Refined tools for micro-modeling in transportation research

Acknowledgments

After completing this thesis, I want to thank everyone who was of essential importance to achieving my goal. My special gratitude goes to the international jury members Prof. Dr. Kai Nagel (TU Berlin), Prof. Dr. Carlo Giacomo Prato (DTU Copenhagen) and Prof. Dr. Ir. Eric van Berkum (Universiteit Twente) for the effort they spent reading my thesis, for their valuable comments and for being inspiring examples in research.

This thesis is the result of an interesting period of research at the Transportation Research Institute IMOB. Five years ago, I changed from a small-scale software development industrial environment to academic research. That required some adaptation. Therefore, I want to thank my promoter Prof. Dr. Ir. Tom Bellemans not only for the opportunity he gave me to join the research team but also for his patience with a student having a strange background. I really appreciated his ability to formulate research questions and his engineering bias during our problem solving discussions. Special thanks go to my co-promoter Prof. Dr. Davy Janssens for the multiple sessions of exchanging ideas while we traveled all over Europe heading to project meetings. I am grateful to my co-promoter Prof. Dr. Geert Wets for addressing interesting research questions to me and for our discussions on management and organization. My appreciation also goes to Prof. Dr. Koen Vanhoof for his remarks and suggestions made during the Ph.D committee meetings.

I owe a lot to Dr. Irith Ben-Arroyo Hartman (University of Haifa, Israel) who acted as an unofficial supervisor of part of my work. We first met on the DATASIM project and later on several conferences where we continued to discuss and cooperate on applications of graph theory in transportation science. I am very grateful for her patience and support.

Thanks to all my colleagues, especially for asking challenging questions. Thanks for the cooperation on small and large projects: making progress in research is only possible by working together. Thanks to all colleagues who supported me to get

things done, administered and organized.

A special word of thank goes to our colleague Dr. Ansar-Ul-Haque Yasar because back in 2011 he convinced me (being a young student) to cooperate on the organization of a workshop on agent-based modeling in transportation. This was the beginning of an active participation in conferences as a workshop or track organizer, which opened a world of fruitful scientific contacts for all of us at IMOB.

Very special thanks go to Brigit, my wife, for supporting my decision to become a student again for several years. She tolerated long periods where there was lack of time for things she estimated to be important. We are not active in the same domain and have different study background which makes her understanding of my focus very remarkable.

Thanks to our children and grand-children, first for having been curious about my work, and second for having been forgiving when I missed another grand-parents day at school or similar event.

Finally, a last word of gratitude goes to my parents who taught me critical reflection.

Luk Knapen

October 30, 2015

Contents

List of Tables

List of Figures

List of Abbreviations

ActBM	Activity-Based Model
AgnBM	Agent-Based Model
BBN	Bayesian Belief Network
BDI	Beliefs, Desires, Intentions
BEV	Battery only Electric Vehicle
BFS-LE	Breadth First Search - Link Elimination
BPC	Basic Path Component
C-TAP	Continuous Target-based Activity Planning
CC	Company Car
CKSP	Constrained K-Shortest Paths
CPP	Carpool Parking
CSP	Compressible Schedule Part
DATASIM	European FP7 Open-FET Project *Data Science for Simulating the Era of Electric Vehicles* (Project Number FP7-ICT-270833)
DDR	Domain of Data Relevance
DOF	Degree Of Freedom
DSS	Decision Support System
EV	Electric Vehicle
FEATHERS	Forecasting Evolutionary Activity Travel of Households and their Environmental RepercussionS
FSM	Finite State Machine
GCPMS	Global CarPooling Matching Service
GIS	Geographic Information System

GPS	Global Positioning System
GUI	Graphical User Interface
IAIS	Fraunhofer IAIS, Bonn
ICEV	Internal Combustion Engine Vehicle
ICOMflex	Flexible work places and the organization of the *New Way of Working* (Flemish project)
IMOB	Instituut voor Mobiliteit, Transportation Research Institute (Hasselt University)
ITN	Integrated Transport Network
ITS	Intelligent Transportation Systems
LTCPP	Long Term Car Pooling Problem
MAS	Multi Agent System
MASWOIC	Maximal Activity Sequence WithOut Internal Constraints
MATSim	Multi-Agent Transport Simulation
MHMM	Multi-Hypothesis Map Matching
MHT	Multi-Hypothesis Technique
MILP	Mixed Integer Linear Program
MNL	Multinomial Logit
MPDS	Minimum Path Decomposition Size
MRI	Mental Representation Item
OD	Origin-Destination
OSGi	Open Services Gateway initiative (former name)
OSM	OpenStreetMap
OVG	Onderzoek Verplaatsingsgedrag Vlaanderen (recurrent Flemish Travel Survey) `http://mobielvlaanderen.be/ovg/ovg04.php`
PDA	Personal Digital Assistant
PDOP	Positional Dilution Of Precision
PHEV	Pluggable Hybrid Electric Vehicle
POC	Privately Owned Car
PTE	Periodic Trip Execution
SLH	Semantic Location History

TA	Traffic Assignment
TAZ	Traffic Analysis Zone
TDM	Travel Demand Measure
TPOW	Tijd en Plaats-Onafhankelijk Werken (Time and Place Independent Working, ICOMflex project context)
TSP	Traveling Salesman Problem
VOT	Value Of Time
VRP	Vehicle Routing Problem
WIDRS	WithIn Day Re-Scheduling

Chapter 1

Introduction

Several aspects of micro-simulation modeling in transportation are covered by this thesis text. The introductory chapter explains the use of Activity-Based Models (ActBMs) and lists a set of pitfalls and challenging open problems at a general level in order to sketch the research context. Two parts will follow.

The first part covers project related research (mainly for the completed DATASIM project). It reports novel tools that enable to answer research questions by using the results generated by micro-simulation. Chapter 2 shows how the travel demand predicted by the FEATHERS Activity-Based Model is used to estimate the power demand generated by Electric Vehicles (EVs) as a function of time and space. Chapters 3 and 4 investigate the feasibility to use Agent-Based Models (AgnBMs) in services that provide advice to find partners for carpooling. Chapter 5 focuses on the combination of micro-simulated schedule adaptation and network performance evaluation by means of aggregated traffic assignment.

The second part focuses on the extraction of information from GPS traces in order to enhance the route choice component used in micro-simulation tools. This research was not project related but was triggered by scientific curiosity. Following hypothesis was formulated and verified using recorded Global Positioning System (GPS) traces: '*for their utilitarian trips (trips having the purpose to perform an activity at the destination location) people tend to compose their route from a small number of least cost components*'. Trips were extracted from GPS traces and for each of them the sequence of road network links is determined by the newly developed map matching tool described in chapter 7. Chapter 8 presents an algorithm that efficiently determines the minimum number of least cost path components required to reconstruct the paths extracted from the data and shows statistics about path composition for several

datasets. Finally, chapter 9 provides a polynomial time algorithm to enumerate all possible ways to split a given path into a minimum set of least cost components. The results of such enumerations can uncover information about routing preferences contained in the GPS traces; this information in turn can be used to support the *route choice set generation* in micro-simulation models.

1.1 Research Context for this Thesis

1.1.1 Agenda Adaptation, Coordination and Integration

Part 1 starts with the description of a research project in which an ActBM is used in the *classical way* by aggregating the individual simulation results using specific selection criteria (segmentation). A different approach is used for the other studies in the first part. They focus on coordination and schedule adaptation and hence make use of spatio-temporal schedule details.

The research presented in the first part focuses on the question how spatio-temporal results produced by ActBMs can be used to solve problems of travel and energy demand that require coordination and schedule adaptation. The reported studies provided input to the *scalability* and *electric vehicles* related work packages in the DATASIM project. The use of electric vehicles introduces the requirement for coordination between the electric energy customer and provider because of the limited power that can be delivered at a given location at any time. The reported spatio-temporal power demand research serves as a basis for ongoing work at IMOB that evaluates agenda adaptation in the context of time dependent electric energy prices.

Carpooling for commuting was studied because of its practical relevance and because it represents several aspects focused by DATASIM. Negotiating and coordinating actors are modeled by an AgnBM. Results predicted by the FEATHERS simulator were used to estimate the scalability issues that are to be expected in real situations. The reported work constitutes the basis for ongoing research and implementation of an Agent-Based Model (AgnBM) at IMOB.

Models for agenda adaptation need to be behaviorally sound and computationally efficient in order to be useful. A framework to evaluate such models is presented.

1.1.2 Route Choice Set Generation

As soon as travel demand is determined, it is to be loaded onto the transportation network. ActBMs provide predictions for individual trips. For each trip a *route choice set* is to be determined from which a route is to be chosen. Such set shall contain plausible alternatives considered by the traveler. A new attribute, quantifying the *structural complexity of a route*, is proposed to be used (i) as a criterion to assess route candidates for inclusion in the choice set and (ii) as an explanatory variable in the choice model. Analysis of large sets of GPS traces shows that actually used routes have a fairly simple structure. Based on this observation we propose two algorithms to extract information useful for both route choice set generation and route selection.

1.2 Activity-Based Models

Aspects of reality can be described by *micro-models*. Such models describe the entity to be investigated as a collection of simple components that in some specific way cooperate to act as a model of the overall entity.

Activity-Based Models (ActBMs) are micro-simulation models in which the simulated entity is a person in the context of a household. For each person in a synthetic population, the model constructs a *schedule*. A *schedule* (agenda) is a non-empty sequence of *episodes* for a given period (e.g. one day or one week) and each episode consists of a trip and an activity performed at the destination location of the trip. Schedule generation consists of at least (i) *planning* (i.e. determining a list of activities to be performed within a given time horizon; such list may already determine the set of possible locations and the expected duration for one or more activities) and (ii) *scheduling* (which fixes all attribute values for a particular activity (start time, duration, location) or trip (mode)).

The outcome of planning, scheduling and travel related decisions made by the individual, needs to be predicted by submodels for specific purposes (location choice, mode choice, etc).

The global spatio-temporal travel demand is determined by aggregating the trips generated by all individuals in the model to a specific level (e.g. by Traffic Analysis Zones (TAZs) in the region being modeled, by periods of time, by subgroups of the simulated population).

Most often, problems described by micro-models can be solved using analytic methods *only if* at least *all* of following conditions are fulfilled: (i) only a single type of entities does exist, (ii) the behavior of the entities can be described by simple

equations, (iii) the relation describing interacting entities is sparse and has a regular structure and (iv) the set of boundary conditions is regular. Specific problems in physics, engineering, econometry, transportation and other fields belong to this class. In many cases however, not all conditions are met.

In the field of transportation, several types of actors are involved and, within the class of human individuals, travel behavior depends on socio-economic and other attributes. Individuals have their own goals and their specific preferences about the way to achieve the goals: consider the difference between *car captives* and *car avoiders*. As a consequence, micro-models in transportation research in general can be evaluated only by *micro-simulation* i.e. by determining the representation of a given system state and then applying the rules (functions) that describe the individual behavior in order to determine the next state. For the problem of schedule generation, the *state* is described by the individual's attributes together with the partially completed agenda. The *rules* then specify how to select the next decision to be taken and how the outcome of that decision depends on the state. In most cases the rules are non-deterministic.

