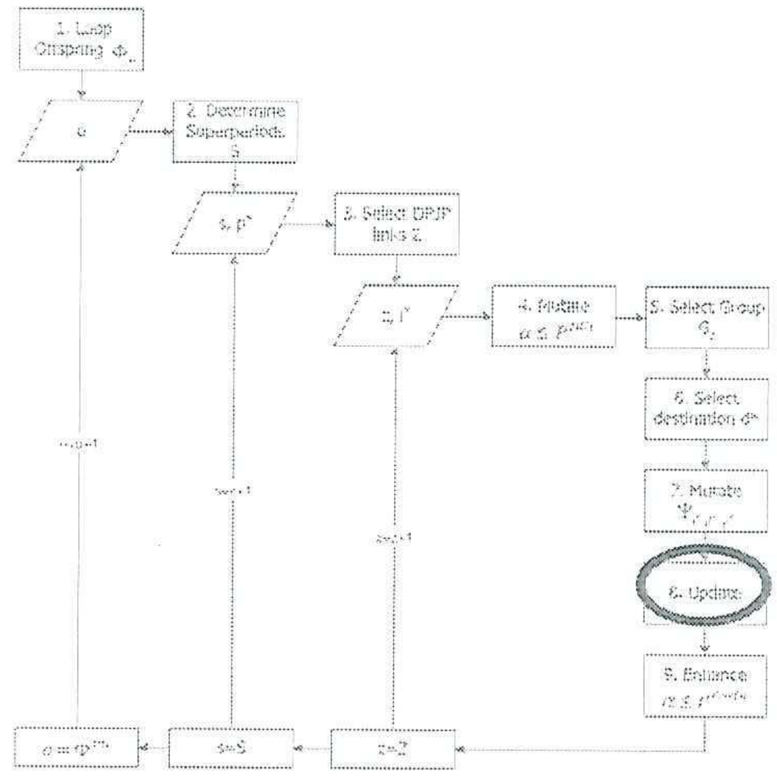
[5]: Spare Q
The *spare capacity mutation operator*, chooses *one* link from the possible set of *outlinks*, based on the amount of *spare capacity* that was available during the representative period p^*. This is an example of *strategic information* that has been gathered during the simulation and was stored in the *link activity* table of the initial parent to this intermediate chromosome Φ_o. The steps are:

i. a chance vector λ^{SpareQ} is created relative to the spare capacity available at that link
ii. a new link is selected based on this chance vector, resulting in a binary split vector

7.4.4.3 [Step 8]: Update the split vector

The actual mutation has taken place and the new split vector $\psi^{new}_{i^*,d^*,p^*}$ information must be stored in the *gene data* of the intermediate chromosome Φ_o we are currently mutating.

For clarity purposes, we display a small version of *figure 7-12* to the right and indicate our current position in the whole functional process of *mutation (green circle)*. Remember this is still the fourth step of the creation process



During this *update step*, the final split vector which will be used is calculated and added to the route choice matrix SPF and SPLIT table for the current chromosome.

The *update* process itself is depicted below:

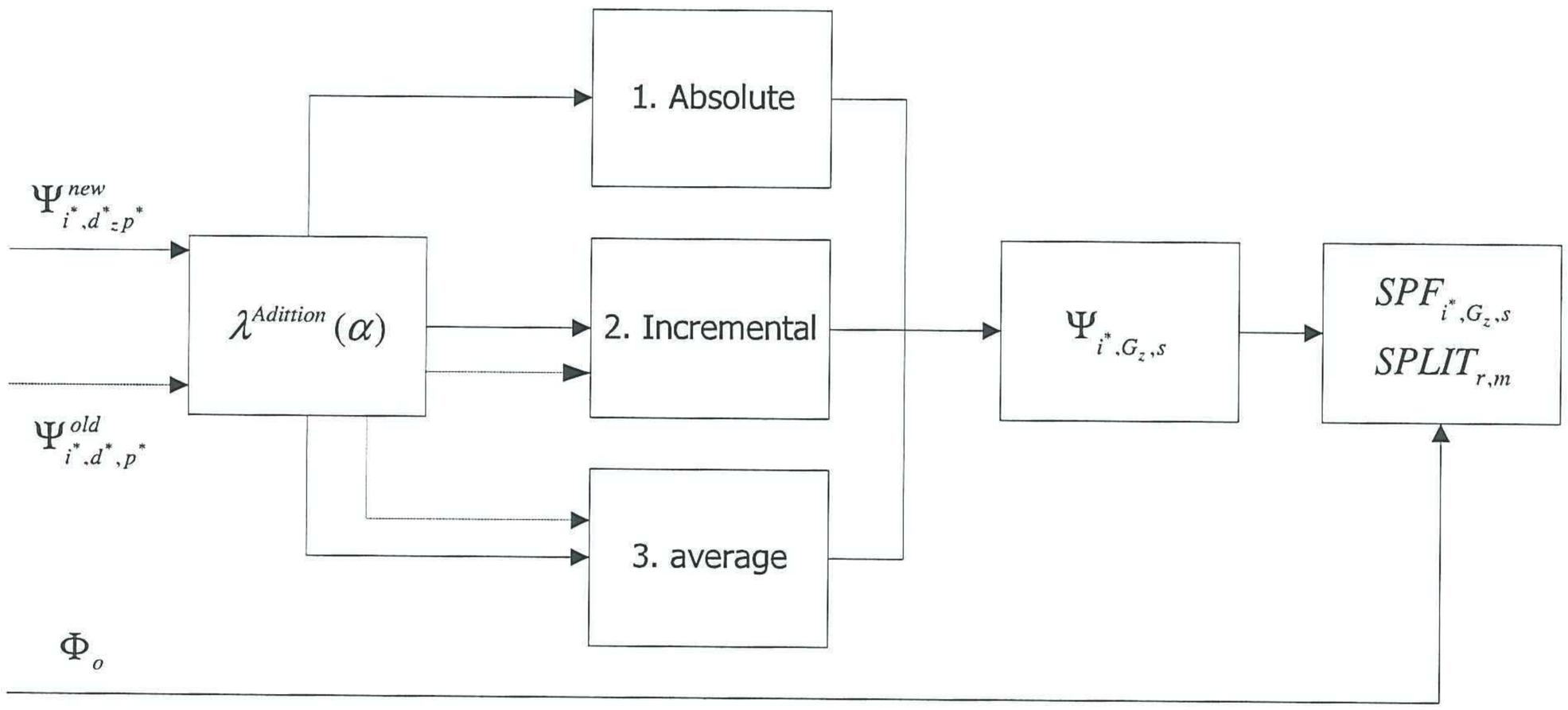


Figure 7-15 Updating new gene data in the chromosomes

The process begins by selecting the *type* of *addition method* we will use. This is done by a random variable and the chance vector $\lambda^{Addition}(\alpha)$, by default this vector is [2 1 1]. The second step is performing a final operation on the *new split vector* $\psi^{new}_{i^*,d^*,p^*}$ based on the data of the old vector $\psi^{old}_{i^*,d^*,p^*}$ and the chosen *addition* method. The result is a *collection* of split vectors $\psi_{i^*,G_z,s}$ that are stored in the route choice data (SPF and SPLIT) of the chromosome Φ_o.

Remember that the new generated split vector was based on the *representative* data for the destination d^* and period p^*. The final resulting *collection of split vectors* are written to all

destinations $d \in G_z$ who are a member of the selected group of destinations and all periods $p \in s$ belonging to the current *superperiod* for this DRIP. The split vector is the same, but it is written to different positions in the genome.

In the reaming text of this paragraph we discuss the three type of addition methods.

[1]: Absolute route replacement
In this case, the new split vector is left unchanged and will function as the new DRIP setting.

$$\psi_{i^*,G_z,s} = \psi^{new}_{i^*,d^* z p^*} \tag{38}$$

[2]: Incremental route replacement
In the case of the incremental route replacement, the old vector and new vector are summed in the following manner:

i. the number of *addition* steps is uniformly picked from the user defined step interval

$$\Omega^{addition}(\alpha) \in [\Omega^{addition}_{\min} \ldots \Omega^{addition}_{\max}], \text{ default [1..3]} \tag{39}$$

ii. The step factor is determined by the random variable $\nu^{stepfactor} = \alpha\nu^{stepfactor}_{\max}$ and the user defined max *step fact*or setting (0.25)

iii. The resulting *splitvector* is calculated by

$$\begin{aligned} \Psi_{i^*,G_z,s} &= (1-\eta)\Psi^{old}_{i^*,d^*,p^*} + \eta\Psi^{new}_{i^*,d^*,p^*} \\ \eta &= \Omega^{addition}\nu^{stepfactor} \end{aligned} \tag{40}$$

[3]: Average route replacement
The Average route replacement is like the name says, the averaging of the old and new split vector to the final DRIP setting.

[3]: Storing the DRIP setting
The final resulting split vector $\Psi_{i^*,G_z,s}$ is updated in the route choice for:

- the current chromosome Φ_o,
- DRIP link position i^*,
- all destinations d belonging to the selected group G_z, and
- during all periods p_s belonging to the super period s

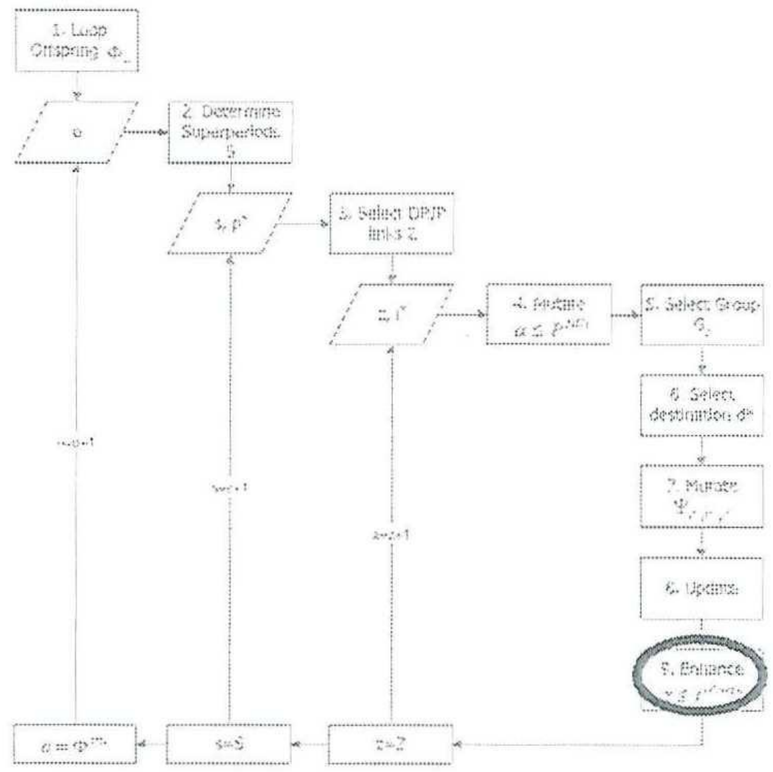
The actual updating is done by simply replacing the $SPF_{i^*,G_z,s}$ reference to the new split vector added to the $SPLIT$ table for the current chromosome Φ_o

7.4.4.4 [Step 9]: Fitting nth order polynomial

The final step in the mutation process is rather experimental and *not fully operational*.

Despite the introduction of the *super period* it still remains difficult to *mutate* a set of harmonic, fluent advices *in time*. Therefore an operator was developed which is aimed at creating a more harmonic, fluent distribution of the temporal advices. In order to do so we try to fit the split vectors for a given DRIP and set of destinations to a nth order polynome in time.

In its current application, it only handles 3rd order which is obviously not applicable to all advices. So the operator should be expanded with a preliminary step determining the desired order of the fit.



In the figure below we illustrate the *smoothing* effects which can be achieved by this type of operation.

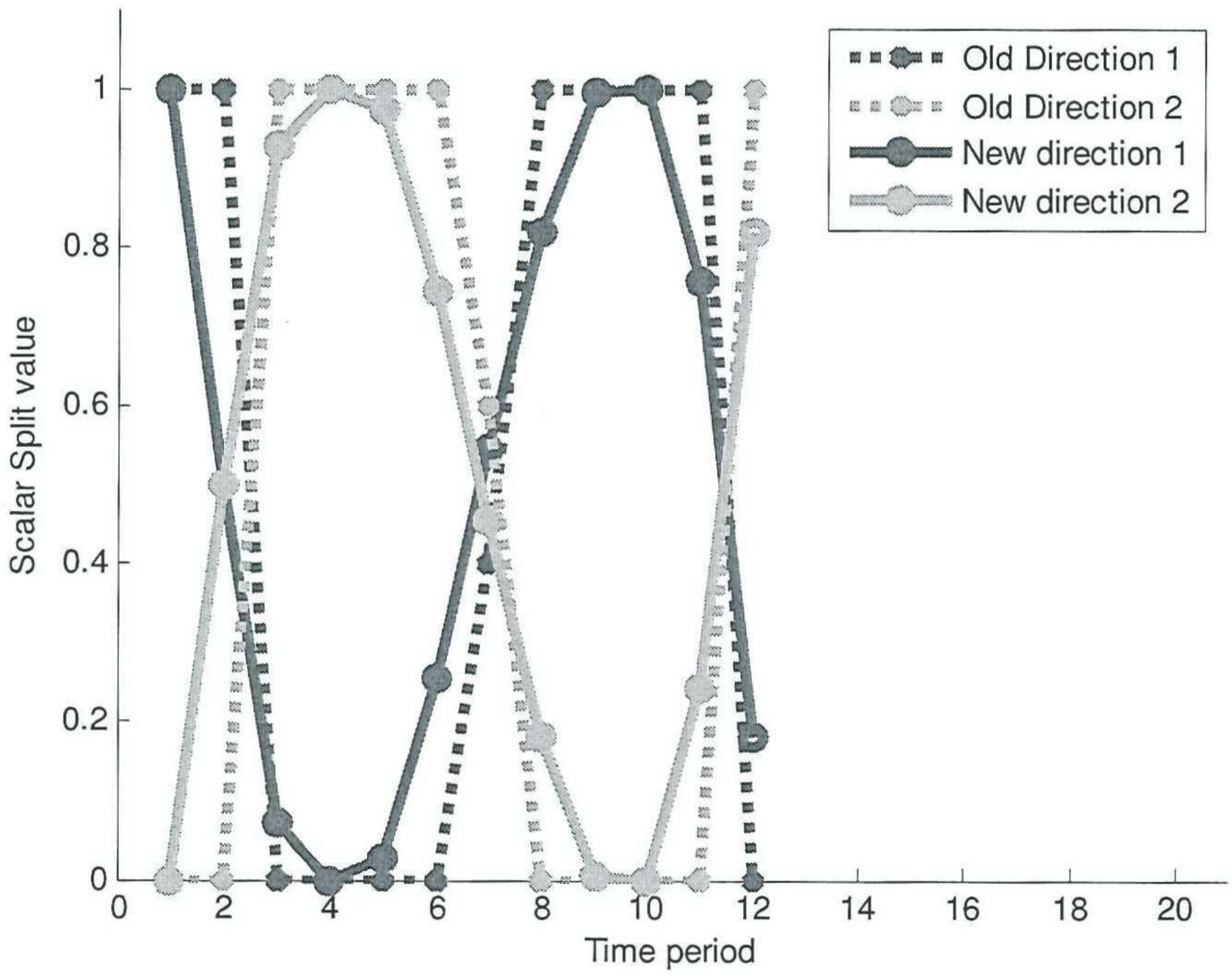


Figure 7-16 Example of smoothing DRIP advices in time by fitting a 3rd order polynome

The process is illustrated by the picture above. The dotted lines represent the old split values for two directions: red and green which are binary advices (*100% right or left*). The solid lines are the new split vectors calculated by a 3rd order fitted polynome to the data.

Due to the experimental nature and the stubbornness of a forced 3rd order polynome, the operator has a very small chance of occurring if $\alpha \leq P^{PolyFit}$ since the default chance value is only 0.025;

7.4.5 Improving split values

The final set in the creation process of new chromosomes is the improvement step. The operation is very simple and holds nothing more that discretizing the split vectors into the desired resolution. Remember that in paragraph 5.5.1 we calculated the total size of the solution space with (4). An important parameter in this formula was the resolution of the generated advice.

We use this resolution to discretize the advices afterward and reduce the search span. Due to the *recombination* and *mutation* processes the stored split vectors in the gene data of the chromosomes have become real valued scalars again. We discretize them into scalars with a resolution of 0.1 (default). This removes unwanted split vectors like [0.01 0.003 0.96] which form an unnecessary computation demand in the model and can never be translated into real life advices. This specific split vector would therefore be translated to [0 0 1].

And with this final operation, we can conclude the *create* process and move on to the next step, where the freshly created DRIP settings are put to the test in our DSMART Model.

7.5 Pushing, evaluating and pulling chromosomes

In this paragraph we will discuss the next *three steps* from in functional process, circled in red.

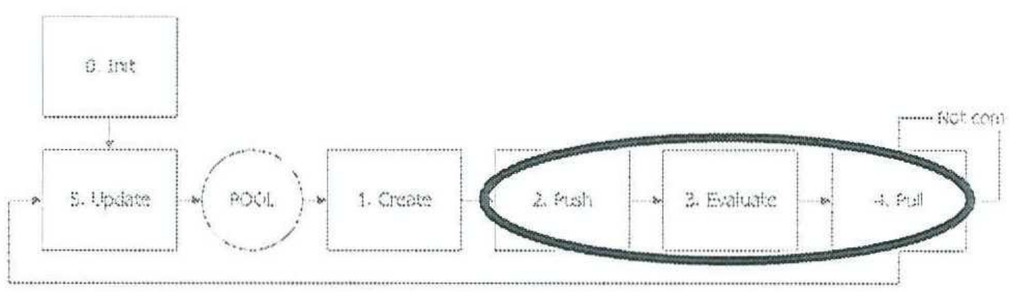


Figure 7-17 Pushing, Evaluating and Pulling chromosomes

The true mathematical nature of the optimization process has been discussed in the previous paragraph. The creation process forms the actual core of the optimization and was therefore considered in great detail. In this paragraph we will discuss the following three steps who have a more *functional nature* and therefore need only be explained briefly.

Parallel implementation
The specific functional structure was chosen because it enables a *parallel implementation* of the optimization process. One of the early explicit demands that arose during the object formulation was the need for a *fast methodology* of generating to make it applicable online. When reviewing the different optimization techniques we chose the *evolutionary algorithm,* with one of the reasons to do so, being the nature of an EA, which lends itself *perfectly* for parallel implementation. Implementing the process in a parallel way is simple, and starts right after a new generation of chromosomes Φ_g has been created. (by default the generation size was 8).

Push
All the individual chromosomes Φ_o belonging to this generation are *pushed* toward the *file server* where they are *picked* up by eight separate instances of the DSMART Model.

Evaluate
Each model *opens* a different chromosome and uses the data stored in the SPF and SPLIT matrix as the *route choices*. Based on this set of route choices, the model makes a dynamic traffic assignment and calculates the value of the objective function during this process. When the simulation is finished, the resulting value of the objective function (generalized gross travel time) is stored in the chromosome together with an updated *link activity table* (strategy variable).

Pull
The chromosomes are pushed back to the same *file server*, this time by the individual DSMART Models, where it wait to be *pulled* back in by the *optimization process* to be evaluate and added to the pool. (this last step is discussed in 7.6)

In order to utilize this parallel implementation feature we have adopted a so called *client – server* structure which is explained in the following paragraph.

7.5.1 Parallel implementation using client-server structure

The parallel implementation is realized in such a way in which the *server* handles the *creation*, *pushing*, *pulling* and *updating* of chromosomes, whereas the clients are only used for the evaluation of a whole generation Φ_g of chromosomes simultaneously.

We illustrate this to the right where the different steps of the overall optimization process which are performed by the server are colored grey.

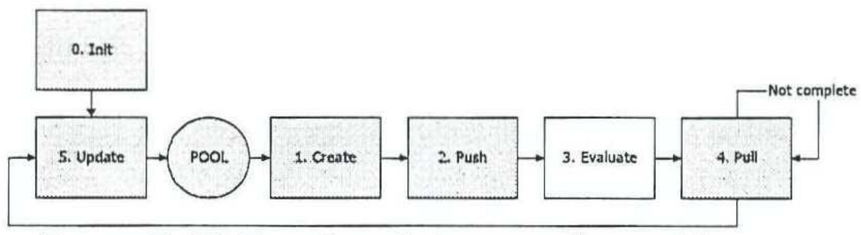


Figure 7-18 Functional steps performed by server (grey)

To realize the *client – server* structure, only a simple home network with enough available computers is needed which is displayed below:

Given a number of computers who are all connected to the same *file server,* and are thus able to share a network drive with read and write capability, we can set it up in the following way.

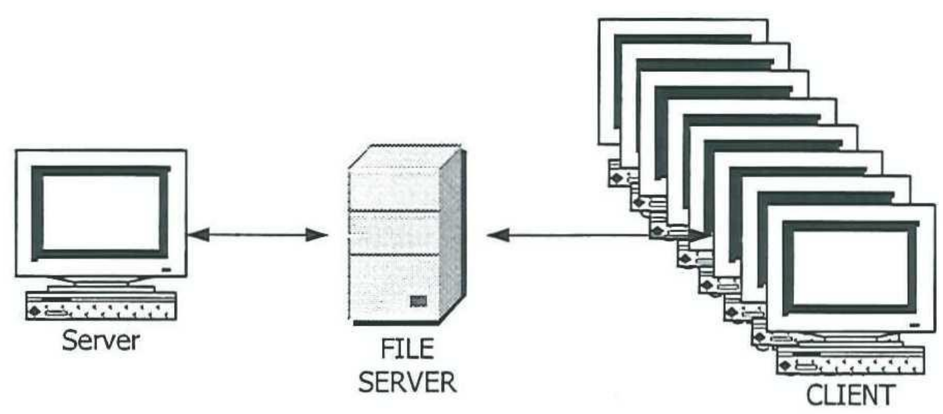


Figure 7-19 Simple network client - server structure

7.6 Adding the generation of evaluated chromosomes to the POOL

We will now discuss the last step of the optimization process which occurs when a new generation of chromosomes has been created, evaluated and *pulled* back into the optimization process.

This last step of the optimization process is *updating* the POOL with the evaluated generation Φ_g^{rated} and indicated by the red circle in the overall optimization process.

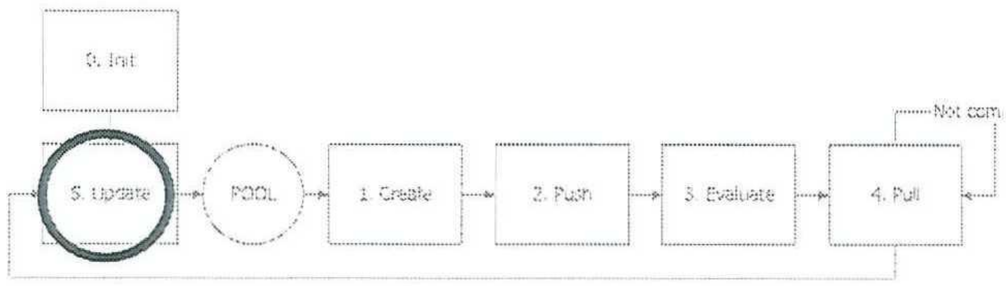


Figure 7-20 Updating the POOL

The actual process of adding the *rated generation* Φ_g^{rated} to the POOL consists of three steps and is depicted in the figure to the right.

We will discuss the three different steps in the text to come.

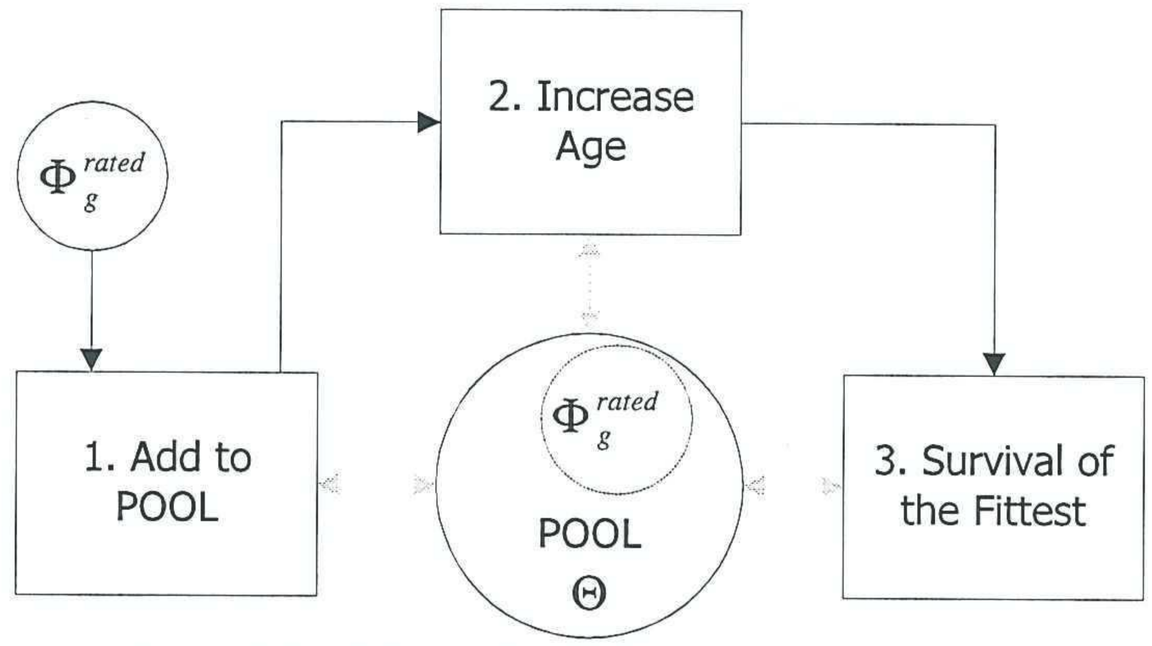


Figure 7-21 Updating the POOL with a new generation

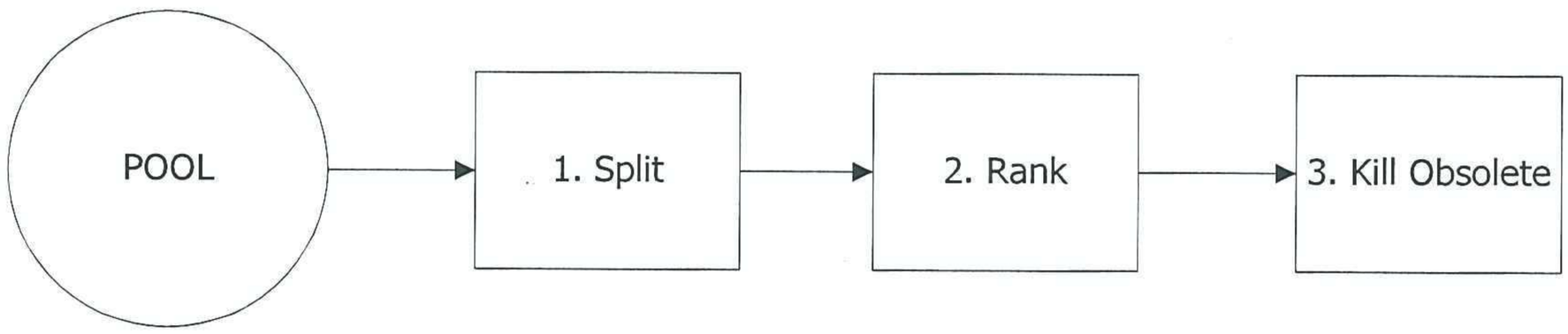
In general the idea of the addition step is that:

1. The entire Φ_g^{rated} generation is added to the POOL
2. The *age* of all chromosomes within the POOL is updated: $\forall C_n \in \Theta,\ C_n.age = C_n.age + 1$
3. In the *survival of the fittest contest* it is determined which chromosomes in the POOL are allowed to stay and which are removed. This is described n the following paragraph.

When step 3. is complete, the main loop starts again with step 1: creating a new set of chromosomes based on the updated POOL information.

7.6.1 Survival of the fittest contest

The *survival of the fittest contest* is used to determine which chromosomes survive to live another generation.



1. Based on the *age*, all mature chromosomes are selected. By default maturity starts with an age of 2.

$$M = C_{n,} \qquad \forall C_n \in \Theta \qquad \wedge \qquad C_n.age > \Theta_g^{maturity} \tag{41}$$

2. All solutions are ranked according to their score from high to low.

$$M = M^{ranked} \tag{42}$$

3. Only the first $\Theta_g^{\max size}$ chromosomes from the M^{ranked} solutions are allowed to stay. The rest is considered obsolete and removed from the POOL. By default the maximum size of the POOL is 30.

   During this step it is also checked if there exist any exact copies of chromosomes, based on their *scores* and the *link activity table*. When there are more then $\Theta_g^{\max Copies} = 3$ copies, the additional copies are removed, leaving extra room for new unique solutions from the next generation, *to boldly go where no chromosome has gone before*.

7.7 Summary

The optimization process as explained in this chapter is a specific application of an evolutionary strategy to our problem. In this paragraph the most important features are summarized:

- Route choices are stored dynamically in a SPF and SPLIT matrix in the form of *destination specific split fractions*. These split fractions are indexed by the link number *i,* destination number *d* and time period *p.*
- The optimization process specifically targets only the links which have a DRIP and from these links, only the destinations for which a DRIP could provide use full information for all periods.
- The DRIP can provide three lines of information and thus provide information for three *directions.* A direction is translated into a group of specific nodes. For example when an advice is given for people heading toward a large node in the network, this advice can be used by all people heading toward that node, or heading even further downstream of that node, knowing they first have to pass this node. Therefore collections of destination nodes are made (the groups) per DRIP which all use the same destinations specific split fraction.
- Not all directions can benefit from the advices on a DRIP so the directions (groups) with a higher chance of benefiting from the information have a higher chance on being selected.
- Travelers passing a DRIP can be sent to their destination in a clockwise or counter clockwise traveling direction of the motorway network. In addition two DRIPS can provide a third alternative through the city.
- The evolutionary algorithm works by selecting parents to a new solution and making a crossover of the gene data (only those split fractions related to the DRIPS). Afterward the mutation operators are used to change the values of split fractions. These operators are custom build and use network data to make a *smart mutation*.
- Providing temporal stable information for a number of consecutive periods is difficult if these advices have to be mutated. Therefore *super periods* are introduced which are mutated in the same way
- User compliance can be incorporated by simply adding the *new splitvector* to the *old value* from the calibrated situation

We will now continue with the implementation of the *generic approach to a route guidance strategy* in the next chapter.

8 Implementation of the generic approach to a route guidance strategy

At this point we are nearing completion of the whole process. A methodology has been developed, based on the continuous evaluation of DRIP settings by a custom build DTA model called DSMART (chapter 6). DRIP settings which are generated and optimized by an adapted *evolutionary algorithm* (chapter 7).

In this chapter the implementation of the methodology within the context of the *model predictive control methodology* (chapter 5) is discussed, together with the most profound boundaries. In addition the process is made suitable for application to our test case Rotterdam of which the results are presented in chapter 9.

We will begin by discussing the implementation of the various components in the *model predictive control framework* by providing a functional scheme. Given this functional implementation the technical implementation in Matlab is discussed and some basic properties and boundaries of the created software are illustrated.

At the end of this chapter one should have a clear idea of how we intend to *generically generate route guidance* for a real life size network and be anxious to read about the *test case* results in the next chapter.

8.1 The model predictive control methodology revisited

This paragraph shows how the *pieces* presented in the previous chapters should be fitted together to form the envisioned online route guidance generation system.

The diagram below presents the different elements needed for such a system divided into four different groups.

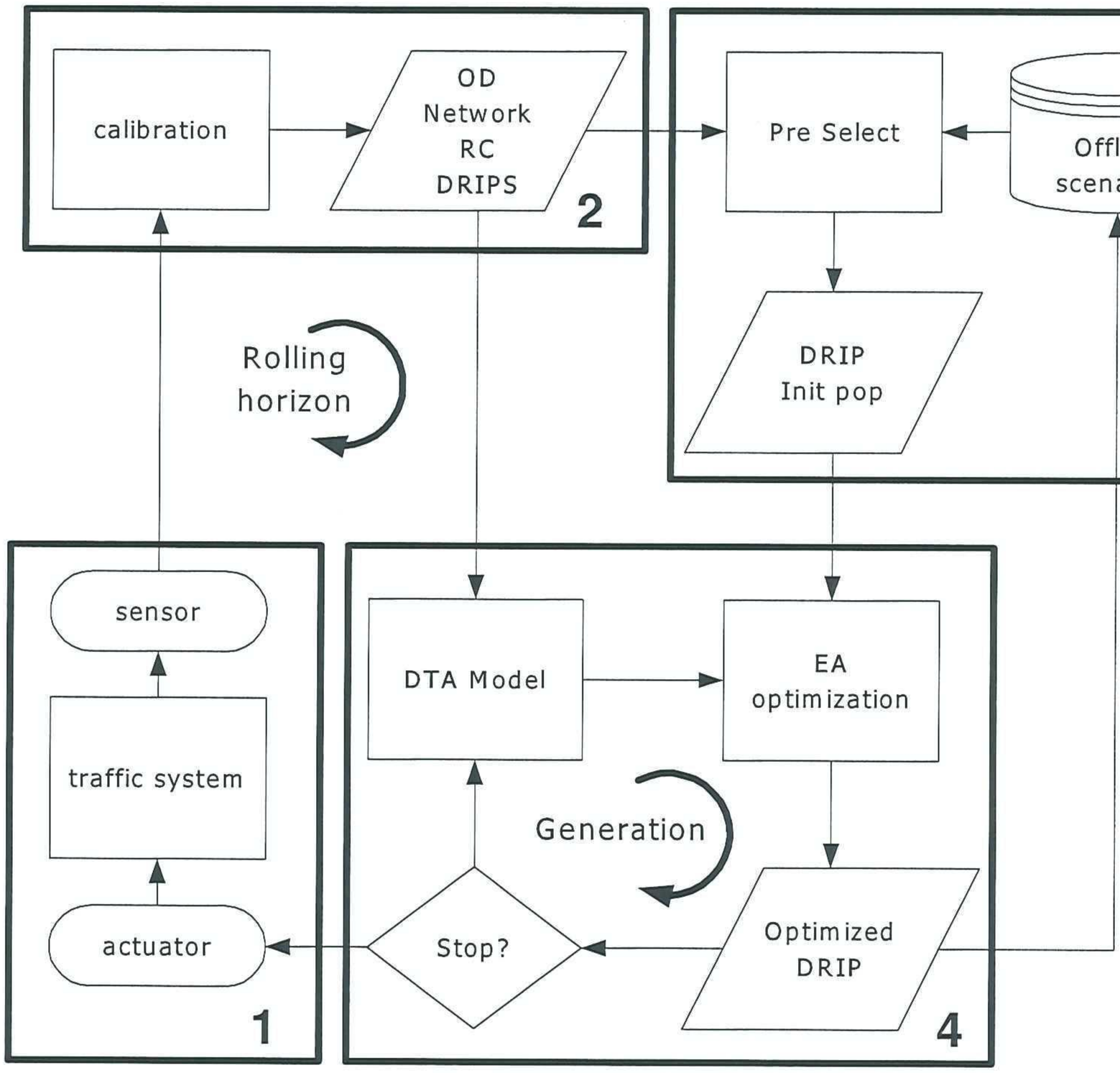


Figure 8-1 Four subsystems in the model predictive control methodology.

1. The traffic system with actuators and sensors
2. Continuous DTA model calibration to represent the current network state
3. Offline defined scenario's which could aid the search process
4. Optimization process

Given the diagram above, we can define two main cycles; the first is the *rolling horizon* which is a cycle aimed at continuously calibrating the model to the current measured state of the traffic system. The second cycle is that of the optimization process, which keeps on *evolving* the *route guidance settings* based on an o*ptimal control* of the short tem future predictions of the traffic system.

In the diagram we have more or less defined four numbered sub systems captured in black squares. The first is the traffic system itself with the *actuators* (DRIPS) and *sensors*, which collect measurement data about the network.

The second sub system is aimed at continuously calibrating the input data of the DTA model, to resemble the measured network state as best as possible. In the case of the *current DRIP settings*,

and *quasi static network data* (link capacity changes, variable speed limits, ramp metering etc) this is not really difficult. The biggest challenge would be in the continuous estimation of the *dynamic* OD matrix and the correct set of *dynamic route choices*. This is however a field of research which has a considerable amount of attention and where a lot of progress has been made in the past, and should therefore pose no fundamental obstacle in attempting to realize. (in [31.] a simple overview on OD estimation approaches is provided)

The third sub system is popular in different dynamic traffic management systems and is known as *scenario management*. When applied in these systems it is usually implemented as a *process that selects a scenario based on the current measured state,* which has been previously defined offline, and contains the set of traffic management settings which should be used. The biggest problem with such an approach to scenario management is the fact that the number of possibilities is of such *an enormous scale*, it becomes impossible to build a database to hold such scenarios, let alone define traffic management settings for it. What is proposed in the system above is not a database containing *all possible scenarios* but rather a *crude selection of a broad range of possible network scenarios* within reasonable limits in terms of the absolute number of scenarios. Such scenarios contain a few *key characteristics* but do not have to match the exact network state as currently measured. By using the DRIP settings for such scenarios as the *initial population*, the search process would be *sped up* enormously. We let the optimization algorithm *detail* the actual DRIP settings but provide a very good initial *guess* by means of scenario selection.

The fourth subsystem has been the focus of this thesis, a methodology that enables the calculation of optimal DRIP settings, independent of the type of network or current conditions. This subsystem is the core of the *model predictive control* approach as sketched above because of its ability to calculate optimal DRIP settings for *any given* situation.

At this point, the *model predictive control methodology* has been detailed into a functional system of which the operation should be clear to the reader, especially the powerful combination between a *crude scenario manager* and the *generic optimization process*, which can use these scenarios as input, to speed up the optimization process. The next paragraph will provide a functional layout of this optimization process and finish with a demonstration of its ability to optimize DRIP settings by discussing the verification results.

8.2 Functional design of the optimization process

With the presented *DSMART Model* and *optimization algorithm* in chapters 6 and 7, and the *route guidance generation methodology* summarized in 5.7 We will illustrate how these different elements have been implemented.

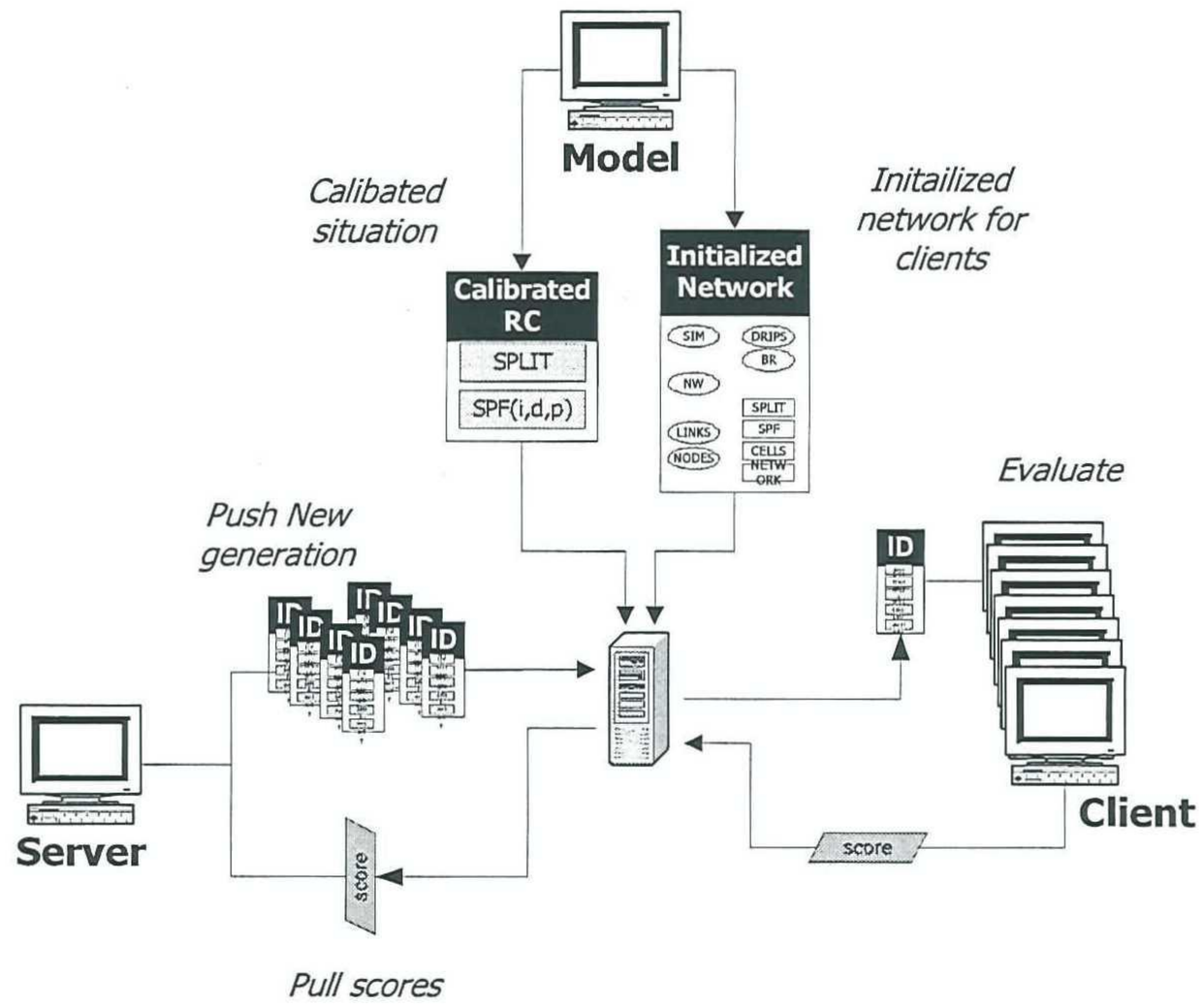


Figure 8-2 Functional design of the optimization process

This diagram displays the *implementation* of the process using parallel computing. The process roughly consists of two steps. in the first step, we calibrate the model to represent the real life traffic situation (In the rolling horizon approach, this step would be repeated periodically). And with this calibrated situation, we start the optimization process in step two.