Travel behavior for individuals is generally described by stochastic models: hence ActBMs make use of stochastic micro-simulation. They shall be used to produce expected values (or distributions) for the quantities the analyst is interested in. If mutually independent entities are considered, parallel processing is easy and model characteristics are computed by simple aggregation.

1.3 The FEATHERS Activity-Based Model

The FEATHERS tool, the results of which are used in the research described in this thesis text, integrates both the *planning* and *scheduling* stages and delivers a complete *agenda* (schedule) for a given day for every member of the synthetic population. The input data consist of (i) a synthetic population specifying socio-demographic attributes (ii) land-use data describing numbers of schools, job opportunities, shops etc (iii) network performance data represented by travel time OD matrices for all transportation modes (iv) decision trees trained using OVG survey results. Those are used to model travel decisions taken by each individual.

1.4 Challenges in Activity-Based Modeling

The focus of current research efforts, some challenges to move the state of the art and suggestions for future development, emerging from practical experience, are listed below.

1. Aggregated results, e.g. the number of activities conducted by individuals during a specific day averaged over the complete population, do not require a large number of runs of the stochastic model in order to find an acceptably small variance. Less aggregated values e.g. traffic flows between specific TAZ can exhibit large variance over several simulation runs and require lots of runs to accurately determine expected values (Qiong et al. (2015)).

2. Individuals are considered to be (almost) mutually independent autonomous actors. This is advantageous at the level of technical implementation since it makes parallel processing of actors trivial. However, if the effect of a specific Travel Demand Measure is expected to change travel timing, schedule adaptation is to be expected. Several activity start and end times are subject to constraints some of which follow from joint activity execution or joint traveling. If coordination between individuals is ignored, the effect of the TDM might be overestimated by the model.

3. Some ActBM consider the decision process conducted by the individuals to be a black box. The outcome of the process is predicted from its inputs using data collected from surveys; the decision process mechanism is not modeled. This can be a limiting factor for travel demand management sensitivity. It raises the question whether or not the ActBM is replaying reality recorded at a given moment in time for a given region. The problem of transparency of sub-models is related to the question of correlation and causality. One way to introduce causal relations in the stochastic model for the decision process is by providing latent variables. This requires the formulation of measurement and model equations (e.g. the equations governing movement related quantities: position, speed and time). Often, this complicates parameter estimation. By introducing the additional equations, part of the correlation is replaced by causality. This comes down to replace *statistical explanation* by *knowledge based explanation* by decomposing a model into submodels. Statistics is the study of properties of collections of data; it explains variation and variance for dependent variables as a function of variation and variance for independent variables without the need to understand anything else about the universe. In transparent models,

universally applicable laws can be added. The time dependent state for a road junction is used as an example: a conservation law which allows to write an equilibrium equation specifying the relation between a junction capacity, its initial state and the number of vehicles entering from and exiting to each connected link during a given period of time can be added to the model of the junction to describe the evolution of its state over time. In a similar way, simulating the travel *decision making process* by itself instead of generating the *decision process outcome* using a statistical model trained by means of data that hold for a specific context, is expected to be more robust i.e. to deliver correct predictions in a larger range of situations. Hussain et al. (2015c) provide an example of this principle. The decision to carpool is not taken as the outcome of a random sample from a distribution determined by a survey; rather the decision process itself is modeled. The common knowledge introduced in the model specifies that a commuter, when evaluating the option of carpooling, allows for limited schedule adaptations only. This model predicts the share of carpooling among other transportation modes observed in the travel surveys; in addition it also provides insight in a possible reason for the outcome and hence contributes to the understanding of the (lack of) effect of dedicated carpooling incentives.

4. In most cases, the planning horizon is one day and consecutive days are handled as mutually independent. This is sufficient as far as network load for normal working or weekend days is to be predicted. Recent models (e.g. Continuous Target-based Activity Planning proposed by Maerki et al. (2014)) separate activity planning (adding to agenda) and scheduling (deciding about timing). Scheduling the next activity is done during schedule execution. The model considers a planning horizon in order to take into account deadlines for specific activities and availability of services (shop opening times). As a consequence, it is able to handle unusual day types (long weekends).

5. Using a specific time-of-day as a fixed time reference for every individual, turns out to introduce artificial restrictions that can be harmful to simulations. The introduction of an individual time reference for every actor is argued by two cases.

   In a city level project, the effect of a large hospital or a dominating employer factory operating in three shifts on travel demand, requires correct modeling of night shift work periods. Breaking a running activity over two consecutive days, introduces technical difficulties.

A second case is *schedule adaptation*: this implies compression and expansion of time periods which induces the concept of time pressure. This time pressure shall be determined only by the begin and end times of the period at hand and not by an artificial fixed time reference within such period. The fixed reference point is harmful in such cases because is creates a technical artifact disturbing the time pressure value. The problem is explained in chapter 5 and occurs as soon as rescheduling is involved (either due to unexpected events (Knapen et al. (2013c)) or deliberately by an individual e.g. in order to optimize Electric Vehicle (EV) charging (Usman et al. (2014a)).

6. Location choice is problematic in regions described by nearly homogeneous land-use characteristics while essential heterogeneity is not captured by the data (e.g. differences in attractiveness induced by marketing efforts). Knapen et al. (2014c) compares three different location choice models: (i) the distance band model used in FEATHERS, (ii) a gravity based model and (iii) the radiation model developed by Simini et al. (2012). This is done by starting from a home-work commuting matrix directly derived from census data. Individual locations (at statistical sector level, $1[km^2]$ on average) were aggregated to a somewhat coarser level (TAZ, $5[km^2]$ on average). Row and columns sums were calculated and the three models were used to create a new home-work commuting matrix from the marginals. All three models seem to accurately predict distribution for the distance driven but the correlation between the original and computed matrices is low (approx. 0.6). Each of the models is distance and population size based. The method used by FEATHERS uses additional information after pre-selection based on a distance-range criterion. The low correlation is explained by the spatial homogeneity of the data used by the models (population size, job opportunities) for Flanders.

7. Business trips (i.e. trips executed as a part of the job and by order of the employer) are ignored and need to be predicted by separate models.

8. Travel behavior models borrow aspects from physics and engineering as well as from mathematical sociology (Gilbert and Troitzsch (2005)). The number of input parameters and their variance are much larger in the latter field than in the former which makes validation of travel behavior micro-simulators difficult.

The aspects mentioned above shall be considered when using ActBM results to feed other models.

1.5 Advantages of Activity-based Models

Following advantages of ActBM (compared to traditional four-step models) originate from the ability to describe the behavior for the constituting entities.

1. The overall behavior of the complete model emerges from the behavior of the constituents and does not need to be modeled explicitly. Each individual actor can be constructed from sub-models in an hierarchical manner. Sub-models can be mutually dependent. As a consequence, specific behavioral details can be described according to the modeling requirements. This allows to study the effect of behavior change for particular segments of the population (sensitivity analysis and travel demand measures (TDM) evaluation).

2. All information for each individual is made available. This allows to report results for particular subsets of the generated schedules by specifying appropriate selections. Such selections can make use of specific attributes of *individuals* but also of *schedule properties*. Examples of the second category are: selections based (i) on departure times for trips towards specific activities and (ii) on specific activity sequences contained in the schedule.

3. Furthermore it is possible to build *relations between entities* by defining similarity concepts: e.g. in carpooling applications, individuals are combined using the spatio-temporal similarity of their commuting trips.

1.6 Research Overview

Research reported in part 1 (see Figure 1.1) takes ActBM results as input. Because predicted trips contain spatio-temporal information for vehicles, it is possible to calculate the energy requirements for each trip in a schedule. This was done for Electric Vehicles (EVs). Detailed technical characteristics of vehicles have been collected and several market shares were considered. Four types of EV charging behavior were used to calculate the energy demand as a function of time and space. The periods during which the vehicle can be connected to the grid follow from the predicted schedules and this allows to calculate the power demand for each TAZ as a function of time.

In a second study, predicted schedules are used to investigate carpooling as a prototypical example of coordination among people in order to cooperate on trip execution. On one hand, an Agent-Based Model (AgnBM) is designed to evaluate the

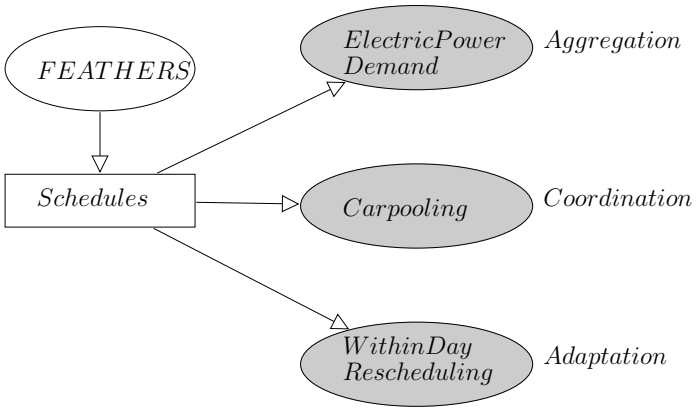


Figure 1.1: Overview part 1: The research consists of the tools represented by shaded ovals. Arrows represent data flows. Schedules (activity-travel agenda's for individuals) predicted by the FEATHERS activity-based model are used as input.

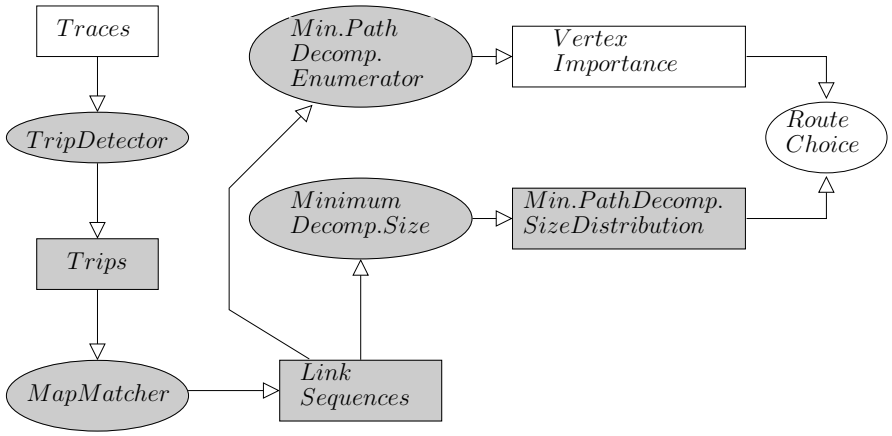


Figure 1.2: Overview part 2: The research consists of the tools represented by shaded ovals and results represented by shaded rectangles. Arrows represent data flows.

negotiation process among carpool candidates and in order to simulate the creation and evolution of the carpools. On the other hand, the back-end engine for a web-based carpooling advisor was designed and evaluated. Such system needs to keep track of a collection of advertised trips for carpooling. The appropriate advice is computed by maximizing the probability for successful negotiation among candidates. The solution method is based on graph theory and leads to an NP-hard problem. In order to support the selection of appropriate heuristics, the properties of the graph have been calculated from schedules predicted for Flanders.

Finally a hybrid framework to evaluate models for schedule adaptation was designed and built. On one hand, schedule adaptation is determined by micro-simulation; on the other hand, the effect on the road network is evaluated by aggregated traffic assignment and skimmed interzonal travel times are fed back to the micro-simulator. This framework supports evaluation for several types of schedule adaptation.

Part 2 (see Figure 1.2) proposes a line of tools aimed at the extraction of route information from GPS traces. This is done by first splitting the trace into pieces by *trip detection*. The resulting GPS sequences for each trip then are converted to sequences of links in the road network by *map matching*. The minimum number of shortest paths required to construct the trip, is determined for each trip that constitutes a path in the network. This number is used to verify the hypothesis that people tend to compose their trips by a small number of shortest paths. A distribution for the size of those minimum decompositions is determined from two sets of GPS traces and confirms the hypothesis. The resulting distribution can be used to enhance the quality of the route choice process. Finally, a technique to enumerate all possible minimum path decompositions in polynomial time, is presented. It is shown how such enumerations can contribute to automated detection of points that have a special meaning in route generation (way-points).

1.7 Thesis Outline

Each part of the thesis contains several chapters each of which covers a particular research topic. In the first part, every chapter starts with a sketch of the research context for the papers constituting the chapter. In the second part, the shared research context is sketched in an introductory chapter because the topics in that part constitute an integrated research effort so that the motivation for the first paper (map matching) can only be understood when the final research goal is known.

Part I

Novel Applications Using Micro-simulation Results

Chapter 2

Spatio-temporal distribution of electric power demand caused by EV

This chapter is based on

Knapen et al. (2012b)	*Using Activity-Based Modeling to Predict Spatial and Temporal Electrical Vehicle Power demand in Flanders*
Knapen et al. (2011)	*Activity-based models for countrywide electric vehicle power demand calculation*

Related co-authored papers	
D'hulst et al. (2012)	Decentralized Coordinated Charging of Electric Vehicles Considering Locational and Temporal Flexibility
Ridder et al. (2013)	Applying an Activity based Model to Explore the Potential of Electrical Vehicles in the Smart Grid
De Ridder et al. (2013)	Electric Vehicles in the Smart Grid
Alvaro et al. (2014)	Vehicle to vehicle energy exchange in smart grid applications
Gonzales et al. (2014)	Determining Electric Vehicle Charging Point Locations Considering Drivers' Daily Activities
Alvaro-Hermana et al. (2015)	Peer to Peer Energy Trading with Electric Vehicles
Usman et al. (2015)	Relationship Between Spatio-temporal Electricity Cost Variability and E-mobility

2.1 Research Context

Several research projects have been conducted in the last five years. Electric Vehicle (EV) research focuses on several aspects: car technology, energy conversion and storage, impact of EV charging on the electric grid, estimation of customer preferences and modeling of influencing effects of early adopters. Main stakeholders in Flanders are customers, energy providers and grid operators. It should be noted that the EV market share seems to be far less than was commonly expected in 2011-2012 and predicted for 2015.

The emergence of EV raises the problem of electric energy distribution but on the other hand contributes to solve the problem of electric energy storage. From the distribution grid point of view, production and consumption of electric energy need to be balanced at every moment in time. The growing amount of solar and wind energy installations can cause balancing problems due to the unpredictability and variability of the production power. The EV fleet represents a floating (in space and time) storage capacity for electric energy that can mitigate the unbalance problem provided that intelligent management systems are in place and a critical storage capacity is available.

FEATHERS predicts a list of about 9 million episodes each one consisting of a trip and an activity. The activity is characterized by the activity type (home activity, work, shopping, etc). Trip attributes are the transportation mode, origin, destination, trip start time and expected duration. Activity attributes can be used to determine the feasibility to charge the EV battery at the activity location. Trip attributes are

used to calculate energy consumption. When the state of charge (SOC) of the battery is known for a given time of day and location, it can be calculated for the location where the EV subsequently is connected to the grid. Hence, the amount of energy that can be charged at a given location during a specific period, can be calculated. This results in the power demand for the electric grid as a function of time and space.

Since it is difficult (and hence expensive) to store large amount of electric energy, since the energy transport is bounded by power limitations in the grid and since the storage capacity is moving around, this leads to complex problems. In the DATASIM project, scientists in the fields of big data, engineering and transportation cooperated on smart-grid and EV charging related topics. Activity-based modeling delivers the input required to estimate the power demand in space and time.

Results predicted by FEATHERS constitute the base for ongoing research on EV charging schemes under assumptions of time dependent energy cost and taking into account charging location specific power supply limitations.

The research described in the following sections enabled me to co-author the papers mentioned above as an active contributor.

2.2 Abstract

Electric power demand for household generated traffic is estimated as a function of time and space for the region of Flanders. An activity-based model is used to predict traffic demand. Electric Vehicle (EV) type and charger characteristics are determined on the basis of car ownership and by assuming that EV categories market shares will be similar to the current ones for Internal Combustion Engine Vehicles (ICEVs) published in government statistics. Charging opportunities at home and work locations are derived from the predicted schedules and by estimating the possibility to charge at work. Simulations are run for several EV market penetration levels and for specific BEV/PHEV (battery-only/pluggable hybrid) ratios. A single car is used to drive all trips in a daily schedule. Most of the Flemish schedules can be driven entirely by a BEV even after reducing published range values to account for range anxiety and for the over-estimated ranges resulting from tests according to standards. The current low tariff electricity period overnight is found to be sufficiently long to allow for individual cost optimizing while peak shaving overall power demand.

2.3 Introduction

2.3.1 Electric vehicles use

The economy's dependency on fossil combustibles is attempted to be decreased for both environmental and strategic reasons. Resulting effects are an expected increase of Electric Vehicle (EV) use and use of alternative sources for electric energy production. Sustainable electric energy sources (wind, solar) deliver power at variable rates that cannot easily be predicted. Furthermore, storing electric energy is a major problem.

The use of EV generates challenging questions but also opportunities: when EV are used in a *vehicle to grid (V2G)* configuration, they can serve as electric energy storage devices. Designing and operating an electricity grid having lots of small unpredictable producers combined with relocatable storage capacity that is time dependent, is a complex problem.

The problem receives more than pure technical attention. White-House-NSTC (2011) states: *President Obama has set a national goal of generating 80% of [the] electricity from clean energy sources by 2035 and has reiterated his goal of putting one million electric vehicles on the road by 2015.*

2.3.2 Activity-based models to predict energy demand by electric vehicles

Activity-based modeling (ActBM) predicts daily schedules for people based on the behavioral characteristics for each individual. As a result, each individual actor can be designed to adapt in its own specific way to changes applied in scenarios when using feedback mechanisms during simulation. Activity-based models therefore allow for policy evaluation. The schedules generated by ActBM contain information about the transport modes used and about the activity kind, duration and location. As a result they provide the tools to investigate the feasibility of goals like the one stated in White-House-NSTC (2011) both by modeling in a closed loop, individual behavior change (adaptation) and the effect thereof on the public infrastructure.

This paper explores the case for Flanders. The region counts 6 million inhabitants on 13000 square kilometers and is part of Belgium (Europe) (11 million inhabitants on 30000 square kilometers). The area is subdivided in 2368 zones. A synthetic population of actors has been built to mimic each inhabitant of the area to be studied. Actor behavior is determined by characteristics of the surroundings like road transportation network, distance between locations suited for specific types of activities, public transport availability, delays induced by congestion. The FEATHERS ActBM described in Bellemans et al. (2010) has been used. Within FEATHERS, actor behavior is modeled by 26 decision trees, each one of which takes as input attributes of both the individual actor and the environment as well as the outcome of decisions already made. The decision trees have been trained by means of the CHAID method using data from regional time specific travel behavior OVG surveys. A single survey covers up to 8800 respondents. The decision trees are used to predict (in the order specified) attributes for work episodes, work locations, work-travel mode, fixed non-work activities, flexible non-work activities, non-work locations, non-work-travel mode. At this moment FEATHERS does not adapt actor behavior to car type (ICEV, EV). Car type is determined after schedule prediction. Resulting schedules are used to predict time and location for travel related electric energy consumption.

First we explain what hypotheses about EV drivers behavior have been made and how EV characteristics have been determined from literature and from available statistical data. Next, calculation details are described. Finally, results for the Flemish region are presented: area specific energy and power requirements as a function of time identify critical parts in the electric grid. The fraction of the household transportation market that can be served by EV without range extenders, is calculated.

2.3.3 Related work

Many research projects are driven by the goals to reduce greenhouse emissions. Recently European research focuses on the problem of matching the supply and demand of electric energy from sustainable sources (solar, wind). Cui et al. (2011) use a car selection model, a budget prediction model and an agent-based simulator (stigmergy) to predict pluggable hybrid electric vehicle (PHEV) market penetration for Knox County (190k households). Davies and Kurani (2011) predict the electric power demand for the PHEV used by 25 households from data recorded in an experiment and from a PHEV car design game conducted by the households: the effect of work location charging is simulated. Kang and Recker (2009) and Recker and Kang (2010) use an activity-based model for California based on statewide travel diaries and several charging scenarios to predict the power demand for the whole area as a function of time. Bliek et al. (2010) describe how PowerMatcher predicts electric energy in a smartgrid containing small unpredictable solar and wind energy sources and tries to match supply and demand using an agent-based auction for electric energy. Clement-Nyns et al. (2009) evaluate *coordinated charging* strategies for a Belgian case. In such systems customers need to specify time limits for charging (which can be produced by ActBM). Waraich and Galus (2009) evaluate energy tariff effects on charging behavior for the city of Berlin by coupling `MATSim-T` (travel demand simulator framework) to `PMPSS` (PHEV Management and Power Systems Simulation). Binding and Sundstroem (2011) describe an agent-based simulator for an auction based energy pricing system aimed at matching sustainable power supply and demand: they plan to integrate the V2G (Vehicle to Grid) concept to temporary store energy in car batteries. Hadley and Tsvetkova (2008) predefine a charging profile and analyze the effect on power demand when applying it to 13 US regions at different times of the day.

2.4 Context

2.4.1 Smart grids and transport engineering

Smart grids are required when trying to meet electric energy demand in networks containing many small production units exposing difficult to predict behavior (solar, wind energy). Several techniques are used to try smoothing power requirement over time and to adapting it to uncontrollable time dependent production. With central coordination based schemes, the energy provider is allowed to turn on/off electric loads remotely. Other schemes rely on intelligence local to the consumer to determine

electric demand at any moment in time: auction based configurations try to adapt demand to production by negotiating location specific prices every 15 minutes. Each one of those schemes requires intelligent components but also a lot of information about the environment and efficient adequate short time forecasting techniques. ActBM in transport engineering can contribute to the problem solution by creating adequate tools to forecast the energy and power demand as a function of time and location in order to decide when and where energy can be delivered proactively or stored in batteries for later retrieval. Several papers mentioned under **Related Work** predict energy demand: they do so either for a small population or as an aggregated value for a wide region. Related work on smartgrid design shows that the auction based pricing system simulators need predictions about *when* and *where* electric power is demanded. Therefor, this paper estimates the electric energy and power requirements for Flanders using activity-based modeling.