Step 1: calibrating DTA model input to match real life situation
The DSMART Model presented in chapter 6, is used to initialize the network data and create a *calibrated* situation based upon the route choice process described in 6.4.

The network data is initially presented in text files determining the position of nodes, vertices and links together with an OD matrix and DRIP information. This data is processed into a collection of data structures with which the model is able to simulate the traffic. These structures are saved on the file server in a file named *"mainnetwork #time stamp#"*. Since the initialization process of the network data takes some time depending on the network size, it is wise to store the *structures* and load them later on in the clients and thus preventing the need for clients to initialize the input data every time they run a simulation. The *dynamic route choices* made in the calibrated situation are stored separately on the file sever in the format of the chromosomes used by the *evolutionary algorithm* and will function as the *initial* POOL inhabitants.

At the end of step 1, we have a calibrated situation stored in two files, one containing the network data and one containing the route choices belonging to this situation.

Step 2: Calculating an optimal set of DRIP settings
We continue by setting up a number of clients who are all connected to the *file server* (default 8) and manually load the *initialized network data* in each client separately.

When this is finished we set up another computer to act as the server and load the *calibrated route choices* into the server as the *first initial chromosomes in the POOL*. With this POOL we create the

first generation of chromosomes to be evaluated by the clients. The first generation is created by adopting such settings of the server that will result in a *severely mutilated version of the initial calibrated route choice chromosome*. This is done deliberately to *diversify the gene data* and prevent the evolution process into *premature convergence* bye the early adoption of locally optimal DRIP settings.

From that point on, each time the server pushes a generation of chromosomes they are picked up by the clients and used as input to the simulation. When a chromosome has been evaluated, the score is returned to *file server* waiting to be picked up by the *EA server* again and processed to the POOL and restarting the process of creating a new generation.

In the following sub paragraphs we will discuss the deliberate absence of *stop criteria* and discuss the default settings used in the *DSMART Model* and the *evolutionary algorithm.*

8.2.1 Deliberate absence of stop criteria

The proposed *optimization process* and its application in the *model predictive approach* in this thesis is by no means complete or finished in the sense that it is operational; more specifically it lacks *stop criteria*. Though the evolution of the average scores of all the chromosomes in the POOL does strongly reveal a typical *hyperbolic* curve, heading toward a certain *limit*, which would suggest an easy implementation of such a criteria, it still remains difficult to implement this in a satisfying way, especially without the *a priori knowledge* of the actual optimal value. I will try to illustrate this in the example below:

Example

In the picture to the right we have shown a typical POOL evolution picture where every position on the x-axis resembles a new generation and the values on the y-axis resemble the scores of the chromosomes in the POOL during that generation.

When dealing with *stop criteria* a frequently proposed method is to determine the difference between the *previous step* or (an average previous value). Whenever this falls beneath a *pre defined boundary value*, the stop criterion has been met. In the picture to the right we have illustrated an oval shaped green region where such a criteria would have been met in order to stop the process.

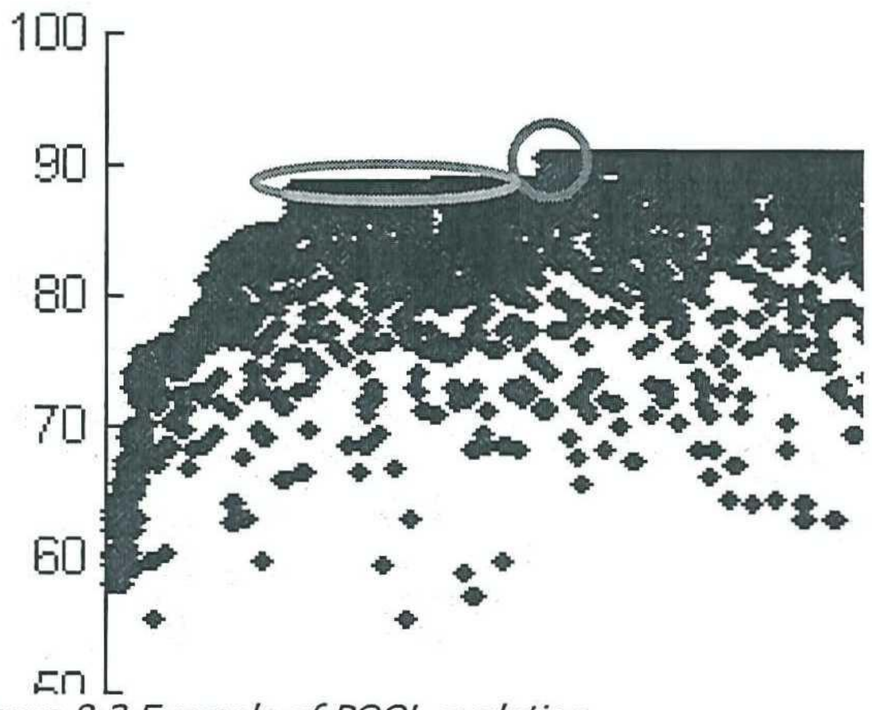


Figure 8-3 Example of POOL evolution

Yet the nature of an *evolutionary programming heuristic* depends on the *mutation* operator to spark so called evolutionary leaps (where the crossover *caries* quality and the recombination operator could *marginally optimizes*). These leaps will occur very often in the early stages of the evolution process as can be seen in the picture (although not solely sparked by mutation). When the number of generations increase the chance on such a leap becomes smaller, but nevertheless when they occur they are beneficial. A stop criterion of the previous proposed kind would have very likely been met *before* and thus preventing the *quality leap* in the red circle.

So without *a priori* knowledge of the *optimal assignment score value* it is very *tricky* to implement a stop criterion; this does bring us to take a closer look at the strength and weaknesses of an EA which we will do in the remainder of this paragraph. The matter of formulating an effective stop criterion is deliberately left out of this thesis, although it is necessary when implementing, it is not essential for developing the methodology.

Strength and weakness of an evolutionary algorithm
The difficulty in formulating a stop criterion is closely related to the weakness of an EA; *quickly* finding the exact optimal value within a region. No one disputes the fact that an EA is best when searching for a global optimum due to the stochastic nature of the search process. And as we will show, the EA suggested in this thesis does a reasonable good job at finding such a global optimum.

But for an EA to find the *exact optimal value* within the region it has evolved itself into (and most likely will contain the global optimum) is a time consuming *process*. It is the *stochastic nature*, the driving force behind the likeliness to find a *global optimum*, which causes the *slow convergence rate* in finding the *actual optimal value* in the region it has evolved itself into. When the *average gene data* of the top ranking chromosomes in the POOL is already in the region of the suspected global optimum, there is no more need for a high rate of mutation, which might extrapolate the gene data into other regions. At this point efforts should be directed at effectively *combining* or *averaging* the existing gene data of these top-ranking chromosomes into an optimal value. Implementing *sliding* EA settings (settings which change as the number of generations increase e.g. a diminishing chance on mutation or higher chance on recombination) could prove beneficial but has not been investigated in this thesis.

Beside the *sliding EA settings* it might prove *fruitful* to investigate a combination of heuristics, where an EA is implemented to search for the *global region* and another (heuristic) search method is used to find the optimal value within this region. This is mainly because another type of heuristic might be better equipped in dealing with the *time continuity of the generated advices*. Since our gene data was mapped in three dimensions, of which one is the time dimension, it remains a chance process in selecting the positions of the genes in this dimension, the DRIP settings in time. By introducing the superperiod we already tried to bring some continuity in the DRIP advices. In advance of the description of the case study in the next chapter, we already present one of the conclusions, which is that the generated DRIP settings are somewhat irregular in time. The deviations are relative small and if the EA runs long enough will disappear, but waiting for this to happen would take a considerable amount of computation time.

The idea of incorporating a *new heuristic*, or perhaps better: *an additional operator* beside the existing crossover, mutation or recombination, which would deal with the time dimension in a *non stochastic* way might prove very beneficial. By approaching the *time dimension* of the split fractions in a more *continuous* way, this operator or heuristic would be better equipped in optimizing these settings a lot quicker. (the *polyfit* mutation operator discussed in 7.4.4.4 is a first attempt)

8.2.2 Default settings for the DSMART Model and the Evolutionary Algorithm

In this paragraph we will discuss the *standard* settings used in the DSMART Model and the evolutionary algorithm. Due to the large number of variables an in depth investigation (ceteris paribus) has not been performed so these settings could very well be non optimal. The *do it yourself* (DIY) approach did increase the insight into EA's significantly and the proposed values are therefore best characterized as *educated guesses* (in addition *Google* proved very useful in finding *rules of thumb* or useful opinions).

We start by listing the DSMART Model parameters in the table below:

Table 8-1 List of the default DSMART Model settings

Name	Description	Symbol	Unit	Value
Time step	The smallest unit of time in which the simulated traffic is considered.	dt	[hr]	$\frac{30}{3600}$
Duration	Duration of the simulation expressed by a start and stop time	T	[hr]	3
Periods	The number of periods used to divide the simulation duration. Standard every quarter is a new period	P	[-]	12

Time slice	A vector consisting of multiplication values used to scale the static OD into a dynamic one	τ_p	[-]	1
Path alpha	The *fraction* of the *total link travel time* (TLTT) which is used as the standard deviation in a normal distribution positioned around the TLTT used in the probit path selection	P^{α}	[-]	0.1
Path trials	The number of times a deterministic route choice set, based on user perceived utility by means of a distorted network representation is calculated at the beginning of a period.	P^{t}	[-]	5
Path type	Paths can be based on link *travel time* or *distance*	P^{type}	[-]	Time
Path gamma	The weight with which the route choice set from the previous period is added to the current period	P^{χ}	[-]	1
Resolution	The minimal resolution of the scalar values in the destination specific split vectors	r	[-]	0.1
Min q	Threshold value for minimal demand. All demand below this value is removed from simulation	$q^{\min}$	[veh/hr]	1
Min k	Threshold value for minimal density. All density below this value is removed from simulation	$k^{\min}$	[veh/km]	0.1
Max. Cardinality	Maximum cardinality of a node expressed in the number of connected links to which the node functions as an input, and therefore the maximum length of a split vector.	$C^{\max}$	[-]	10

The listed settings, which are used in the model to create an initial chromosome, are saved together with the initialized network structure and as a result of this, the client's share exactly the same settings as the DSMART Model did.

In the next table we list the default settings for the *evolutionary algorithm*

Table 8-2 List of the default evolutionary algorithm settings

Name	Description	Symbol	Unit	Value
User compliance	A weight factor indicating the percentage of travelers that will follow the advice	$\eta^{compliance}$	[-]	1
Contestant size	Factor used for calculating the number of contestants who enter the tournament selection based on the current size of the POOL.	$\upsilon^{contes\tan ts}$	[-]	0.6
Generation size	The size of a new generation in number of chromosomes	Φ^{size}	[-]	8
DRIP select interval	Interval from which a factor is randomly drawn which is used for calculating the number of DRIPS	$[\upsilon_{\min}^{DRIP}..\upsilon_{\max}^{DRIP}]$	[-]	[0.2..1]
Crosscombine chance	Chance on crosscombine	P^{CC}	[-]	0.9
Mutation chance	Chance on mutation	P^{MUT}	[-]	0.4
Poly fit chance	Chance on polyfitting operator	P^{POLY}	[-]	0.025
Crosscombine operator chance	Vector describing the probability of the crossover or recombination operator to be chosen	λ^{CC}	[-]	[5 3]
Mutation operator chance	Vector describing the probability of the mutation operators: *non BR, newvector, scale, hustle, spare Q* to be chosen	λ^{MUT}	[-]	[2 3 2 1 2]
Super period select interval	Interval used for calculating the number of super periods based on the selected fraction and the number of periods	$[\upsilon_{\min}^{SP}..\upsilon_{\max}^{SP}]$	[-]	[0,5..1]
Maximum new vector values	Fraction used for calculating the maximum number of new elements in a split vector	$\upsilon_{\max}^{NV}$	[-]	[1]
Maximum scale value	Fraction used to calculate the maximum scale size	$\upsilon_{\max}^{SCALE}$	[-]	0.7
Addition operator chance	Vector describing the probability of the addition methods: *absolute, incremental, average*	λ^{ADD}	[-]	[2 1 1]
Incremental add step interval	Interval describing the possible number of steps used in the incremental addition	$[\Omega_{\min}^{ADD}..\Omega_{\max}^{ADD}]$	[-]	[1..3]
Incremental add maximum step size	Factor used for calculating the maximum step size	$\upsilon_{\max}^{Stepfactor}$	[-]	0.25
Poly fit order	Fixed order of the polynomial to use	Ω_{order}^{Poly}	[-]	2

Resolution	The level of detail allowed in the elements of a split vector. (the same as *r* in the DSMART Model)	$\Phi^{resolution}$	[-]	0.1
Maturity	The age at which the chromosomes enter the survival of the fittest contest	$\Theta^{maturity}$	[-]	3
Twins	The number of identical twins which may exist in the POOL at any given time. (a.k.a. *maxcopies*)	Θ^{Twins}	[-]	3
Maximum POOL size	The maximum allowed number of mature chromosomes in the POOL	$\Theta^{\max .size}$	[-]	32

8.3 Technical implementation

This paragraph discusses the computer implementation of the optimization process in Matlab by describing the implementation of the various components, which have been presented in the second paragraph of this chapter.

In order to keep our focus on the methodology we will not deviate into detailed descriptions of GUI's, error handling, code listings, debugging, programming approach etc but briefly address the most important issues related to the implementation. At the end of this paragraph we will discuss a *step plan* which should be followed when setting up a network for DRIP optimization using the developed software.

8.3.1 Matlab Implementation

The programming of the *DSMART Model*, *clients* and *server* has been realized in Matlab 14; and this language has been chosen for a number of reasons. The scripting language of Matlab is easy to use and read and widely supported throughout the world. The language has a great deal of included mathematical functions, which makes programming easier. The graphical functions within the package are very suitable for making plots and graphics. And perhaps the most convenient feature is the save and load possibility for various variables. This made the saving and importing of chromosomes to/from the file server and setting variables for the EA a simple task by simply loading a set of saved variables.

8.3.1.1 DSMART Model

In the picture we display the GUI belonging to the DSMART Model. The code of the model adds up to 150kb of text distributed over 50 files.

When starting Matlab, the command *main* would pop up the following GUI which enables the user to load a network and make an assignment / create a initial chromosome. The input requires four text files describing the nodes, links, vertices and OD matrix. With this the network is initialized and set up for assignment. Only a few variables can be set by GUI, but most must be edited in a special script.

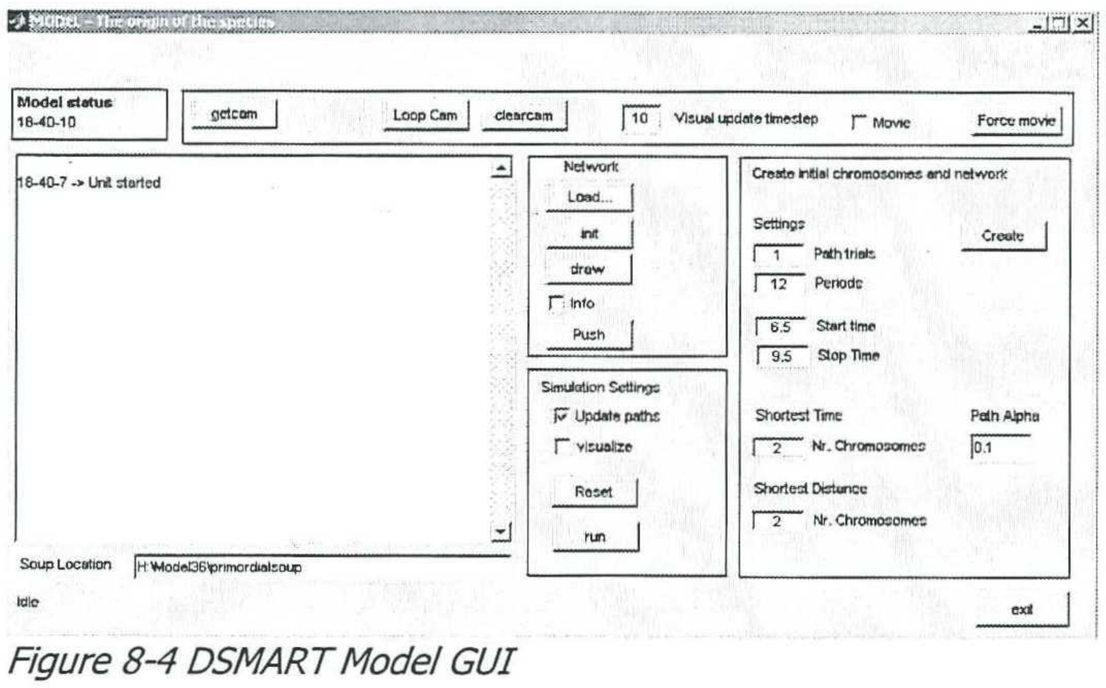


Figure 8-4 DSMART Model GUI

In the next chapter we will discuss the calibration process, which has been performed in order to create a reasonable representation of the Rotterdam network. Since the model only has a limited set of variables mostly aimed at the route choice process, the actual calibration is realized by adjusting the network properties and the OD table.

The input data is *generic* in the sense that *any network*, as long as its coded correctly in the text files, can be used. The size of the network does heavily influence the usability. When developing the DSMART Model it has never been used on networks exceeding 500 nodes.

All GUI's developed have the same basic build. The large white square is used to display information regarding the processes it has performed and the status bar below displays the process its currently working on. The smaller white rectangles are used for user input together and together with the few buttons they are used to control the process.

8.3.1.2 Server

The main GUI from the server is shown to the right and has the same characteristic as described above. The server comprises 130kb spread over 38 files. All server settings can be edited, saved or loaded during execution making it possible to implement *sliding* settings as suggested earlier by means of the GUI below. The editing is realized by a third party piece of Matlab code called *CRC-Structbrowser. (thanks for sharing!)*

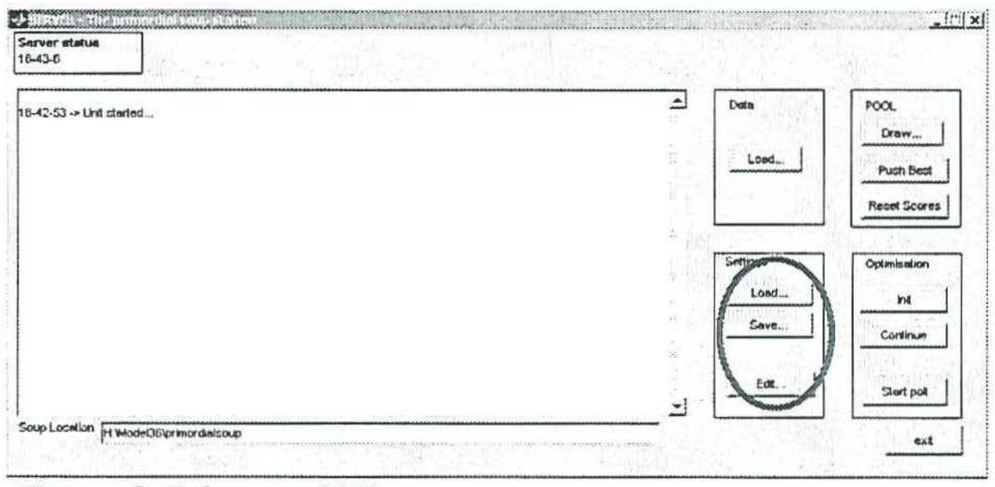


Figure 8-5 Server GUI

Settings can be edited and from that point forward used in the generation process. The *save, load* and *edit* features are located in the red circled area and can be accessed at any time.

We will discuss later on how we use this feature to load *extreme* EA settings to severely mutate, in fact, it would be more appropriate to call it mutilate, the first generation which is created. By doing so we aim at diversifying the gene data to limit the chances on premature convergence on a local optimum.

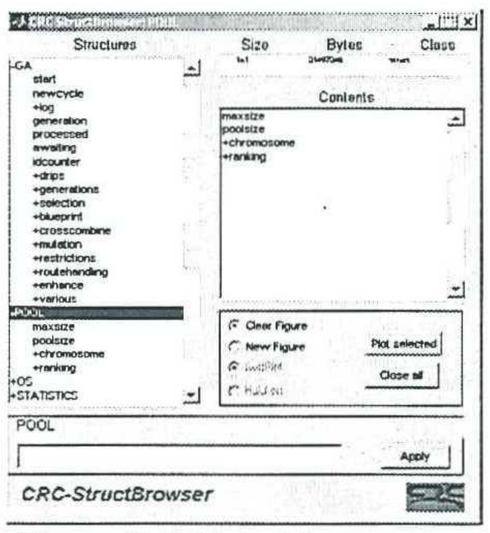


Figure 8-6 Struct browser

8.3.1.3 Client

The total code of the clients adds up to 80Kb spread over 28 files and is in fact a stripped clone of the DSMART Model. It has *no initializing capabilities* of the network based on text data, since it loads an initialized network structure. In addition there is *no route choice module* present, since the chromosomes themselves provide this information.

The client can however evaluate, visualize and animate the assignment it is currently performing.

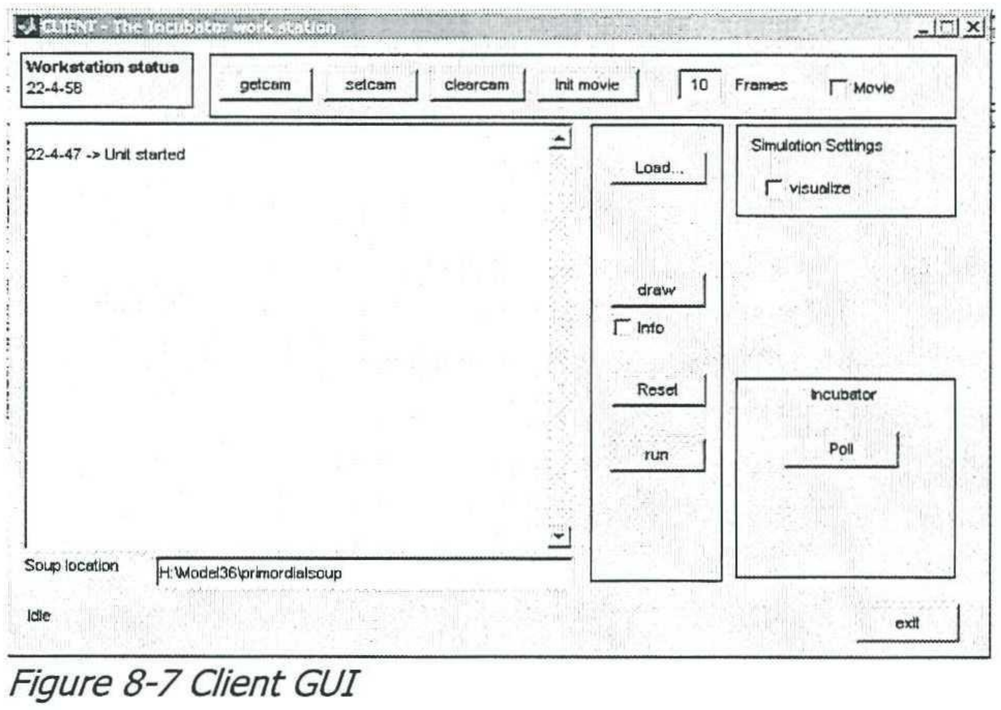


Figure 8-7 Client GUI

8.3.2 The necessary steps for generating route guidance

In this paragraph we will list the consecutive steps needed to optimize the DRIPS within a network using the Matlab implementation of the developed optimization process.

Pre process

i. Make a description of the network and OD demand in text files
ii. Make a calibration of the network representation and OD matrix by *trial assignments* in the DSMART Model until it sufficiently resembles the required network state.

Creating initial chromosomes

i. The script containing the different variables (as described earlier) used by the DSMART Model is edited where necessary.
ii. Using the *calibrated network data* and the *DSMART Model settings* the network is *initialized* and saved to the file server.
iii. The DSMART Model is run and makes an assignment using the calibrated network. The result in terms of route choices, score, and strategy variables (link activity table) is saved as the initial chromosome.
In this step we make an assignment of the calibrated network and OD matrix using the route choice module to calculate the actual traveled routes. During this phase the dynamic routes used, are actually calculated and stored in the chromosome by means of the SPF matrix and SPLIT table.

Setting up the clients

i. By default eight different computers are used who all run a separate instance of the client
ii. The *initialized network data* created by the DSMART Model is loaded into the client
iii. The clients are placed in *poll mode*
The poll mode is a state where each client scans the file server for chromosome data. When found, the client uses this as input for the assignment and calculates the corresponding score. When finished with the assignment, the score is saved on the file server and the client is reset and starts scanning the file server again for new chromosome data.

Setting up the evolutionary algorithm

i. The *initial chromosome* created in the previous step is loaded into the pool where they have their score set to 0, and the basic settings of the EA are initialized.
ii. A set of parameters is loaded which will result in a severe mutilation of the gene data.
iii. With these settings we force the creation of a new generation
iv. The optimization process is started and the generation from the previous step is pushed to the file server.
v. A set of *normal settings* are loaded into the EA and will be used during the creation of the next generation

Starting the optimization process

i. The server is placed in *poll* mode
The poll mode is the state where the EA checks the file server for evaluated chromosome data. When this is found, the server imports this data until the whole generation is evaluated. The chromosome data is added to the POOL, which in term is updated and a new generation is created. This generation is pushed onto the file server and the EA starts scanning the file server for evaluated chromosome data.

Stopping the process

i. Based on user interference, we simply exit the poll mode of the client and server and exit the software.
ii. We push the best performing chromosome to the server where it can be evaluated
iii. The workspace of the EA server and POOL evolution is saved

Post processing

i. The *best chromosome data* is compared to the *initial chromosome data* by making an *animated* assignment of both. This way, the resulting video can be interpreted
ii. The data of both chromosomes is compared in the *testlab* (again a piece of software in Matlab), which results in the DRIP plots illustrating the old and new periodically generated DRIP settings

8.4 Discussion of the verification results

The verification of both the DSMART Model and the optimization process is discussed in appendix 1 in detail. Some results:

Though the generated route guidance advices may not always seem logical in the sense of continuity and simplicity, they do always have the desired effect in terms of optimizing the objective function and creating an assignment without congestion. The algorithm is able to generate the correct advices but needs quite a few generations to smooth them into the optimal values. (this can de deducted from the typical hyperbolic shaped evolution). Although some non optimal advices are generated as well, the overall assignment evaluation by means of the generalized gross travel time improves considerably!

To illustrate this effect, we present two DRIP diagrams taken from the second test network in the verification of the optimization process from appendix I §1.2. We can see that 7 of the twelve generated advices are continuous in time, but some deviations exist.

The picture below illustrates the destination specific split fractions which have been generated for DRIP numbers 3&7 for travelers heading toward destination 8. The simulation period was divided into twelve periods which must be read from top left to bottom right. The new split vector values are indicated by the green values together with their scalar values (which always sum to 1). The black arrows indicate the old split vector values used in the calibrated situation and are horizontally mirrored.

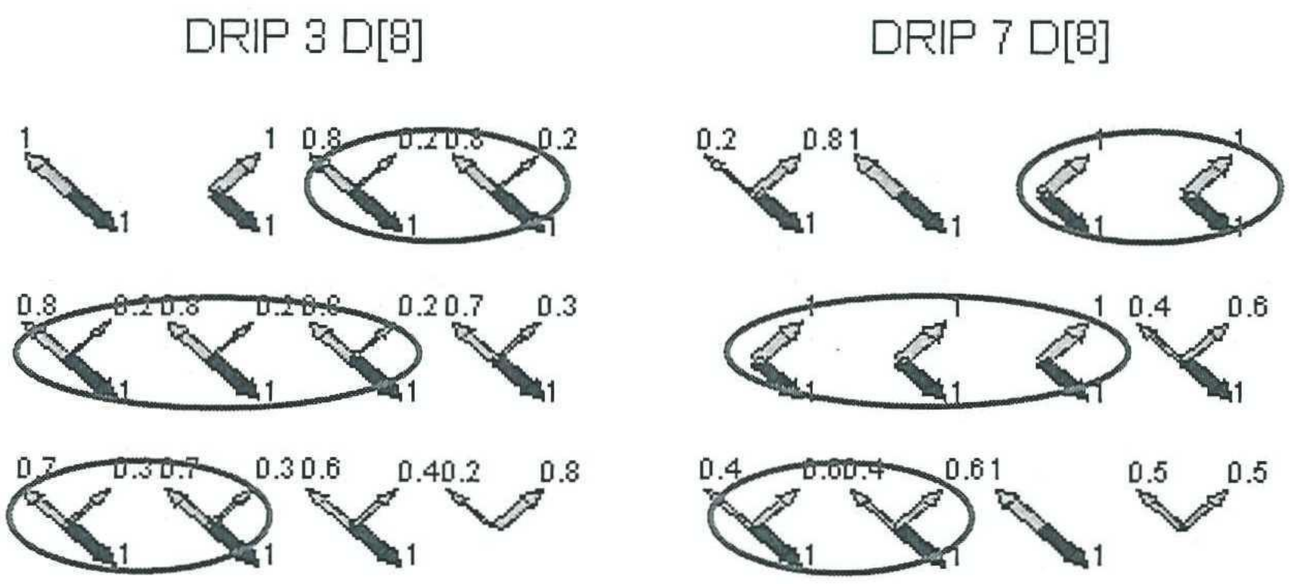


Figure 8-8 Two example DRIP diagrams

These diagrams have been developed in order to *visualize* the generated DRIP settings. The actual settings are displayed in green, whereas the *old* route choice is displayed in black. Now without getting into detail of the exact nature of the network it can be assumed that both DRIPS have to cooperate. The settings are displayed in time from upper left period 1, to lower right period 12.

The first two settings do not have to be taken into account (the DRIPS are located downstream of the origin and only start having effect during the second period) The circled settings in red are in equilibrium, and optimal, the non circled are *very near optimal.* The point of this example is to show that although the settings are in equilibrium they can very well behave *irregular in time*. The type of

assignment for this network did not require the DRIP settings to change over time, so all settings could have looked like the first 5 settings circled in red. But as we discussed earlier the result of the stochastic nature of the process is non continuous DRIP settings in time, despite the fact they are still optimal!

9 Case study Rotterdam

This chapter describes the application of route guidance generation methodology to our test case; the Rotterdam network. The case study is chosen to illustrate the application of the method on a real life network which has been calibrated to crudely resemble real network traffic conditions during the morning peak hour. It is explicitly stated that the calibration is crude since it was done in a quick and dirty way in an attempt to mimic real life observed queue behavior (the location and length of the queues). A crude calibration suffices to *illustrate* the effectiveness of the proposed route guidance generation methodology.

This chapter starts by summarizing the (calibrated) network and the used settings for both the DSMART Model and the evolutionary algorithm in the cases where they differ from the standard settings discussed in 8.2.2. We also provide an overview of the *custom routes*, described in 6.5, we have added to the route choice process in order to correctly model the effect of DRIPS.

When familiar with the network and its routing, the three different scenarios which will be optimized by adjusting the DRIP settings are discussed. These scenarios are chosen to resemble the *everyday recurrent congestion, a non recurring extreme event and an accident.* For these scenarios we will discuss the results in terms of the *evolution* of the solutions and the resulting *optimized assignment* by introducing the *assignment plot* and a table with some assignment characteristics. The actual DRIP settings for the various scenarios are listed in the Appendix.

At the end of this chapter we will discuss the results of the case study using the proposed methodology which in turn will bring us to the conclusions in the next and final chapter.

9.1 The Rotterdam motorway network

This paragraph discusses the Rotterdam network and the way it has been modeled.

The picture to the right shows a schematic map of the motorway network within the Netherlands together with the two largest cities: Amsterdam and Rotterdam.

This case study is centered on the ring Rotterdam which is indicated by the black rectangle.

The city map is provided in the picture below indicating the motorway network in red and the urban network in yellow. The port of Rotterdam is the biggest container terminal in Europe and plays an important role in the city. The river (maas) divides the city in a north and south bank, the motorway crosses the river on the east side by means of the *Brienoord bridge* and on the west side by means of the *Benelux tunnel*.

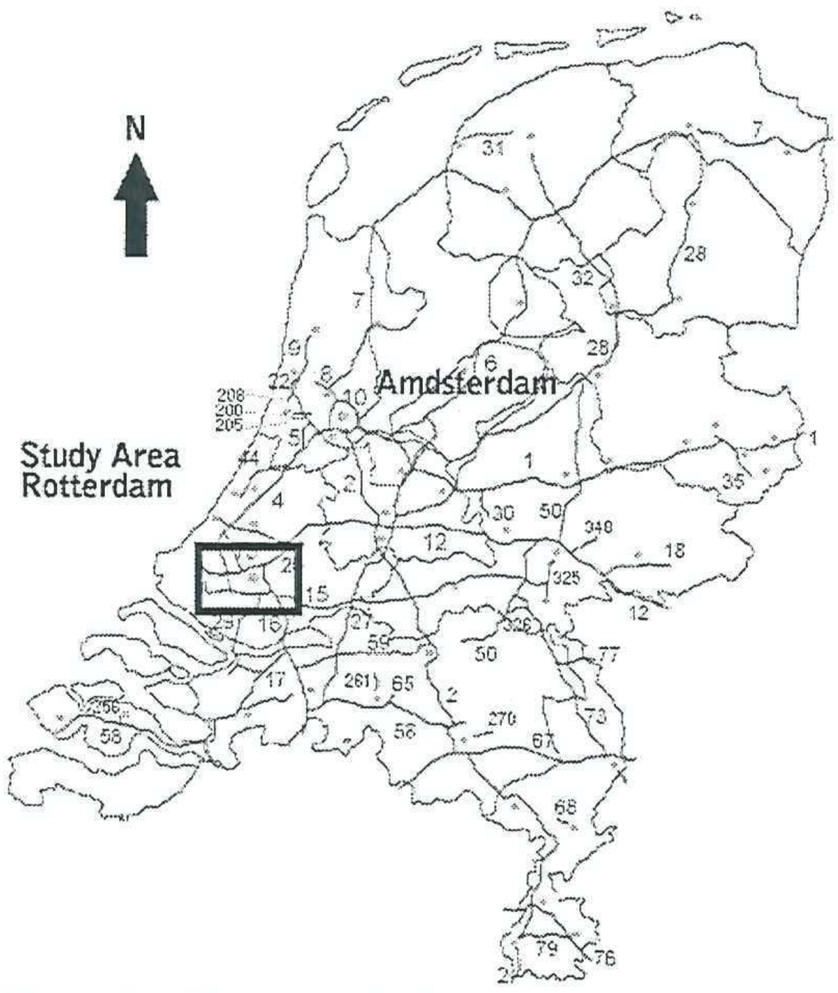


Figure 9-1 The Netherlands

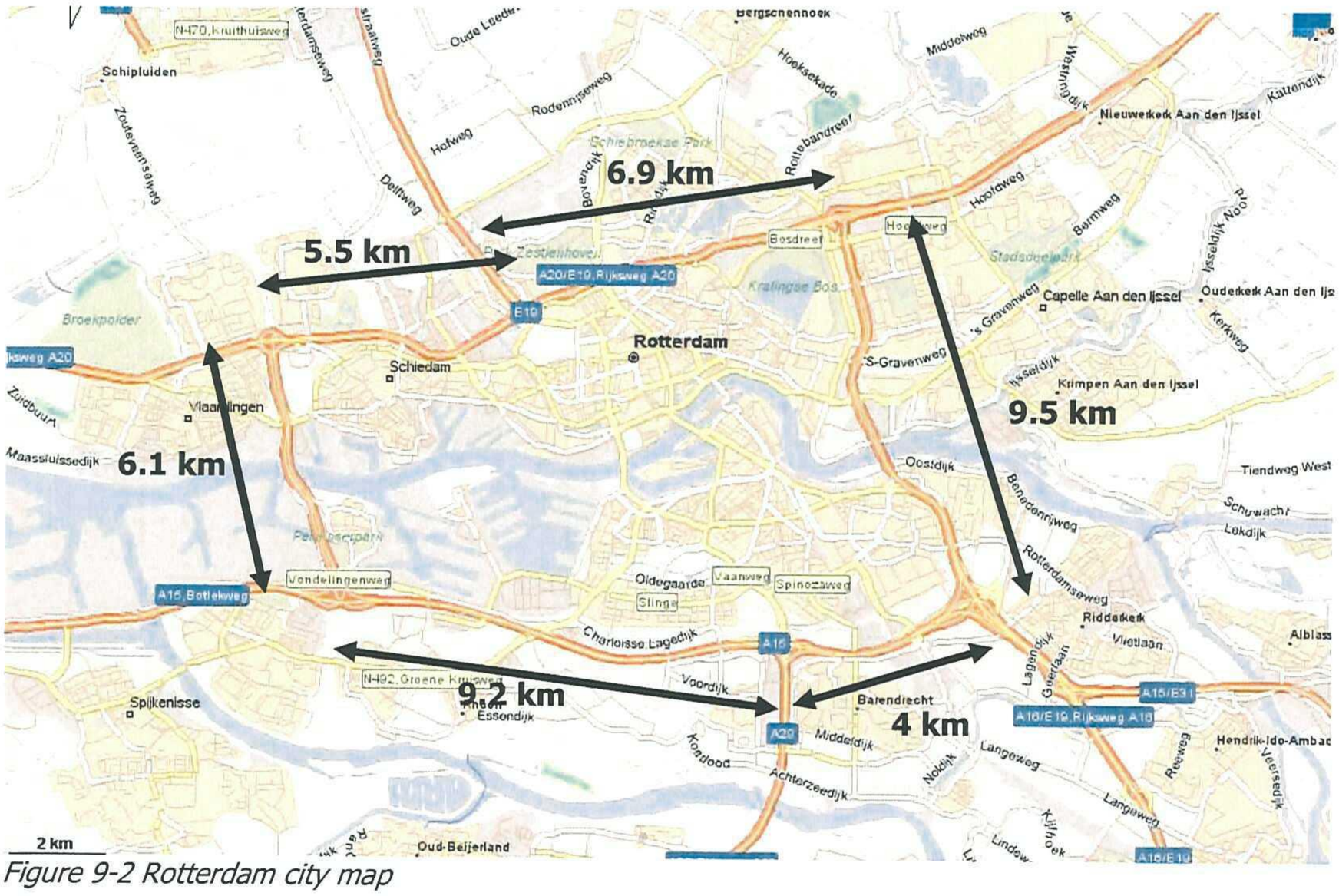


Figure 9-2 Rotterdam city map

The picture on the next page shows a schematic representation of the network annotated with the *main traveling directions* (at the end of the intersected highways), the motorway names (A numbers), the location of the harbor, the major infrastructure connections (*maastunnel* and *Erasmusbrug*), the football dome *the kuip* and the six dominant intersections.

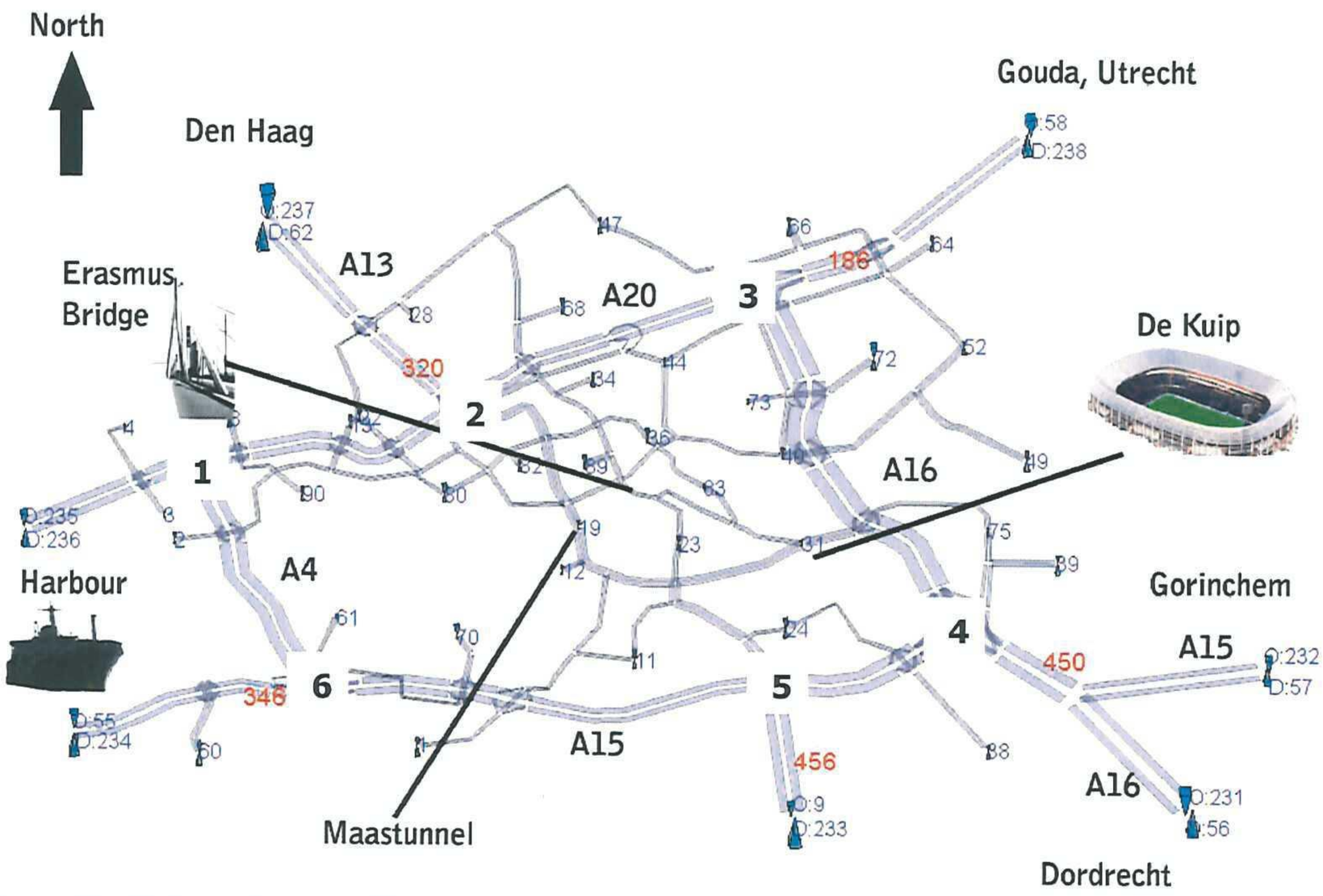


Figure 9-3 Schematization of Rotterdam Ring way and key infrastructures

The six major intersections are illustrated below and correspond to the numbers in figure 9-3.