2.4.2 Electric energy demand evolution - Power demand

2.4.2.1 Energy demand

According to several sources (Perujo Mateos Del Parque and Ciuffo (2009), Perujo and Ciuffo (2010)) the total amount of energy drawn from the grid by electric vehicles is relatively small: a 30% market share EV would represent 3% of the total annual electric energy consumption for the region of Milan, Italy.

For a Flemish household, the estimated yearly amount of electric energy required by the car (0.2 kWh/km, 15000 km) is of the same order of magnitude as the amount of electric energy consumed by the household for other purposes (current electric energy consumption value). According to figures published in *Oxford University Environmental Change Institute* website statistics pages UO_ECI (2011) the average yearly consumption for a Belgian household amounts to 3899 [kWh/year]. A similar figure (3500 [kWh/year]) for Belgium is mentioned by EABEV (2010). As a consequence, the relative contribution of transport in the overall demand, will grow significantly with increasing EV market penetration.

The evolution of electric energy demand per sector for Belgium is given in William D'haeseleer et al. (2007). Total consumption in 2005 was 80.2 TWh. The transport sector contribution increases but amounted to only 2.12% in 2005. According to several sources (Ramage (2010), Perujo Mateos Del Parque and Ciuffo (2009)) the energy demand by EV is not expected to cause problems on the electricity grid provided it is distributed over time.

2.4.2.2 Power demand

Activity-based models help to assess where and when peak power demand would exceed limits imposed by the grid. Perujo and Ciuffo Perujo and Ciuffo (2010) studied power demand for the Milan region using the assumptions that people will not charge their car batteries every day but only when needed and that charging starts between 16:00h and 19:00h in the evening obeying a uniform distribution over time. Perujo Mateos Del Parque and Ciuffo (2009) recognize the need for statistical values (estimated distributions) on daily commuter trips for a particular region. Our study uses ActBM to calculate charging time and location resulting in a prediction of EV power demand.

2.4.3 The use of activity-based models

Electric energy demand estimates require detailed data about location and timing as well as trip purpose and activity information for each simulated individual. This paper investigates following scenarios for charging of both Battery only Electric Vehicles (BEVs) and Pluggable Hybrid Electric Vehicles (PHEVs) in order to calculate peak power demand as a function of time and location starting from FEATHERS predicted schedules:

- Scenario **EarlyLowTariff**: people start charging as soon as possible during the low tariff period (night-time, reduced-rate electricity).
- Scenario **UniformLowCost**: people start charging at a uniformly distributed moment in time but so that their cost is minimal (maximum use of low tariff period).
- Scenario **LastHome**: people start charging batteries as soon as the car gets parked at the *last* home arrival of the day irrespective of any low-tariff period.
- Scenario **AlwaysAtHome**: people charge batteries immediately after *each* home arrival.

In all cases, charging period is assumed to be contiguous (uninterrupted) which means that no auction based dynamic pricing for fifteen minute charging blocks has been considered. Furthermore we hypothesize

- that everyone recharges batteries every day due to *range anxiety*
- that all cars are charged at home with additional charging at the work location in well defined cases only

2.5 Electric vehicle fleet attributes

Since the EV market is only emerging, predictions cannot be based on extensive statistics. Assumptions made in the paper have been explained and argued below.

2.5.1 Vehicle categories

Electric cars are subdivided into the categories `small, medium, large` similar to what is done in Perujo and Ciuffo (2010). In order to estimate the energy requirement, one needs to know the contribution of each category to the complete vehicle fleet. Belgian government statistics provide the distribution of registered cars along a classification based on the IICEV cylinder volume. We state the one-to-one mapping of categories given in table 2.1 that shows market share and technical characteristics for each category. Vehicle characteristics in the table have been derived from data in Perujo and Ciuffo (2010) and Nemry et al. (2009). The market share figures have been taken from the Belgian federal government 2009 *PARC010 Transport Indicator* statistics Federal Planning Bureau (2009)

2.5.2 Available Chargers

Locally available 3.3 [kW] and 7.2 [kW] chargers are considered. Our model distinguishes between *home* and *work location* chargers (see table 2.2). Charger type occurrence probability is given in table 2.1. The power value for home chargers is assumed to depend on the car category: smaller cars are equipped with a less powerful charger. On the other hand, companies offering car charging facilities are assumed to provide powerful chargers in order to save time and to extend the distance that can be bridged during one day. The company investment in a powerful charger is assumed to be a profitable one.

2.5.3 Company cars in Belgium - Vehicle ownership

Employers are believed to allow Company Car (CC) drivers to charge at the work location because that is less expensive than providing fuel cards to employees. However, for technical reasons, not all companies can provide the required infrastructure. The fraction of actors who can charge batteries at the work location has been determined as a fraction of company car drivers. It has been assumed that 50% of the CC drivers can charge at the work location.

	Vehicle categories		
Equivalent engine cylinder volume [cc] (ICEV category)	$V < 1400$	$1400 \leq V \leq 2000$	$2000 < V$
Market share (from Belgian government statistics)	0.496	0.364	0.140
EV category	small	medium	large
Battery capacity (kWh)	10	20	35
Range (km)	100	130	180
Energy consumption (kWh/km) : lower limit	0.090	0.138	0.175
Energy consumption (kWh/km) : upper limit	0.110	0.169	0.214
Charger type at home : prob(3.3[kW])	0.8	0.4	0.1
Charger type at home : prob(7.2[kW])	0.2	0.6	0.9
Charger type at work : prob(3.3[kW])	0.1	0.1	0.1
Charger type at work : prob(7.2[kW])	0.9	0.9	0.9

Table 2.1: Correspondence between EV and ICEV for categories specified in Belgian Government statistics

The FEATHERS ActBM predicts trips and provides information about car availability but not about car ownership (private vs. company owned). In order to estimate the number of people able to charge batteries at the work location, we need to estimate the fraction of work trips traveled by company car. The COCA (Company Car analysis) report Cornelis et al. (2007) states that, depending on the context, multiple definitions of a *company car* (voiture de société) are in use because both fiscal and operational aspects are concerned. The COCA definition (*A company car is made available by a company to an employee for both professional and private use*) is used in our study. The same COCA report states that, based on two Belgian reports (*OVG* for Flanders and *ERMMW* for Wallonia), it can be concluded from data up to 2005, that 6% ... 7% of the car fleet in use by Belgian households, is company owned (source Cornelis et al. (2007) page 31/80). The OVG42 report Cools et al. (2011) estimates the fraction of company cars available to households in 2009 to 10%.

Our model assumes that 10% of the actors driving to work, make use of a company

Charge location	Car category	Prob(3.3[kW])	Prob(7.2[kW])
Home	small	0.8	0.2
	medium	0.4	0.6
	large	0.1	0.9
Work	small	0.1	0.9
	medium	0.1	0.9
	large	0.1	0.9

Table 2.2: Charger Type Distribution

car. Cars used in schedules without any work trips, are assumed to be Privately Owned Car (POC).

2.5.4 Relation between EV ownership and EV type

The portions of EV being PHEV are assumed to differ between privately owned and company cars. Currently no data about the respective expected market shares are available. PHEV rates 0.0, 0.5 and 1.0 for both CC and POC have been combined to run simulations.

PHEV do not have practical range limitations but long All-Electric-Range (AER) versions are more expensive than BEV. Temporal unavailability of a car induces high hourly costs for a company: the investment in a more expensive PHEV is assumed to be a profitable one. Private owners, on the other hand, are assumed to be more reluctant against large initial investments for private use.

2.6 Simulations

2.6.1 Method overview

The FEATHERS ActBM (Bellemans et al. (2010)) created by the Transportation Research Institute (IMOB) has been used to generate *activity-travel schedules* (daily agenda for each individual of the Flemish population). Each schedule consists of trips and activities. For each trip, departure time, trip duration, origin and destination zones are predicted. For each activity, the purpose (work, shop, bring-get, ...) is predicted. In this study, only work and non-work activities are distinguished. FEATHERS results apply to a single 24-hour period. A working day simulation has

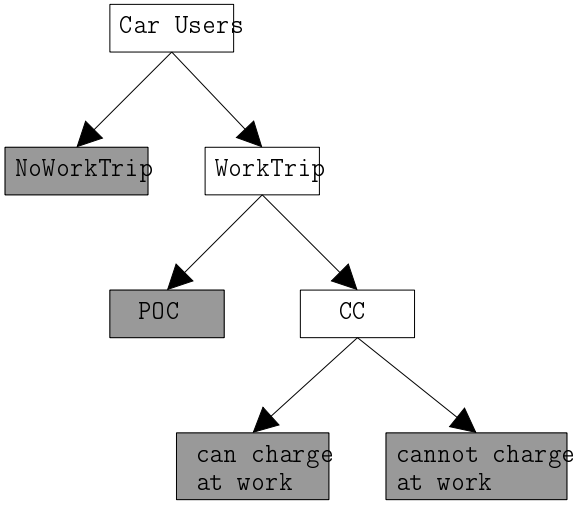


Figure 2.1: Car users partitioning. (1) workTrip based partitioning follows from the AB-Model-generated schedules. (2) ownership (POC, CC) and canChargeAtWork are specified by parameters.

been used.

Energy and power demand are computed from FEATHERS results as follows:

- In a first step, schedules having at least one car trip are extracted and data structures are set up.
- In the second step, car ownership, possibility of work location charging, car characteristics (range, distance specific energy consumption, battery capacity) and the types of home and work location charger used, are determined. Both a BEV and a PHEV belonging to a same category, are assigned to the schedule. A feasibility indicator is calculated which tells whether or not the schedule can be executed using the assigned BEV electric car (PHEV always is feasible since the internal combustion engine (ICE) always is available as a range extender). Each individual schedule is assumed to be executed using a single car and a predefined fraction of the company cars can get recharged at the work location. The set of schedules is partitioned as specified in figure 2.1. For each one of the leaf node parts, the market share has been specified: the results shown in this report hold for 10% *no-work trip* and 10% *work trip* electrification.
- In the third step, *charging scenarios* are evaluated. Schedules are sampled from the partitions set up in the second step and the start time for each charging operation is determined. Energy requirement and power demand are accumulated

for every minute of the day for each one of the 2368 zones in Flanders.

2.6.2 Vehicle characteristics determination

Vehicle characteristics for each schedule are determined by random selection using the joint probabilities shown in the Bayesian network in figure 2.2 . Arrows designate dependencies between probability densities. For example, the EV *type* depends on the *ownership* and on the fact that the schedule can be executed using a BEV (block *BEV-feasibility*). The shaded rectangle *Electrified* represents the probability density from which EV are sampled. The shaded rectangle *EnergyReq* represents the probability density for the electric energy required to complete all trips in the schedule. The ovals represent *change of variable functions*. Function *f(schedule,consumption)* calculates whether or not the sequence of trips in a given schedule can be driven by a BEV given the stochastic value for the distance specific consumption of the vehicle and the charge opportunities in the schedule. Function *f(schedule,consumption)* corresponds to the conditions detailed in equations 2.1 and 2.2.

The function *g(schedule,consumption)* calculates the stochastic value for the energy required during each minute of the day for the given schedule.

Vehicle characteristics are determined as follows:

- Vehicle *category* is randomly selected from the distribution specified in table 2.1
- Vehicle *range* is selected from table 2.1.
- Work location charging is allowed for 0.50 of the company car drivers. Privately owned cars cannot be recharged at work. The charger power is randomly selected for both home and work location chargers using the distribution specified in table 2.1.
- Vehicle *consumption* is randomly selected using a uniform distribution in the interval specified for the vehicle category (from table 2.1). This is the consumption determined by official US and European standard (FTP, WP.29) test suites that do not account for cabin clima (heating, airco) nor for frequent acceleration and deceleration.
- The specific energy consumption as determined by European (UNECE WP.29 R101) and US standard methods is argued to be an underestimation (Elgowainy et al. (2010)). The standardized test conditions differ from operating conditions: hence, a *range reduction coefficient* of 0.75 has been applied. The range reduction coefficient is used to adjust the specific consumption (which is used in schedule feasibility and energy demand calculations). This is done for

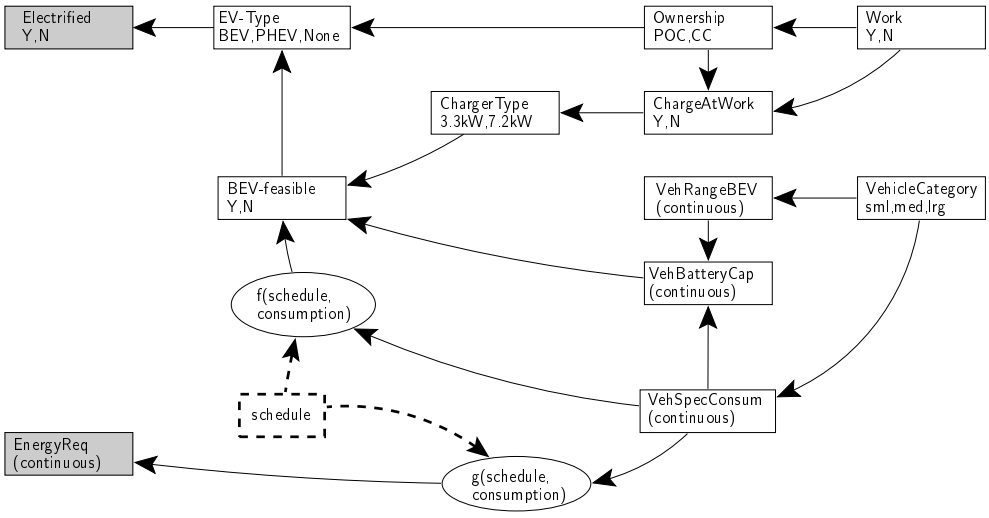


Figure 2.2: Bayesian network showing conditional dependencies for stochastic variables. Continuous line rectangles designate probability densities. The domain for the variable is listed between curly braces. Each continuous line arrow designates a conditional dependency. Ovals designate *change of variable* functions. Dashed lines represent regular functional dependencies.

both BEV and PHEV in the same way.

- The battery capacity is derived from range and distance specific consumption and has been verified with data found in literature (Nemry et al. (2009), Wu et al. (2010), Kromer and Heywood (2007)).
- PHEV categories PHEV48, PHEV64 and PHEV96 are considered and have been mapped to the categories *small*, *medium* and *large* respectively in order to determine the relative market shares (see table 2.1). The number in the category identifier designates the AER in kilometers.
- Finally, the charger power is randomly selected for both home and work location chargers using the distribution specified in table 2.1.

2.6.3 BEV-feasibility

In order to be feasible for a BEV, each location in the schedule shall be reachable when starting with a fully charged battery in the morning: this is expressed by the

condition (set of $\#\mathbb{L}$ inequalities)

$$\forall i, j \in [1, \#\mathbb{L}] : C_b - d_{O,i} * cons + \sum_{j=1}^{j<i} t_j * p_j \geq C_b * DCD \tag{2.1}$$

where i and j are location indexes, C_b is the battery capacity, $\mathbb{L}$ is the set of all locations used in the schedule, t_j is the charge-period duration at the j-th location and p_j is corresponding power, $d_{O,i}$ is the total distance from the first origin to the i-th destination, *cons* is the distance specific energy consumption and $DCD = 0.1$ is the maximal *deep charge depletion* coefficient. DCD has been applied to specify the minimal battery level that shall be available at all times; it is used to model *range anxiety* and is used in BEV-electrification feasibility calculation only. The condition that the battery cannot get over-charged is given by following set of inequalities using the same symbols

$$\forall i, j \in [1, \#\mathbb{L}] : C_b - d_{O,i} * cons + \sum_{j=1}^{j\leq i} t_j * p_j \leq C_b \tag{2.2}$$

2.6.4 Vehicle sampling

The vehicle type (BEV, PHEV) is determined using the conditional probability values specified under *Relation between EV ownership and EV type* above. The probability for a vehicle to be a PHEV is given following expressions containing given probabilities in the right hand sides

$$P_{EV} = P(EV|CC) \cdot P_{CC} + P(EV|POC) \cdot P_{POC} \tag{2.3}$$

$$P_{PHEV} = P_{CC} \cdot P(EV|CC) \cdot P(PHEV|EV \wedge CC) + \tag{2.4}$$

$$P_{POC} \cdot P(EV|POC) \cdot P(PHEV|EV \wedge POC) \tag{2.5}$$

where EV designates Electric Vehicle, CC designates Company Car, POC designates Privately Owned Car, $PHEV$ designates Pluggable Hybrid Electric Vehicle. It follows that

$$P_{BEV} = P_{EV} \cdot (1 - P(PHEV|EV)) = P_{EV} \cdot (1 - P_{PHEV}/P_{EV}) = P_{EV} - P_{PHEV} \tag{2.6}$$

where BEV designates Battery Electric Vehicle. Let N_v be the number of cars. A set of $P_{BEV} \cdot N_v$ elements is sampled from the set of schedules that can be executed by a BEV (the *BEV-feasible* schedules); then $P_{PHEV} \cdot N_v$ cars are sampled from all remaining schedules (BEV-feasible and BEV-infeasible ones).

2.6.5 Charging parameters - Scenarios

2.6.5.1 Assumptions valid for all scenarios concerned

- Energy cost is assumed to conform to the current tariff scheme used in Belgium: it consists of one contiguous *regular tariff* period and one contiguous *low tariff* period during the night (from 22:00h to 07:00h).
- The schedules apply to a working day and schedules are assumed to repeat on successive days. This assumption allows to determine the period of time available for recharging overnight. Everyone is assumed to recharge batteries everyday.
- When plugged to the electric grid, charging occurs during a single uninterrupted period of time.

For each schedule and each charging opportunity, the *required* charge duration for full recharge and the *available* charge period are calculated. The available charge period is determined from the arrival and departure times at the charge location. If the available period length is larger than the required charge duration, their difference is the *slack time* (otherwise slack time equals zero). A non-zero slack time implies a degree of freedom for selecting the time to start charging. In many cases, there is an interval $\Delta t = [t_0, t_1]$ of starting times t_s such that $\forall t_s \in \Delta t$ the energy cost is the same.

2.6.5.2 Scenario specific assumptions

- Scenario **EarlyLowTariff**: If Δt is contained in the low tariff period, the actor starts charging as soon as possible; otherwise (the case where the charge period contains the low-tariff period), the actor starts charging as late as possible thereby pushing energy demand to the morning hours. This scenario conforms to the situation where people are using simple timers to start charging.
- Scenario **UniformLowCost**: Each actor tries to minimize energy cost by charging during the low tariff period as much as possible. The charge period start time t_s is chosen from Δt by random selection using a uniform distribution.
- Scenario **LastHome**: All actors ignore the existence of a low-tariff period and start charging immediately when arriving at home after the last trip of the day.
- Scenario **AlwaysAtHome**: All actors ignore the existence of a low-tariff period and start charging immediately when arriving at home after each home arrival.

Note that scenarios *EarlyLowTariff* and *UniformLowCost* are energy cost minimizing scenarios at the individual actor level, but the other ones are not.

Scenario	FEATHERS ActBM prediction	
All	Fraction of actors performing work trips	0.406
All	Fraction of actors performing car trips	0.555
All	Fraction of car using schedules containing work activity	0.531
All	Average work related car trip distance (km)	19.376
All	Fraction of trips that are work trips	0.160
	EV Energy demand calculation	
CC=0.0 and POC=0.0	Total energy demand	1380[MWh]
CC=0.5 and POC=0.5	Total energy demand	1652[MWh]
CC=1.0 and POC=1.0	Total energy demand	1829[MWh]

Table 2.3: FEATHERS Results Statistics.

2.6.5.3 Aggregation of micro-simulation results

Battery charging opportunities are identified during micro-simulation and inserted in the schedules according to the applied scenario. For each charge opportunity, the required power is accumulated and recorded for each minute in the charging period. This process results in a power requirement time series for each zone. Plots are generated for the zones having

- maximal energy requirement (power integrated over time)
- maximal power peak value

for the full day, the normal-tariff period and the low-tariff periods respectively.

2.7 Summary of results for Flemish region

- FEATHERS statistics and energy demands have been summarized in table 2.3. Scenarios are identified by the ratio of the EV fleet being a PHEV for company cars (CC) and privately owned cars (POC) respectively. Replacing BEV by PHEV increases power demand since longer distances are driven on electricity. PHEV can exhaust the full AER while BEV can drive distances strictly smaller than the *anxiety reduced* range only.
- Table 2.4 shows the fractions of BEV-feasible schedules determined in the second step (accounting for work location recharge). Note that only 10% of the

Partition	Fraction of the car using schedules	
	When charging after last home arrival	When charging at each home arrival
BEV-feasible schedules without work trips POC (NW)	0.364	0.371
BEV-feasible schedules with work trips POC (W_POC)	0.357	0.371
BEV-feasible schedules with work trips CC, chargeAtWork (W_CC_CAW)	0.020	0.021
BEV-feasible schedules with work trips CC, no chargeAtWork (W_CC_NCAW)	0.024	0.024
BEV-Infeasible	0.235	0.213

Table 2.4: Car-using Schedule Partitions with respect to Feasibility for Electrification

schedules having a work trip have been assigned a company car in the scenarios considered. Almost 78% of the trips is BEV-feasible when the EV category coincides with actual ICEV market shares given in table 2.1.