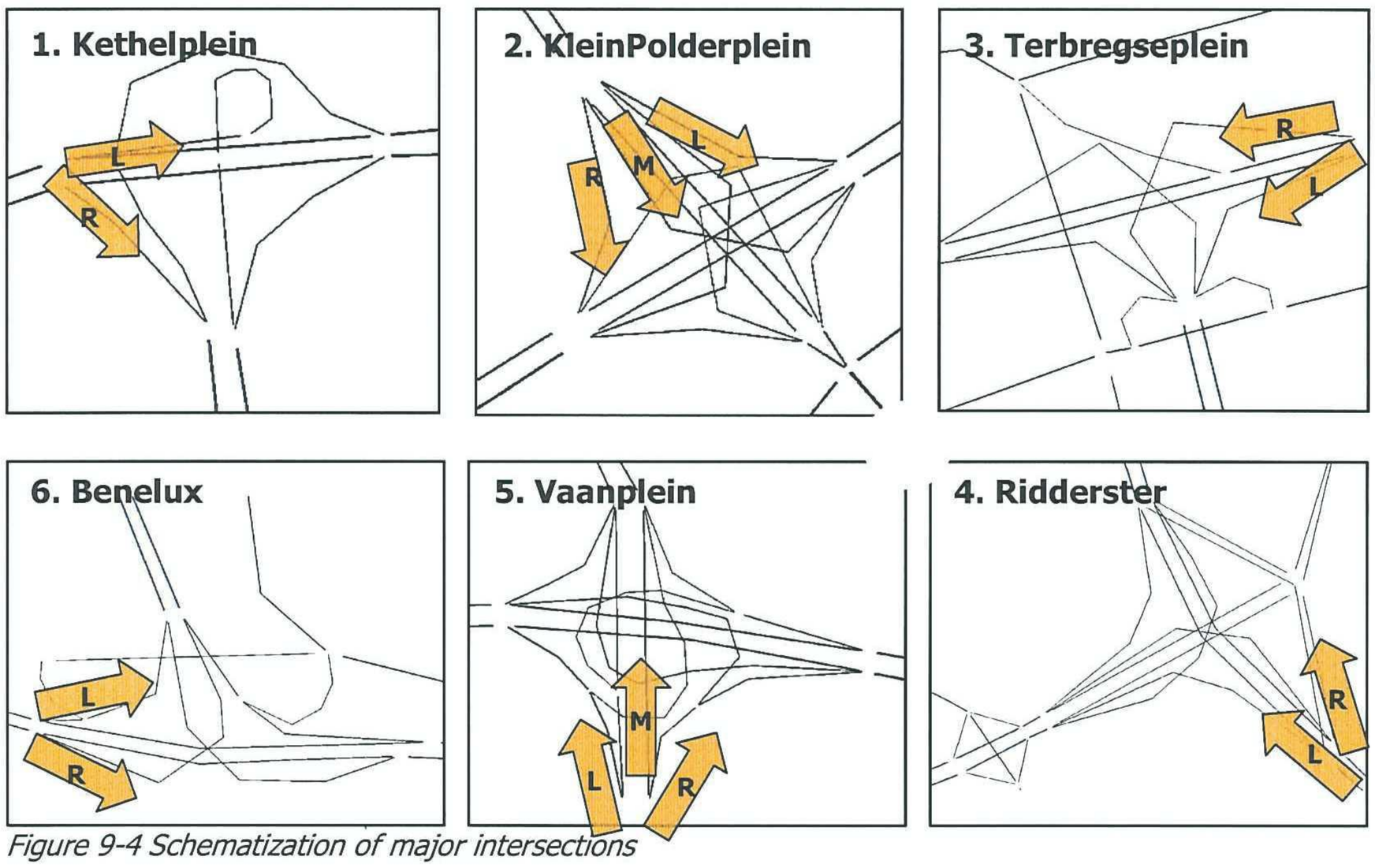


Figure 9-4 Schematization of major intersections

The orange arrows indicate the possibilities for rerouting traffic.

9.1.1 Schematized network

The network used for this study is an adaptation of the *Zuidvleugel network* (the south of Netherlands) which was made available by the courtesy of DHV.

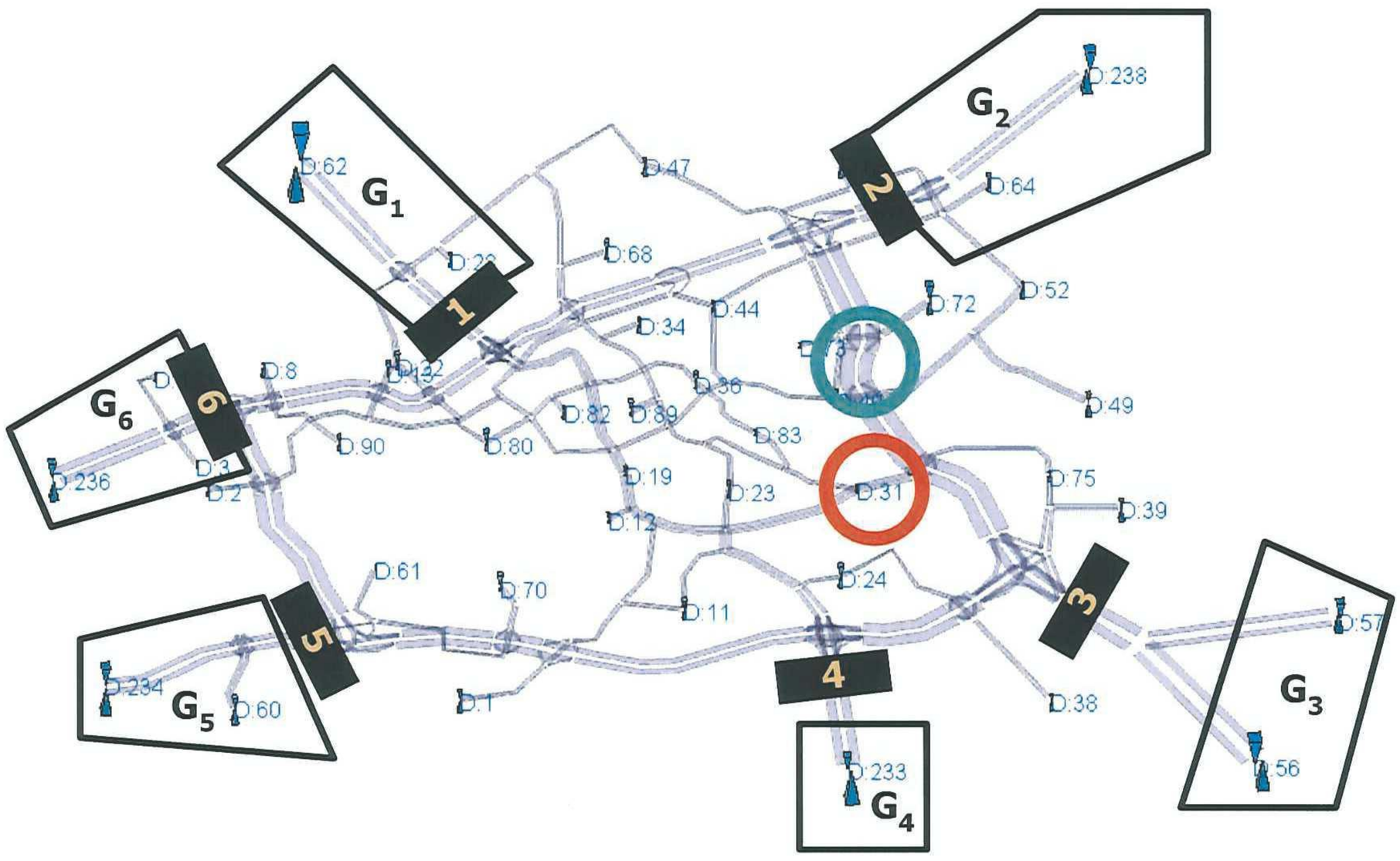


Figure 9-5 The Rotterdam network used in DSMART

The original network and OD matrix where calibrated for the morning peak situation in 2003 and used in the static traffic assignment model QUESTOR.

In the picture we can see the *destination nodes* with their numbers in blue, we also show the *destination groups* with their *node members encaptured* in the black polygons and indicated by *G#*. and we also show the locations of the DRIPS by means of the black squared numbers.

In addition two areas of interest have been circled; the red circle represents *the football dome "de kuip"* and is used as the destination in the *extreme demand* scenario. The Green circled area indicates the location of *an accident* which is used in the last scenario.

The network model as presented is explicitly chosen to include the effect of the *urban road network* connected to the on- and off-ramps and their effect on the queue development. Although this effect would be far greater if traffic regulation was also simulated, which is not the case, by merely allowing it to have effect we demonstrate the possibilities. (Future possibilities might be simultaneously setting *optimal traffic regulation settings* on the urban network or generate route guidance for *urban DRIPS*).

It also enables us to investigate other scenarios where other *extreme events are scheduled somewhere within the city* and perhaps most important it also allows us to use *urban city roads* as an alternative for routing purposes through the city.

9.1.2 DRIPS, destination groups and alternative routes

DRIPS
In picture 9-4 the intersections are illustrated together with the possibilities they provide for alternative routing by means of the orange arrows. This way a traveler can be sent around the network in a clockwise (CW) or counter clockwise (CCW) manner. The DRIPS themselves are located upstream of the orange arrows in such a way that their split vector connects to the links indicated by the orange arrows and thus enables travelers passing these DRIPS to be sent in the desired direction.

For the DRIPs located near *Kleinpolderplein* and *Vaanplein*, there exists a third alternative which would allow people to travel through the city.

Destination groups
Remember that in paragraph 7.3.1 we have already illustrated the position of the DRIPS and the layout of the different *groups of destination nodes* for which a DRIP can provide information. Generally speaking, these destinations are located near or downstream of the intersections as discussed above. To determine for which group a *DRIP setting* is generated in the evolutionary algorithm we introduced the chance vector λ_z which describes the probability of choosing such a group during the *crossover* and *mutation* phase. These chances are based on the difference between the length of the alternative routes and displayed in the graph below.

Custom routes
In the figure to the right we have displayed a schematic version of the Rotterdam network, where in grey the *actual roads* used for the rerouting are drawn.
We have connected the different DRIPS with the destination groups and indicated the likeliness of these destination groups by the line type and thickness. A thick black solid line has twice the probability compared to the thin dotted line. This likeliness is also used in the chance vector.

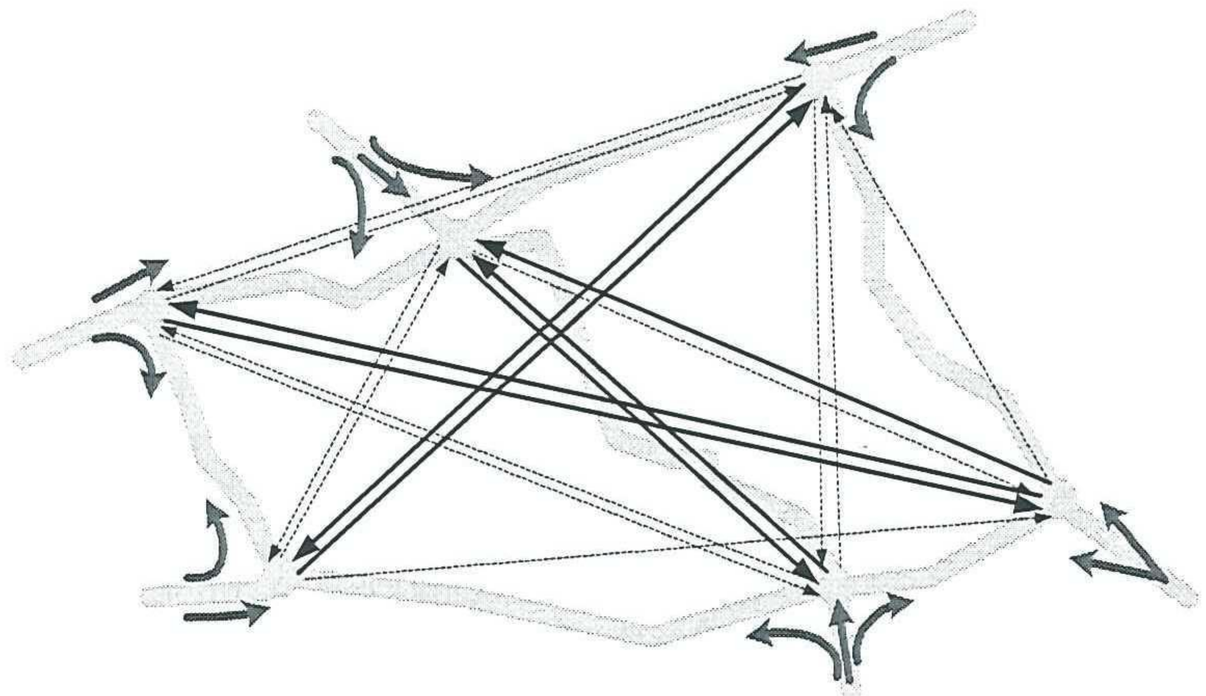


Figure 9-6 Relation between DRIP and likeliness of destination

Some alternative destinations might look very unlikely at first glance since they would require people to travel more than twice the distance. The main reason why we do include these groups is because we're talking about *travel times* and not *travel distance*. Extreme network conditions such as large accidents fully blocking the motorway could result in conditions where traveling the ring in another direction could still be better.

Off course unnecessary re-routing which would lead to longer travel times does increase the summed generalized gross travel times and therefore reduces the objective function. However as we have explained in 5.3.1 we apply classic utility optimization to keep it simple and can therefore not guarantee there will be no exploitation (a *constrained pareto optimization* should be applied). This could result in behavior where certain group(s) might be re-routed and forced to travel a longer route for the benefit of the travel time of other destinations.

9.1.3 Network statistics

In the picture below we have displayed the motorway, on- and off-ramps and links used for the intersections to the left, together with the motorway names and the urban road network to the right.

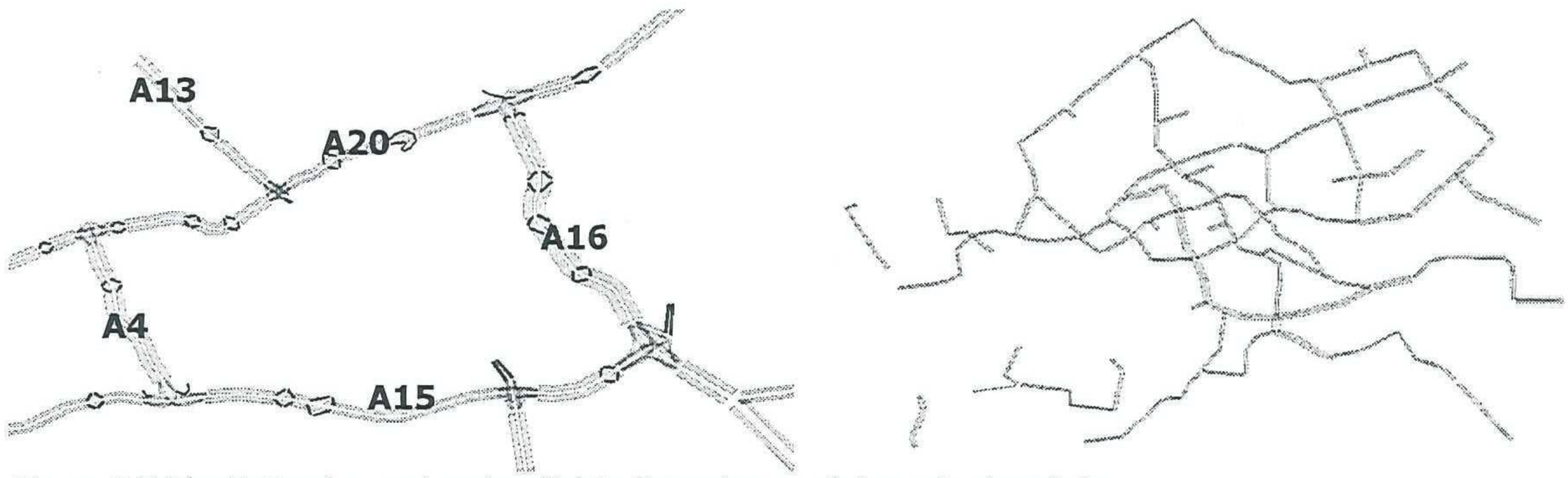


Figure 9-7 The Rotterdam network split into its motorway links and urban links

As we can see Rotterdam has a typical ring way structure in terms of the *motorway links* . The network consists of 234 nodes, from which 47 act as origin and destination and are represented by the upward and *downward facing pyramids* in figure 9-5. The size of the summed origin or destination is represented by the size of the pyramid.

In the table below we have summarized the statistics of the 468 links and one of the most important conclusions is that *most of the cells are used for the urban network*. If this inner network could be limited further it would increase the calculation speed considerably! The statistics are derived from

the DSMART model which initializes the input data (text files describing nodes and links) into cells. The capacity is the total summed link capacity.

Table 9-1 Rotterdam network link statistics

	Motorway links			Urban links		
	Sum	Mean	Stdev	Sum	Mean	Stdev
Nr links, [-]	229			239		
Nr lanes, [-]		2.67	1.41		1.35	0.48
Nr cells. [-]	345.00	1.51	0.90	1025	4.3	2.9
Capacity, [veh/hr]	1309375	5717.79	3280.77	638000	2669.46	779.37
Speed, [km/hr]		88.95	21.84		32.85	8.52
Length, [km]	260.83	1.14	0.81	269.58	1.13	0.72

The low average maximum speed on the motorway network can be explained by the speed on the on- and off ramps which is mostly 50 and the speed restrictions near *kleinpolderplein.*

9.2 Calibrated network conditions

In order to create a calibrated situation, we used a *global* image[1] of the reoccurring morning queues on the network and tried to simulate the *position* and *length* of these queues together with the *buildup* and *dissemination* behavior. This means that less attention was paid to the actual *intensity, speed* and *time of day* of the queues for a number of reasons which we will address. But before we do, we have to keep in mind that the main objective is to develop a *methodology* and prove it will work for a life sized network. The most important aspect is its *ability* to generate optimal route guidance which we try to illustrate using our case study area Rotterdam and a number of different scenarios. We try to *mimic* the behavior of the queues in order to give some credibility to the case study itself to which the reader or familiars with this network can relate. This does not change the fact that by proving the methodology to work for this situation, it will also work for a qualitatively better calibrated situation.

Preload
To keep it simple, we avoid the common practice of setting a *pre-load* on the network which means we start the simulation with an empty network. Congestion effects are therefore bound to set in on a later time compared to reality.

9.2.1 Network, OD and DSMART Model Settings

The network and OD matrix we created, stems from a larger network for the whole of the province *South Holland* and has been provided by *DHV*. This network and the accompanying OD matrix, has been calibrated for the morning peak using a static model (for an average recurrent morning peak workday congestion in 2003) . While translating this network a number of difficulties had to be overcome, such as:

- The detail of the DHV network model had to be scaled down in order to be calculated quickly enough in the DSMART Model by means of:
  - Simplifying the network in terms of number of nodes and links (factor 10)
  - Uniquely connecting the 600 zones to 47 nodes
  - Adjusting the OD matrix for a part of the intra zonal traffic which had now become *obsolete* because the zones are now connected to the same nodes
  - Reducing the inter urban traffic demand for zones which became connected to *nearby nodes* and where now forced to travel via the simplified urban network.
  - Upsizing this urban network capacity to deal with the large demand, which now has few or no alternatives to travel to the onramps.

[1] The recurrent morning work day congestion for 2003 is described in Appendix III

- A dynamic time pattern had to be estimated for the OD matrix
- The original 2 user class OD matrix was summed with a truck PCE value of 1.7
- Due to the absence of traffic regulation and the inability for the DSMART Model to simulate instability effects of traffic flowing at near capacity, the on- and off-ramps had to be adjusted in terms of speed and capacity to reproduce congestion effects

Calibration variables

To calibrate the model we where limited to the OD and the network. The most important specific variables used to reproduce the *general morning peak congestion image* where:

- The Speed and capacity on the on- and off ramps (70 [km/hr] and 2500 [veh/hr])
- Capacity of the *bottleneck* near Delft zuid.
  The three lane motorway A13 from Rotterdam to The Hague has an artificial bottleneck to reproduce congestion effects near Delft zuid. The original capacity of 6600 [veh/hr] is reduced locally to 5500 [veh/hr]
- OD demand for motorway nodes.
  The OD demands for the motorway origin and destination nodes have been increased to stress the congestion on the motorways a little.

OD matrix

The specific DSMART Model settings we used are equal to the standard settings, discussed in 8.2.2 except for the *time slice values* to dynamize the OD matrix. It became very clear from early on that a *dynamic OD was unavailable* for this network (which is mostly the case for any network for that matter). To still capture the dynamic nature of demand we already introduced the time slice value τ_p in 6.2.1. The values we used for this vector are depicted in the picture below.

The original calibrated static OD matrix from the DHV model we used was calibrated for an average *two hour* morning peak period between 7:00 and 9:00.

The values in this OD where given in [veh/hr] and considered representative for this two hour period.

In order to dynamize this, we used detector data on the motorway network near *Kethelplein* between 06:30 and 09:30 expressed in [veh/hr], to create the following pattern of relative intensity divided over 12 periods.

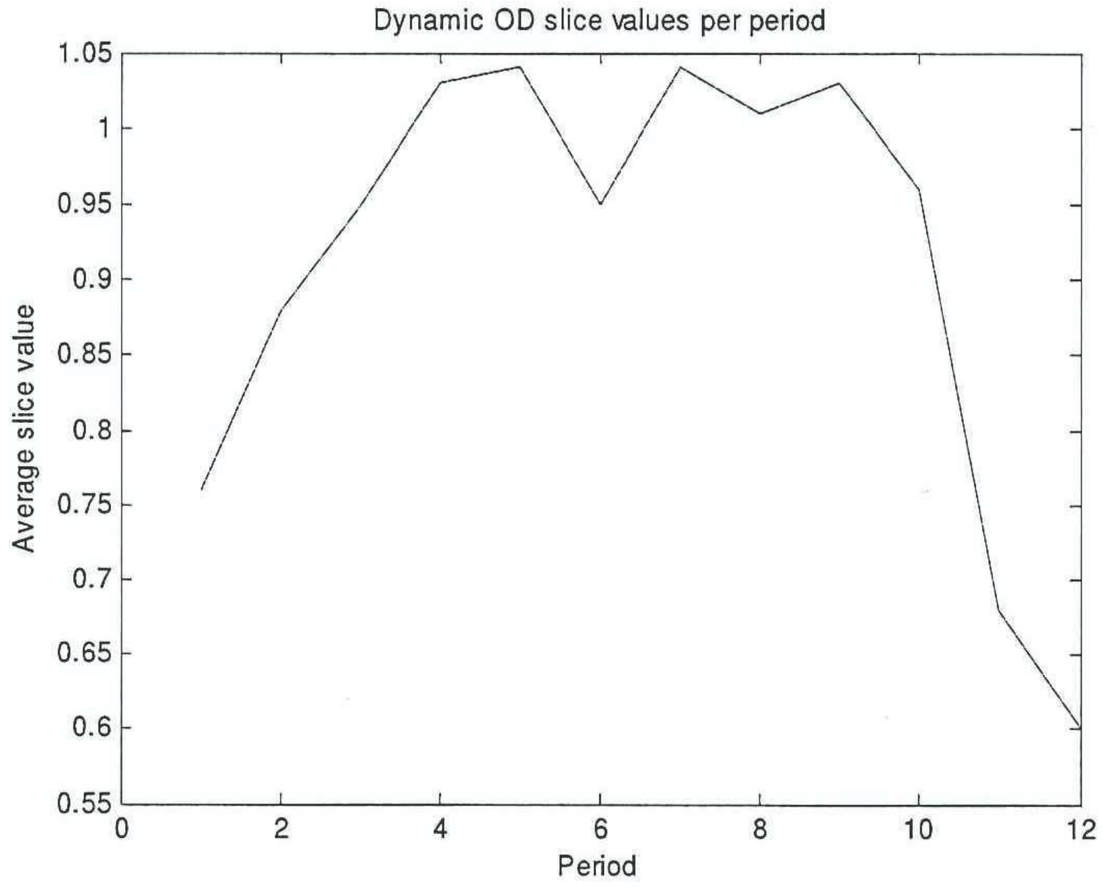


Figure 9-8 OD time slice pattern

Because no *pre load* is used, the OD matrix is assigned for a three hour period to the network (instead of a two hour period) with the OD time slice pattern shown in figure 9-9. The first 3 periods and the last period are used to gradually build up/down and therefore the simulation period is prolonged to 3 hours divided into twelve periods.

9.2.2 Assignment of the calibrated situation

The resulting calibrated assignment is difficult to describe with a single picture due to its dynamic nature. To give some idea about the development of the queues we provide four still images below which describe the position of the queues after 45, 90, 135 and 180 minutes starting with the top left picture. The original start time of the simulation is 06:30, yet the formation of queues is delayed in comparison to real life because the assignment starts with an empty network (no pre load).

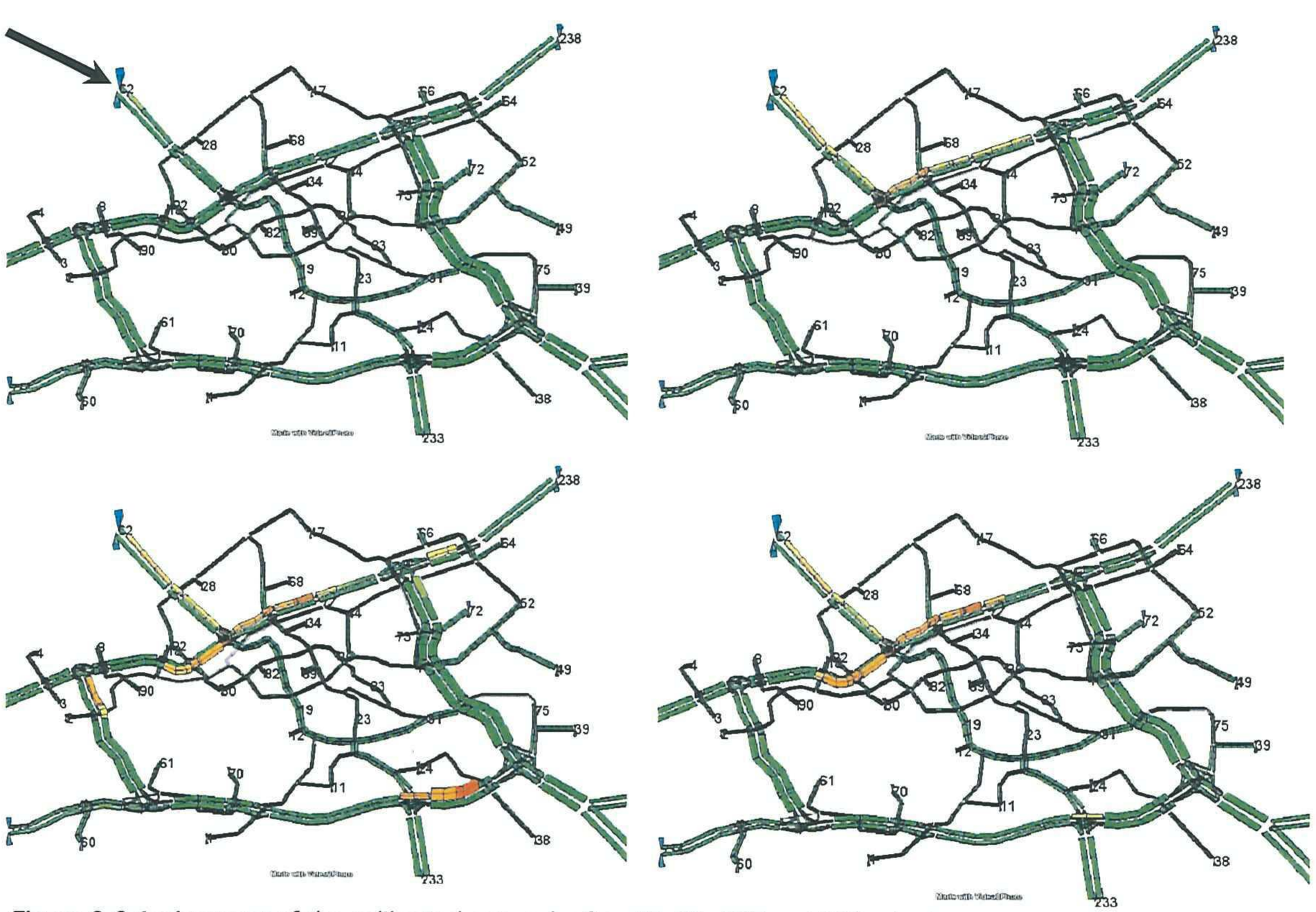


Figure 9-9 Assignment of the calibrated network after 45, 90, 135 and 180 minutes

The above *stills* and the *simulation time they reflect* are also used to describe the *calibrated* and *optimized* assignment in the other *scenarios* discussed in the following paragraphs which allow them to be visually compared by the reader.

The speed as a percentage of the maximum allowed speed is indicated in the picture below, ranging from green (100% free flow speed) to red (blocked, 0% free flow speed).

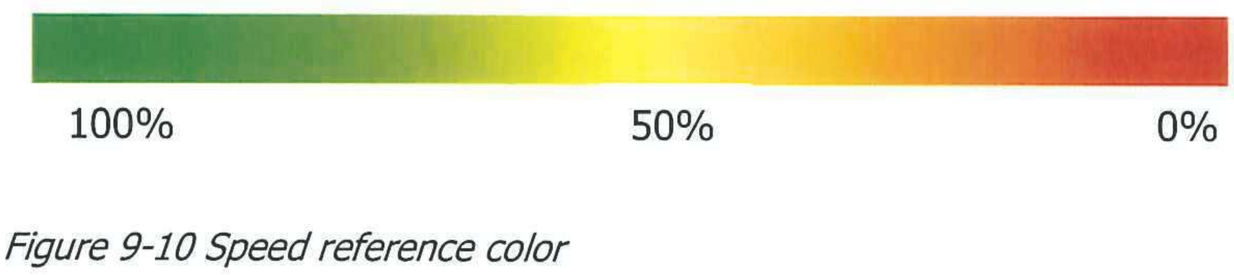


Figure 9-10 Speed reference color

In the following text we will explain *two measures* used for describing the assignment. The *assignment plot,* which graphically depicts some interesting qualities by means of absolute values, and the *assignment total* table, which lists various assignment characteristic summed variables.

When discussing the various scenarios we will *first* discuss the assignment resulting from the *route choices in the calibrated situation* applied to the different *circumstances* in the various scenarios. These results are presented using the *plot* and *total table* with their *absolute values*. When

presenting the *optimized situation* we use the same *plot* and *total table* but illustrate the difference between the optimized situation minus the calibrated situation.

9.2.2.1 Assignment plot

To make the different assignments comparable to each other, we introduce the following summary of the assignment depicted in the figure below. The plot details are discussed in the text below.

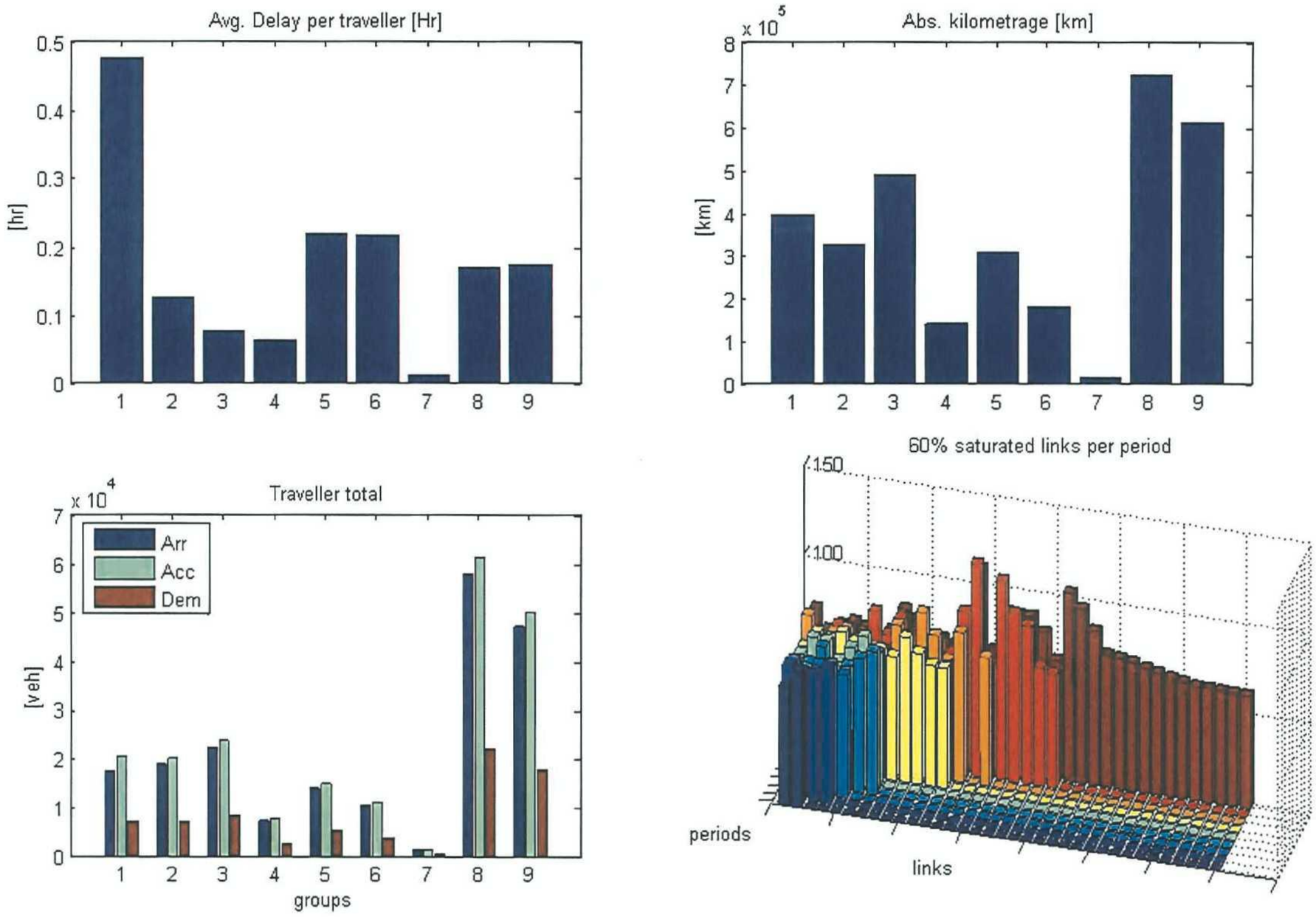


Figure 9-11 Absolute assignment plot for the calibrated situation

This diagram is called *an absolute assignment plot* and is used to illustrate the assignment results for a scenario. Both the top two figures and the bottom left figure work with an x-axis of *destination groups*, which we will first explain.

Description of the destination groups
A *destination group* is a collection of *destination nodes* and the *groups 1 to 6* are the same as the destination groups used for the DRIPS displayed in figure 9-5.
Destination group 7 consists of only one destination, namely "*de Kuip" (*football dome) which is used in *scenario 1: extreme event* (destination number 31, circled in red in the figure 9-5).
Destination group 8 is the collection of *all nodes located within the city*, or in this case inside of the ring way and destination group 9 is the collection of *all remaining nodes outside the ring way*. Note that all groups are *unique* and every destination node can only be a part of *one group*.

Average delay per traveler
The top left figure displays the average encountered delay per traveler per destination group in hours. The encountered delay is based on the links chosen and the difference between the actual encountered link travel time and the free flow travel time for that link.

Absolute number of kilometers traveled
The top right figure describes the absolute number of kilometers traveled by all cars heading toward a destination within one of the groups. It is the summed value of all travelers accessing a link multiplied by the link length and summed per destination.

Traveler totals
The bottom left figure describes some statistics regarding the total number of travelers. In blue the absolute number of *arrivals per destination group (ARR)*, in green the *total number of people who could access* (ACC) the network heading for a destination within one of the groups, and in red the *static OD demand per destination group* (DEM) expressed in [veh/hr] for that group.

Collection of saturated links
The bottom right figure gives an impression about the number of links and the number of time periods these links have been saturated over 50% of their *jam density per lane*. The maximum value for this is 125 [veh/km] and the displayed values are given per lane on the z-axis. The number of links is given on the x-axis and the number of time periods is given on the y-axis. It is hard to derive any absolute values from this figure, but it is able to express some very distinct values for the different scenarios.

9.2.2.2 Assignment total table

The second measure to describe the assignment is the *total table* which holds some macroscopic descriptive statistics for the assignment.

Table 9-2 Calibrated Assignment descriptive statistics

Name	Value	Unit	Name	Value	Unit
Link Travelers	2.4410e+006	[veh]	Time	2.8210e+006	[veh*min]
Kilometrage	3.1971e+006	[veh*km]	Travel delay	2.2874e+005	[veh*min]
Arrivals	1.9849e+005	[veh]	CS arrival	3.3175e+007	[veh]
Accessed	2.1206e+005	[veh]	CS departure	3.9060e+007	[veh]
Travel demand	7.6597e+004	[veh/hr]	Objective	902.8927	[-]
Remainder	1.3574e+004	[veh]			

Where *link travelers* is the total number of registered travelers summed for all links and destinations. (Because a route consists of multiple links, this value is far more than the number of arrivals for example). The *kilometrage* is the total value of all vehicle traveled kilometers during the simulation. The *arrivals*[1] is the value of all arrived travelers at their destination, the *accessed* is the value of all travelers who managed to enter the network from one of the origins. The *travel demand* describes the summed value of the demand for traffic per hour for all destinations and the *remainder* describes the total number of vehicles still in the network at the end of the simulation. The *time* describes the total travel times used in the network for all travelers and the *delay* describes the total delay encountered for all travelers. The *CS arrival* describes the cumulative sum of arrivals (formula 4) and the *CS departure* (formula 3) those of departure used in the objective function and the average distance is a measure for the average distance traveled in the network. And finally *score* is the value of the objective function for the assignment.

[1] By an arrival a person is meant which has finished his/her journey and has arrived at the destination node, where it is added to the total travelers on the destination link. By an accessed a person is meant which has accessed the network through the origin node and is starting his/her journey toward the desired destination.

9.2.2.3 Discussion of the calibrated assignment

The calibrated assignment shows some distinct dynamic behavior which is partially explained in this paragraph. Since this behavior is difficult to illustrate by means of tables, stills or plots this paragraph is added.

Artificial bottleneck near Delft Zuid
If we look at the stills from *figure 9-10* we have illustrated the location of our *artificial bottle neck* (BN) by means of a black arrow. As we have discussed (9.2.1) this BN was one of the calibration variables which enabled us to *mimic* some observed queue behavior near *Delft Zuid*. Without this being thoroughly investigated, the general idea is that due to the traffic flowing their almost at capacity small *disturbances* lead to queue behavior. What we have done is forced a *static* bottleneck by reducing the capacity from 6600 to 5500 [veh/hr]. The queue spills back all the way to *kleinpolderplein* which can be seen in the stills at later times.

Other small queues near large intersections
If we keep the OD time slice profile in mind (figure 9-8) we can see that the overall demand is increasing right until period 6, where it is followed by a small drop and sustains this high *time slice* until period 10. The effect of this varying OD demand and the fact that all users will choose their perceived shortest paths will lead to some small congestive effects near the other intersections as well; in specific *Kethelplein* and *Vaanplein*.

Overall dissipation
When the OD demand decreases, we witness the *culmination* of the queues near *Kleinpolderplein* due to the artificial BN which then slowly resolves until the demand can be handled by the network capacity.

Discussion of the assignment plot and total table
With this in mind we briefly discuss the absolute *plot assignment* and the *total table* results. If we look at the first figure from the plot assignment we see an average delay of 0.5 hours for people heading toward *destination group 1*, which is off course caused by the artificial bottleneck. Groups 5 and 6 also encounter some delay which could be caused by the queue they have to pass located on the A20 due to travelers heading for the A13.

If we look at the figure illustrating the *kilometrage* we can explain the high values for groups 8 and nine if we look at the *brown bar*, illustrating the demand in [veh/hr], for these groups. Since the kilometrage is an absolute value, the sheer demand difference in [veh/hr] between groups 8 and 9 is also expressed in the absolute kilometrage.

The third figure also describes the number of people *accessing* the network and *arriving* at their destination. We could say that the difference between access and arrivals of group 1 is in the absolute sense almost as large as the difference between groups 8 and 9 with a far smaller absolute demand in [veh/hr] which is off course directly related to the bottleneck,

The fourth figure should be approached intuitively, what we see is a collection of links (x-axis) with a high density (z-axis), however no where near jam density (!), during a large number of periods (y-axis). This clearly indicates those links being affected by the bottleneck which still allow traffic to flow but have an increased density due to the upstream bottleneck.

The assignment totals listed in the table speak for themselves and can be used as a reference to compare other scenarios or assignments.

In the ensuing of this chapter we will discuss the three scenarios and their optimized assignment and finish this chapter with some general conclusions regarding the case study.

9.3 Scenario 0: Recurrent everyday morning congestion

In this scenario, we will try to optimize the calibrated situation as described in the previous paragraph using the optimization methodology.

We will do so by describing the *evolution* during the optimization process by means of the POOL, present four stills of the resulting assignment and use the *assignment plot and total table,* as discussed in the previous paragraph, to illustrate the difference. In the appendix we will provide the actual generated DRIP settings for the different scenarios. At the end of this paragraph we will discuss the achieved results for this scenario.

Since the calibrated situation has already been discussed in the previous paragraph we can start immediately with presenting the optimized assignment.

9.3.1 Optimized assignment

We will first discuss the pattern of evolution and the time it took and continue by discussing the resulting assignment by means of the *plot* and *total* table. Evolution

9.3.1.1 Evolution

In the figure we display the *evolution process* as it occurred by describing the chromosome scores from all chromosomes within the POOL for each generation.

The x-axis displays the generation number whereas the y-axis displays the value of each chromosome for the objective function.

As mentioned before we mutilate the first generation to diversify the gene data which is reflected in the low value of the very first generation.

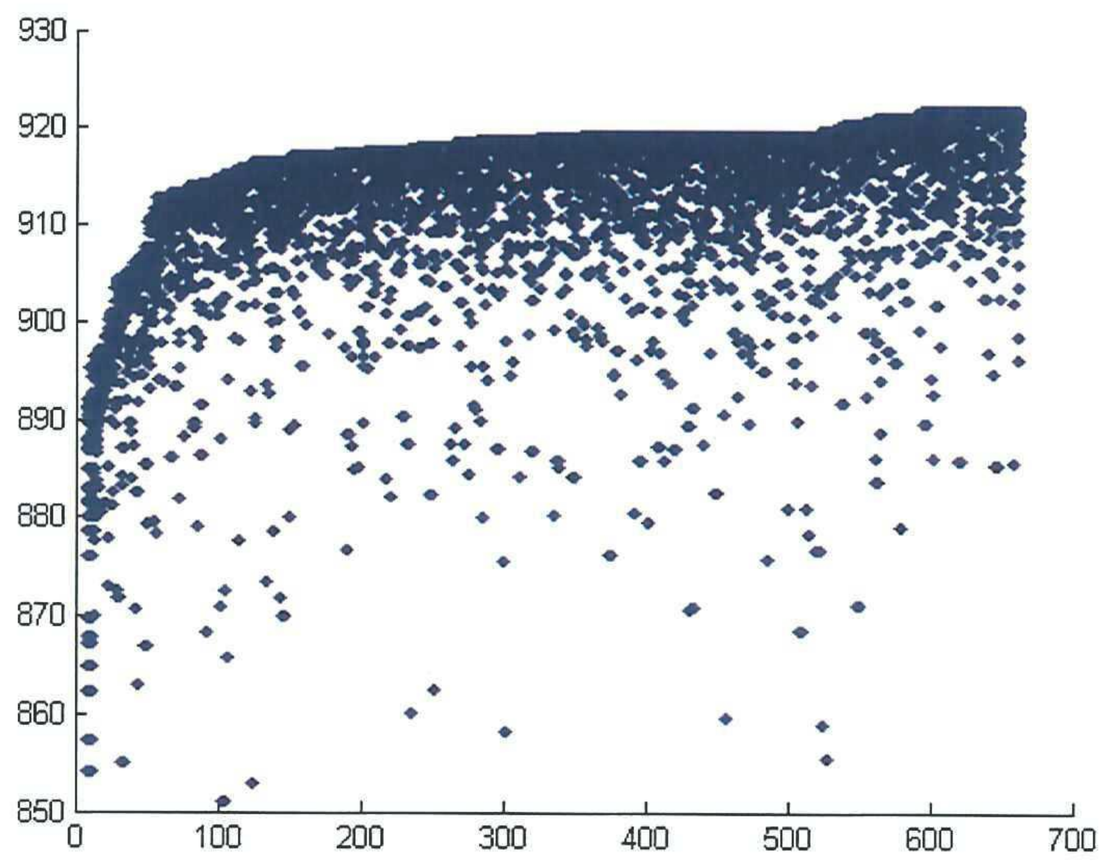


Figure 9-12 POOL evolution for optimizing the everyday congestion

The computers used as clients took about 300 seconds to finish calculating an assignment using a generated chromosome as input, the above evolution therefore was realized in approximately 60 hours.

Somewhere around generation number 600 the highest score value occurred and we will discuss the proposed DRIP settings in this chromosome and the resulting assignment in the next paragraph.

9.3.1.2 Resulting assignment

We will first display the resulting assignment using the four still images at different times.

Assignment stills

The pictures below give an idea about the network at different times during the assignment.

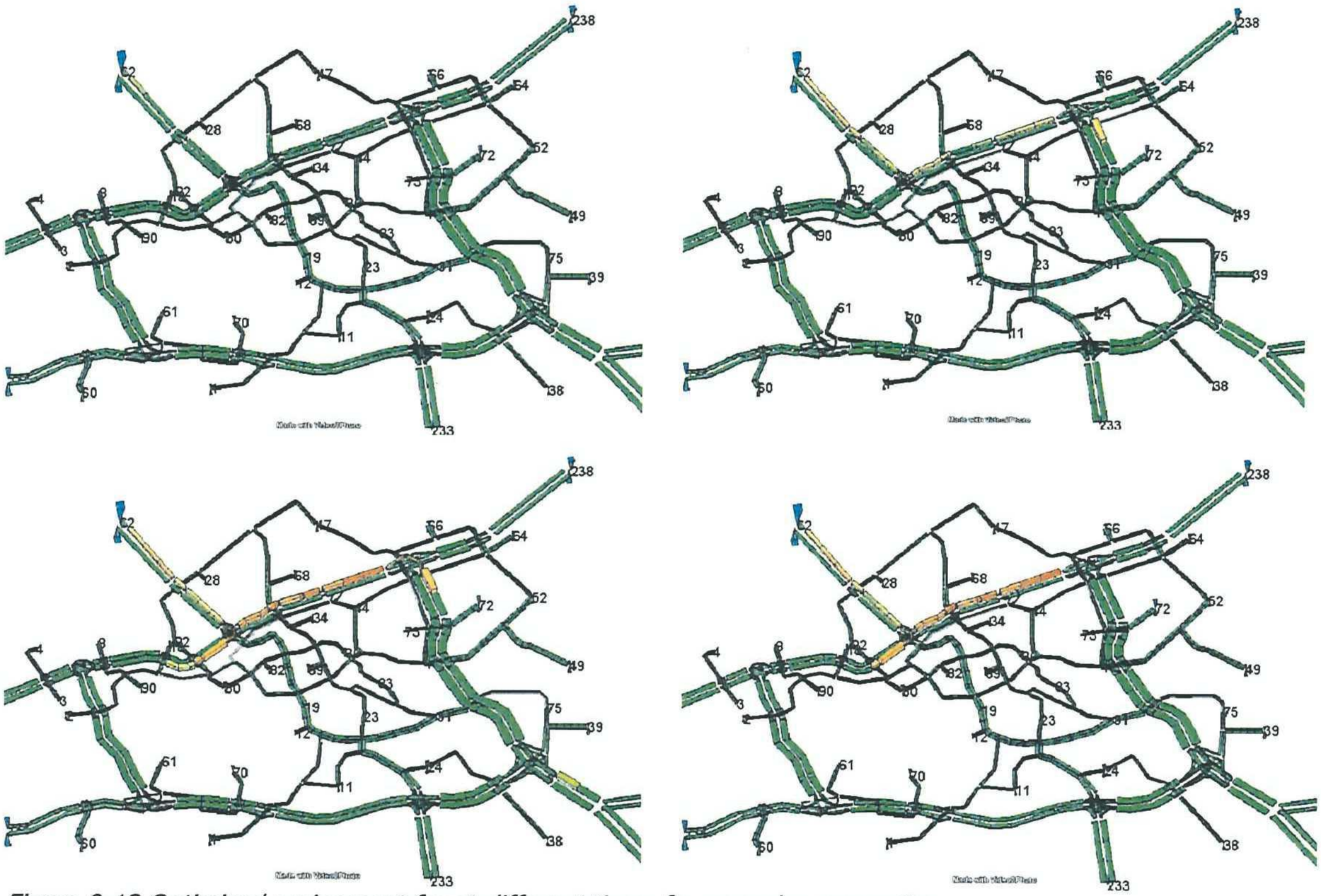


Figure 9-13 Optimized assignment for at different times for everyday congestion

From the figure we can conclude that in comparison to the calibrated situation, the queue is now concentrated around the bottleneck *kleinpolderplein*, this becomes especially clear if we compare the third still image between this figure and the calibrated situation in figure 9-9 where additional small queues are located near *Kethelplein* and *Vaanplein* which are now absent.

In the remainder of this paragraph we will display the *assignment plot* and *total table* but instead of using the absolute values for this assignment, we display the difference between the absolute values of the calibrated situation and those for the optimized situation, and thus effectively facilitating comparison and interpretation of the results. An exception is the *link saturation* which displays the absolute values for the optimized assignment and should be compared manually.
As mentioned earlier this practice is repeated in all scenarios.

Assignment difference totals

In the table below we describe the difference between the assignment totals.

Table 9-3 Assignment different network totals for everyday congestion

Name	Value	Unit	Name	Value	Unit
Link Travelers	1.5246e+003	[veh]	Time	-6.0120e+004	[veh*min]
Kilometrage	-3.6043e+003	[veh*km]	Travel delay	-6.4549e+004	[veh*min]
Arrivals	475.6489	[veh]	CS arrival	1.2183e+005	[veh]
Accessed	0	[veh]	CS departure	0	[veh]
Travel demand	0	[veh/hr]	Objective	19.0837 (921.98)	[-]
Remainder	-475.6489	[veh]			

Assignment difference plot

The assignment plot in the figure below describes the absolute difference values for the optimized situation minus the calibrated situation.

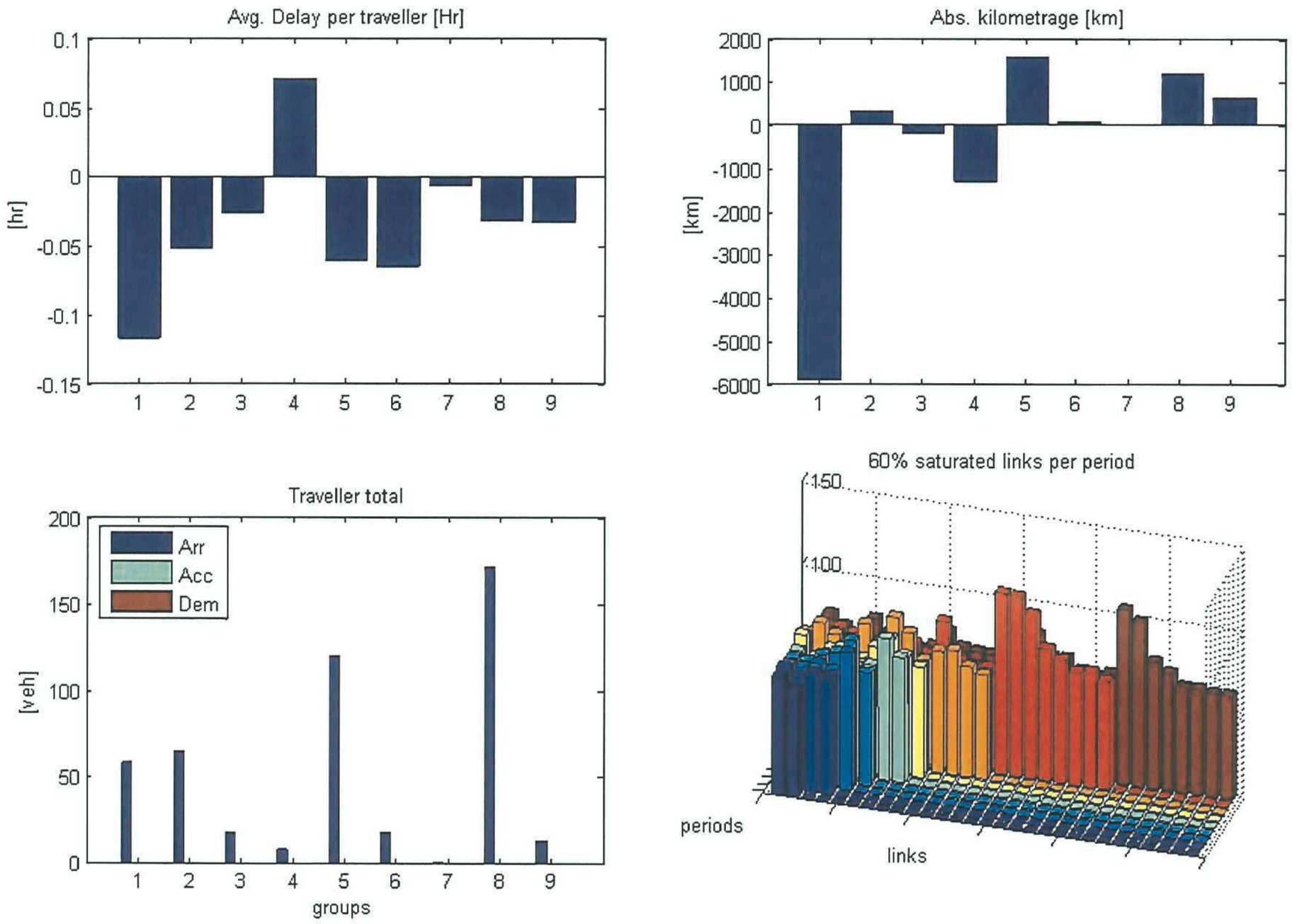


Figure 9-14 Assignment difference plot for the everyday congestion

9.3.2 Discussion of the optimization results for the everyday congestion

At first glance we could conclude that the average encountered delay dropped for all but one group of destinations with a maximum of 6 minutes for group one. The number of arrivals has increased while there is no difference between the number of accessed vehicles, this would lead to conclude that the throughput has increased. The kilometrage for group 1 dropped which is strange if we consider more people have traveled toward group 1. An explanation could be given by the fact that more people where re-routed through the city resulting in a shorter trip distance. The link saturation plot is different in that it shows less high peaks and a lower absolute number of saturated links.

The results presented support the notion the assignment has indeed been optimized, this is quite a good result if we consider the fact that the travelers in the calibrated situation base their routes on periodic user perceived utility based on *full network knowledge*.

If we use the objective function as a measure we could say the assignment has been increased by 2%(if we where to quantify the decrease in delay by means of the VOT the value for the achieved optimization would be far more spectacular).

9.4 Scenario 1: Extreme demand; UEFA final 2002

In this scenario we will try to use the route guidance settings on the DRIPS to optimize the assignment resulting from an extreme demand heading for D*e Kuip* instigated by the UEFA finals held in Rotterdam in 2002.

We will start by discussing the superimposed OD demand which will be used in the assignment. Following we will discuss the resulting assignment using *the route choices from the calibrated situation* with the super imposed traffic demand for *De Kuip*. Next we present the optimized assignment using the changed DRIP settings and discuss the achieved results at the end of this paragraph.

9.4.1 Description of the additional demand

The extra demand we super impose is an additional 5000 [veh/hr] (this number is fictive and not based on reality although Feyenoord starred in the finals that year and won in front of the home audience). This demand is distributed over the origin nodes as displayed in the figure below:

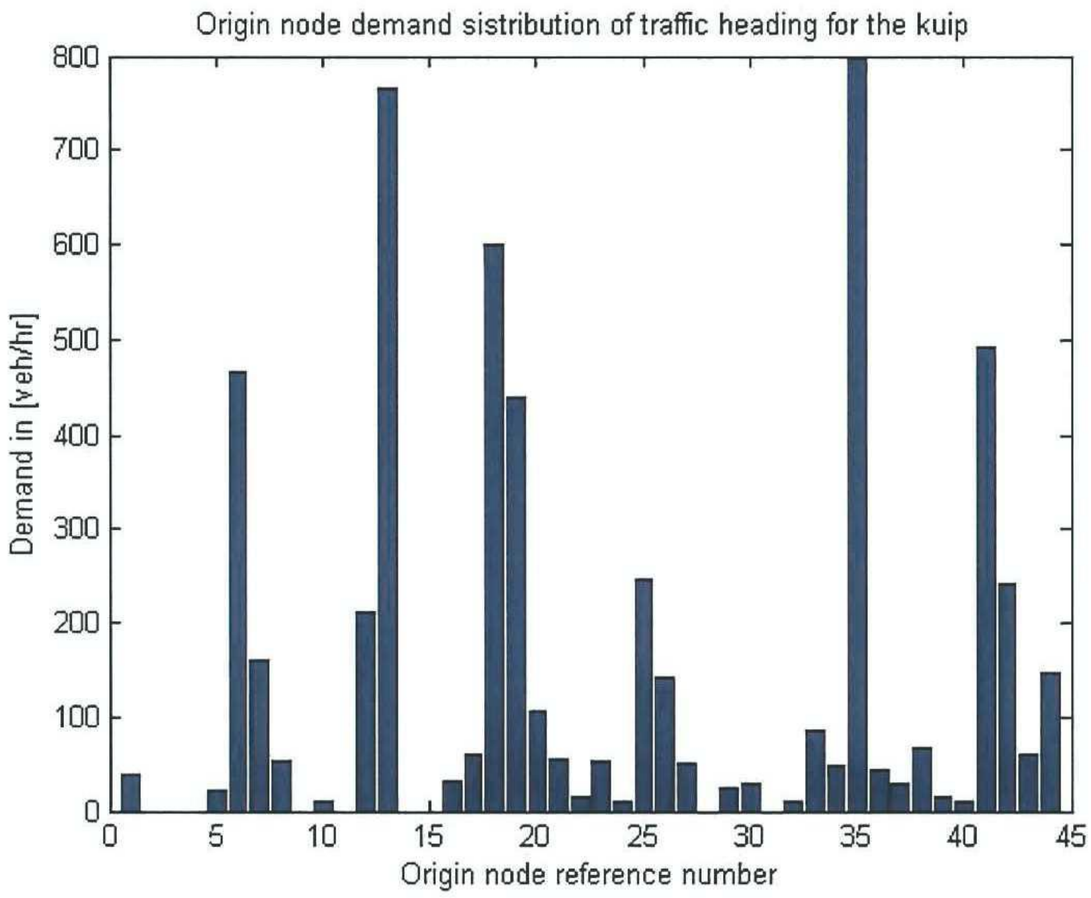


Figure 9-15 Distribution of the extreme demand heading for the Kuip by the origin nodes

The seven *peaks* in the demand distribution resemble the seven *origin nodes connected to the motor way*. This means that a large portion of the super imposed demand is coming from *out of town*, traveling by motorway and can thus be influenced by the DRIPS.

9.4.2 Calibrated situation

We will begin by presenting the four assignment stills to give some idea about the troubles ahead.

Assignment stills

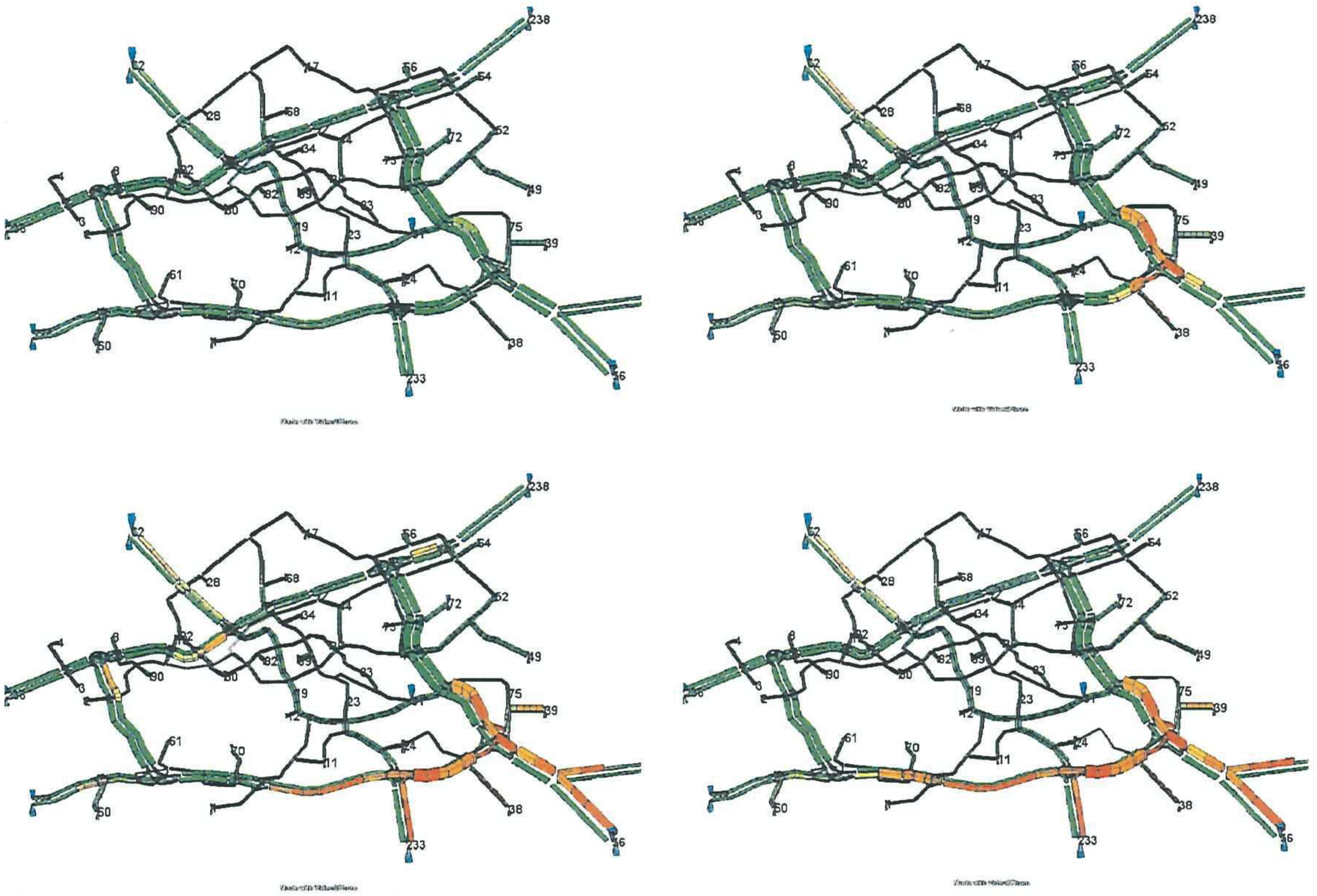


Figure 9-16 Assignment of the calibrated situation under extreme demand at four different times

We can see clearly how traffic starts to build up before the exit heading toward *de Kuip,* instigated by demand coming from the south, and the effects of the spilled back queue.

Assignment totals
In the table below we list the calculated totals for this assignment.

Table 9-4 Assignment totals for the calibrated situation under extreme demand

Name	Value	Unit	Name	Value	Unit
Link Travelers	2.3965e+006	[veh]	Time	3.3986e+006	[veh*min]
Kilometrage	3.1112e+006	[veh*km]	Travel delay	8.4763e+005	[veh*min]
Arrivals	1.9884e+005	[veh]	CS arrival	3.4032e+007	[veh]
Accessed	2.2074e+005	[veh]	CS departure	4.1610e+007	[veh]
Travel demand	8.1597e+004	[veh/hr]	Objective	747.0598	[-]
Remainder	2.1895e+004	[veh]			

Assignment plot

We will now display the assignment plot and the assignment statistics.

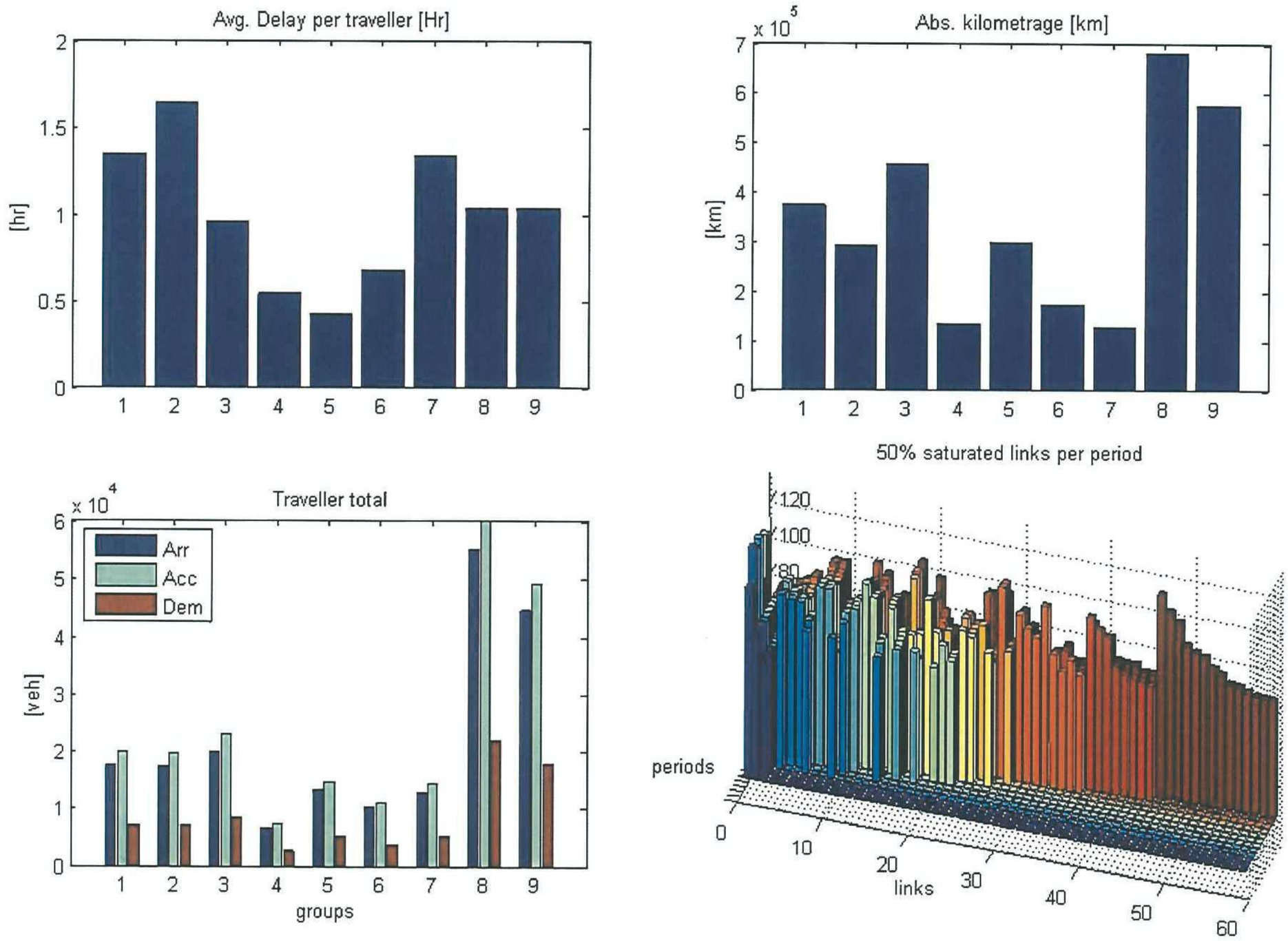


Figure 9-17 Assignment plot for the calibrated situation under extreme event conditions

If we would compare the absolute values and assignment stills belonging to the calibrated situation described in 9.2.2, the severity of the impact of this *super imposed demand football demand* should be quite startling.

If we compare the color *of the cells to the earlier presented* color bar *(*

Figure 9-10) we can say that the orange colored cells flow at 20-40% of the maximum allowed speed.

9.4.3 Optimized situation

In this paragraph we will discuss the calculated optimized assignment for this extreme demand scenario.

9.4.3.1 Evolution

The picture displays the evolution process which has been realized in almost 1000 generations. We can also see that no progression has been made since the 500th generation and conclude we could have stopped the process much earlier.

The low initial values of the chromosomes are directly related to the practice of mutilating the initial chromosomes aiming for diversified gene data. In all this process took about 60 hours and was performed on slightly faster PC's. (PIV, 2ghz)

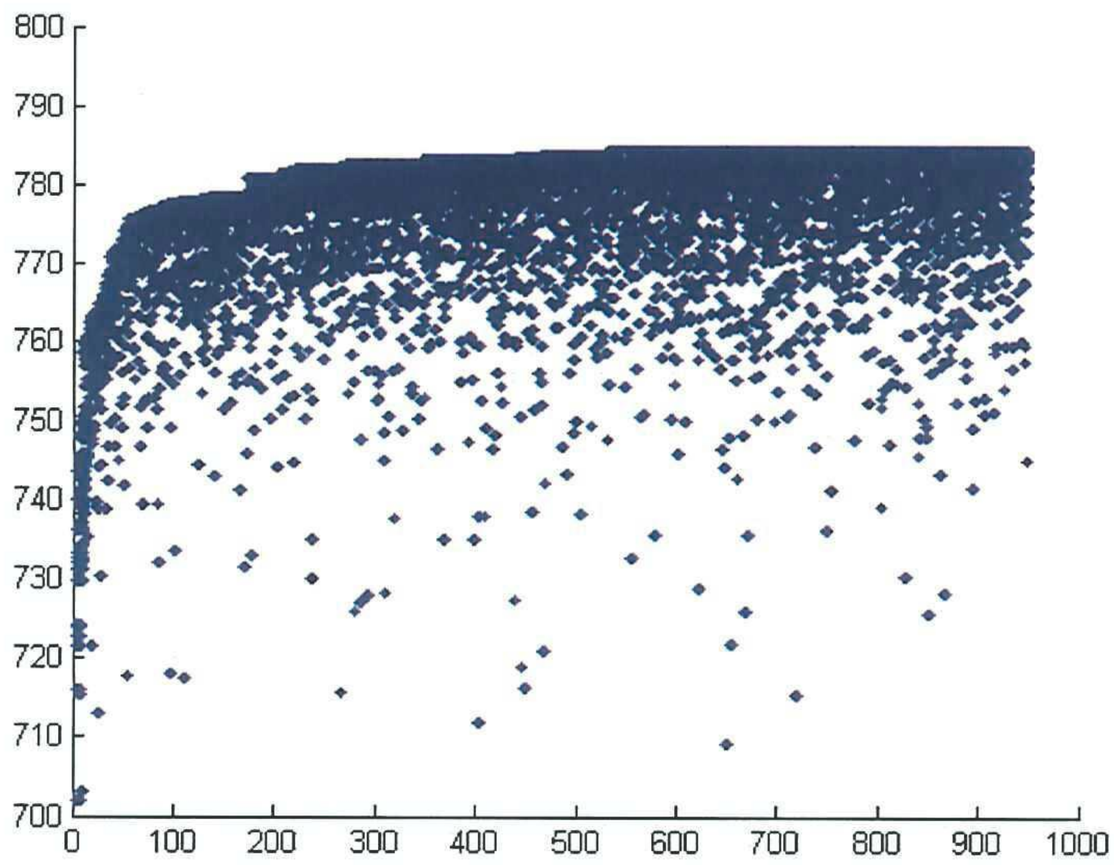


Figure 9-18 Evolution of the POOL for the extreme demand scenario

With the highest scoring chromosome in the POOL at the time of generation 950 when the optimization process was stopped, we have made an assignment and discuss the findings in the text below.

9.4.3.2 Resulting assignment

We will now provide the assignment stills, total table and assignment plot to give an idea about the changes in the assignment.

Still images
The resulting assignment is described by the four still images.

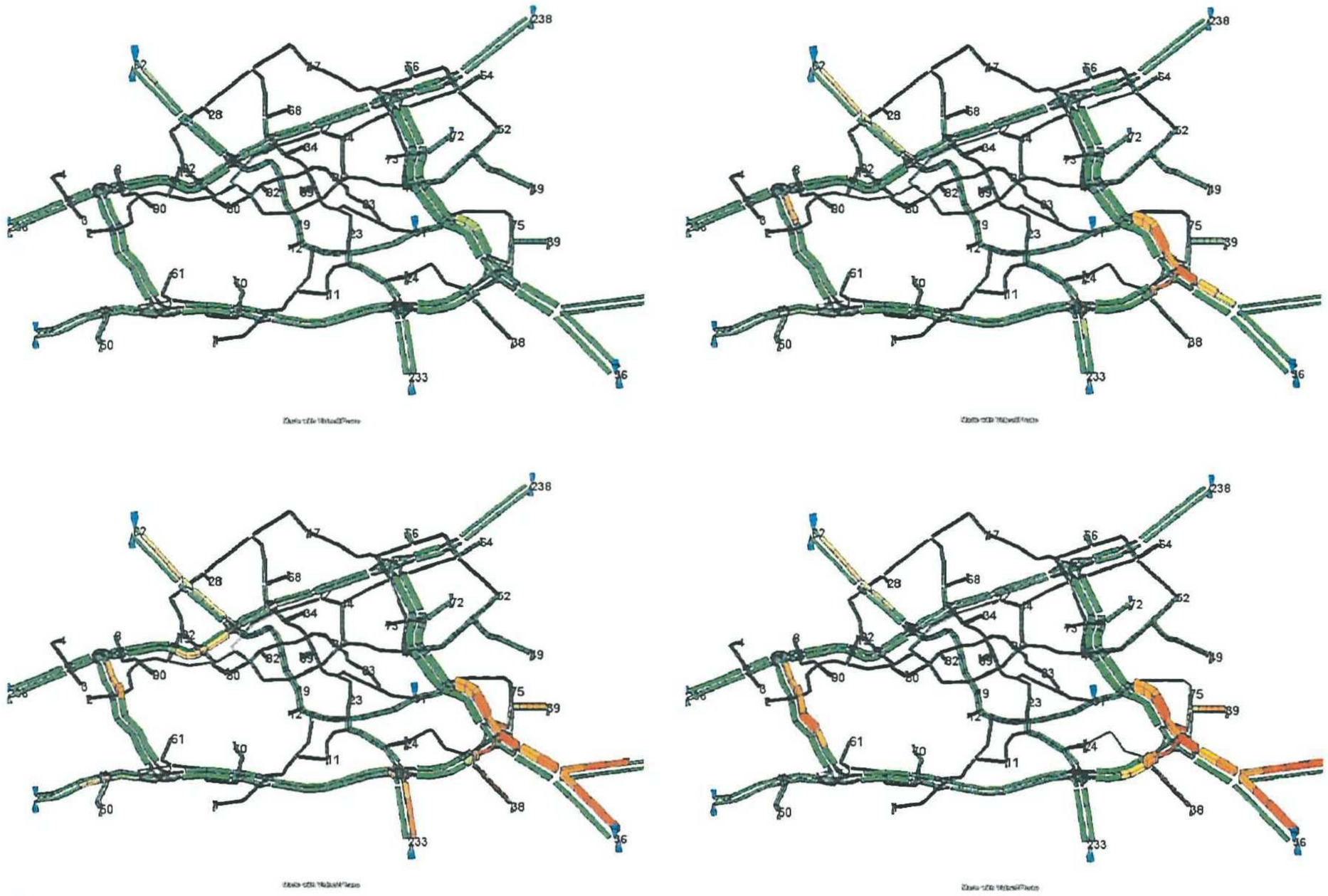


Figure 9-19 Resulting Assignment of the optimized extreme demand situation during four time periods

Assignment difference totals
In the table below we list the absolute difference between the optimized situation and the original calibrated situation.

Table 9-5 Assignment difference total values

Name	Value	Unit	Name	Value	Unit
Link Travelers	4.7093e+004	[veh]	Time	-1.1091e+005	[veh*min]
Kilometrage	7.0980e+004	[veh*km]	Travel delay	-1.6595e+005	[veh*min]
Arrivals	4.0595e+003	[veh]	CS arrival	3.6055e+005	[veh]
Accessed	1.6566e+003	[veh]	CS departure	0	[veh]
Travel demand	0	[veh/hr]	Objective	37.3215	[-]
Remainder	-2.4029e+003	[veh]			

Assignment plot

In the figure below the absolute difference between the optimized situation minus the calibrated situation is illustrated using the assignment plot.

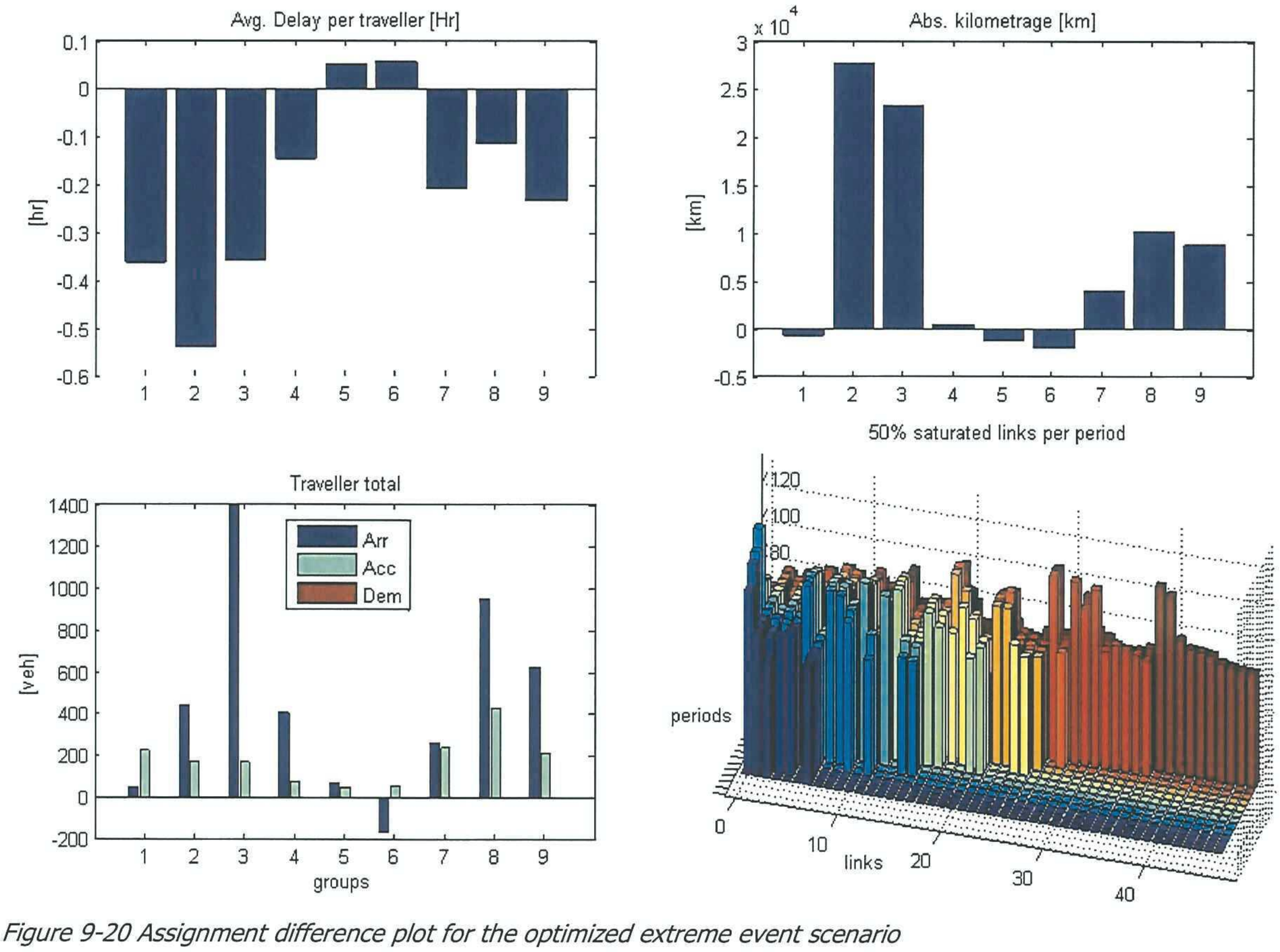


Figure 9-20 Assignment difference plot for the optimized extreme event scenario

9.4.4 Discussion of the optimization results

If we look at the assignment stills we can see the difference between the large portion of a congested A15 in the calibrated situation compared to a much small queue in the optimized situation. We can also identify some effects of the re-routing in the second still image indicated by the queue near intersection *Kethelplein.*

The assignment difference plot shows quite some progress in terms of the reduction in travel delays as high as 30mins for destination group 2. It is interesting to see the small increase in delay for groups 5 and 6 which could be considered exploitation (of the desirable kind obviously).

If we look at the kilometrage pot we can see some considerable increases in the number of traveled kilometers. If we take destination group 3 as example we see an increase of 2.5e+4 km. This increase could also be attributed to the high number of additional arrivals and people who managed to access the network. The 25000 [km] divided over the additional 1400 arrivals would mean these people would have traveled an average 17 [km]. Since the average distance in the network is 13.5 [km] we conclude that indeed some extra mileage has been made due to the re routing!

The link saturation plot shows a decrease in the absolute number of saturated links (approx 15) with a small decrease in the average lane density.

In all we can state that optimization again has been successful with an increase in the objective score of 5%.