- Figure 2.3 shows the power demand for an area with 5835 inhabitants for scenarios *UniformLowCost*, *LastHome* and *AlwaysAtHome*. The power peak for *UniformLowCost* (individual actor cost minimizing) is the bigger one and the peak shifts from about 20:00h to about 02:30 between scenarios. Note that the power demand shown is to be added to the already existing *zone-specific* demand but at the time of writing only countrywide aggregated time dependent electricity consumption data are available; hence data have not yet been presented geographically to pinpoint problematic areas. The result shows that it is worth extending the ActBM actor behavior model to make it sensitive to electricity prices.
- The power peak for scenario *EarlyLowTariff* at 22:00h amounts to eight times the *UniformLowCost* peak value because everyone is assumed to start charging at the same moment using a timer. This peak is expected to cause problems for the electric grid and has not been included in the diagram.
- Table 2.5 shows the fraction of charge opportunities used and the daily charge

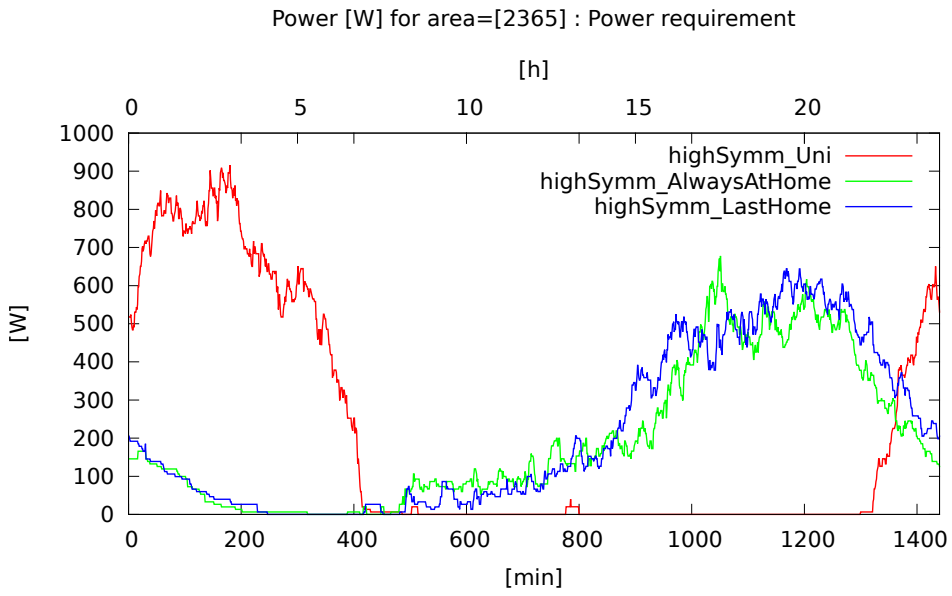


Figure 2.3: Power demand for EV charging as a function of time. The red line holds for *UniformLowCost* (cost minimizing, random), the green line for *AlwaysAtHome* and the blue line for *LastHome*. Market share EV is 0.1 of total fleet and CC share is 0.1 , PHEV shares of EV fleet are: 1.0 for CC and 0.5 for POC.

frequency for each usecase partition and scenario. BEV and PHEV owners are assumed to share the same charging behavior.

- Table 2.6 shows absolute and relative energy demand for the scenario where 10% of the cars are EV and BEV/PHEV ratio is 50/50. Almost 60% of the energy consumption is by PHEV, almost 94% by privately owned cars.

2.8 Conclusion

Schedules predicted by the FEATHERS ActBM have been used to predict energy demand and power peaks due to electric vehicle charging as a function of time and location for several EV market penetration scenarios and PHEV/BEV ratios. For the Flanders case, 78% of distances traveled daily using a single car on working days, seem to be BEV-feasible assuming that EV categories deployment conforms to current one for ICEV. Secondly, replacing BEV by PHEV increases electric energy consumption because PHEV can exploit their full electric range. Finally, the current reduced rate

Partition	Home charging scenario							
	EarlyLowTariff		UniformLowCost		LastHome		AlwaysAtHome	
	FracOp	NumCh	FracOp	NumCh	FracOp	NumCh	FracOp	NumCh
NW	0.853	1.000	0.850	1.000	0.854	1.000	1.000	1.196
W_POC	0.822	1.000	0.822	1.000	0.823	1.000	1.000	1.262
W_CC_CAW	0.911	2.194	0.914	2.199	0.905	2.203	1.000	2.450
W_CC_NCAW	0.817	1.000	0.828	1.000	0.821	1.000	1.000	1.260

Table 2.5: Fraction of charge opportunities used (*FracOp*) and number of charge operations per day (*NumCh*) for each scenario and partition (N: No, W: Work, POC: Privately Owned Car, CC: Company Car, CAW:Can Charge at Work)

Partition	Energy [MWh]			Relative
	BEV	PHEV	Total	
NW	280.346	414.969	695.315	0.421
W_POC	363.328	486.132	849.460	0.515
W_CC_CAW	25.846	31.915	57.761	0.035
W_CC_NCAW	20.126	28.110	48.236	0.029
Total	689.647	961.126	1650.773	
Relative	0.418	0.582		1.000

Table 2.6: Absolute and relative daily energy demand when 10% of cars are EV and 50% of the EV are PHEV both for Privately Owned Car (POC) and Company Car (CC) for scenario *AlwaysAtHome*

electricity period is sufficiently long to allow for charging period distribution over time in order to avoid unwanted power peak demand while allowing people to minimize cost.

2.9 Future research

Although activity-based models have a firm statistical basis, some aspects of reality do not yet have been translated to AB-model parameters. Therefore, this study shall be the base for following research paths.

On one hand, more accurate technical and market related data need to be determined from literature, surveys and experimentation. Data about distance specific energy consumption in real situations are based on measurements that use official standards and are underestimated: they need to be refined (cabin clima effects). The amount of car users who are able to charge at home has not been considered a limiting factor for the current study but could be one of the main factors when estimating EV market share.

The software will be extended to remove the constraint of using a single vehicle for schedule trips executed by multi-car households. The behavioral model is to be extended to integrate car selection decisions based on the actor specific charging decision strategy.

Finally, AB-models and smartgrid models need to get integrated in a closed loop. Since typical activity-based models account for price elasticity and allow for learning, results feedback allows for evaluation of smartgrid strategies for *charging timeslot* allocation. Evaluation of the *vehicle to grid* (V2G) concept requires integration of smartgrid controllers with AB-models.

2.10 Critical Reflection

1. The possible inaccuracy for the location choice is not an issue in this research because the number of arrivals at each location and the distance driven seem to be predicted quite well according to the research reported in Knapen et al. (2014c) (the inaccuracy mentioned in section 1.4 applies to *location pairs*).

2. According to Qiong et al. (2015) the relative error on the average daily distance driven is less than 10% with a confidence interval of 95% for 70% of the TAZ only after averaging the results for about 40 FEATHERS runs using a 10% population fraction. Since in the scenarios considered for EV, most of the charging is done at the home locations, this figure can be used to estimate the required computational effort. The paper constituting this chapter, was prepared using a single prediction run only whereas the results of 40 runs using 10% of the population (or 4 runs using the full population) should have been averaged in order to achieve accurate results for 70% of the TAZ. For the time dependent power demand (Figure 2.3), cases having the highest peak demand were considered. Those correspond to zones having a large population. The figures reported in Qiong et al. (2015) apply to the distribution for the mean calculated over the inhabitants in the TAZ. Hence, the cases exposing uncertainty, correspond to TAZ having a small population.

   The results in table 2.6 are aggregated over the full population and the full day and hence can be assumed to be accurate.

3. The zoning used for connection to grid transformers does not coincide with the TAZ but no detailed information can be obtained from the grid operators. Therefore, subsequent papers used a synthetic grid constructed by electrical engineers using scarce data; this was done in order to account for cases where local transformers act as limiting factors.

4. It was assumed that everyone using an EV can connect the car to the power grid for the complete parking duration in each charge opportunity location. This is realistic because low market shares (10%) were used and charging occurs only at home and work locations.

   (a) The possibility to charge at work was modeled explicitly.

   (b) Due to the near complete lack of public charging points in Flanders, all EV owners are assumed to be able to charge at home. When higher market shares are considered, the model needs to be extended. People who cannot

charge at home would not buy an EV: this requires data describing the proportion of home locations having a private car box equipped with a suitable power outlet.

5. Business trips have not been accounted for since the simulation was fed by FEATHERS predictions that cover household travel behavior only.

6. Finally, Usman et al. (2015) investigates the case of time dependent electric energy cost. The maximal amount of money that can be saved when the agenda adaptation is limited so that no activity or trip start time is changed more than 15[min] (hence particular activity duration can be compressed or expanded by at most 30[min]), is at most one euro per day. This was calculated for a hypothetical realistic time dependent variable energy cost derived from (i) the currently used domestic customer tariffs and (ii) published time dependent wind and solar power production figures. Under the given conditions, no travel behavior change by rescheduling is expected. As a consequence, the considered scenarios are plausible. It is to be expected that electric power demand peak shaving needs to be realized by intelligent car controllers. This is feasible in case sufficient charging stations are available so that electric vehicles can be connected to the grid most of the time they are parked. Typical parking duration is much longer than typical charging time which provides the required degrees of freedom for optimization by centralized control.

Chapter 3

Agent-based Models in Carpooling: Components Design

This chapter consists of paper

Knapen et al. (2013d) *Exploiting Graph-theoretic Tools for Matching in Carpooling Applications*

Related co-authored papers	
Galland et al. (2013a)	Simulation Model of Carpooling with the Janus Multiagent Platform
Galland et al. (2013b)	Multi-Agent Simulation of Individual Mobility Behavior in Carpooling using the Janus and JaSim Platforms
Galland et al. (2014)	Simulation of Carpooling Agents with the Janus Platform
Hussain et al. (2014)	Organizational and Agent-based Automated Negotiation Model for Carpooling
Hussain et al. (2015a)	An Agent-based Negotiation Model for Carpooling: A Case Study for Flanders (Belgium)
Hussain et al. (2015c)	Agent-based Negotiation Model for Long-term Carpooling: A Flexible Mechanism for Trip Departure Times
Hussain et al. (2015b)	Agent-based Simulation Model for Long-term Carpooling: Effect of Activity Planning Constraints
Hussain et al. (2015d)	An Agent-based Model for Carpooling: Effect of Strict Timing Constraints on Carpooling Trips
Related book chapter (DATASIM)	
Knapen et al. (2014e)	Agent-based modeling for carpooling
Related IMOB reports	
Knapen and Galland (2013)	Design Note : Agent-based Carpooling Model

3.1 Research Context

3.1.1 Research focusing on Coordinating Individuals

Coordination and synchronization between schedules of different individuals is largely ignored in current Activity-Based Models (ActBMs). However, it is possible that they constitute an influencing factor that cannot be ignored in particular cases of Travel Demand Measure (TDM) effect evaluation. At the time the paper presented in this chapter was written, very few research on cooperating actors in ActBMs was going on. Cooperation and coordination in ActBMs was only emerging as a research object. Ronald (2012) investigated coordination for joint social *activity* execution. In the DATASIM project it was decided to investigate joint *trip* execution.