9.5 Scenario 2: An accident near Terbregseplein

In this scenario we will use our optimization scheme to calculate optimal DRIP settings to deal with the following accident:

In the morning of Tuesday 9 march 2006 Jan Jaap Julius will borrow the Daimler Dart SP 250, a restoration project by his father finished just weeks ago, and use it to travel TU Delft University together with a friend. Unfortunately while using the on-ramp Kralingen at 7:00 in the morning, bad maneuvering, a broken taillight and fog will lead to a non lethal accident resulting in a crashed foreign truck carrying oranges and effectively blocking all lanes of the motorway.


Figure 9-21 Daimler Dart SP 250

The accident is spotted quickly and by 7.15 the first arriving police vehicles have restored the use of one lane. By 7:45 the tow vehicle has arrived and two lanes have been restored. At 8:30 salvage work has rendered three lanes open to the public again and two hours after the accident, at 09:00, all 5 lanes are once again open to the public.

The location of the crash site is indicated by the green circle in figure 9-5 and shown in detail to the right.

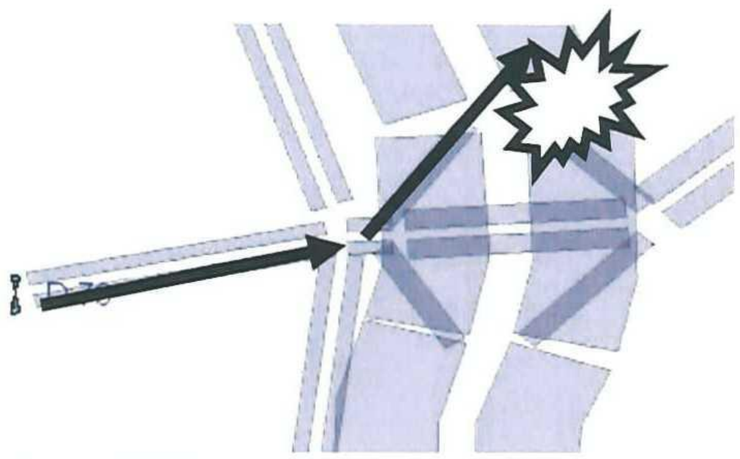


Figure 9-22 Location of crash site

9.5.1 Calibrated situation

The resulting assignment as the result of his accident using the calibrated route choices is shown in the figure below for 7:45, 8:30, 9:15 and 9:30 starting in the top left corner.

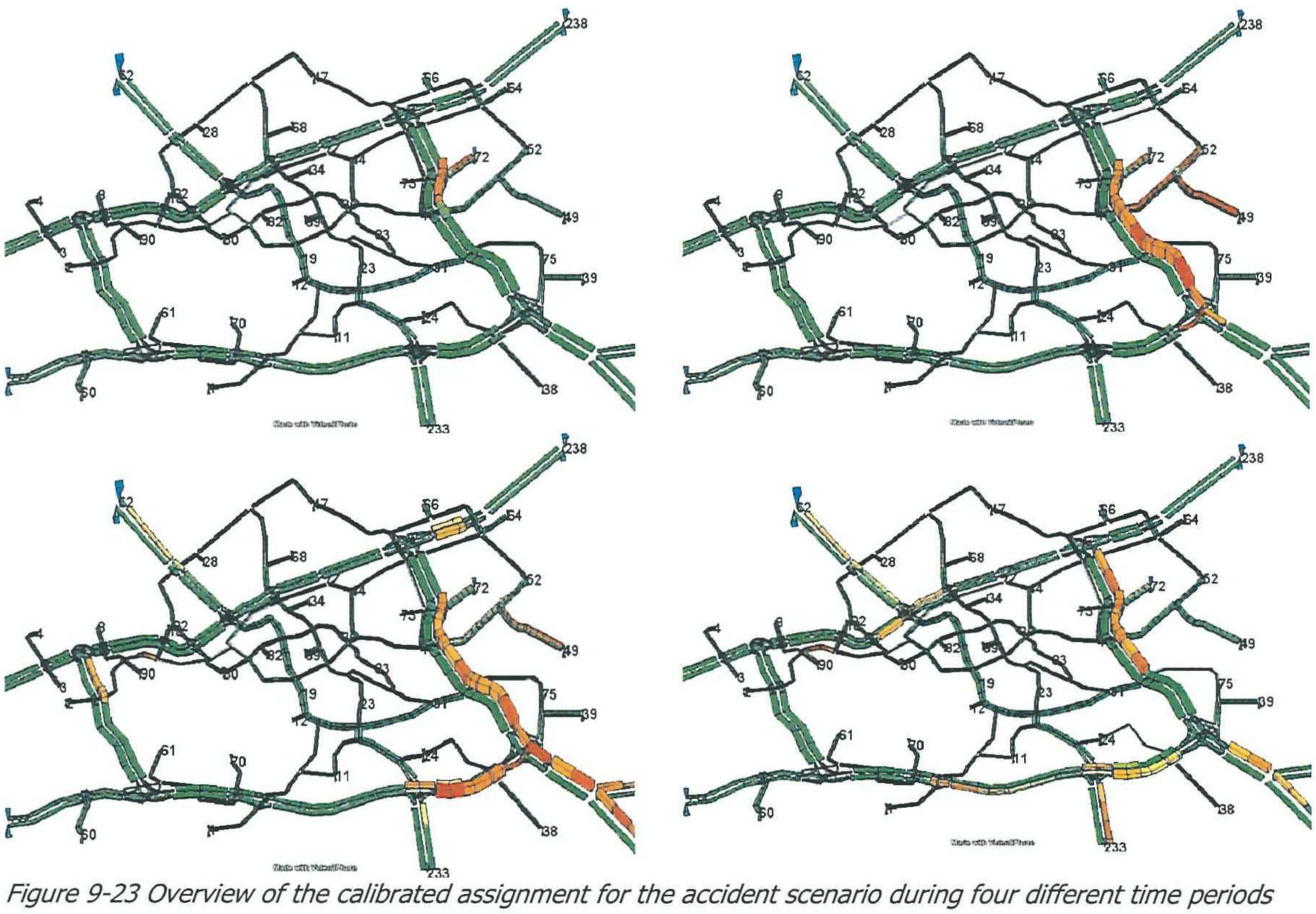


Figure 9-23 Overview of the calibrated assignment for the accident scenario during four different time periods

The accompanying *assignment plot* is displayed below:

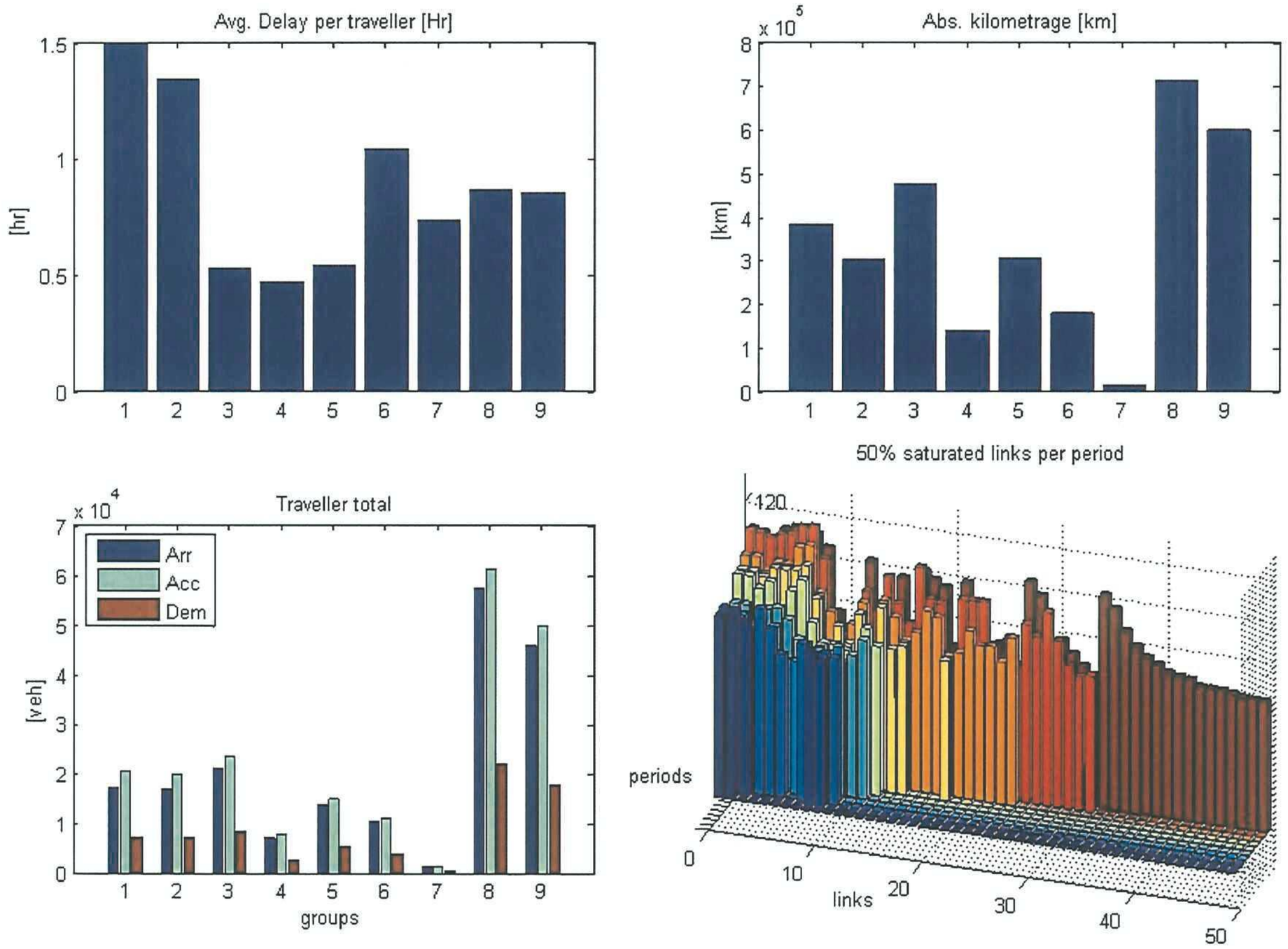


Figure 9-24 Assignment plot for the calibrated accident situation

Note that the *link saturation* plot shows a large collection of links which have been blocked a 100%, a fact which is revealed by their lane density during one period being almost equal to 125 [veh/km]. This plot also reveals the gradual steps in the availability of the capacity for the links closes to the accident.

With the corresponding network statistics being:

Table 9-6 Assignment statistics corresponding to the calibrated accident scenario

Name	Value	Unit	Name	Value	Unit
Link Travelers	2.3913e+006	[veh]	Time	3.3183e+006	[veh*min]
Kilometrage	3.1190e+006	[veh*km]	Travel delay	788787	[veh*min]
Arrivals	1.9243e+005	[veh]	CS arrival	3.1768e+007	[veh]
Accessed	2.1103e+005	[veh]	CS departure	3.9060e+007	[veh]
Travel demand	7.6597e+004	[veh/hr]	Objective	728.7306	[-]
Remainder	1.8596e+004	[veh]			

9.5.2 Optimized situation

The accident scenario has been optimized using the developed methodology and the results are discussed in this paragraph.

9.5.2.1 Evolution

In the figure we can see the evolution of this scenario was halted after 200 generations already, which is quite fast in comparison to the other scenarios (which were halted after 600 and 1000 generations).

The general *path* of evolution is the same in all scenarios and by illustrating the results already achieved during the *early years* in this scenario we try to illustrate that most of the computing time is used on minor improvements.

In 8.2.1 we already mentioned this fact while discussing strength and weaknesses of the realized EA.

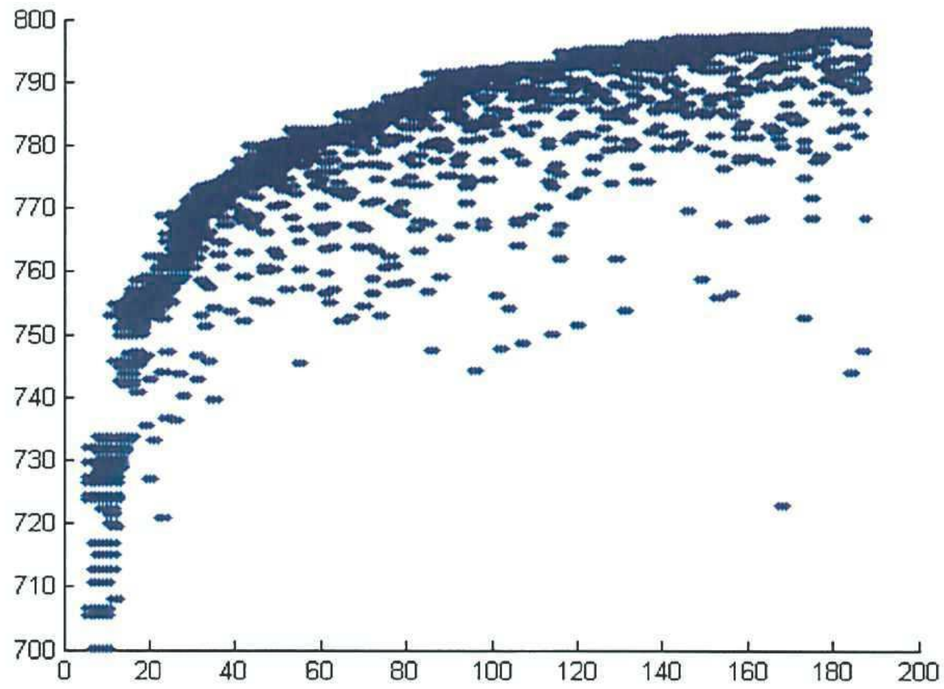


Figure 9-25 POOL evolution for the accident scenario

It was there we already stated that the EA performs excellent in finding the *region where the global optimum is located* but the way it is implemented now would take quite some time in finding the exact optimal value within this region.

By showing that for this scenario some quite good results are already booked in less then 200 generations we hope to illustrate this fact. If we look at the evolution curve in the picture we can see it is already heading for a horizontal path, waiting for the mutation operator to spark an improvement and make a small leap in evolution. Given the evolution paths in the other scenario's we could say that the expected improvements are small in their score values but could have considerable impact on the assignment. Before we go further it is time to take a look at the achieved optimization results so far.

9.5.2.2 Resulting optimized assignment

Assignment stills
In the figure below the assignment corresponding to the optimized accident situation is shown for the four time periods mentioned earlier.

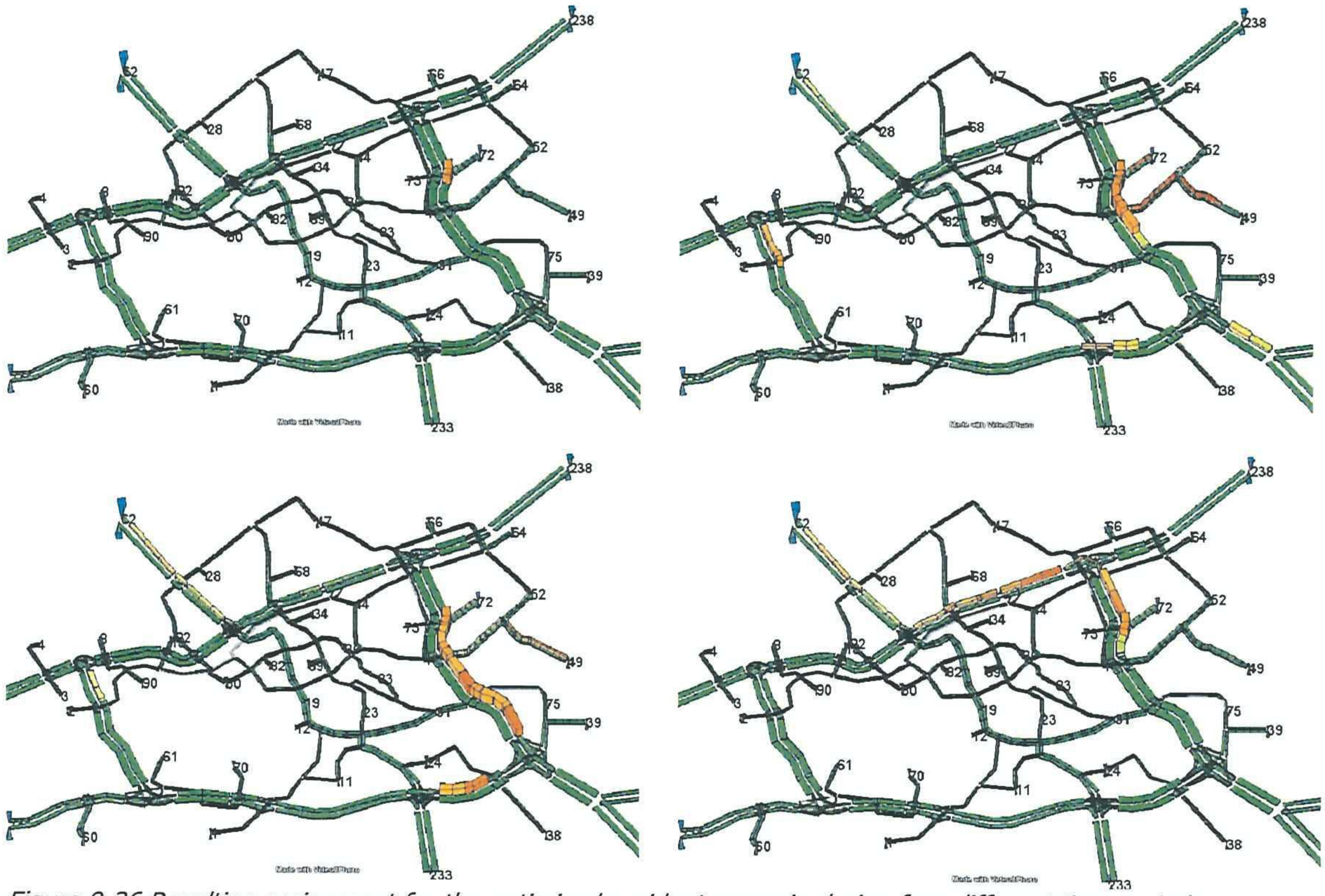


Figure 9-26 Resulting assignment for the optimized accident scenario during four different time periods

Assignment difference total values
We continue by providing the absolute assignment difference total values in the table below.

Table 9-7 Assignment difference statistics for the optimized accident scenario

Name	Value	Unit	Name	Value	Unit
Link Travelers	5.9644e+004	[veh]	Time	-2.5685e+005	[veh*min]
Kilometrage	7.9668e+004	[veh*km]	Travel delay	-3.3141e+005	[veh*min]
Arrivals	4.7434e+003	[veh]	CS arrival	6.3695e+005	[veh]
Accessed	863.2932	[veh]	CS departure	0	[veh]
Travel demand	0	[veh/hr]	Objective	69.7425	[-]
Remainder	-3.8801e+003	[veh]			

And on the next page present he difference plot for this scenario.

Difference plot

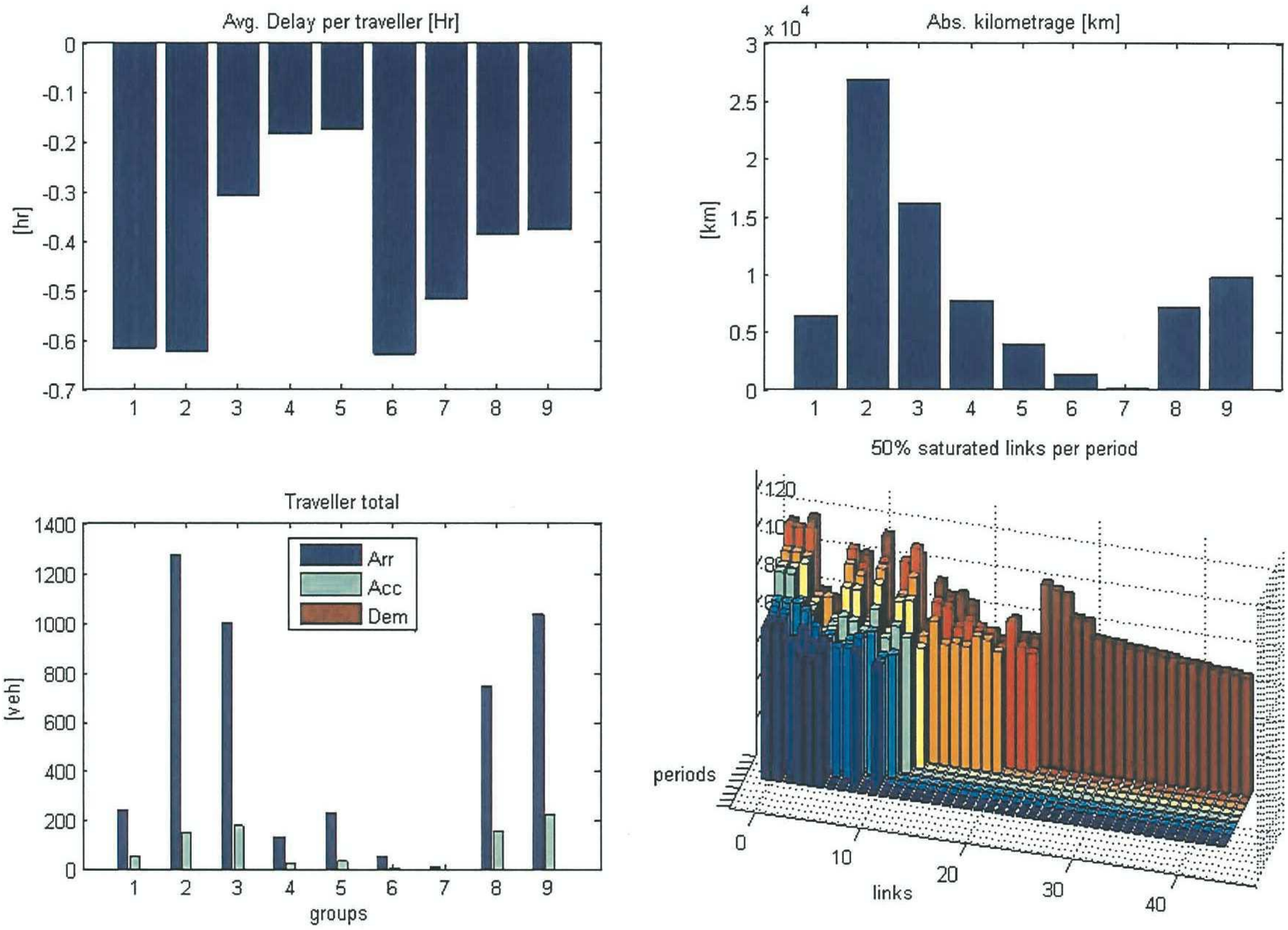


Figure 9-27 Assignment difference plot for the optimized accident scenario

9.5.3 Discussion of the optimized situation in the accident scenario

If we look at the difference plot we can see quite some improvement has been made regarding the encountered delay values for all destination groups. Also the number of people who managed to arrive at their destination increased considerably in cooperation with a higher number of people who managed to enter the network. The total kilometrage increased but again this is partially attributable to the increased number of travelers in the network.

The link difference plot shows a far more *relaxed* image of the saturated links. On the one hand, the absolute number of affected links dropped slightly but more important the number of links close to the accident (in the picture close to the origin) portraying the same steps in density reduction as described by the salvage work in restoring capacity is smaller; meaning due to the rerouting less links where directly affected by the accident.

The objective function increased by approximately 10% rendering it the highest optimization level so far. And results expressed in euro by using the VOT would be fare more appealing.

This scenario was clearly the best *scenario equipped* to be optimized because of the severity of the congestion and due to the easy re routing opportunity of people coming from *Ridderster* (the largest origin) via the A15 to the north side of the ring resulting in great improvement.

9.6 General Case study conclusions

In the case study we showed the ability of the optimization process to optimize all three scenarios using the proposed methodology.

When addressing the initial *mission statement* for this thesis: *a generic approach to a route guidance strategy*, we could say the case study allowed us to illustrate the *generic nature* of the developed methodology.

On the one hand, we are able to use any *given network and OD table* with a variable number of DRIPS on a variable number of locations as long as we can define *groups of destinations* for which the DRIPS could provide useful guidance. If we manage to calibrate the model appropriately we have the base situation needed for practically applicable optimization.

On the other hand we showed that *three different scenarios* where optimized using *one single methodology* without the need for formulating intricate exceptions and rules, which illustrates another side of the generic nature of the optimization methodology.

However, if we are to realize an online system as sketched in 8.1 the computation times would have to drop considerably. In the next paragraph we identify the major areas for improvement and make a guess about the optimization time under ideal conditions.

9.6.1 Speed Performance

One of the major problems facing the online application of the proposed methodology in the *rolling horizon context* is the performance of the whole methodology in terms of *speed*.

From the case study we can conclude the following in terms of performance:

- A client takes about 300 seconds to make an assignment
- In the first 75 – 100 generations the most improvement per generation is made
- From the total POOL evolution it follows that most generations are needed to improve a minimal part of the score.

The developed implementation of the proposed methodology is unfit to implement online, yet it proves *three* different areas where speed optimization can be gained which are discussed below. The first area is off course the *computation speed of the client* which is the main bottleneck in the whole process. The second area is the *speed of convergence* as a result of the generation process in the evolutionary algorithm and the third area is the settings needed in the *rolling horizon context*.

DSMART Model implementation

We will now list some areas of suspected improvement for the client together with an estimation of the possible increase in speed.

- As we already stated in 8.3.1 a minimal factor of 3 is expected when implementation of the model is realized in another language beside matlab.
- By carefully re modeling the urban network in such a way that: the behavior of the *onramps* and *off ramps* is still modeled, *urban routing alternatives* would exist and the destination *nodes for extreme demand is still there*, but:
  - reducing the total number of destination nodes to 70% (general complexity)
  - reducing the total number of links in the network to 70%
  - reducing the total number of nodes in the network to 70%

  Would increase[1] performance by $\left(\frac{1}{0.7}\right)^2 \approx 2$

[1] This reduction is explained in appendix IV

- Setting the time step from 30 seconds to 45 seconds would reduce the number of time steos needed, but in addition would also have a considerable effect on the division of the network into discrete cells. The combined effect is likely to be larger than $\frac{45}{30}$ and therefore estimated to be 2.
- If instead of a desktop computer, dedicated work clients would be used this might improve the performance of the model by an additional factor 2.
-

Given these estimates of improvement, we could say we can increase the performance of the clients with a factor $3*2*2*2=24$

EA optimization improvement
The second area of improvement is in the efficiency of the evolutionary algorithm.

In the figure we have used the POOL evolution of the incident scenario and identified three regions.

The first region is where the mutilated chromosomes experience the first beneficial mutations and are mutated into very crude solutions. These first generations can be replaced by using a crude scenario manager. This is depicted in a very modest way in the picture, but one could also imagine a crude scenario which would skip the need for the first 30 generations.

The second indicated area is where the average increase in score divided by the increase in generation is equal to 1 (45 degrees). This is where the EA is at its best and evolution is steered toward the region where the global optimum will most likely reside.

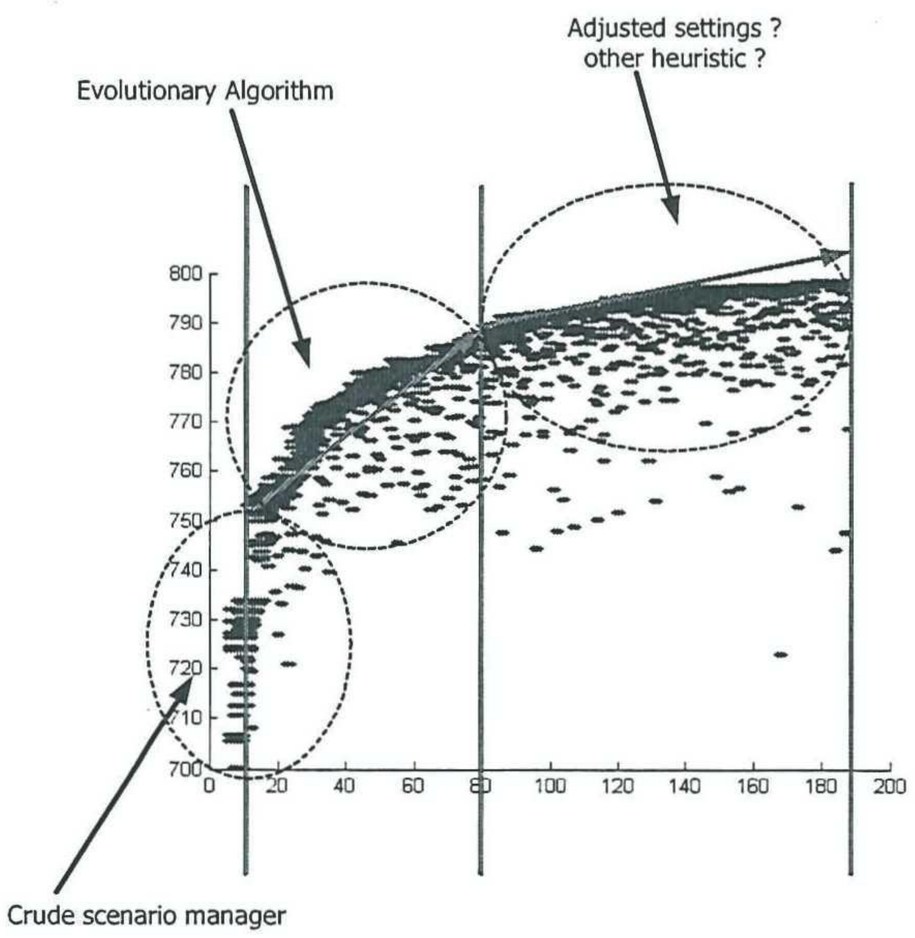


Figure 9-28 Areas for improving the Evolutionary Algorithm

As mentioned earlier after the initial evolution the *innovation* rated drops considerable when the EA as we developed it is used to find the *optimal values* in the supposedly global region. If *sliding settings* in combination with *additional operators* could be implemented, or *another heuristic* is used, which would allow the *optimum*, in the region the evolution process has been steered by the EA, to be found much quicker this would benefit the optimization process considerably.

Let's say by implementing the *scenario manager* and a *smarter way of local optimization*, we could reduce the number of needed generations to 200 (this was already the case for the incident scenario!). This number is based on a generation size of 8, and if we were to use *24 clients* instead of 8, and for simplicity reasons assume this would increase the efficiency by a *factor 2*, we would now need to calculate only *100 generations*.

Rolling horizon context
The final area of speed optimization is in the settings needed for the rolling horizon context.

- In the rolling horizon context it is unnecessary to model a simulation period of three hours; generating DRIP settings for the next 45 minutes should be enough if we assume the length of the horizon to be 30 minutes. The total simulation duration would therefore decrease with a factor $\frac{12}{3}=4$

Overall improvement
If we where to actually realize and combine all of the proposed speed improvement factors we would find a total optimization time needed of approx. $\frac{300[\text{sec}]}{24*4}*100 = 5.2\,\text{min}$.

Note that the actual time will be higher because the time it takes to create a generation and the communication of the chromosomes via the network is neglected, and to include these effects we round the optimization time to 6 minutes.

Off course the proposed improvements are not that simple, especially the reduction in the number of needed generations to 100. However it can not be denied that an expected optimization time of 6 minutes would bring the online application within grasp.

10 Conclusions and recommendations

This chapter presents the general conclusions related to the proposed methodology aimed at developing *a generic approach to a route guidance strategy* and lists the most important recommendations and provides a SWOT analysis.

In the first paragraph the *thesis objective* as formulated in chapter 3 is revisited and the first conclusion, that the *objective is accomplished*, can be drawn. In the next paragraph a number of generalized conclusions related to the developed methodology and its applicability in the *rolling horizon* context will be drawn. In the following paragraph the most important recommendations concerning the proposed methodology are formulated and the chapter is finished with a SWOT analysis of the proposed methodology and its intended implementation.

10.1 Thesis objective

In 3.2 the objective was formulated as:

> "Develop a methodology which is able to generate appropriate en-route route guidance in terms of prescriptive information. This methodology should preferably be independent of various network conditions such as traffic density, topology, link types etc and be able to generate route guidance for different scenario as they occur in practice. To further improve the quality of the route guidance, it should be investigated if a better alternative for feedback control could be applied."

In conclusion of this thesis we can state that we have indeed developed a methodology which is independent of network topology. As stated in 9.6 and illustrated by the case studies in chapter 9, the methodology is able to generate route guidance for *various scenarios and network conditions*. As for investigating an alternative for the currently used feedback control, we have shown in 5.1 that the *optimal control* used in the *rolling horizon approach* will lead to the best results.

We have shown that the *proposed methodology* is able to generate *optimal control based route guidance* which can be disseminated quite easily by means of *prescriptive advices* on DRIPS. The route guidance can be generated for *any given type of network* and *any given type of scenario as long* as we have a calibrated model and feasible alternatives exist in the network.

By introducing the compliance factor η as explained in 6.5.1 we are able to model the short term effect of compliance to the prescriptive DRIP advices in terms of users who will follow and those who won't. If over time user faith might increases/decrease or vary between DRIP locations this long term effect can also be expressed in a change in the factor η .

If the proposed speed improvement suggestions in 9.6.1 would be realized, the proposed *route guidance generation methodology* could be implemented in the *model predictive control* framework sketched in 8.1 and used in an online system.

10.2 Conclusions regarding the proposed methodology

In this paragraph we formulate the most important *overall conclusions* regarding the proposed methodology and implementation.

10.2.1 Route guidance generation methodology;

As to the nature of the route guidance generation methodology we can conclude the following:

- The open structure of the generation methodology summarized in 5.7 allows *other elements* to be used or *individual optimization* of the elements
  - For example a better suited DTA model (in terms of qualitative of simulation and performance) could be implemented as long as it is able to provide the necessary input for the EA and objective function
  - A new objective function can be implemented which would be better at translating the demands of different stakeholders etc.
  - The *evolutionary algorithm* responsible for generating and optimizing the DRIP settings can be the focus of study resulting in optimized performance of this algorithm and *smarter operators*.
- The developed methodology is network independent and can therefore be applied to any given traffic network as long as it is kept in mind that:
  - The network size determines the speed of optimization
  - Feasible alternatives must exist
- The developed methodology is *generic* in the sense it can easily deal with *varying network conditions* or *management scenarios* without additional constraints.
- The objective function can accommodate anything as long as it can be quantified using ex post, running or ex-ante simulation, network or assignment variables
- By introducing the cost of a DRIP in the objective function; in theory the optimal *number* of DRIPS together with the *location* and optimal *dynamic settings* could be generated as well.

10.2.2 Model predictive Control framework

Based on the results so far and the implementation of the RG generation methodology in the model predictive / rolling horizon approach sketched in 8.1 we can conclude the following:

- The idea of a *crude scenario manager* is valuable in the sense it can hold a realistic number of network scenario's because network conditions do not need to match exactly. (whereas other scenario managers need an astronomical amount of scenarios to exactly match network conditions)
- Using a *crude scenario* could decrease the number of generations needed initially and therefore speed up the overall optimization process.
  A crude scenario would render the mutilation of the first generation obsolete since its gene date is roughly what we need and there is no chance on premature convergence.
- The generation methodology as implemented in this thesis is inadequate for online application. However if the proposed speed improvements in 9.6.1 could be achieved, the total time needed for optimization could be reduced to a minimal 6 minutes which could very well be implemented in the rolling horizon context.

10.2.3 DSMART model

- The DSMART model is the determining factor in both the speed and quality of the generation of route guidance
- The combination between the DSMART Model and the EA by means of *strategy* variables, such as a measure for the *link activities* and the *basic free flow route choices,* proved very useful
- Simulating route choices by means of *destination specific split fractions* is a very effective way in terms of computation but also provides an excellent opportunity for simulating DRIP advices.
- The 1st order numerical solution provides understandable behavior of the simulated traffic
- The probit route choice process proved to model the traffic very well, when compared to the data in appendix III, and the addition of highway routes and custom routes to the route choices enabled effective simulation of DRIP advices

10.2.4 Evolutionary Algorithm

- The *determining factor* in the resulting evolution is the objective function
- The Evolutionary algorithm proved very well equipped in optimizing the DRIP settings under varying network conditions which makes it a generic approach
- The *open nature* of the EA allowed *custom build* operators to be implemented which proved very beneficial for the generation of DRIP settings. (the distinct mutation operators in 7.4.4.2)
- Most of the generations of the EA are used for a marginal optimization of the score.
- *Sliding EA settings* or *additional operators* could very well reduce the number of needed generations to marginally optimize (the *tail*) and be able to find the actual optimal values a lot quicker
- Another direction for optimizing the EA is in dealing with the *time dimension* of the DRIP settings. The large number of generations *in the* tail is partially the result of a stochastic process which needs to mutate individual time periods into optimal values. Dealing with the time dimension in a smarter way (as attempted by the super periods in 7.4.4.1) could yield some very good results and should be investigated.

10.3 Recommendations

In this paragraph provides recommendations regarding further research with respect to the proposed *methodology (not* regarding the *implementation.*)

Reducing the number of needed generations in the EA to find the optimal value

- Investigate the effect of *sliding settings* in the EA to reduce the number of needed generations to find the actual optimum and thus reduce the length of the *tail* in the POOL evolution
- Develop an additional operator, similar to the *Polyfit* proposed in 7.4.4.4, which is able to *smooth* DRIP settings over time and therefore aiding in the reduction of the number of generations needed
- All implemented *mutation operators* are now based on a single *link choice,* meaning that the operators only look at one connected downstream link to the DRIP link (7.4.4.2; spare Q operator and NON BR). By adding additional operators who do not solely consider the condition of the downstream *links*, but focus on the conditions of the *downstream routes*[1] could be very beneficial.
- A simple facility should be created which enables a *human expert* or *traffic manager* to quickly and intuitively create (a set of) DRIP settings and insert or override them into the next generation to be evaluated. This could increase the optimization tremendously.

Increase the quality of DRIP settings

- it should be interesting to see whether *an island implementation* (several clusters consisting of one server and a number of clients optimizing a scenario where the best performing solutions are periodically migrated from/to different clusters) of the evolutionary algorithm would increase the fitness value.

Improving the quality of the DSMART Model

- By implementing the effect of downstream traffic regulation on off-ramps more realistic queue behavior could be realized.
- Additional simulation detail considering *weaving behavior* and *near capacity disturbances* could be implemented
- Adding *maximum node capacities* could add some realism to the progression of flows via connected links. This way the links in- and output are no longer decisive but the capacity of an intersection would be
- It should be investigated if adding another dimension to the cells to accommodate different *user classes* would be beneficial or become computationally intractable.
- Instead of a *general time* slice τ_p, *origin node specific time slicing vectors* $\tau_{o,p}$ could be added to further capture the dynamic effects of transport demand

Rolling horizon implementation

- in figure 8-1 a feedback loop is drawn from the generated DRIP settings to the *crude scenario* manager. This feature could improve the quality of the used crude scenarios and enable the system to adapt to long term changes. (e.g. changes elsewhere in the network in the Netherlands will have their effect on the demand pattern and size on the ring Rotterdam network. A feedback loop of crude scenario's could ensure adaptation to such changes in the crude scenario database)

---

[1] As a suggestion for such an additional strategy variable, I would recommend the clients to calculate *the travel time of the alternative routes* used for *each DRIP* and *corresponding groups* during *every period*. This way, when mutating DRIP settings they can be based on the previous travel times of these alternative routes for the considered period.

10.4 SWOT

In the following text a SWOT (strength, weakness, opportunity and threat) analysis is carried out on the proposed route guidance generation methodology and envisioned implementation. Generally speaking the S&W apply to internal aspects whereas O&T apply to external factors.

Strength
The optimal route guidance generation methodology is:

- Versatile to the possible scenario's for which it can be applied and generic in the sense it can be applied to any network and any number of DRIPS as long as feasible alternative routes exist.
- the object function can accommodate any type of objective as long as it can be quantified using ex-post, running or ex-ante simulation, network or assignment variables in any combination.

Weaknesses

- Finding a global optimal solutions is not guaranteed (however, a better one is)
- Large numbers of DRIPS and periods may take a long time to optimize
- The DRIPS themselves limit the possibilities for (more) specific prescriptive routing instructions
- The DRIP settings are based on a calibrated situation and a prediction based on the expected OD demand and network state. Bad calibration will lead to bad routing advices

Opportunities

- Under optimized circumstances this could be used in real time environment
- The methodology is excellent in combination with evacuation planning[1].
- Communicating the proposed routing advices to in car route guidance systems. E.g:
    - Specific instructions could be broadcasted locally near a DRIP via TMC or Bluetooth
- The evolutionary algorithm and the crude scenario manager allow

Threats

- the cost of the proposed system in comparison to the low expected revenue[2]
- skeptic attitude from travelers toward the system

---

[1] If mobile DRIPS would be positioned at strategic points in the network accompanied by an authority figure the compliance rate would be a 100%
[2] In Appendix VI a crude estimation of the benefit of such a system per year for the case study area is made, adding up to about 170k€ a year.

Literature

References

1. Zuylen H.J., *CT5800 Dynamic Traffic and Transport Management*, TUDelft CITG sectie Verkeerskunde, Oct 2001
2. Hoogendoorn, S.P., *Theoretical Framework for DTM, Course notes of PAO-Course DTM*, TUDelft faculty CITG
3. Papageriou M., *Review of road traffic control strategies, contributed paper* Proceedings of the IEEE vol. 91 no. 12, December 2003)
4. Luk J.Y. K., Yang C. , *Comparing Driver Information Systems in a Dynamic Modeling Framework*, Journal of Transportation Engineering, Jan/Feb 2003, pg 43-50)
5. Meijden E. van der, *handboek incident management rijkswaterstaat, Document TT98, Traffic Test BV*
6. Schiesel R., Demetsky M., *Evaluation of Traveler Diversion Due to En-Route Information*, Virginia University, MAUTC, may 2000
7. Arthur & Passini, R.,*wayfinding :people ,signs and architecture*,NewYork,MacGraw Hill Ryerson,1992
8. Jackson P., *Behavioral responses to dynamic route guidance (DRG) systems*, Paper to be presented at the PICT International Doctoral Conference, 28th-30th March 1994
9. Mahmassani H., et al, *Integrated arterial and freeway control strategies for IVHS advanced traffic management systems*, Center for transportation research, University of Texas, 1998
10. *Potentiele reikwijdte voor het gebruik van GRIPS*, AVV RWS, Wittenveen and Bos, oct 2003
11. *DACCORD, Consolidate DACCORD architecture*, v 1.0 , HCG, June 1998
12. *Kerncijfers Verkeer*, AVV RWS, dec 2003
13. M. Ben-Akiva, M. Bierlaire et al., *Development of a route guidance generation system for real time application (DYNAMIT)*, MIT, Aug 2000?
14. Crittin F., Bierlaire M., *New algorithmic approaches for the anticipatory route guidance generation problem,* STRC 2001 conference paper, ITS sessions, march 2001
15. Zuylen H.J., et al., *CT5804 ITS For dynamic Road Traffic Management,* TUDelft CITG, Traffic Planning and engineering section, 2002.
16. Dudek C. et al, *Improved Dynamic Message Sign Messages and Operations*, Texas transportation institute October 2000
17. Ross T., et al, *HARDIE, Harmonization of ATT roadside and driver information in Europe*, Drive II project V2008, Dec 1994
18. *Motto's op DRIPS*, Ministerie van Verkeer en Waterstaat, Adviesdienst Verkeer en Vervoer, Jan 2003
19. Messmera A., Papageorgiou M., N. Mackenzie, *Automatic control of variable message signs in the interurban Scottish highway network*, Transportation research part C, 1998
20. Bottom J., Ben-Akiva M. et al, *Investigation of route guidance generation issues by simulation with Dynamit*, Transportation and traffic theory, proceedings of the 14th international symposium on transportation and traffic theory, pg577-600, pergamon 1999
21. Ben-Akiva M., *Methodology for Dynamic Traffic Management systems, Demand modeling and traffic simulation,* seminar in operations research ppt show, MIT, may 31 2002,
22. Peeta S., Ziliaskopoulos A.K., *Foundations of Dynamic Traffic Assignment, the past, the present and the future,* Networks and spatioal economics I, 2001 pg 233-265, Kluwer academic publishers

23. Bovy P.H.L., Bliemer M., *CT4801 Transportation Modeling*, TUDelft CITG, Transportation and Planning, Aug 2002
24. Wan E.A. , *Control Systems: Classical, Neural, and Fuzzy*, Oregon Graduate Institute, Lecture Notes – 1998
25. Hegyi A., *Model predictive control for integrating traffic control measures*, TRAIL Thesis series, 2004
26. Peeta S., *System Optimal Dynamic Traffic Assignment in congested networks with advanced information systems*, Dissertation university of Texas Austin, sep. 1994
27. Kroes P.A. et al, *WM0312CT Filosofie, Technology assessment en Ethiek voor CT*, TUDelft TBM sectie Filososfie, Sep 2000
28. Moonesinghe K., Ranasinghe D.N., *Implementation and performance evaluation of heuristic optimization techniques On a network of workstations*, FAST-IEEE Student Conference, Pakistan, 1998
29. Hoogendoorn S.P., Botma H., Minderhoud M.M, *CT4821 Traffic flow theory and simulation*, TUDelft CITG, Transportation and Traffic engineering section, 2000
30. *State of the art distributie reisinformatie,* N. Radewalt, Ministerie VWS, AVV, Afdeling verkeerskundige analyses mens en maatschappij, feb 2002
31. *Implementing A Dynamic O-D Estimation Algorithm within the Microscopic Traffic Simulator Paramics*, California PATH Working Paper, UCB-ITS-PWP-2002-4 , Reinaldo C. Garcia, University of California, Irvine, 2002

Internet

www.amici.tudelft.nl
www.tsl.tudelft.nl

Bibliography

32. *Motorway Traffic flow analysis, New Methodologies and recent empirical findings*, Delft university press, 1998
33. Romph E. de, *A Dynamic Traffic Assignment Model, theory and applications*, Dissertation, TUDelft CITG, sectie verkeerskunde, july 1994
34. Ceylan H., Bell M. G.H., *Genetic algorithm solutions for the stochastic equilibrium transportation networks under congestion*, TR B, vol. 2, February 2005, pg 170
35. *Werkboek gebiedsgericht benutten, Met de architectuur voor verkeersbeheersing*, RWS Rotterdam, 2002
36. Pirsig R.M., *Zen and the Art of Motorcycle Maintenance: An Inquiry into Values*, Bantam, April 1984
37. Carey M., GE Y.E., *Bibliography of dynamic traffic assignment (DTA) related papers (1970 – Aug 2003) in alphabetical order with abstracts*, School of business and management, Queens University Belfast, Nov 2003
38. Farver, J., Tradeoffs in the Design of route Guidance systems, Lecture 8 PPT sheet, MIT, March 2003
39. Chabini I., Dean B., *Shortest path problems in discrete-time dynamic networks: Complexity, Algorithms and implementations*, MIT, 1999
40. Kotsialos A., Papageorgiou M., *The importance of traffic flow modeling for motorway traffic control*, Networks and spatial economics, Vol. 1, 2000, pg 179-203
41. Ben-Akiva M., Bottom J., Ramming M.S., *Route guidance and information systems,* Proc Instn Mech Engnrs Vol 215 Part I, 2001
42. Oh J., Jayakrishnan R., *Temporal control of Variable Message Signs towards achieving dynamic system optimum,* University of Carolina, Department of civil engineering, Institute of transportation studies, Dec 2000
43. Barabasi, A., Bonabeau E., *Scale free networks*, Scientific American, May 2003
44. Mahmassani H., *Dynamic network traffic assignment and simulation methodology for advanced system management applications,* Networks and spatial economics, I: 2001 pg 267-292, Kluwer academic publishers, 2001
45. Daganzo C.F., *The Cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory*, University of Carolina, Department of civil engineering and institute of transportation studies, october 1992
46. Berg, De Schutter, Heygi, Hellendoorn, *Model predictive control for mixed urban and freeway networks,* nov 2003

[i] "kijk maar hoe het met pinkeltje is afgelopen"
[ii] "Phaedrus, a thick glassed female student, an essay, and the bricks on the Opera house of the main street of Bozeman, Montana"; I hope the *quality* of this thesis is appreciated.

Appendix I: Verification

In this appendix we will verify the DSMART Model and the evolutionary algorithm, or at least make the outcomes they generate plausible. In order to do so, we use simple theoretical situations at which calculations can be made by hand and compare them to the outcome produced by the model and/or optimization algorithm. We start with a verification of the model in the next paragraph by addressing issues such as node progression, link progression, number of vehicles in a queue, queue speeds, queue buildup and route choice. In the following paragraph we will discuss the outcome of the *route guidance generation methodology* applied to optimize the settings of three DRIPS during 12 periods on a simple straight forward network. At the end of this appendix one should have enough trust in the operation of both the DSMART Model and the evolutionary algorithm used to optimize it.

1.1. Model Verification

To verify the model, we will start by showing that the implemented *node progression* by means of the *split fractions* and the *fprod* multiplication, as discussed in chapter 6, produces plausible outcome. As a result of this we can conclude that the model is able to progress travelers past nodes using destination specific split fractions and nodes of *m* cardinality. In the next paragraph we will show that the traffic assignment model is able to capture the essence of *queues* in terms of *number of queued vehicles* and the *queue dynamics* of build up.

1.1.1. Node progression

In this example we will verify the n*ode progression* by means of the example network below.

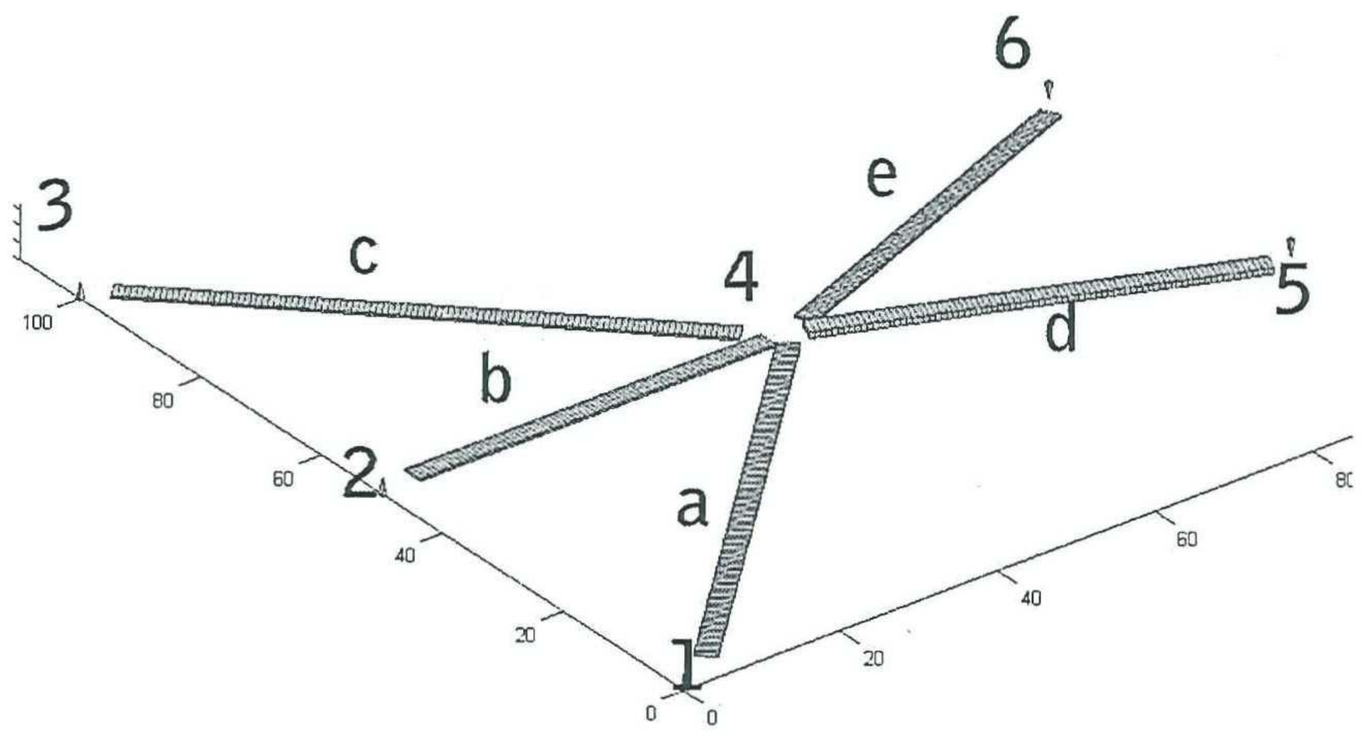


Figure I-1 Theoretical node to illustrate node progression

The network consists of 5 *links* indicated by the letters a to e. The links all have a jam density of $k_{jam} = 125$ [veh/km], a capacity $Q = 5000$ [veh/hour] and a maximum speed of $v_{\max} = 120$ [km/hr]. The table to the left shows the connection of each link in terms of *from node* and *to node* and the *rounded* length of each link in [km]. These links are connected by six nodes, indicated in the picture above by numbers and displayed in the middle table below with their geographic coordinates in [km]. Also shown is the OD matrix in the right table in [veh/hr] where the origins and destinations are listed row and column- wise respectively.

Table I-1 Node progression verification test network properties

Link	From	To	Length
A	1	4	71
B	2	4	50
C	3	4	71
D	4	5	54
E	4	6	54

Node	X	Y
1	0	0
2	0	50
3	0	100
4	50	50
5	100	30
6	100	70

	1	2	3	4	5	6
1	0	0	0	0	1750	750
2	0	0	0	0	1500	500
3	0	0	0	0	1250	250
4	0	0	0	0	0	0
5	0	0	0	0	0	0
6	0	0	0	0	0	0

The simulation time step chosen is $dt = 20$ [sec] and the total simulated period T takes three hours. There is no dynamic OD change so the demand is constant for the given period. One can verify manually that the intensity of link *d* should be 4500 [veh/hr] (1750+1500+1250) and that of link *e* should be 1500 [veh/hr] (750+500+250).

We can calculate the actual intensity per link by means of the following formula:

$$I_i = L^i_{V\max} \frac{\sum_{c=j0}^{j1} k_{i,c,d} L^i_{cellength}}{L^i_{length}} \quad (1)$$

$$L^i_{ncells} = \frac{3600 L^i_{length}}{L^i_{V\max}\, dt} \quad (2)$$

We find for the $k_{i,c,d}$ of link *d* a value of 37.5 [veh/km] with the actual cell length $L^i_{cellength}$ being 0.6667 [km]. The summation term has the references *j0* to *j1* which stem from the implementation and refer to the *actual* cells for this link. Since the cell density is *equal* for all cells in this link, we can also calculate the total number of vehicles on the link by using the total number of cells which is 81. (This number can be verified by (2)). With these numbers we can verify the actual intensity as being 4500 [veh/hr]. The same approach holds for link *e* where the cell density was $k_{i,c,d} = 12.5$ [veh/km]). With this result we make the following generalization:

Generalization:
Since the 1st order kinematic wave model, approximated by the Godunov scheme, resulted in the given values for $k_{i,c,d}$ in the example above, which we are able to reconstruct to the corresponding *link intensities* of which we can say they are in correspondence to the expected values, we state that the node progression as implemented in our model is plausible.

1.1.2. Link progression

In this example we will evaluate the simple network below consisting of two links and three nodes and show that the queue which is drawn in red as a result of the capacity difference between both links is in concordance with what we would expect. (The color indicates the speed varying from 120 as green to 0 as dark red, and the height resembles the actual cell density)

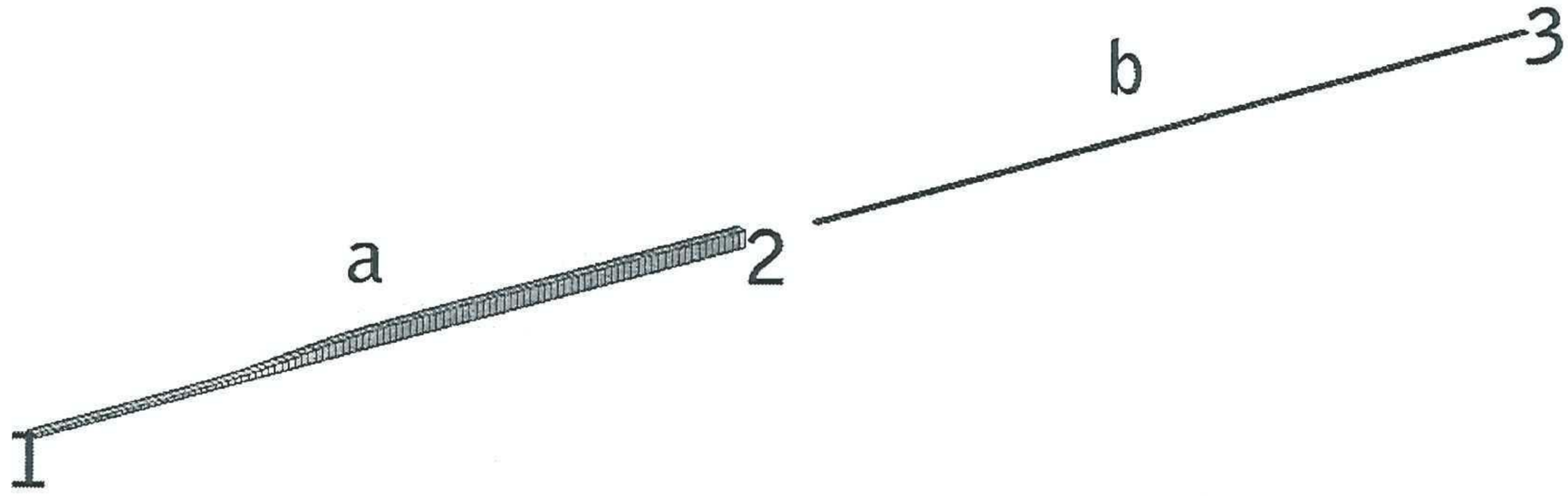


Figure I-2 Theoretical network consisting of two links to illustrate link progression

Both links all have a jam density of $k_{jam} = 125$ [veh/km] and a maximum speed of $v_{\max} = 120$. Link *b* has a capacity $Q = 1000$ [veh/hour] whereas link a has a capacity of $Q = 2000$.

The total simulation period is 3 hours with a time step *dt* equals 20 seconds. The network properties are listed below:

Table I-2 Node progression verification test network properties

Link	From	To	Length
A	1	2	70.71
B	2	3	70.71

Node	X	Y
1	0	0
2	50	50
3	100	100

	1	2	3
1	0	0	2000
2	0	0	0
3	0	0	0

Since we start with an empty network we can calculate the time it takes to arrive at the bottleneck.

$$t_{arr} = \frac{L^i_{length}}{L^i_{V\max}} = \frac{50\sqrt{2}}{120} \forall i = link\ a \qquad (3)$$

This takes about 0.589 hours and could be verified in the model itself where the second link was accessed during the 107th time step.

1.1.3. Number of Queued vehicles

The difference in the capacity of links *a* & *b* together with the given OD demand result in a systematic demand difference of $dq = 1000$ [veh/hr] at the end of link *a*.

Since we know the *arrival time* and the *simulation duration*, we know the *extra demand time* during which the demand difference *dq* is presented.

$$t_{exdemand} = T - t_{arr} = 3 - \frac{50\sqrt{2}}{120} \tag{4}$$

We are gone check this with the cell information we have for link a. Link 1 has a total of 150 cells with the following densities:

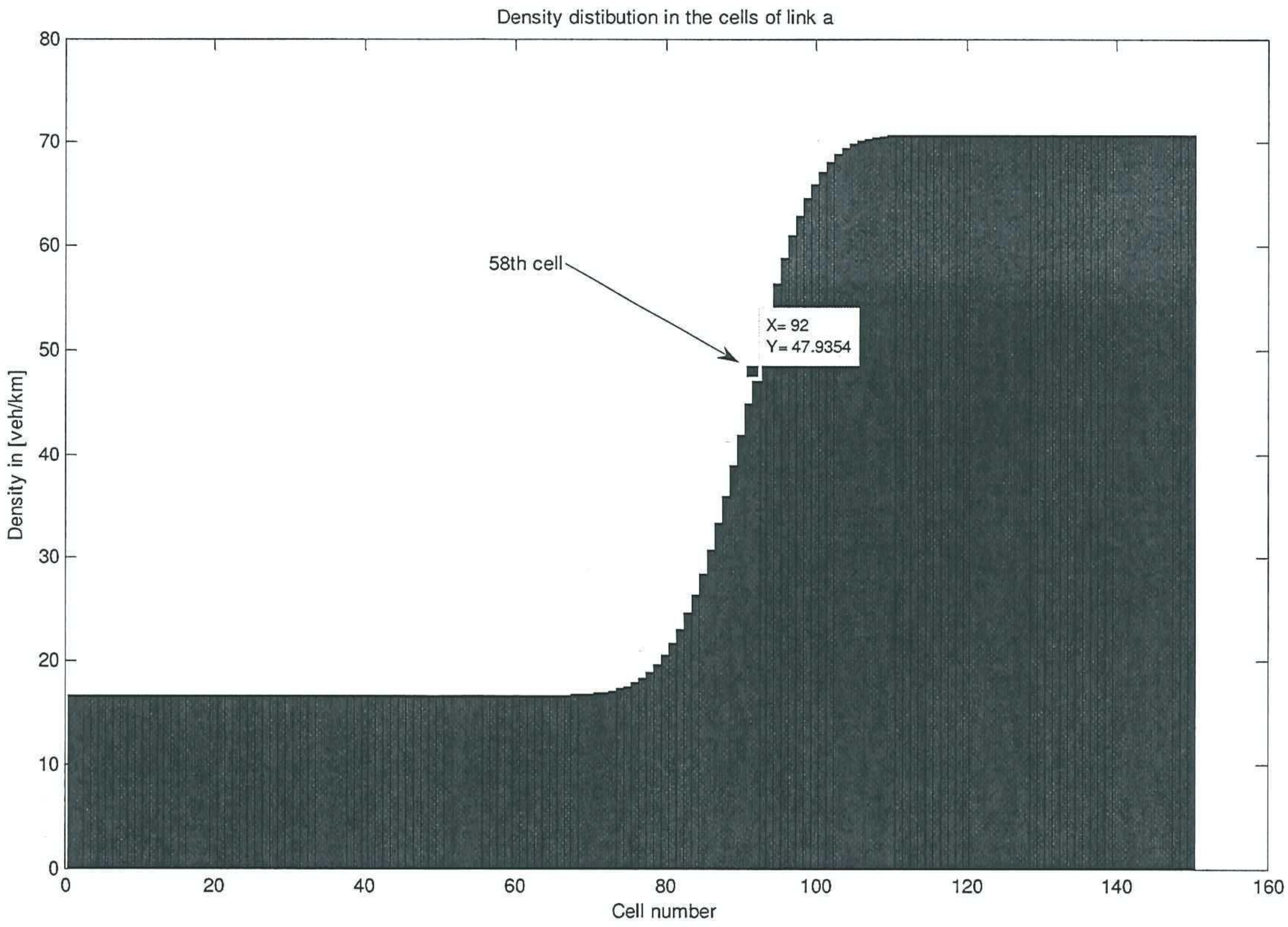


Figure I-3 Cell densities of the first link

If we look at the total sum of these individual cell densities, we find a total value of 5383 [veh/km] and each cell has a cell length (0.6667 [veh/km] (since the maximum speed is constant). With this, we can conclude that the total number of vehicles on this link is:

$$n^{i}_{vehicles} = L^{i}_{cellength} \sum_{c=j0}^{j1} k_{i,c,d} = 0.6667 * 5383 = 3589 \tag{5}$$

We can verify this by assuming the inflow and outflow of link *a* during the whole assignment period as:

$$n^i_{vehicles} = T(q^i_{in} - q^i_{out}) = Tq^i_{in} - (T - t_{arr})q^i_{out} = 3 * 2000 - (3 - \frac{50\sqrt{2}}{120})1000 = 3589 \qquad (6)$$

Therefore, the *amount* of people on the queued link based on the *calculated densities* is confirmed by the amount we would expect based on the *demand* and *simulation duration*, as calculated in (6).

In the example above we used the summed density to calculate the total number of vehicles. Since we use a triangular fundamental diagram, we should be able to estimate the densities ourselves.

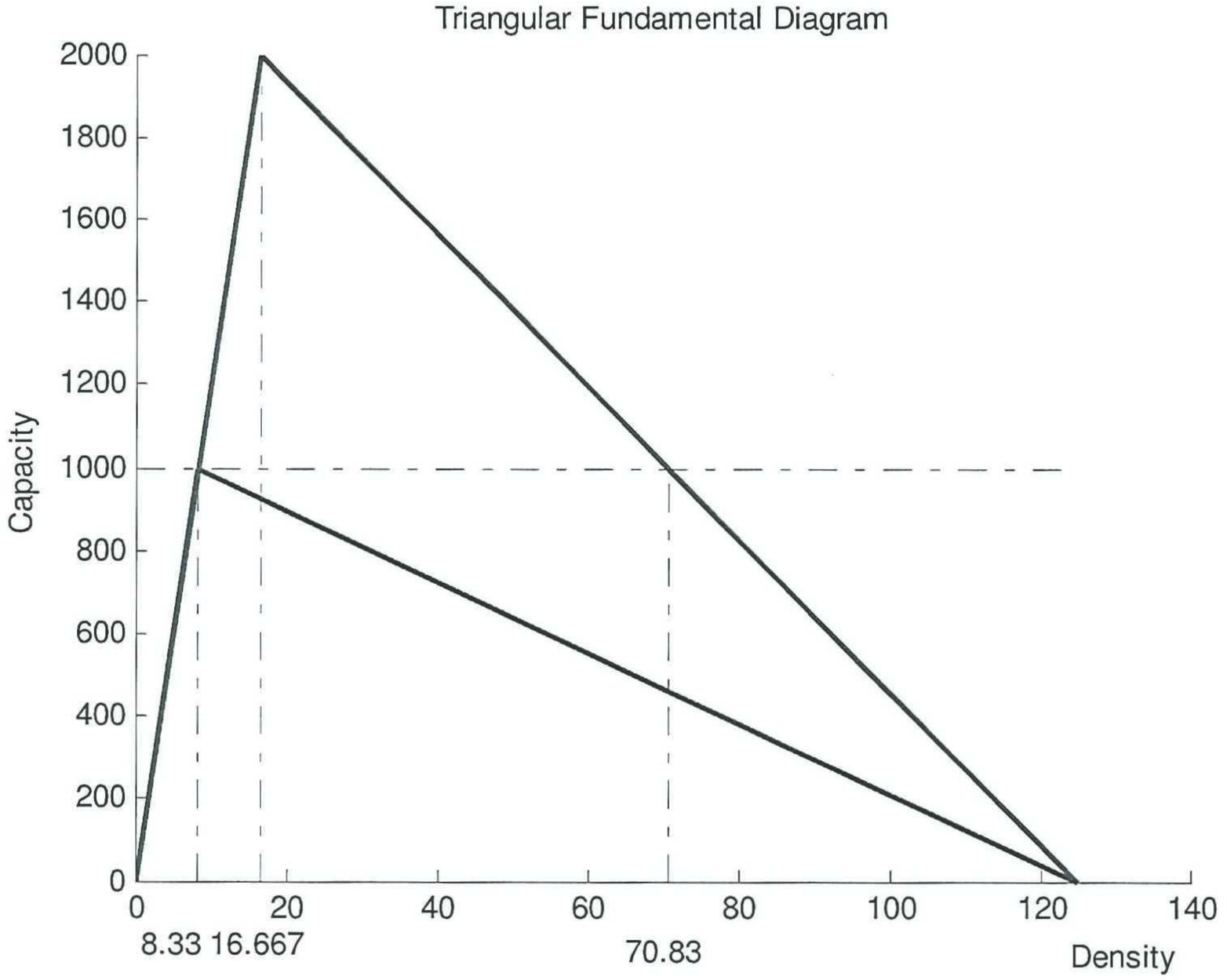


Figure I-4 Triangular fundamental diagram

In the diagram above we have drawn the *triangular fundamental diagram* which we use in our model as well. We can see that the density of the first link if when flowing at capacity should be equal to 16.667 [veh/km]. This matches with the *measured* density of the first cells drawn in figure I-3. In the congested region, we know the flow is equal to the maximum allowed flow in link *b*. We can reconstruct the density there to be 70.83 [veh/km] which also corresponds to the *measured* data.

Generalization:
Since the number of vehicles *measured* in the *cells,* created by the Godunov numerical approximation of the 1st order kinematic wave model, *under congestive conditions* matches our manually calculated values, we state that this approximation at least yields *plausible results during congestion* in terms of modeling the *absolute number of congested travelers.*

This assumption is significant, because the calculated route guidance is largely dependant upon the number of people being queued and their effect on the gross travel time objective function.

1.1.4. Queue buildup

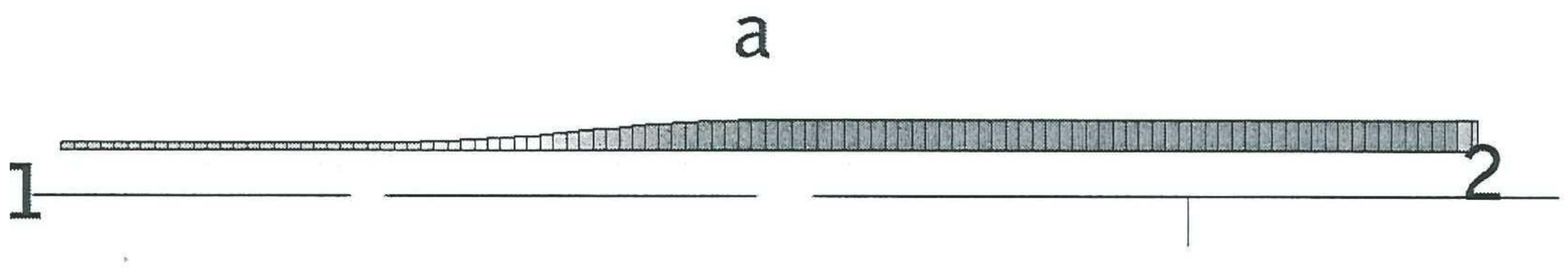


Figure I-5 First link and cell densities / speeds in detail

In the picture above we have taken a closer look at the cells of link a. The color indicates the speed and the height indicates the density. We will now try to illustrate the queue dynamics in terms of *build up* being in concordance to the expected values based on the 1st order approximation and with that, justify the implementation. (Whether the 1st order approximation itself provides a good enough representation of *real life traffic* has already been discussed and considered adequate)

From kinematic wave theory we can calculate the speed of a shock wave by:

$$\omega = \frac{q_1 - q_2}{k_1 - k_2} \tag{7}$$

Where the indices 1 and 2 are equal to the links *a* and *b*. With this formulation of the shockwave we find:

$$\omega = \frac{2000 - 1000}{70.83 - 8.33} = 16 \text{ [km/hr]}$$

Off course it takes some time to reach this density in a cell and before this happens, congestion effects are already made visible and spread stream upward (the green-yellow-red curved *tail* at the end of the queue in figure I-5).

But still, we'd expect the queue to be at $t_{exemand}\varpi = 38.57$ km stream upward or at the $\frac{t_{exdemand}\varpi}{L^i_{cellength}} = 58$ cell measured from the end of the link.

In figure I-3 this cell has been highlighted and sort of resides in the *middle* of the *s* curve at the end of the queue. The location of the queue *does in fact* match the position we'd expect by the crude estimation we made.

However the build of the queue is far more fluidly resembled by the Godunov approximation. This is because traffic *does still react instantaneously*, but does so *per cell*! In our crude manual estimation we only consider two regions and base our shockwave on the difference between those regions. By adopting the discretisation in space we have effectively created 150 regions (the number of cells for this link) and apply the shockwave theory to each cell/region individually. So the essence remains the same (with the inevitable drawback of instantaneous reaction) but the effect is far more *fluid*.

Generalization:
The implemented discretisation of time and space as a result of the Godunov approximation in our model enables a *fluid* representation of *queue build up*, which produces results that are roughly confirmed by the expected values calculated using a crude shock wave estimation.

1.1.5. Queue speeds

And finally we address the vehicle speeds in the queue.

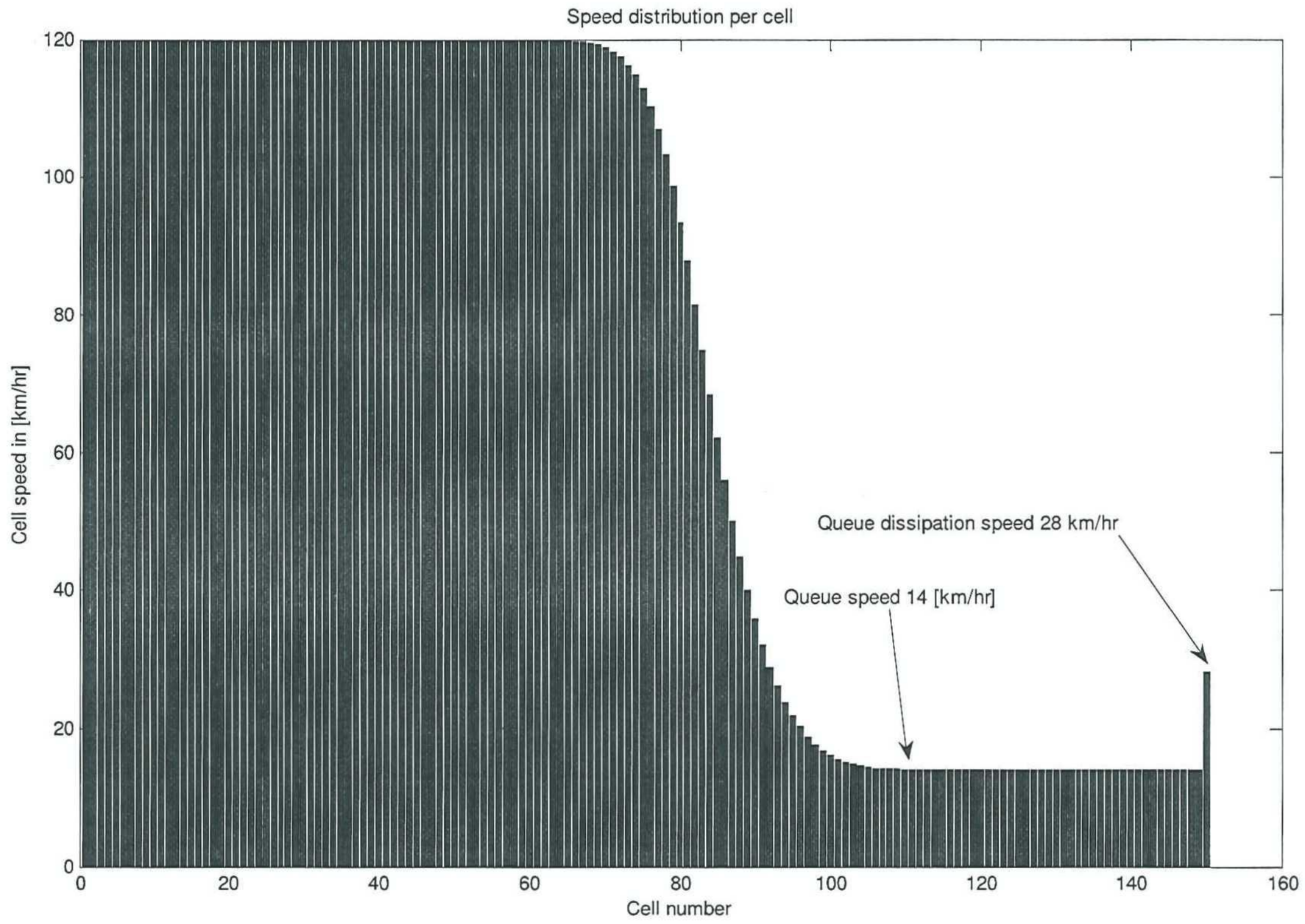


Figure I-6 Measured speed of cells of link a

In the diagram above, we illustrate the *measured cell speeds* per cell. We can see that the normal *free flow speed* is represented in the first cells, being 120 [km/hr], and witness a smooth deceleration curve until the queue speed of 14 km/hr in the last cells.

We also witness the high exit speed at the end of the last cell entering link 2 and explain this later on. But first of all, since the bottleneck formed by link *b* is stationary, the queue dissipation process behaves likewise. (If the BN would be removed or increase / decrease in time we would witness a similar shaped curve in terms of acceleration we now see when travelers catch up with the queue.

Speeds are calculated by means of:

$$v = \frac{q_{i,c}^{out}}{k_{i,c,\sum d}} \tag{8}$$

Where $q_{i,c}^{out}$ is the *outflow* of cell *c* belonging to link *i* measured in [veh/hr] and $k_{i,c,\sum d}$ is the aggregated total density for all destinations in cell *c* belonging to link *i*.

With this formula we can explain the *constant speed region* of the queue. Eventually when a cell is filled the density remains constant at 70.83 [veh/km], due to the flow in the bottleneck. These cells all behave the same in terms of *outflow* and *density* and therefore have the same speed. The speed can be calculated by the density and the *bottleneck capacity* being 1000 [veh/hr], and when substituted in (8) we find a cell speed of 14 [km/hr].

The last cell is calculated different from the other cells on a link. The reasons for this is the fact we don't know if the vehicles exiting the last cell of a link are able to progress past a node. This depends upon the available capacity in the first cell of the link to which the flow is directed by means of the destination specific split fractions. The result of this is an overestimation of the outflow speed of this cell, because the speed is based on the desired $q_{i,c}^{out}$ of the cell and the density $k_{i,c,\sum d}$ by which it is generated.

1.2. Verification of the optimization process

In this paragraph we will try to make the optimization process plausible by applying the process to the theoretical network below. In the first step we will use the model, as presented in chapter 6, and an OD matrix to dynamically assign this network with travelers heading from node 1 to node 5.

Given the normal shortest path route choice process and a small path alpha factor, resulting in small differences between the user perceived and the actual shortest paths, we will likely witness people traveling from node 1 to node 5 in a straight line. However, due to the specifically chosen link capacities, the routing of travelers using the single straight path in terms of the generalized gross travel time objective function is non optimal. The OD demand presented is 5000 [veh/hr] and this will cause link 1 to become congested and link 2 to function as a bottleneck and flow at capacity whereas the links 3 and 4 will flow below capacity.

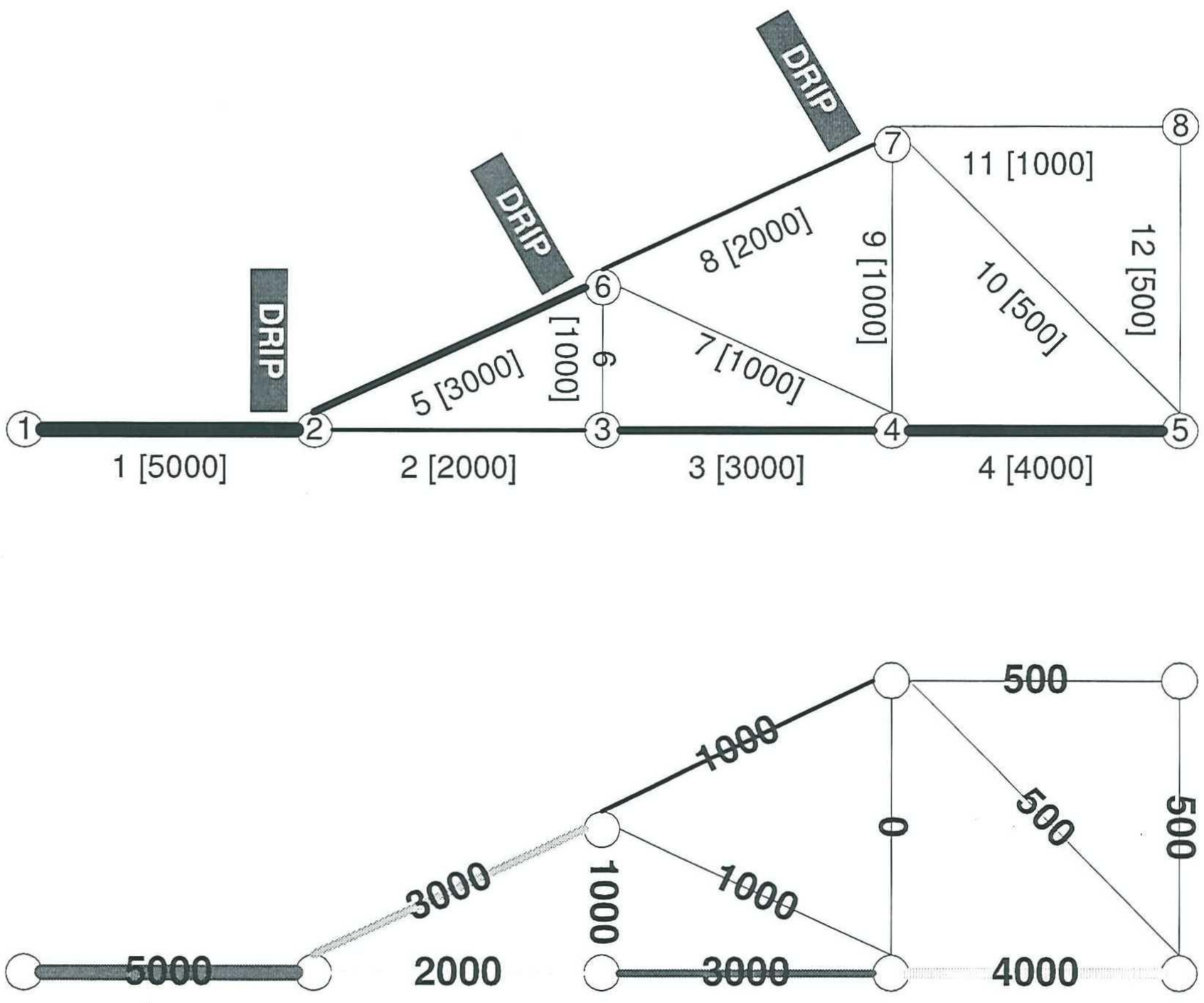


Figure I-7 Test network to verify optimization process. Network characteristics (above) and optimal flow (below)

In the lower half of the figure we illustrate the routing scheme which can be considered as optimal. We see that DRIP 1 has to split the flow over links 2 and 5. DRIP 5 has to split the flow over links 6, 7 and 8 and DRIP 8 has to split the flows over links 10 and 11. If traffic would be routed this way, the full OD demand would be satisfied and no congestion effects would occur in the network.

We start by listing the most important characteristics of the network, the simulation settings and the optimization settings before discussing the results.

Network settings
The network consists of eight nodes and twelve links. All links all have a jam density of $k_{jam} = 125$ [veh/km] and a maximum speed of $v_{\max} = 120$ [km/hr]. The different link capacities are drawn next to the links between brackets and links (1,2,3,4,6,11,12) all have length 10 [km], links (5,8,7) have length 12 [km], link 6 has a length of 6 [km] and link 10 has a length of 14 [km].

Demand
The OD demand is 5000 [veh/hr] wanting to travel from node 1 to node 5.

Simulation settings
The simulation time step $dt = 60$ [sec]. The simulation duration T is 3 hours and there is a total number of 12 periods defined. Based on the time step and formula (2) from this appendix we determine the cells to have a length of 2 [km].

1.2.1. The calibrated situation

When we apply the model to the calibrated situation and use a path alpha of 0 we see the following result of the assignment using the periodic stochastic route choice process and the settings listed above.