One of the objectives of the DATASIM project was to investigate the use of large scale ActBMs in transportation research. The case of carpooling was chosen because

on one hand in it basic form it is conceptually simple but on the other hand it generates challenging research questions that were not answered in ActBM research.

The paper presented in this chapter represents early DATASIM work and describes the design of an Agent-Based Model (AgnBM).

3.1.2 Synchronization and Coordination Complexity

When introducing cooperation in ActBMs both the creation and adaptation (during execution simulation) of schedules are required to handle the synchronization problem.

1. Joint execution of both activities and trips heavily depends on space and time requirements for a set of people. While manipulating the schedules all possible combinations need to be considered (mandatory and discretionary activities, fixed and flexible times, shared and disjoint activities to be executed by the cooperating individuals). Synchronization is required
   (a) using *immutable* moments in time defined by the environment (school begin and end times, shop opening times, etc); those *authority constraints* are handled by current ActBMs
   (b) using *negotiated* moments in time (e.g. to start social activity or to schedule a carpooled trip). This in general can lead to very large networks consisting of large connected components of involved people whose schedule are mutually dependent. Creating such network constitutes a scientific challenge and finding computationally feasible ways to handle it is a second one.
2. Introducing coordination between individuals adds a level of complexity to the models. However, since this complexity effectively exists for the individual who is building a schedule, it is assumed to be worth the effort to integrate coordination and mutual synchronization in the model. This is because of the mental effort required for scheduling the planned activities. It is yet unknown whether this complexity is a factor that is as important as other factors taken into account while evaluating TDMs (like travel cost, public transport service level, VOT, etc). The higher the schedule independence, the lower the mental effort required for replanning when something unexpectedly goes wrong in a cooperation. There might be a *value of independence* which causes a trend to avoid interconnecting schedules.

3.1.3 Motivation to focus on Carpooling

Alternatives for solo-driving are focused by research because of environmental considerations and problems of saturated road networks. Furthermore, budgetary constraints that apply to time-table based public transportation recently caused new kinds of collective transportation to receive research focus.

1. Carpooling is a variant of collective transportation that is among the cases that are simple to be used as an initial research model (at least the case where only two people are involved in a carpool). In its basic form it requires an exploration phase, a negotiation phase and a trip execution and evaluation phase. The structure of the basic model is well understood although data collection to feed the behavioral models is hard. The basic model also allows for straightforward extensions raising non-trivial research questions.
   Once the carpooling mode is selected as an option, following steps need to be performed in order to come to a final decision:
   (a) *participants selection*: this implies the need for an exploration phase where the individual makes use of her/his own social network or consults web-based tools that advertise requested and supplied (recurrent) trips
   (b) *trip structure*: either a simple trip consisting of chained sub-trips or a tree structured trip is used.
      i. In the *chained* trip case, the complete trip is driven using a single car. Determination of the optimal trip consists of driver assignment and trip start time selection. The driver has the longest trip among all participants and sequentially picks up and drops passengers.
      ii. In the *structured* trip case, several cars are used (Knapen et al. (2012a)). Partial trips join at Carpool Parkings (CPPs) and one car is used to continue the trip. In general such trips consist of a *join* and a *fork* tree (in practice often a join tree only).
   (c) *driver selection*: a driver is to be selected for each car involved (normally the car owner)
   (d) *car selection*: the participants need to decide which car will be used. Car ownership, car availability and car capacity play a role.
   (e) *route selection*: In the *chained trip* case, multiple (very small) vehicle routing problems need to be solved. In the *structured trip* case, multiple cars are involved. For each of them (in general multiple) of such small VRP need to be solved (see item 1f)
   (f) *join and fork locations*: in the multi-car case people meet each other at a

Carpool Parking (CPP), *join* and continue driving part of the trip together. At another location, which is not the destination for at least some of the participants, the trip can *fork* into a set of sub-trips and people continue their trips in another constellation (some might continue walking, taking a train etc). The problem of co-routing and CPP selection is discussed in Knapen et al. (2012a). The problem is to find the optimal use of cars and carpool parkings for a set of people using similar but not identical routes. One of cars is used to drive the shared part of the trips. The paper discusses the *join tree*: it consists of several trips driven by several cars joining at several CPP. When joining on a CPP, all but one of the cars is left at that location and all involved people continue the trip together in one car. After the last *join*, all people are on board of the shared car. A participant can either be picked up at home or join at a CPP. Each of the participants specifies maximum detour criteria so that a set of feasible CPPs can be determined for each traveler in advance. Finding a solution requires combinatorial optimization but the problem size in general is small because the set of candidates is known in advance and limited in size by the largest of the available cars.

(g) *trip start times*: the start time for each part of the trip is to be determined. The trip start time for each participant then can be derived. For each participant, the earliest departure times and latest arrival times can be available as hard constraints. However, in general not every moment in the feasible time window is equally preferred by the traveler.

(h) *required schedule adaptation*: in general, trip duration and timing to meet the constraints will differ from the values that hold for the solo-driving case. As a consequence, deriving a carpooled schedule from a solo-driven one, requires adaptation.

The sub-problems mentioned in general are intensely interwoven. Passenger pick-up order has an impact on the distance driven and on the trip start times. The locations involved and their mutual distances impact the car (and driver) selection because of their effect on the total distance to be driven by the car. In the *structured* trip case, CPP selection affects car selection as well as the composition of the passenger groups in partial trips (e.g. in the *join* tree). Some of the problems mentioned require the solution of (moderately sized) combinatorial optimization problems each one of which is not expected to cause computational problems. Their interconnection via the constraints in the participants schedules however can raise challenging problems.

2. On one hand people can largely reduce travel cost and on the other hand the observed share of carpooling in travel surveys is about 7 to 9% for commuting despite incentive programs. This raises the research question '*why is solo-driving that popular ?*' The low share of carpooling observed in travel surveys might be caused by spatio-temporal lack of similarity in travel patterns, by schedule synchronization difficulties, by VOT consideration or by still other factors. Trasarti et al. (2011) investigated spatio-temporal similarity of recorded trips in Italy and conclude that up to 32% of the trips can be carpooled provided that people accept a 2.5[km] walk and a schedule adaptation by one of them of 1[h] (or to be divided among the participants in a pair of carpoolers).
   The low share of carpooling in relation to the achievable cost savings, suggests that *reluctance to schedule adaptation* or *conservation of independence* might be a determining factors in the carpooling decision and hence in TDM effects research.
   The bachelor thesis project reported in Van Aerschot (2014), conducted a survey among effective carpoolers. From the 136 respondents, 111 claimed that no schedule adaptation was required to enable carpooling. This could mean that carpooling is evaluated as a travel mode option only in case it fits the schedule people are used to.
3. Several studies investigate factors influencing carpooling. The majority of them are based on stated preference surveys. None of them focuses on schedule adaptation and synchronization problems.

It was concluded that (i) the model is sufficiently well defined to start a research effort, (ii) coordination is considered a possible determining factor and (iii) the effect of coordination among agents required for joint traveling, was not investigated before.

3.1.4 Carpooling Projects Context

The context for the work on carpooling is summarized in general terms as follows:

1. Several *exploration phase support* systems are operational where people can post their trips (supply and demand). In general, planned or required trips need to be matched interactively by making use of queries based on time, location and other attribute values filtering.
2. Matching problems related to carpooling have been researched by the computer science community but behavioral aspects and the requirement for schedule adaptation are ignored.
3. Several studies try to determine factors influencing the decision to carpool from

surveys. None of them was found to focus on coordination and none was found to deliver behavioral data useful to feed a negotiation simulator.

A notable exception is the work by Kamar and Horvitz (2009) describing a comprehensive agent-based model for carpooling which focuses on a fair payment system.

The paper which constitutes this chapter is based on preliminary research reported in Knapen and Galland (2013) which is an internal report (software design document focusing more on the behavioral model and data sources to meet the associated data requirements). This report also served as a design document for the work mentioned in section 3.1.4.2. Two lines of research emerging from this internal report are described in the following two subsections: (i) the matching problem in carpooling advisor systems and (ii) AgnBMs to simulate decision making w.r.t. carpooling.

The last subsection covers projects supporting the carpooling research at IMOB, all of which I contributed to and most of which I managed.

3.1.4.1 Advisor Service

The case of an *advisor service* for carpooling is considered: people post their requests and offers and the automatic service is expected to supply optimal matching advice.

Testing such service and evaluating its behavior during the startup phase are non-trivial problems. Therefore, an AgnBM is planned to be used as an exerciser for the advisor service.

This chapter describes both the advisor and the exerciser. The next chapter focuses on the estimation of the size of the matching problem to be solved by the exerciser.

Ben-Arroyo Hartman et al. (2014) significantly extends the work presented in this chapter that only touches the matching problem in bipartite graphs (which is unrealistic but was done in order to explore the domain).

3.1.4.2 Agent-based Model

An AgnBM is being developed starting from Knapen and Galland (2013). The purpose is to build an operational model containing agents that model decisions to carpool in a realistic way so that the model is capable to generate predictions about the carpooling situation in Flanders and to evaluate TDM measures. The papers mentioned in the introduction to this chapter show the evolution of the development. The model starts from TAZ based schedules predicted by FEATHERS. Hussain et al. (2014, 2015a,c,b,d) describe a model that simulates the *exploration*, *negotiation* and *execution* phases for a simulated period of several months (current experiments use

a period of 150 working days). Carpooling for commuting is modeled. In the exploration phase, agents living in the same TAZ and working in the same TAZ (different from the home-TAZ) can send invitations to carpool to each other. Currently the negotiation process itself is not modeled: there are no proposals and amendments sent back and forth between agents in order to evolve to an agreement. Rather, all agents involved in a negotiation submit information about their timing constraints and supply a trip start preference function. The *outcome* of the negotiation process (the final decision) then is determined by calculating the probability that the negotiation will succeed. If the probability exceeds a given threshold, a carpool is formed. Carpools are restricted by the capacity of the car used (not limited to pairs of individuals).

The model assumes that the trips are executed as agreed. Actual trip execution, unsafe driving and individuals being late to start the trip are not modeled. Hence the system does not include a reputation evaluator.

The main purpose is to find out under which assumptions about schedule flexibility the observed carpooling share in the modal split can be reproduced. After this, the model will be usable for TDM measures evaluation.

Trip start preference functions have been recently added to the model. Details are described in Khan (2015). The model uses a *suitability function* similar to the one described in this chapter. The trip start time for the carpool is the time for which the product of the preference functions is maximal (which is slightly different from the method described later in this chapter).

3.1.4.3 Supporting and Related Work

Following theses under my supervision, support the carpooling research and the project results are expected to contribute to the construction of models being built.