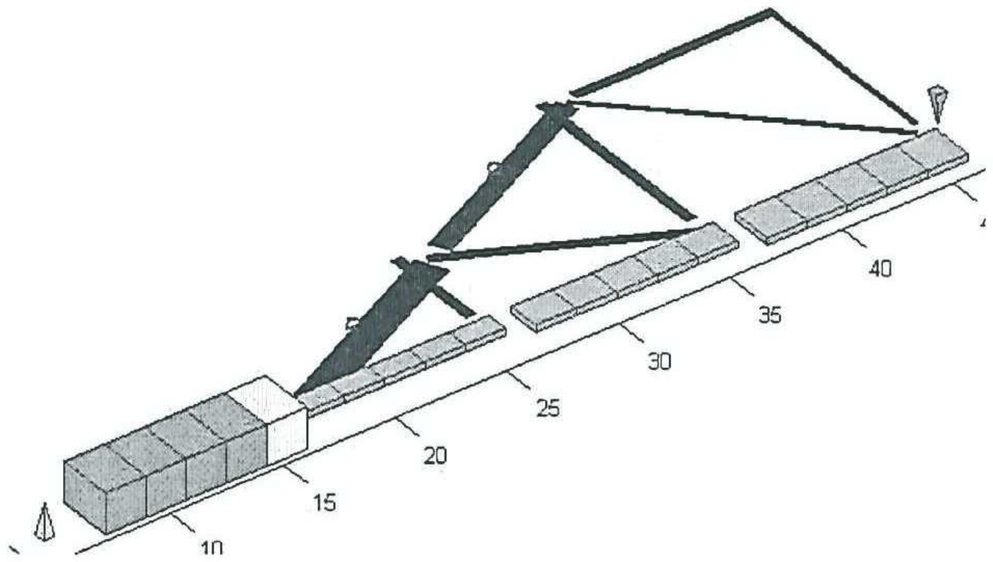


Figure I-8 Assignment using periodic stochastic route choice process

This is exactly the kind of behavior we would expect. Note that by changing the path alpha parameter to higher values, would lead to other links being used as well, namely link 5. However the resulting assignment would still be far from optimal. The figure shown here is at the end of the three hour simulation process and the assignment pattern remained unchanged during the twelve periods.

With the objective function defined as:

$$F_{OBJ} = \frac{\int_0^T \sum_O \sum_D ODdt}{D_d - A_d} \tag{9}$$

(see thesis 5.3) the value for this typical assignment would be: 99.8097, and functions as the reference value to compare future assignments with.

1.2.2. Optimizing the verification network

Finally a first real optimization will be made and in order to do so, we use the default settings for the evolutionary algorithm as described in chapter 6. We start by loading the calibrated solution into the optimization algorithm and *severely mutilate* the gene data to diversify the pool. Then we started the optimization process where each generation consisted of 8 chromosomes. Due to the small scale of the problem, the chromosomes where calculated one after each other because only one client was used which did about 70 generations per hour.

The process led to the following *pattern of evolution*.

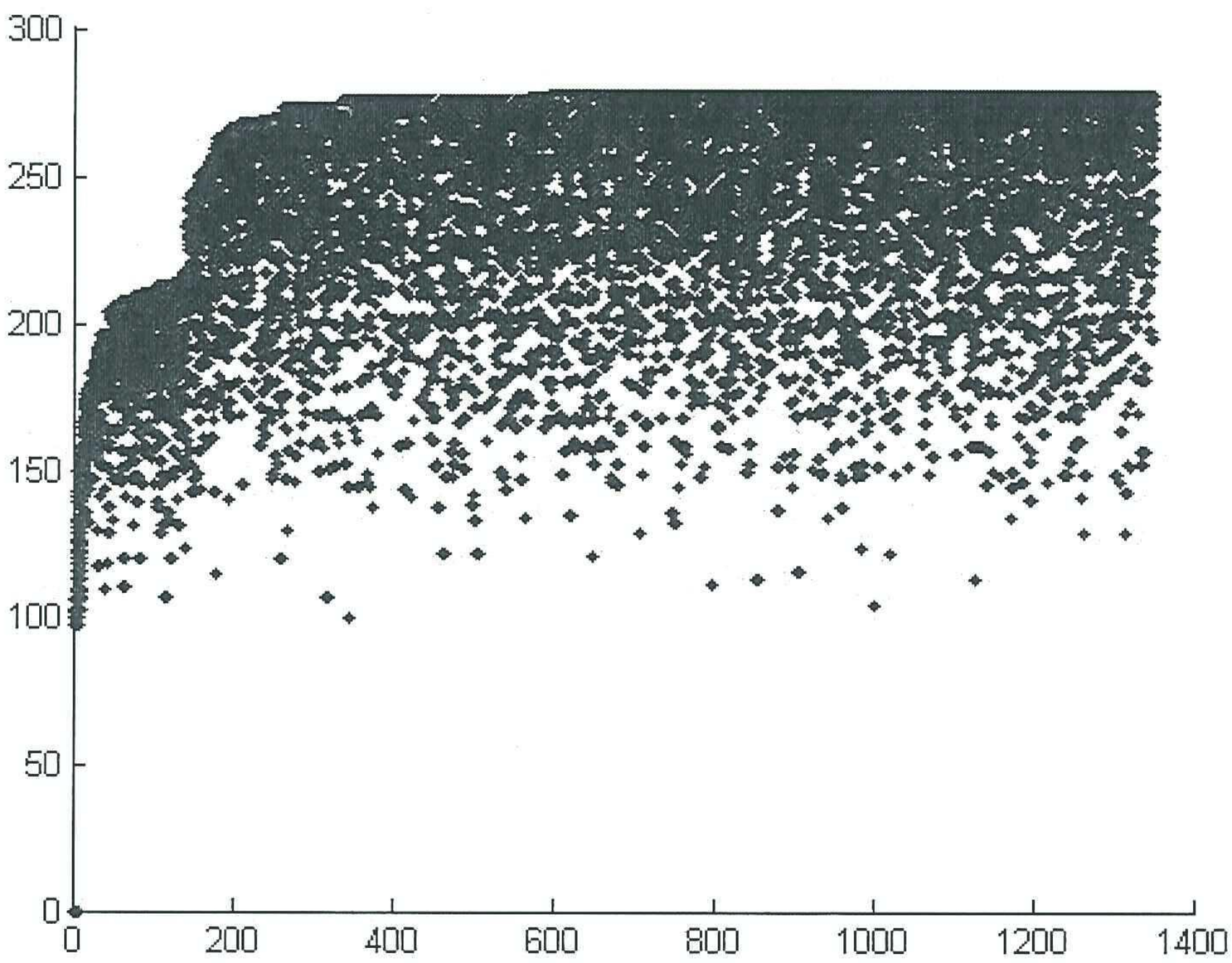


Figure I-9 Pool evolution during the optimization of the verification network

In the figure above we have plotted the scores for every chromosome in the POOL on the y axis from generation 1 to generation 1300. If we look at the figure we see that the collection of POOL scores increased significantly in the first 300 generations. It was during this period that the most progress was made in terms of efficient DRIP settings. Since generation 600 no new better chromosomes have been developed and evolution has stopped.

Now we have shown how the average POOL scores increased during the first 600 generations until all development stopped. Would this mean we have attained the final OPTIMAL state of DRIP settings, thus proving our methodology to be plausible?

The resulting *assignment after optimization* (AaO) is shown in the figure below which was constant for the whole period.

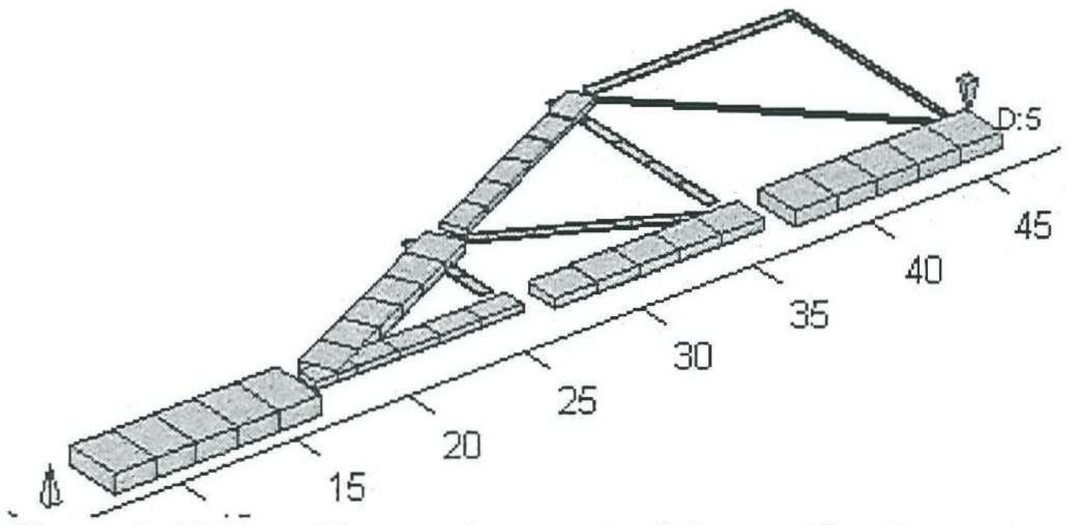


Figure I-10 Resulting assignment of the verification network using optimized DRIP settings

As we can see, the resulting assignment is quite good when visually compared to the calibrated normal stochastic route choice case. The score for this assignment was 277.4793 and can thus be interpreted as a 177% increase, however this would be very superficial.

To put things to the test, we determine the actual split fractions which have been generated for the DRIPS and compare them to what they should be. The latter can easily be deducted from the desired capacity per link displayed in figure I-7.

In the following pictures we will draw the split fractions which have been generated for the three DRIP locations. In green the new generated split fractions are displayed pointing upward whereas the old split fractions used in the calibrated situation are displayed in black and facing downward. The way the links are connected to a node, and drawn into the *split fraction scheme* is determined by the link reference number and should be interpreted from left to right. (smallest outgoing link number is displayed by the left arrow and the highest outgoing link number by the right arrow) The settings are drawn twelve times for every period starting from top left to bottom right.

To make things simple, we have displayed the *layout* of the split fractions belonging to the three different DRIPS in the verification network. In this picture we see the arrow pointing toward the corresponding link numbers, which can be verified manually by the network picture displayed earlier on in this paragraph.

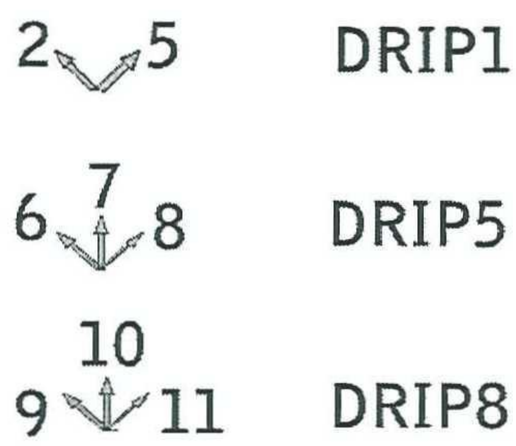


Figure I-11 DRIP Layout

Drip 1:
In the calibrated case all travelers coming from node 1 followed a straight line from link 1 – 2 -3 – 4 until they reached node 5. This is correctly displayed in the single black arrow, indicating that all travel from link 1 always took the same next link (2) for every period.

Based on the connected links and their capacity we can conclude that the optimal split fraction should be [0.4 0.6] ([2/5 3/5]) which we can conclude has actually been generated for every period except the first!

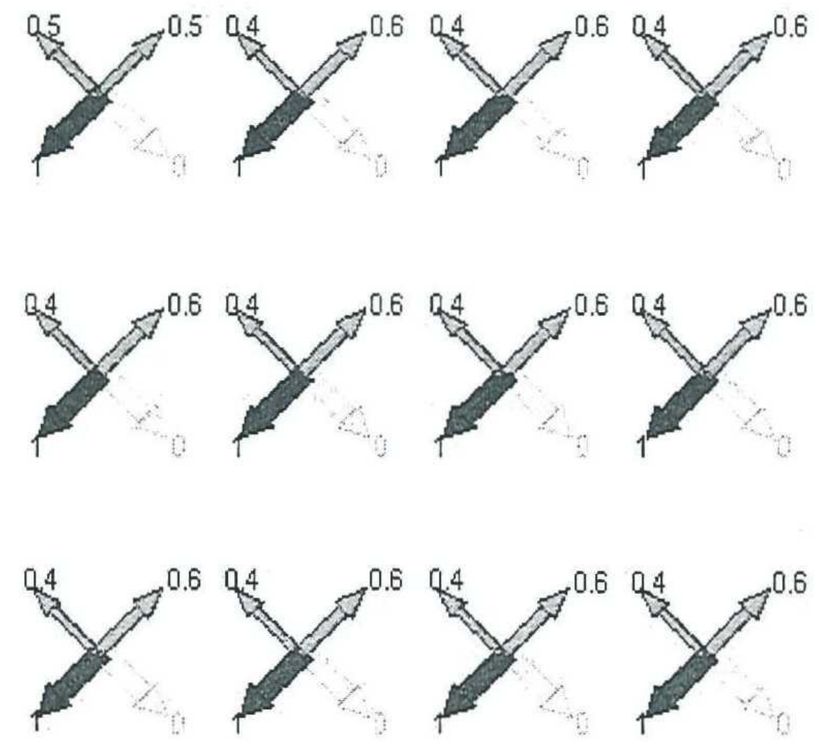


Figure I-12 DRIP 1

Drip 5:.
In the calibrated case none of the travelers actually passed this DRIP. If they would have, their shortest path would lead them to take link 7 which always was the case (black)

Based on the connected links and their capacity we can conclude that the optimal split fraction should be *[1/3 1/3 1/3]*.

Since our resolution is 0.1 this particular split fraction can never be calculated. However if we look at the network we can see there is only one link which could take more than 0.3 part of the incoming flow of 3000 and that is link 8 (most right).

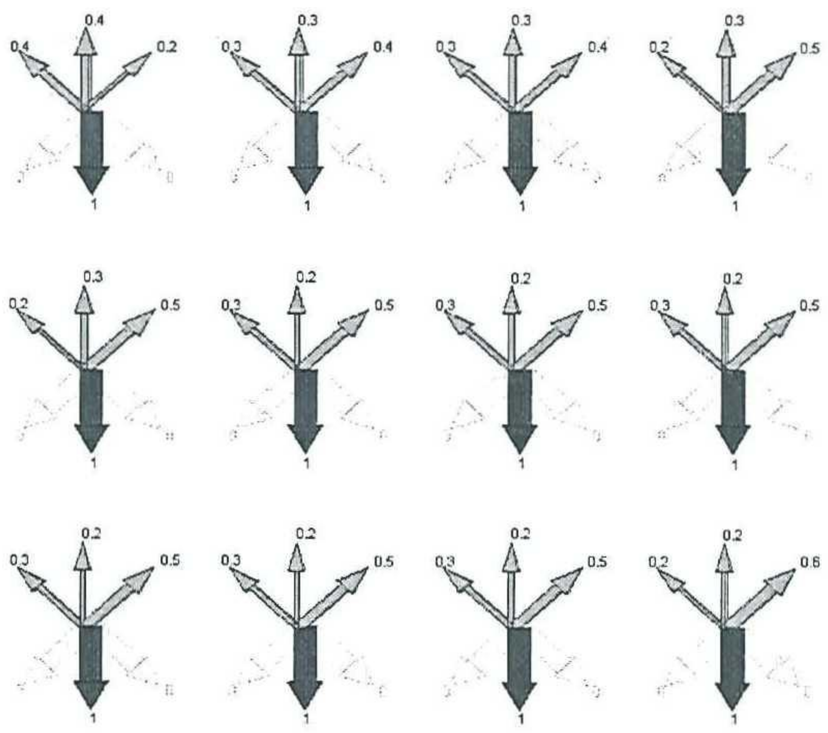


Figure I-13 DRIP 5

We can conclude from the picture that links 6 and 7 (the most left and middle arrows) never receive more than 0.3 part of the incoming load of 3000, rendering their usage never above capacity. Except for the first period one might say since the SPF is 0.4, however if we look at the settings for DRIP 1 we see that the total load during this first period was actually 0.5 * 5000 = 2500. So apparently since the optimal situation could never be achieved the algorithm developed these settings as second best.

To the right we now display the optimal settings calculated for DRIP 8.

In the calibrated none of the travelers actually passed this DRIP. If they would have, their shortest path would lead them to take link 10 which always was the case.

Based on the connected links and their capacity we can conclude that the optimal split fraction should be [0 1/2 1/2]. However this would be the case if all capacity would be used because the previous DRIP settings would spread the traffic by 1/3 over each outlink. Since this was impossible due to our own set restriction, what is best now?

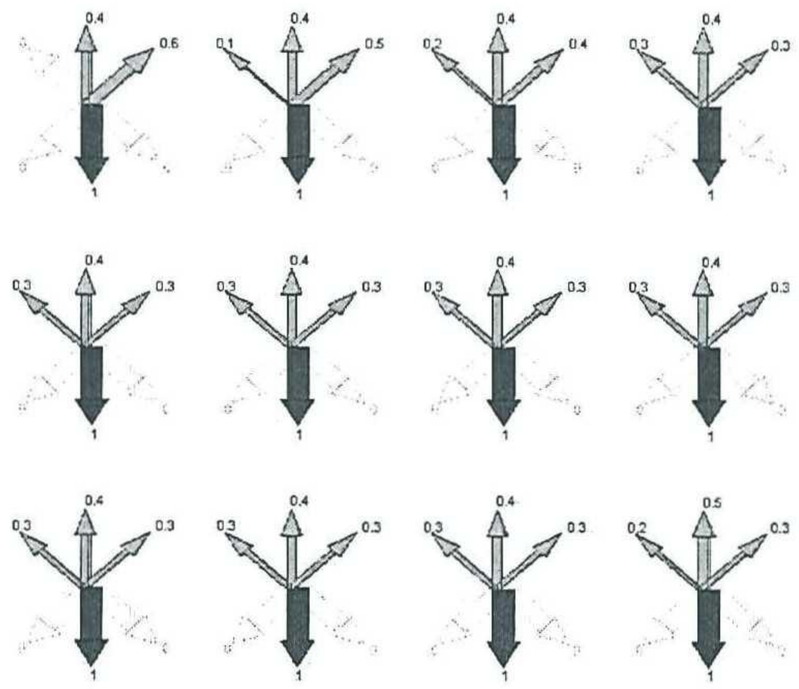


Figure I-14 DRIP 8

Due to the settings on DRIP 5, there is some capacity left which could be used to route the traffic. If we examine the settings from DRIP 5 we can conclude that the fraction pointing toward link 6 or 7 was combined never more than 0.5. Since the incoming flow was 3000 (DRIP settings 1) we could calculate there is a spare capacity of 500 left on link 4. If we look at the settings of DRIP 5 we can also conclude that the incoming flow toward DRIP 8 is approx. 0.5 * 3000 = 1500 [veh/hr].

So what is optimal under these conditions? Well lets see. The left arrow points toward link 9 and has a capacity of 1000, the flow it receives is determined by its split fraction which is mostly 0.3, rendering the load on this link 0.3 * 1500 = 450 [veh/hr], just below the maximum free spare capacity of 500 [veh/hr]. The fastest route to the destination is the middle link but has a maximum capacity of 500 and as we can see the split fraction is almost every period 0.4. This would mean a 0.4 * 1500 = 600 [veh/hr] load. Apparently there should be some congestion noticeable, although relative small.

In the picture to the right we have drawn node 7 together with the cells in more detail. The congestion effect is *noticeable* due to the slightly elevated last cell of link 8. (The height represents density, whereas color represents maximum speed)

Since the extra demand is small, its effect on the density c.q. queue is small as well.

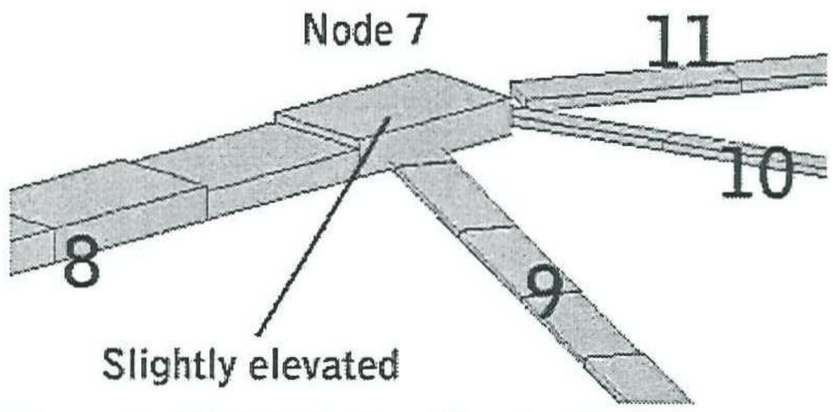


Figure I-15 detail of spillback effects

Conclusion

Due to a badly chosen example, we where unable to verify whether the theoretical optimal situation was a 100% attained, yet the calculated DRIP settings seemed very logical under the given network properties. In addition, the fact that the optimization algorithm was able to overcome difficulties like the badly chosen split proportions and come up with a *way around* should spark some interest and confidence in the *generic* nature of the process.

1.2.2.1. Another example

Since the previous example did prove the generic nature of the optimization algorithm and showed its ability to work *around problems*, we quickly provide another test case to see how close the optimal situation can actually be achieved.

Our new network is depicted to the right.
The idea is that 6000 [veh/hr] will want to travel from node 1 to node 8. To do so, we have set up three DRIPS in the network at links a, c and g. (DRIP 1, DRIP 3, DRIP 7)

What we see is the result at *the end of the assignment of the calibrated situation*. The dominant route that was chosen was a-c-i-k. Due to the limited capacity of I, spillback effects occurred on link c. At the 9th period these effects became so severe that the alternative to travel via a-d-f-k (despite the limited capacity) became dominant. At the beginning of the twelfth period, the congestion had dissipated and the route a-c-i-k became dominant again. However the simulation took only twelve periods and therefore stopped with the traffic being halfway link c.

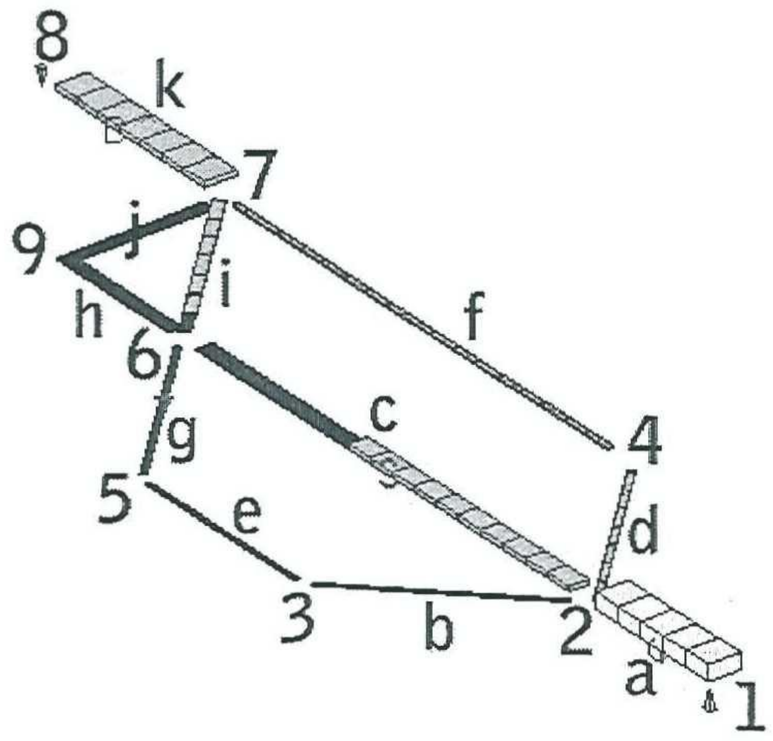


Figure I-16 Verification network 2

Again we use the default settings of the optimization algorithm, a time step of 60 seconds, a simulation duration of three hours and twelve periods.

POOL evolution
From the picture to the right we can determine that most of the *evolution* in the DRIP settings took place during the first 200 generations. From that point on no real progression was booked in terms of the absolute value of the objective function.

We will continue by discussing the resulting assignment based on the highest scoring solution in the POOL. For our reference the calibrated solution scored 59.88 whereas the highest solution score 90.56. (an increase of 51%)

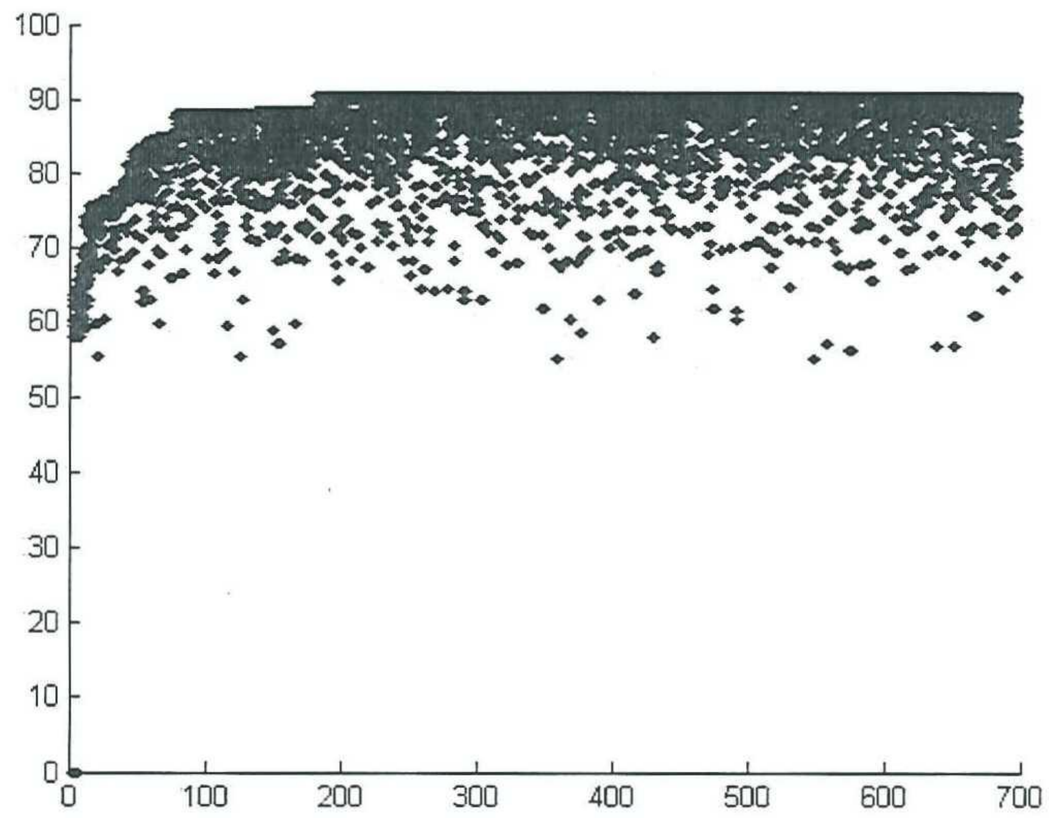


Figure I-17 POOL evolution

Resulting assignment
To the right, the resulting assignment has been drawn. As we can see, all links are used and no congestion is noticeable.

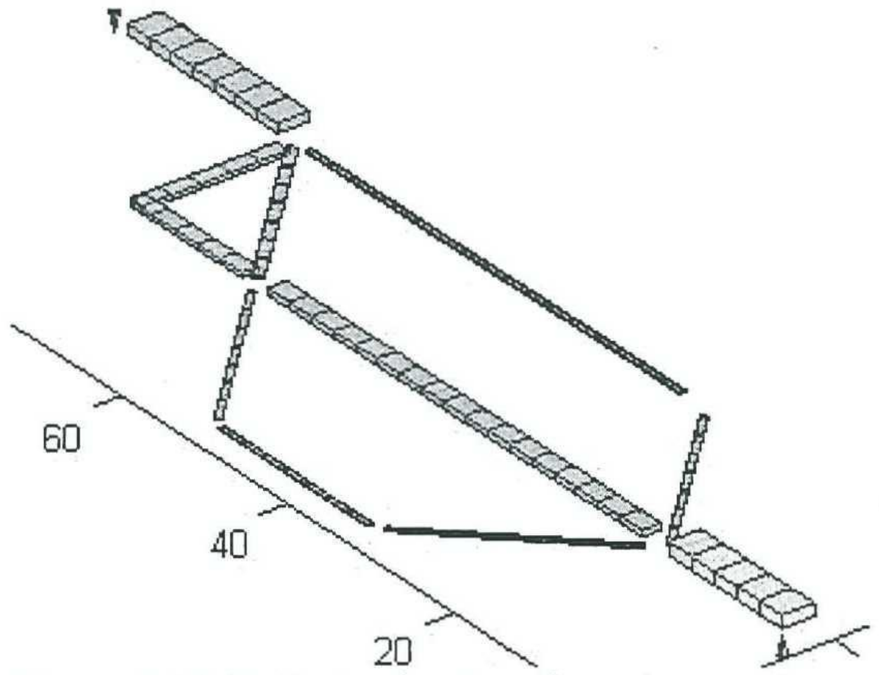


Figure I-18 Optimized network assignment

Before we continue, it is important to note the difference in capacity between the links. Link a has a capacity of 6000, link b-c-d respectively 1200-3600-1200. This means that the desired split fraction for DRIP 1 should be [0.2 0.6 0.2]

Link h and I have a capacity of 3000 and 2000 which limits the possible DRIP settings for DRIPS 3 and 7.

In the figure below we have displayed the resulting calculated DRIP settings.

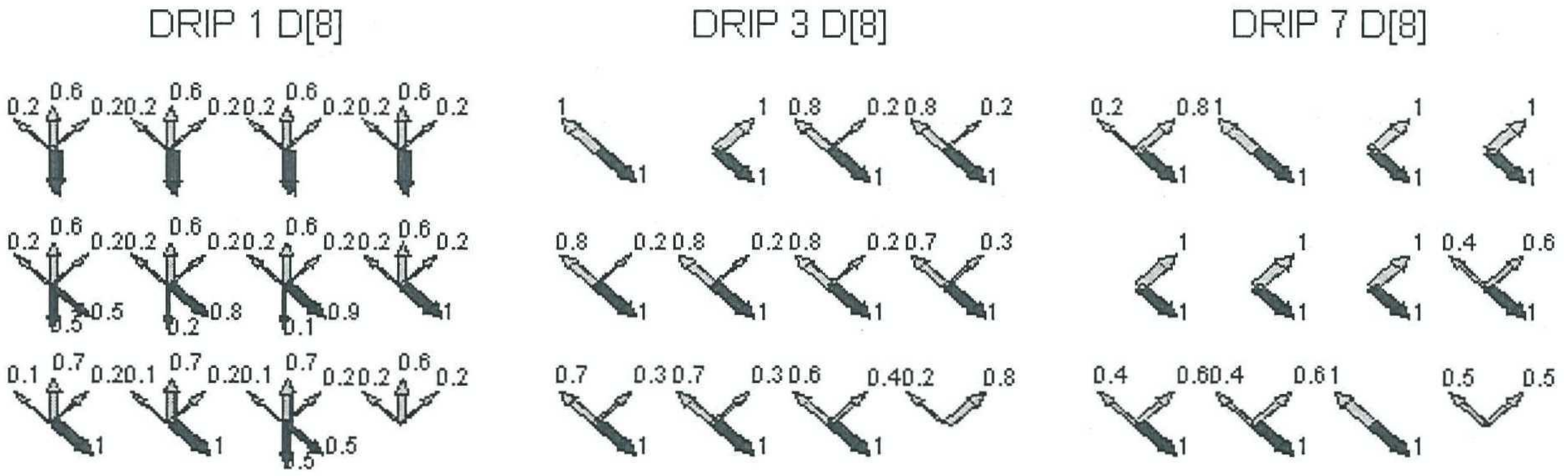


Figure I-19 Overview of DRIP settings verification network 2

Link 1 has for almost all periods the desired split fraction [0.2 0.6 0.2]. (Except for three periods where some deviation occurs which leads to small spillback). The combined effects of DRIP 3 and DRIP 7 heading toward link I (the arrow pointing toward the right) should never be more then 2000 [veh/hr], fortunately this is never the case (at best 1920 [veh/hr]). The capacity of link h is 3000 which is equal to the maximum assigned load (0.4 * 1200 + 0.7 * 3600). The DRIP settings of the last period of DRIP 1 do not effect the objective function (travelers passing DRIP 1 will never reach their destination during one period), as do the DRIP settings of the first period of DRIPS 3 and 7. In all we can say that there are three non optimal advices given. These are for DRIP link 1 during period 9, 10 and 11. But still, they affect 10% of the travelers during 25% of the time.

Conclusion:
Though the generated route guidance advices may not always seem logical in the sense of continuity and simplicity, they do always have the desired effect in terms of optimizing the objective function and creating an assignment without congestion. The algorithm is able to generate the correct advices but needs quite a few generations to smooth them into the optimal values. (this can de deducted from the typical hyperbolic curve shaped evolution). Although some non optimal advices are generated as well, the bigger picture does improve significantly!

Appendix II. Optimized DRIP Settings for the case study

In the following pages, we will graphically depict the *prescriptive route guidance* which should be disseminated by the DRIPS in order to optimize the assignment.

Perhaps superfluous we, stress the fact that these *guidance advices* where calculated under the assumption that $\eta^{compliance} = 1$; *all* travelers passing a DRIP and heading for one of the destinations follow the advices. And remember that the *prescriptive route guidance advice* is a direct translation of the *optimized destination specific split fraction* as described in 5.6, we therefore only depict the split vectors since they are easy to interpret.

In the tables below, the actual generated *destination specific split fractions* at the DRIP locations are presented graphically in green (arrows pointing up). The calibrated split fractions are printed in black (and pointing down). By comparing these two, the difference between both is more easy to interpret.

The advice should be interpreted in terms of left and right, if one was to pass the DRIP and the there would be a split value pointing to the left, this would mean one should access the ring way in a clockwise manner. (whereas split vector values pointing to the right mean traveling the ring way CCW). For the two DRIPS who can guide travelers through the city as well, an arrow pointing straight up means traveling the city route.

The DRIPS are addressed *row wise* in the table below (the same order as described in *figure 9-1*)

Table II-1 DRIP link numbers and the location in the network

DRIP 320	1; Klein polder plein	DRIP 456	4; Vaanplein
DRIP 186	2; Terbregseplein	DRIP 346	5; Benelux
DRIP 450	3; Ridderster	DRIP 416	6; Kethelplein

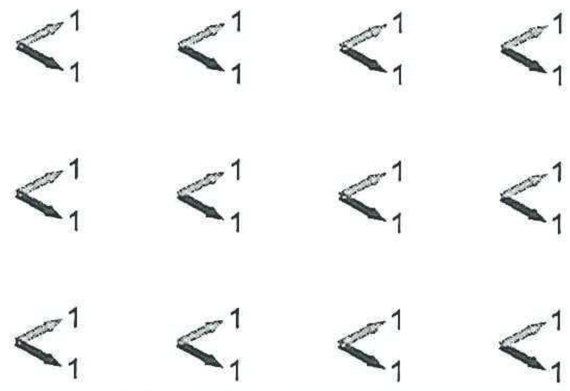
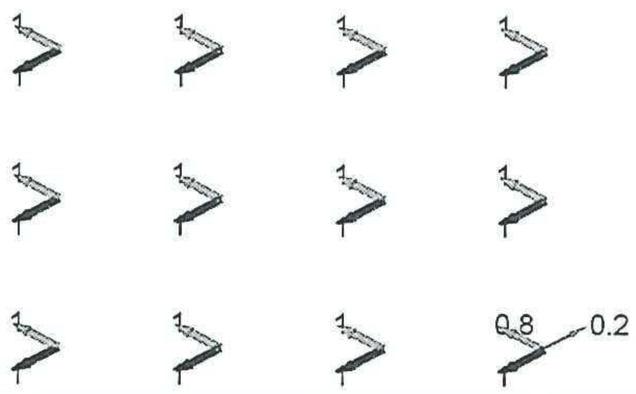
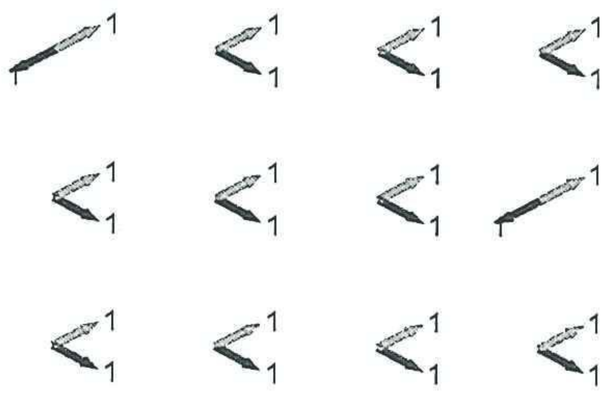
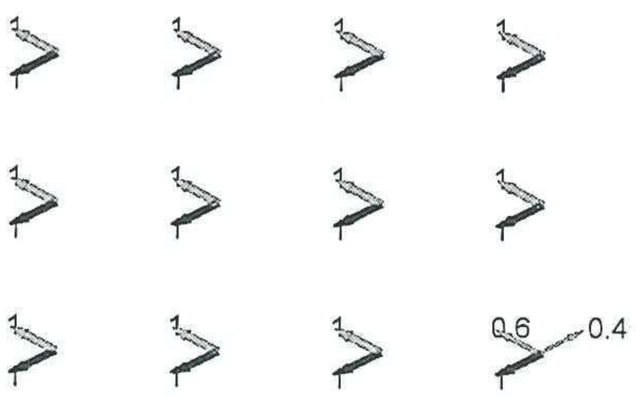
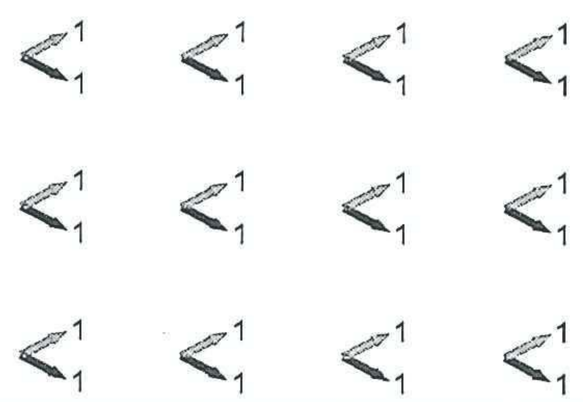
The three columns represent the three *destination groups* for which guidance is calculated. The figures are annotated with the following information:

- DRIP ###; describing the link number at which the DRIP is located
- [destination nodes]; describing the collection of *individual destination nodes* who make up for a *group* for which the DRIP information is generated
- scalar fraction values. For all splits, the scalar value is printed as well

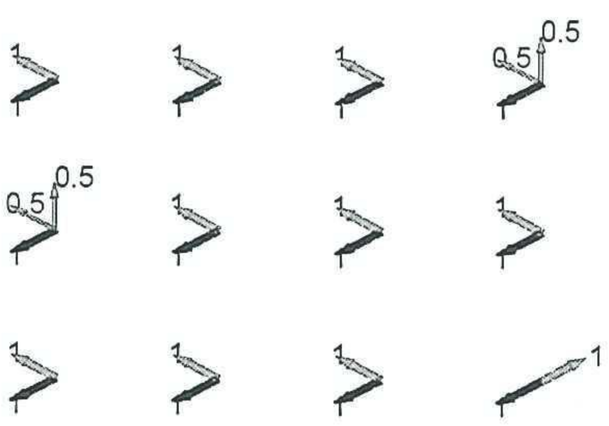
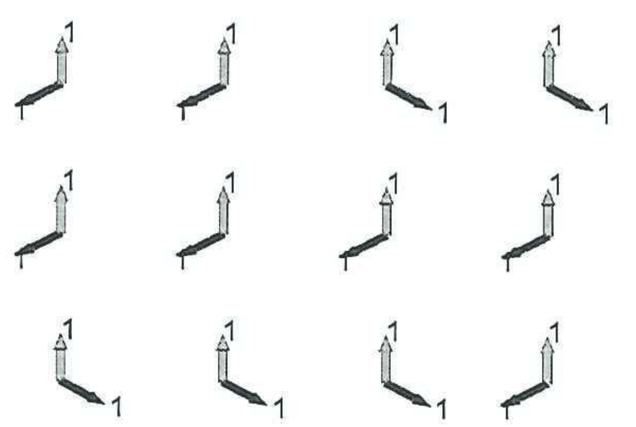
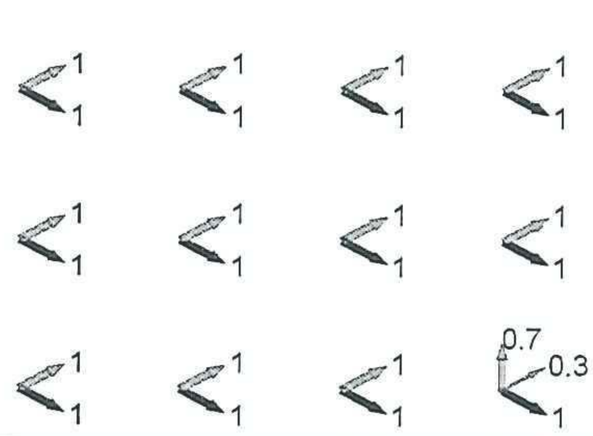
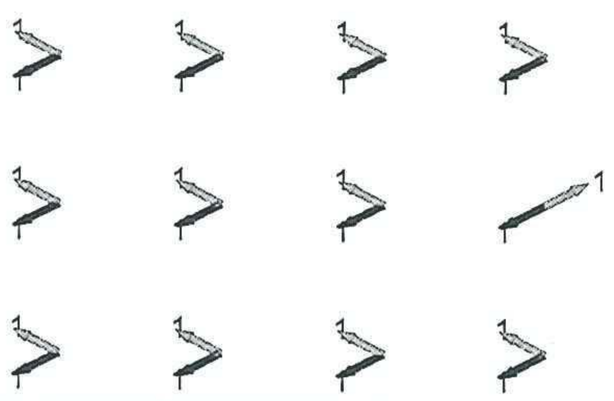
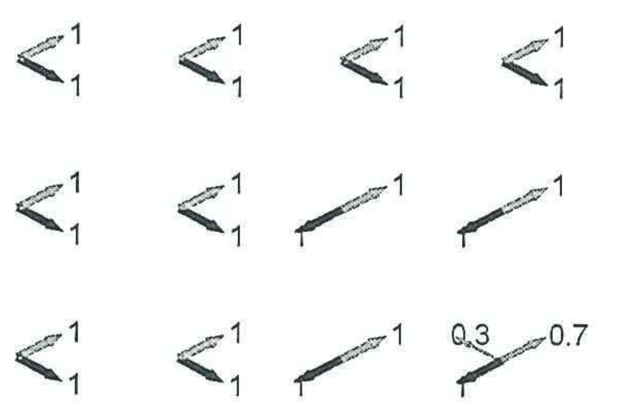
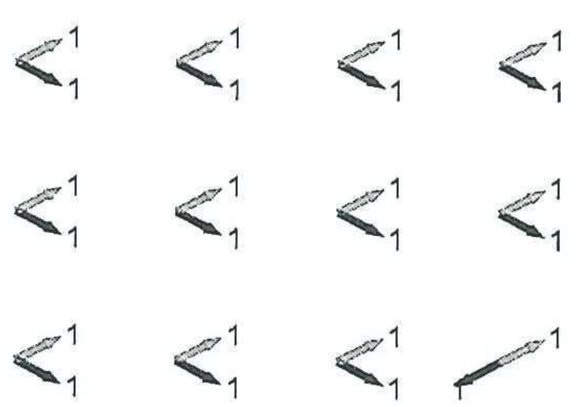
The simulation duration of three hours was divided into twelve periods of 15 minutes. For each period the settings are given and should be read from *top left* to *bottom right* in a normal reading fashion.

1.1 DRIP Settings scenario 0; Recurrent morning congestion

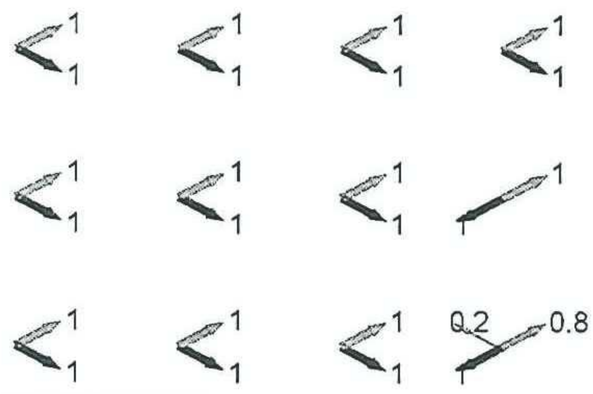
DRIP 320 D[57 56]	DRIP 320 D[233]	DRIP 320 D[234 60]
DRIP 186 D[234 60]	DRIP 186 D[233]	DRIP 186 D[236 3 4]
DRIP 450 D[66 64 238]	DRIP 450 D[62 28]	DRIP 450 D[236 3 4]
DRIP 456 D[62 28]	DRIP 456 D[236 3 4]	DRIP 456 D[66 64 238]

DRIP 346 D[57 56]	DRIP 346 D[62 28]	DRIP 346 D[66 64 238]
		
DRIP 416 D[57 56]	DRIP 416 D[66 64 238]	DRIP 416 D[233]
		

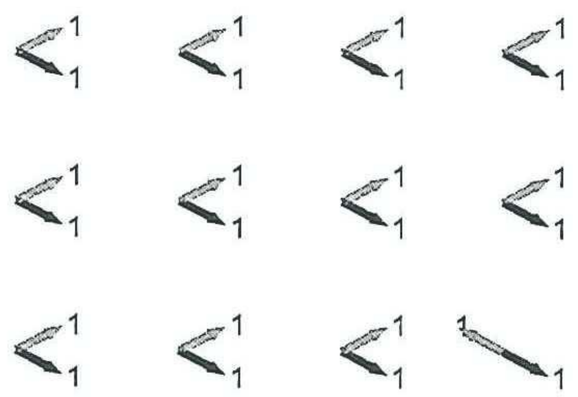
1.2 Drip setting scenario 1: Extreme demand at the kuip

DRIP 320 D[57 56]	DRIP 320 D[233]	DRIP 320 D[234 60]
		
DRIP 186 D[233]	DRIP 186 D[234 60]	DRIP 186 D[236 3 4]
		

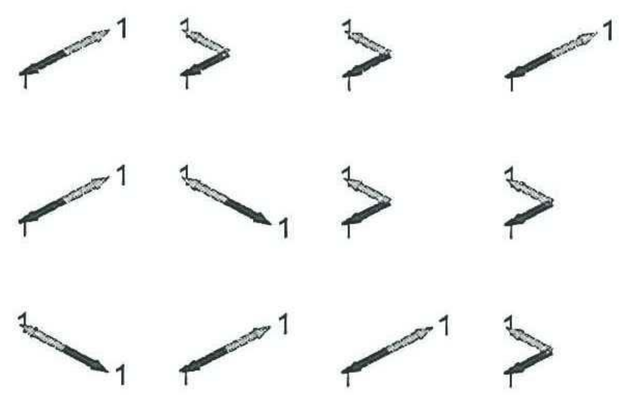
DRIP 450 D[62 28]



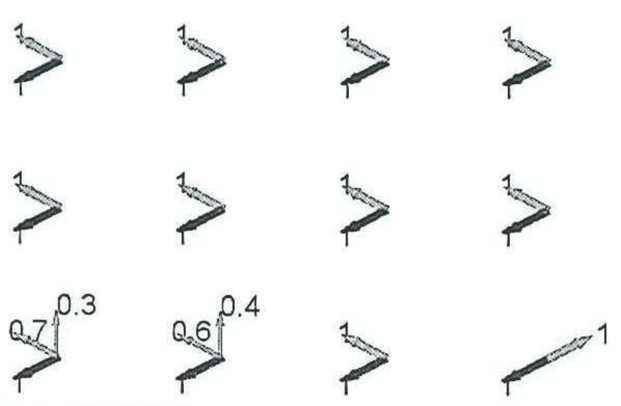
DRIP 450 D[66 64 238]



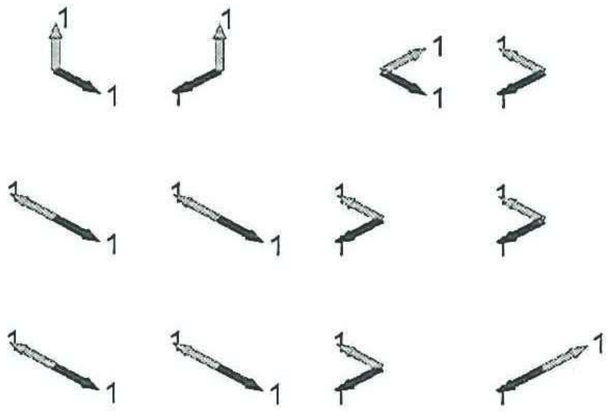
DRIP 450 D[236 3 4]



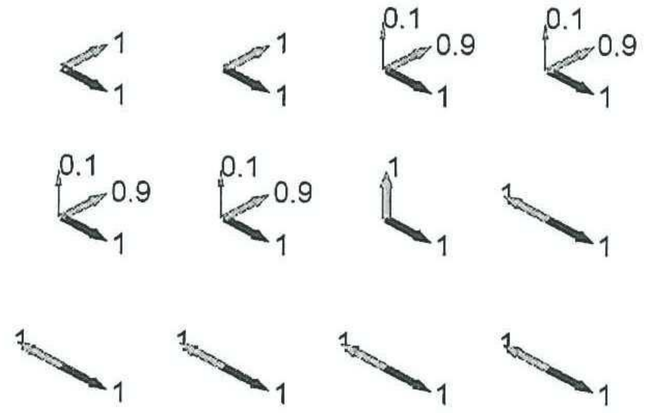
DRIP 456 D[236 3 4]



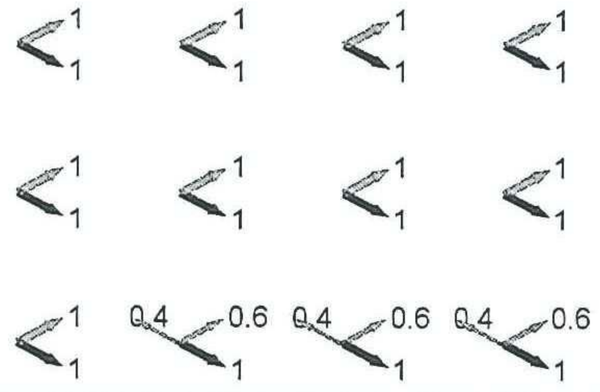
DRIP 456 D[62 28]



DRIP 456 D[66 64 238]

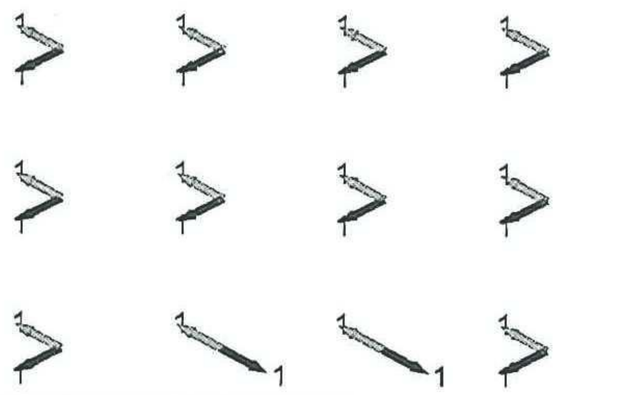


DRIP 346 D[57 56]

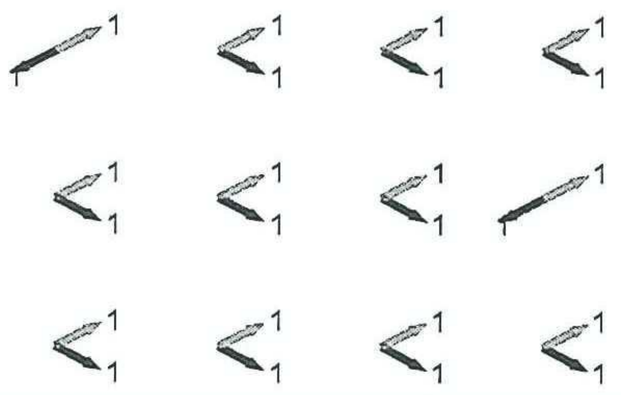


DRIP 346 D[62 28]

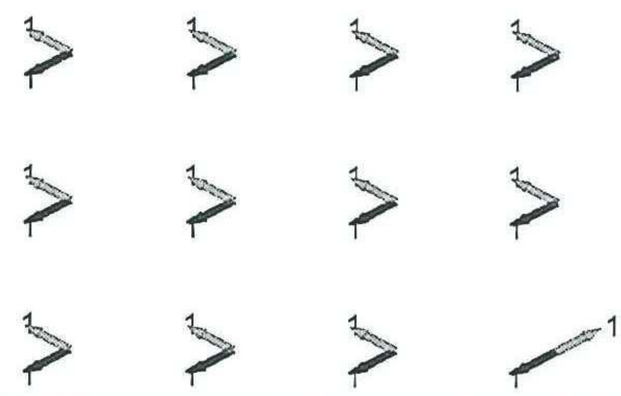
DRIP 346 D[66 64 238]



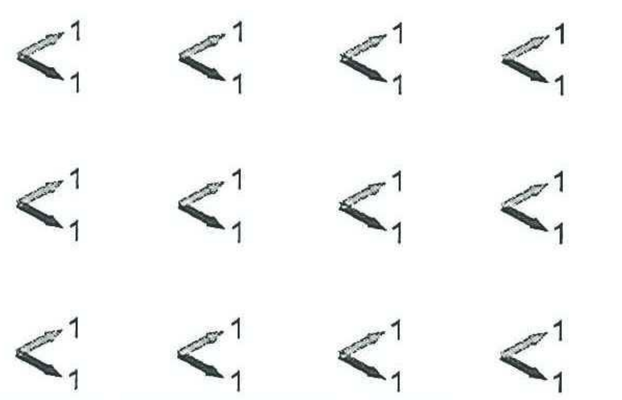
DRIP 416 D[57 56]



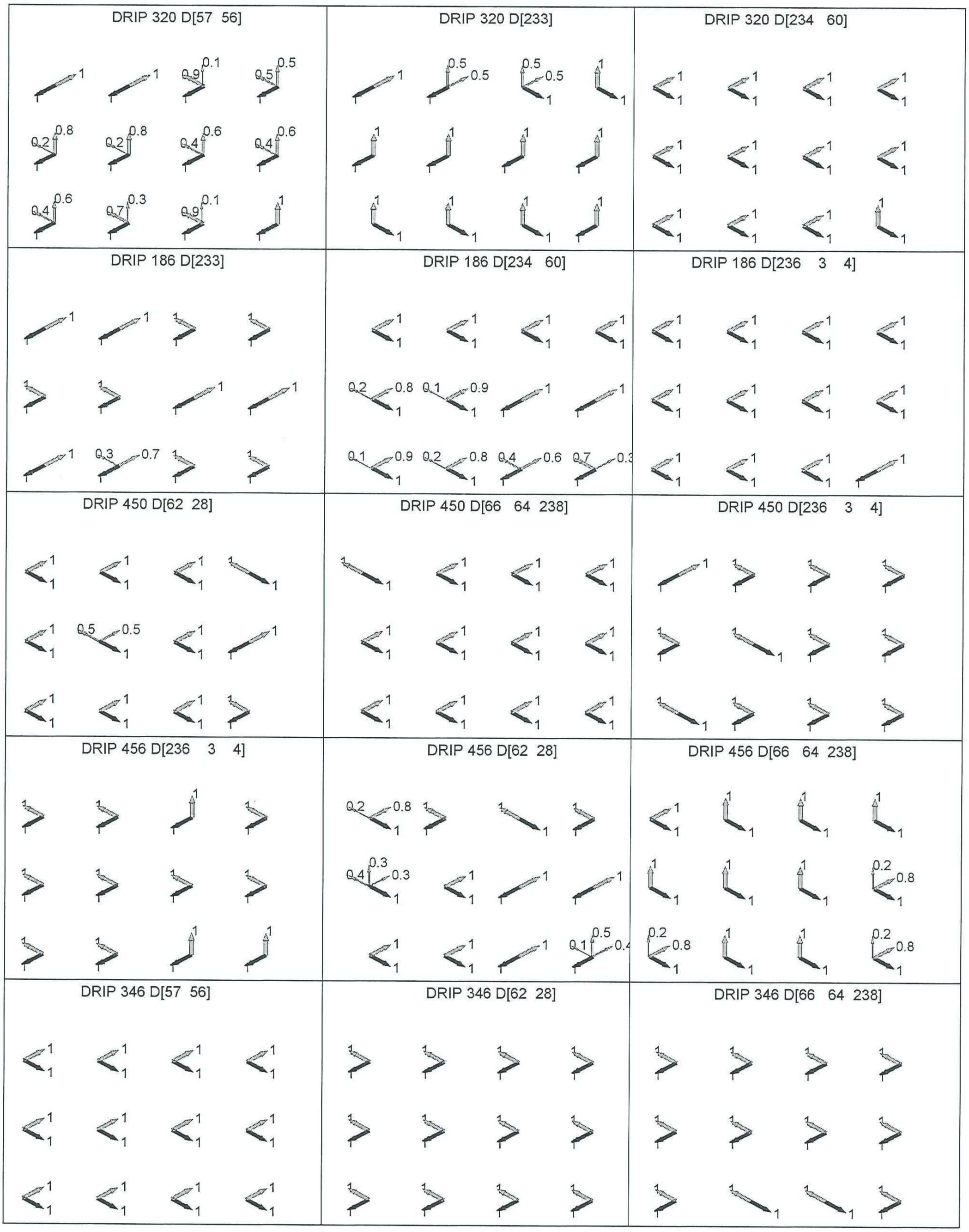
DRIP 416 D[66 64 238]

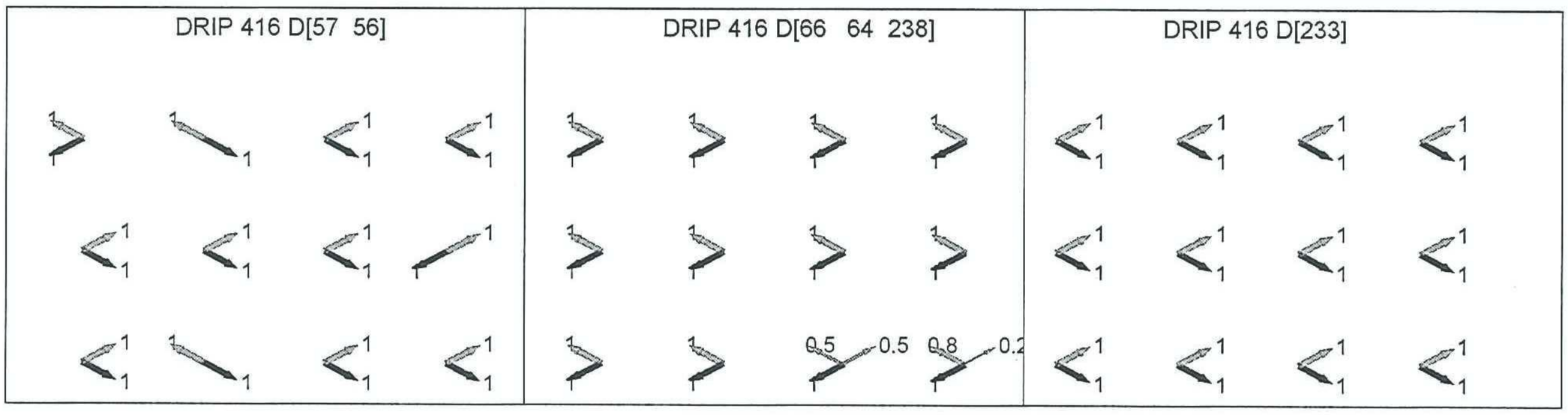


DRIP 416 D[233]



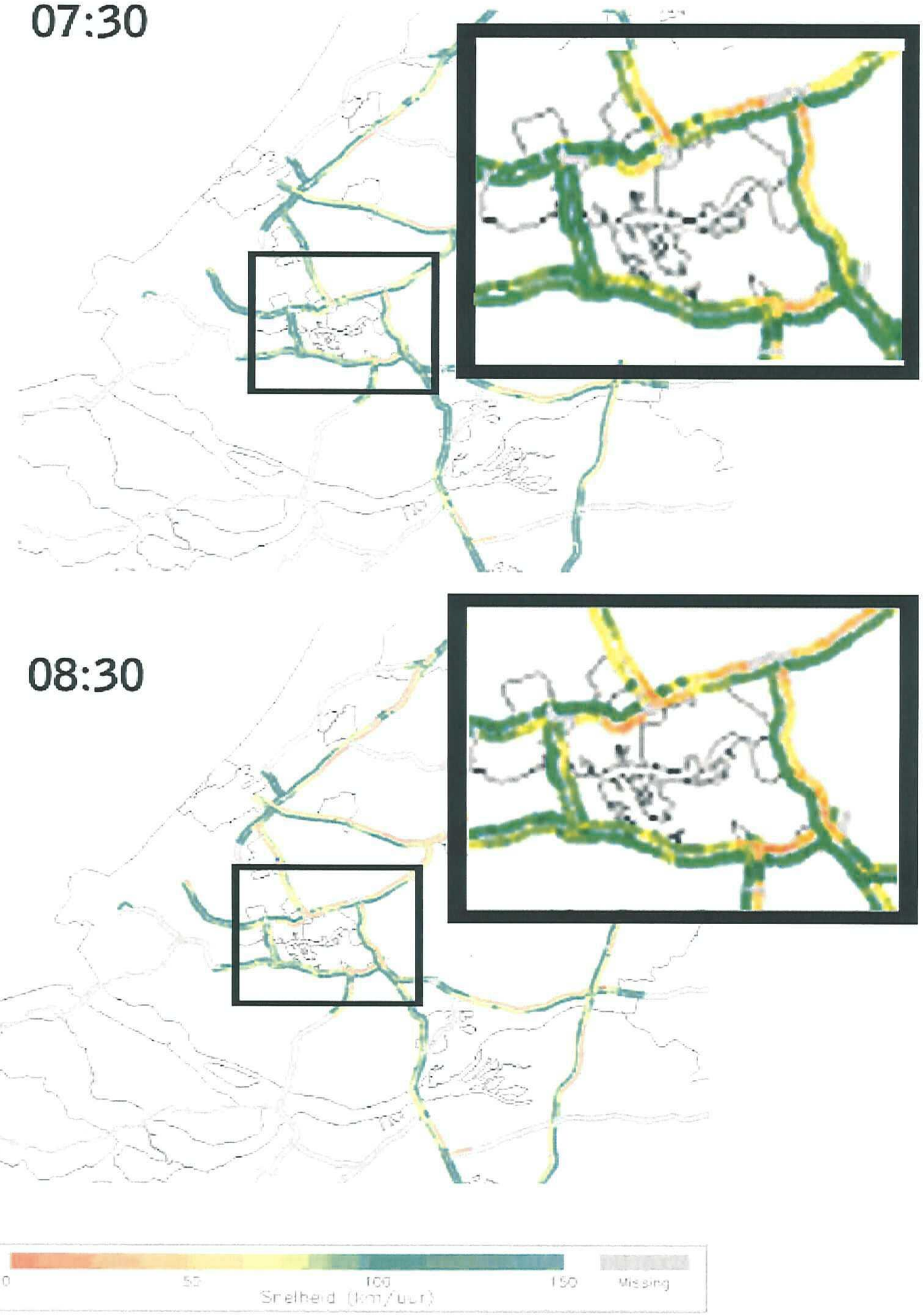
1.3 Scenario 2: Accident at Terbregseplein




DRIP 416 D[57 56]
DRIP 416 D[66 64 238]
DRIP 416 D[233]
0.5
0.5
0.8
0.2

Appendix III: Real life recurrent morning congestion

In the picture below the average speed and location of the recurrent morning peak for an average work day in the south of the Netherlands are displayed at 07:30 and 08:30 hours. These images where used to calibrate the model.



Source: DZH, Rijskwaterstaat, Ministerie van verkeer en waterstaat

Appendix IV: Increasing DSMART speed by reduction of network complexity

The diagram below displays the different steps in the DMSART model used to progress the traffic through the network.

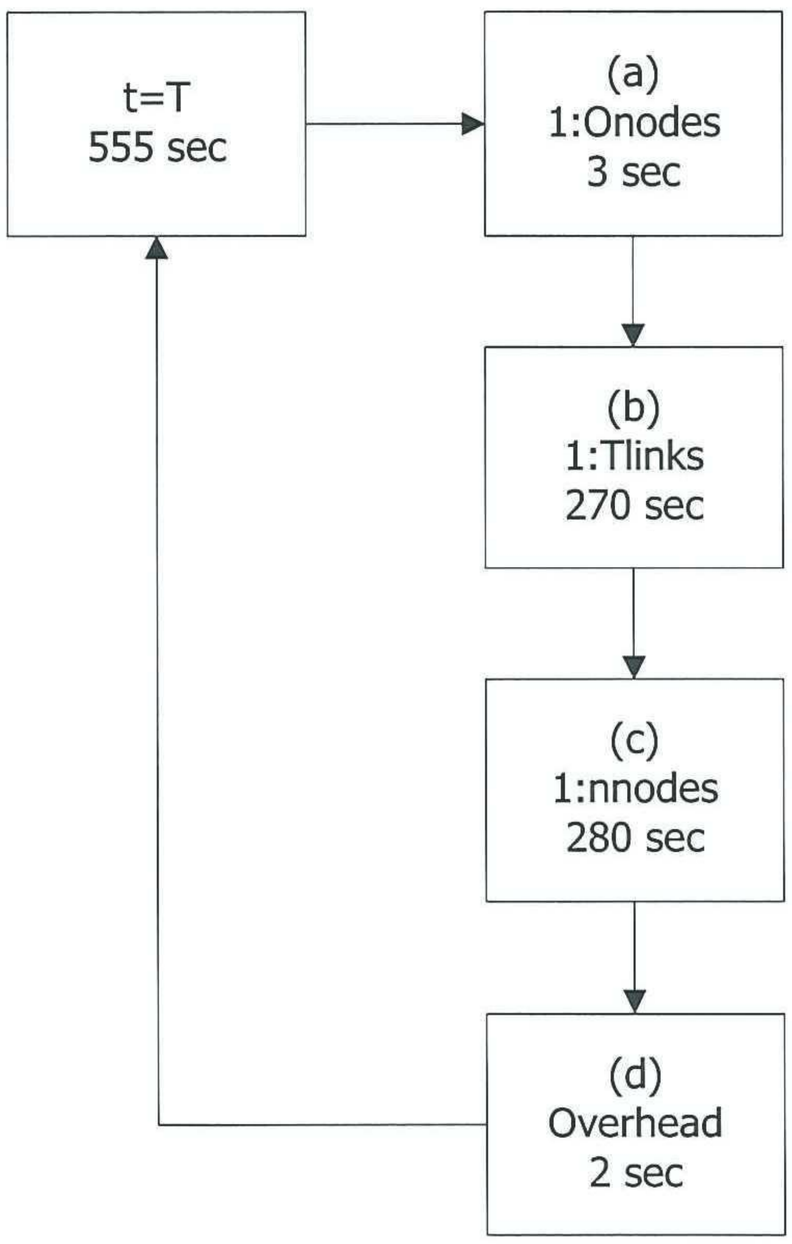


Figure IV-1 Processing time needed for simulation of the Rotterdam network

The total computer load in terms of computation time per process for the Rotterdam simulation is given above and can be expressed by $T = a + b + c + d \approx b + c$. Where (b) is a process which is run every time step dt and consists of one loop addressing all transport links (1 to number of transport links), and (c) is a process which is also run every time step dt and consists of one loop addressing all nodes (1: number of nodes).

In addition, both (b) and (c) have to deal with the *general complexity of the problem* which is the number of destinations in use, or the width of the cells.

If we assume a reduction of:

i. the number of transport links to 70%
ii. the number of nodes to 70%
iii. the number of destinations to 70% (general complexity)

Now since the process (b) and (c) both use about the same computation time, we can reduce the computation time by a factor $dT = \frac{1}{0.7^2} \approx 2$

Appendix V: Hind sight

At this point we are ready to evaluate the overall process which led to the development of the proposed optimization methodology.

Background and literature study
In the first phases, a lot of attention was paid to the *background* of the problem in general and the situation in the Netherlands in particular. It became clear that in current practice DRIPS are fed with simple straightforward descriptive network information. One of the main problems mentioned in literature is *user faith* in the system. The notion that *descriptive network information* on requires network knowledge, and provides a *quantitative base* for comparison between expected and encountered network conditions (which could easily be used by a traveler to justify his/her the faith or disbelief in the system) led to believe this was not the best. In addition, no instances where known which dealt with pure *prescriptive information* on the motorway so the focus shifted toward prescriptive information.

Methodology development
Initial effort was aimed at developing a *smart heuristic,* which led to the investigation of various shortest path algorithms, which would be able to generate DRIP settings. This idea was quickly abandoned and replaced by the control approaches which where researched at that time with their iterative nature and the inevitable need for a DTA model also became clear.

From literature research the difficulty of dealing with *the fixed point problem* or generating *consistent* information led to the conclusion that optimal control would likely provide the best results. Later on the *rolling horizon* or *model predictive control* approach provided a framework for real life implementation of the optimal control methodology.

Custom model
It became very clear early on that the available DTA models to be use would either be *Dynasmart,* M*etanet* or a *custom model*. The idea of working with a third party model was never appealing because assignment information would either have to be reconstructed afterwards or be unavailable at all, however two models where investigated because custom development would mean a lot of extra work. When investigating Metanet it also became clear that if route guidance was to be developed using Metanet, the format of this guidance would also be compatible to Metanet which meant at most bifurcation nodes could be used in the network modeling. When investigating Dynasmart a similar argument arose due to the fixed nature of the way a VMS was implemented. Soon after these conclusion efforts where aimed at developing a custom dynamic traffic assignment model for over 6 months.

In this time the model steadily grew starting from a one link model solved by the Godunov scheme. This one link model was quickly expanded into a *multiple destination model* by adding another dimension to the *cells*. The research by Lebaque led to the idea of introducing a *heading* vector which would enable fast calculations over this extra dimension. Soon after that the *destination specific split fraction* and the *fprod* multiplication where implemented and the model was able to progress traffic. The previously made investigations into *shortest paths* proved not to be a waste and in little time the *Floyd Warshall algorithm* was implemented. After this, a long period of *debugging* the code, and *optimizing* the route choice process and overall speed was due.

Optimization using evolutionary algorithms
Quickly after the notion of an *iterative nature* of the control methodology a *genetic algorithm* was seen as a possible solution in optimizing DRIP settings due to previous experience with this type of heuristic. At first a basic cycle was of a GA was implemented which almost immediately revealed a problem in the way route choices where initially stored in the

DSMART Model. Soon after both the model and GA where adapted into working with the SPF reference structure and SPLIT table and the mapping of the gene data on three dimensions became a fact. With this implementation, the basic cycle of the GA was steadily expanded by means of more and more operators. The symbiotic nature between the model and the GA was exploited more and more when the chromosomes started carrying information about the link activities and the free flow shortest paths. By now the optimization technique could be better characterized by the term *evolutionary algorithm* and more and more additions where added while always keeping the stochastic nature in mind (hence the chance vectors). This initial combination of DSMART Model and EA was used to optimize *all split fractions on all links* in small test networks. During this process a lot of problems manifested themselves in a dramatic way because the optimization was aimed at all links. As a result the problems could easily be identified and attempts to solve them where made. It was during this phase that the *super period*, *the split fraction enhancement*, *multiple mutation operators* and *active genome selection* was developed.

Case study

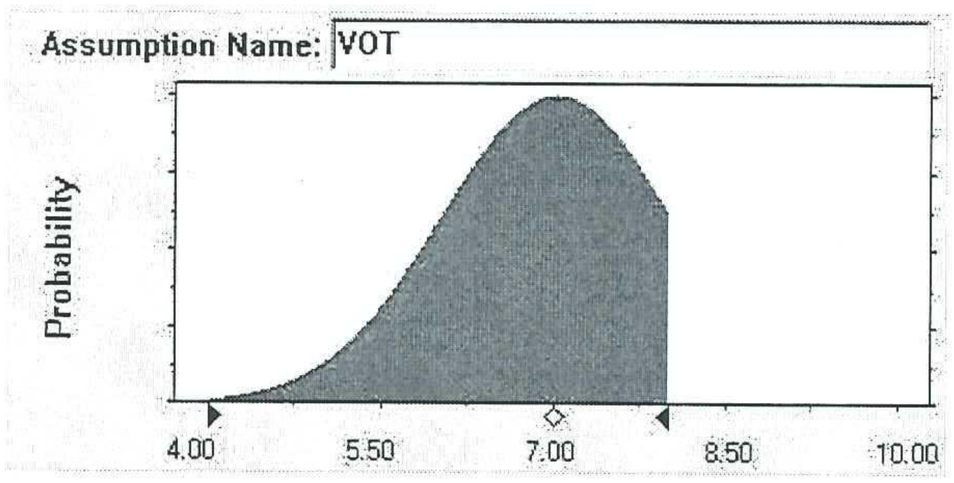
By now the DSMART Model and EA where mature and efforts where directed at developing a Rotterdam network model and calibrating this. During this process the *zuidvleugel network model*, which was made available by DHV, was simplified and translated into the format necessary to be implemented in the custom built DSMART Model. This proved an iterative optimization process itself because it took about 50 versions before one could speak of *a crude calibration*. Finally when the network model was built as well, the EA was adjusted in order to work with a fixed number of DRIPS instead of all the links. Previously developed features like active genome selection where obsolete and a smart way of selecting gene data belonging to the DRIPS was developed. By now, the first attempts to optimize the Rotterdam network proved successful and efforts where aimed at finding the correct EA settings.

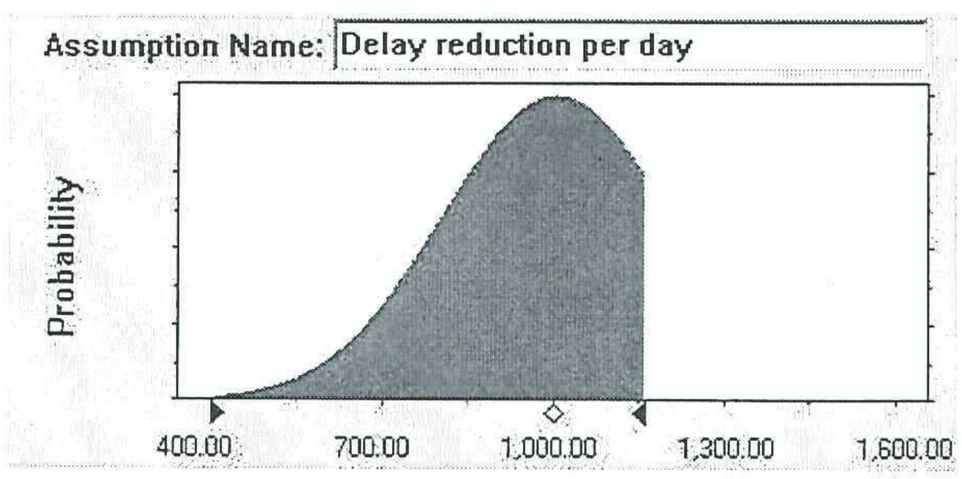
Appendix VI: Benefit

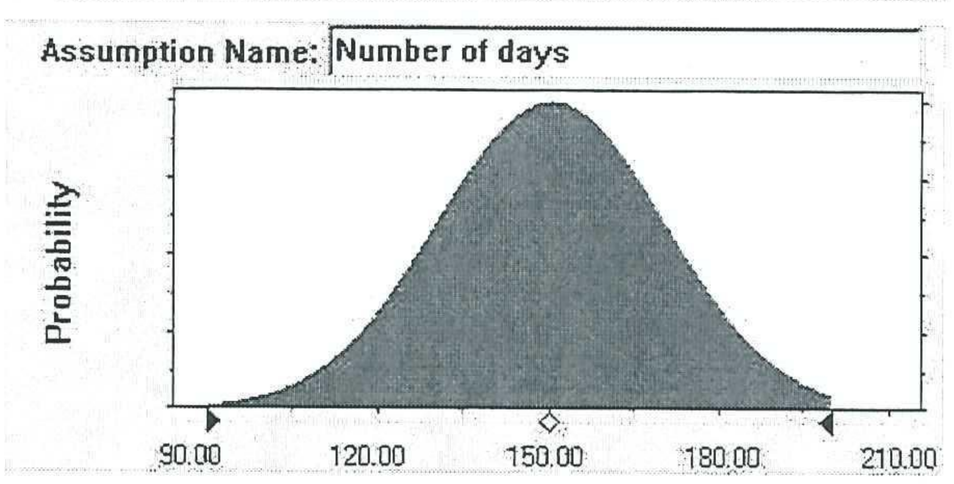
To give some idea about the possible societal benefit of such a system a simple *monte carlo* analysis is carried out to estimate the annual savings in euro based on the results in the recurrent congestion scenario.

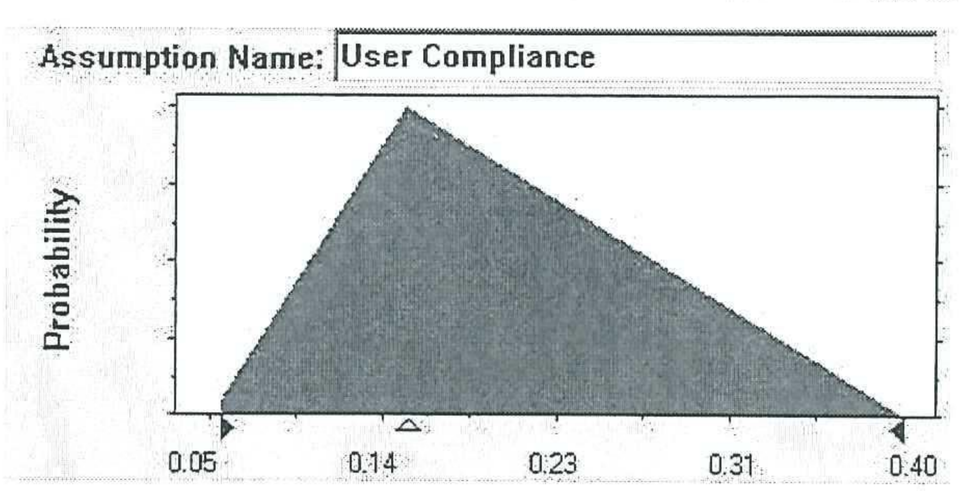
Assumptions

- The VOT is assumed to be normally distributed around 7 and limited to 8 euro.
- The expected delay reduction in total [traveller*hours] is normally distributed around 1000 hours per morning peak recurrent congestion and limited to 1150.
- The number of expected normal recurrent congestion is normally distributed with an average of 150
- The User compliance rate is estimated between 0.05 an 0.3 with 0.15 being the likeliest

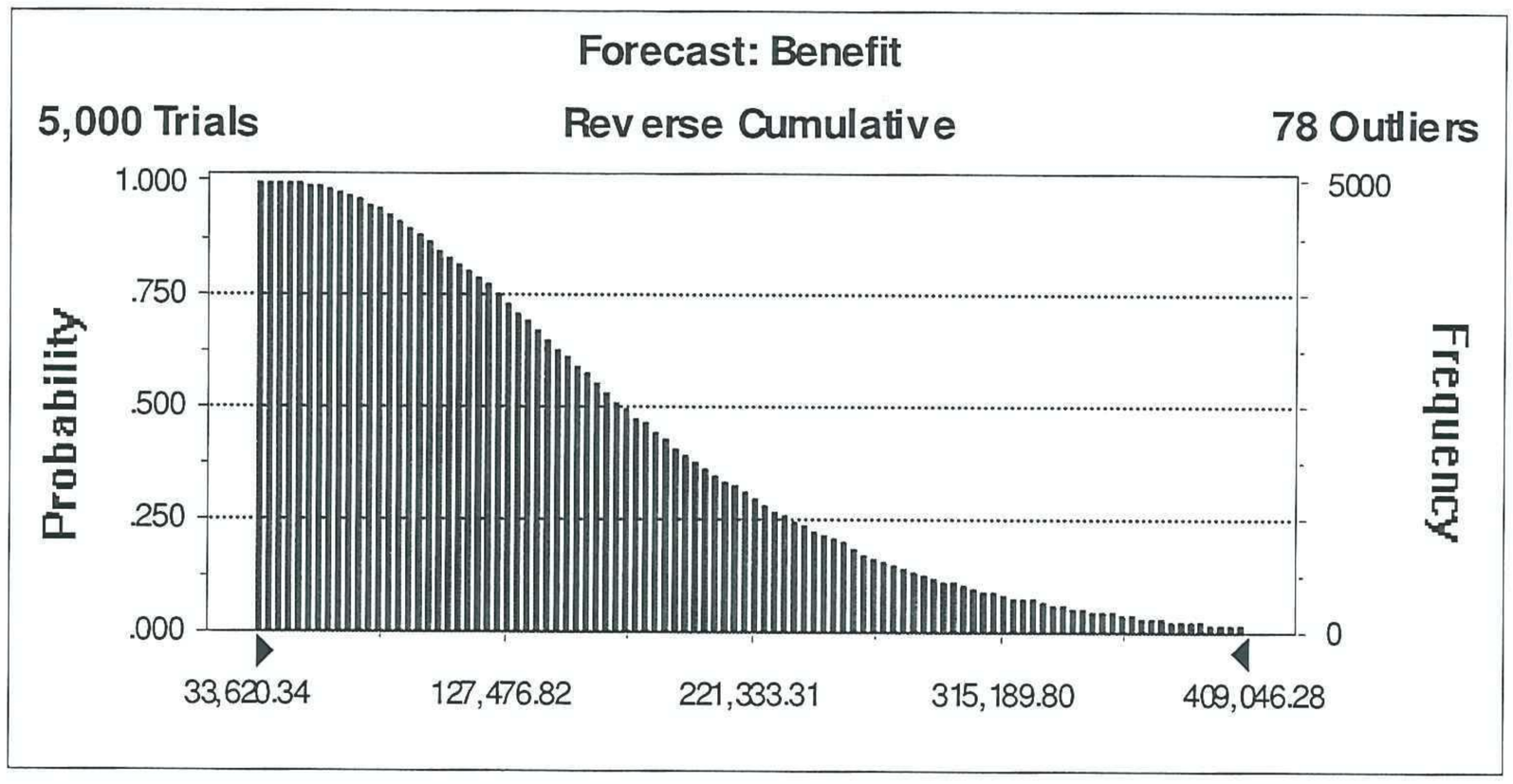








The actual saved hours is calculated by multiplication of the *expected total delay in hours* of a recurrent congestion scenario times the *user compliance.*



Based on the case study results, we could say with 50% certainty that the expected revenues will not exceed 170.000 euro a year.

This number is based on a lot of assumptions

- the improvement of the everyday congestion could be larger in real life since this scenario is the most difficult to optimize and the route choices used where based on full network knowledge which is not that realistic to start with
- The effect of the additional number of people who arrived at their destination during the optimized DRIP settings has not been taken into account
- During extreme events, the revenues will be much more