1. van Haperen (2013)) investigates the usability of OpenStreetMap (OSM) in carpooling research. He integrated all Flemish carpool parkings in OSM starting from a publicly available data study published by the government.
2. Van Aerschot (2014) conducted a survey among effective carpoolers in order to find out whether schedule inflexibility is an inhibitor for carpooling.
3. Huijbregts (2015) assigns each Carpool Parking (CPP) to the TAZ it is contained in. Then home-work and work-home trips are extracted from schedules predicted by FEATHERS. A CPP is inserted in each trip converting *home-work* to *home-CPP-work*. The resulting two-part trips for which the *CPP-work* parts are sufficiently similar in time and space are combined in pairs. The gain of the resulting trip is defined by the cost savings resulting from carpooling (the

longer the co-driven distance, the better). High gain trips only are retained in order to maximize overall gain. Specific heuristics are used since the problem is equivalent to finding a maximum matching in a graph. It is not required that carpool partners are homed in the same TAZ. The combined trips allow to calculate the number of cars on each CPP. Results are compared to observed values. The correlation is low: this can be caused by using TAZ and hence ignoring local details of CPP accessibility. A greedy algorithm was used by selecting high gain pairs first, hence probably the optimum is not found. Further research is required to find out whether the use of the greedy algorithm causes the low correlation.

The currently ongoing ICOMflex project (in which I actively participate) focuses on the transition of company organization to flexible work time and location. Details are found at `http://www.vim.be/projects/icomflex`. In this project, IMOB is responsible for the collection of diary data and for the estimation of the effect of the introduction of time and location independent working (Tijd en Plaats-Onafhankelijk Werken (Time and Place Independent Working, ICOMflex project context) (TPOW)) on travel demand. The technical infrastructure to collect diary data using a *prompted recall* survey is operational (but at the time of writing, bulk data collection did not yet start). GPS traces are recorded using smartphones and uploaded to a server. Stops are detected automatically and recorded in a database. Each participant annotates her/his trips and stops via a web application (`http://imobwww.uhasselt.be/ICOMflex/`). For each trip the user specifies the mode and for each stop, some attributes of the activity need to be entered. For each activity period the participant is asked to specify for both the start and end times, whether or not the moment in time could be chosen autonomously by the individual. Autonomous selection of the moment in time means that the selection was not restricted by coordination with other people or constrained by any rule. This survey aims to find out to what measure individual schedules are interconnected because this information is not available in the Belgian time use survey. The results are expected to be useful for carpooling research too.

3.2 Abstract

An automatic service to match commuting trips has been designed. Candidate carpoolers register their personal profile and a set of periodically recurring trips. The *Global CarPooling Matching Service (GCPMS)* shall advise registered candidates how to combine their commuting trips by carpooling. Planned periodic trips correspond to nodes in a graph; the edges are labeled with the probability for success while negotiating to merge two planned trips by carpooling. The probability values are calculated by a learning mechanism using on one hand the registered person and trip characteristics and on the other hand the negotiation feedback. The probability values vary over time due to repetitive execution of the learning mechanism. As a consequence, the matcher needs to cope with a dynamically changing graph both with respect to topology and edge weights. In order to evaluate the matcher performance before deployment in the real world, it will be exercised using a large scale agent-based model. This paper describes both the exercising model and the matcher.

3.3 Introduction

An advisory service for carpooling while commuting is to be built. People will register their periodic commuting trips: the base period typically is one week i.e. a specific pattern valid for working days is repeated after every seventh day. Considering one week periods accommodates for most situations (including part-time workers).

People who are able to fulfill all their carpooling needs within their own social network of acquaintances (*local exploration*), are assumed not to need the service.

Others will need to explore the set of carpooling candidates yet unknown to them and managed by a web based Global CarPooling Matching Service (GCPMS): this is called *global exploration*. The matching service integrated in the GCPMS shall determine which trips are best suited to be merged for carpooling and shall provide advice by suggesting people to start a negotiation with respect to a specific periodically executed trip.

Testing software that implements a GCPMS is essential because providing inaccurate or wrong advice initiates negotiations having a large probability to fail, which can expel customers. Deployment of a GCPMS shall go flawlessly because lost customers will be reluctant to return. However, the integrated matching mechanism can only be verified for operational fitness by testing under real world conditions. This complicates testing since (i) the GCPMS requires a critical mass of registered users in order to operate effectively and efficiently, (ii) performance and effectiveness need

to be evaluated on a running system because they are very difficult to predict from design data only and finally (iii) the behavior of the advisor (with respect to accuracy) during the initial phase is difficult to predict and observations made are difficult to interpret; this phase corresponds the startup transient phenomenon where stable operation has not yet been reached.

Therefore we propose the use of an agent-based model simulating the customer community in order to exercise the matching service for testing and system validation. This paper describes aspects of the combined setup of the Multi Agent System (MAS) exerciser and the GCPMS under test and focuses on the required matching component.

The remainder of the paper is organized as follows. Section 3.4 presents related work. Section 3.5 explains the principle of operation for the GCPMS and shows the test environment setup using a MAS. Section 3.6 discusses several functions used to model domain specific concepts and shows how different functions can be required to implement a particular concept in the GCPMS and the MAS respectively. Section 3.7 explains how the GCPMS determines the probability for the negotiations to succeed. Section 3.8 describes the problem of matching along with some proposed solutions. It also presents an early experiment to estimate computational performance. Finally sections 3.9 and 3.10 present future research directions and conclusions respectively.

3.4 Related Work

In recent years, agent-based simulation has entered the field of transportation science because of its capability to analyze aggregated consequences of individual specific behavior changes.

Luetzenberger et al. (2011) investigate the effect of environmental conditions and plans to incorporate the agent interactions required when carpooling.

Kamar and Horvitz (2009) describe an agent-based model aiming to optimally combine demand and supply in an advisory system for *repeated ride-sharing*. The authors focus on the mechanisms required to model users cooperating on joint plans and on the economic value of the shared plans; this research focuses on the fairness of the payment system but does not consider the ride-share demand and supply change in time.

Agatz et al. (2010) focus on the problem of dynamic non-recurring trips which is related to commuting carpooling but requires different solution concepts. Both maximal individual advantage and system wide optimum are considered.

Chun and Wong (2003) describe a group negotiation protocol for agreement on agenda schedules. A group can consist of two or more agents. The negotiation mechanism is based on ideas drawn from the A* shortest path algorithm. Each agent is assumed to specify its most preferred option first and to specify consecutive new proposals in non-increasing order of preference. Each one uses a private (i.e. not published) utility function. The protocol initiator makes use of a proposal evaluation function that is based on the assumption that agents behave as mentioned before. Versions using preference feedback by agents and conflict resolution by the initiator are reported to result in nearly optimal solutions using a quite small number of negotiation rounds.

Knapen et al. (2012a) study the problem of finding an optimal route for co-traveling. The origin (home) and destination (work) locations are given for each individual as well as a set of carpool parkings. Each of those home, work and parking locations are possible transferia (locations where to change travel mode or to change vehicle) where one can join or leave a carpool. Each individual declares the maximal time and/or distance that is acceptable to move from origin to destination. The combined route (co-route) that solves the problem consists of a join part and a fork part. In the join tree, carpoolers enter the main driver's car at several locations and times. In the fork tree they successively leave the car and, if not at their destination, continue their trip by other means. The paper proposes an algorithm to find the optimal solution for the join tree.

Varrentrapp et al. (2002) provide an informal and formal problem statement for the LTCPP. Then the soundness of the problem formulation is argued and some properties of the LCPP are proved. Finally the problem is proved to be NP-complete. This paper assumes that pools are stable in time and that every member in turn acts as the driver (round robin concept).

Manzini and Pareschi (2012) describe an interactive system to support the mobility manager (officer) operating on the LTCPP. The proposed methods and models make use of clustering analysis (CA). The basic hypothesis is that in a group the driver of the shared car turns among the participants (similar to (Varrentrapp et al., 2002)). Clustering procedures using methods available in standard Decision Support System (DSS) are proposed. After clustering, for each driver a Traveling Salesman Problem (TSP) is to be solved. Similarity measures are used but not discussed in the paper. The result is a Graphical User Interface (GUI) based interactive system that can be applied to company employees. A case study for a public service in the city of Bologna is presented. Experiments show that the overall relative saving in distance and time increases with the number of participants.

Iwan and Safar (2010) describe mining algorithms to discover *user link* and *location link* patterns respectively. User link patterns focus on similarity between the sequences of locations visited by individuals. Location link patterns apply to sequences of locations. Both are relevant when trying to estimate the probability for people to be able to carpool.

Trasarti et al. (2011) derive *travel routine* from sets of GPS traces. Similar trips are extracted (based on space and time-of-day). A routine is defined as a sufficiently large set of similar traces belonging to an individual. A profile is a set of routines. Based on the assumption that the passenger walks for a given maximal distance to (from) a location where (s)he is picked up (dropped off) by a driver, an upper bond for carpooling is determined using travel profiles.

Person traces can provide more information than car traces. Carpooling induces mutual dependency and hence additional uncertainty about the drivers and passengers timeliness. Exchanging location information is used to help solving this problem. This however requires energy efficient localization techniques and ubiquitous coverage like the one presented in Papandrea and Giordano (2013).

Xiao et al. (2012) infer social ties between people from Semantic Location Histories (SLHs). GPS trajectories first are annotated by assigning a meaning to each visited location. Then user's movements are modeled as sequences of semantically annotated locations. Finally, similarity between users is calculated by comparing their semantic location sequences. The method is demonstrated using the public GeoLife GPS trajectories data set.

Ronald (2012) presents a multi-agent system that models joint social activity execution. Although not focusing carpooling directly, it is important in this context because it focuses on cooperative activity execution while simulating daily agendas which is related to the concept of co-traveling (cooperative trip execution).

Finally, a large body of literature (Nijland et al. (2009), Guo et al. (2012), etc) has been published about the concept of rescheduling activities in a daily agenda. This however, considers agenda adaptation to unexpected events as opposed to rescheduling in the context of negotiation to cooperate. Arentze et al. (2005) present an overview of the Aurora activity-based model for schedule generation and adaptation. People are simulated as individual agents. A comprehensive model has been specified describing the insertion, re positioning, deletion and substitution of activities as well as changing locations, trip chaining options and transport modes. Models of this level of detail are required to integrate carpooling concepts in a simulator. The paper describes the use of *Aurora* in an experimental setup to study schedules consisting of work activities and green activities in several scenarios.

3.5 Carpooling Model

3.5.1 Problem Context

This paper focuses on carpooling for commuting i.e. planned periodic cooperative traveling, not on ad-hoc ride-sharing where people try to find companions for a single ride in the very near future (usually within the same day). In order to find carpooling companions, people who did not find a suitable partner by exploring their private network, register themselves with the GCPMS service. Registration implies first posting some characteristics describing the individual like age, gender, education level, special interests (like music style preferences), job category, driver license availability, etc. Those qualifiers are used because it is known that continued successful cooperation between people requires a minimal level of similarity (McPherson et al., 2001).

Secondly, people post information about each trip they periodically plan to execute: those data consist of source and destination locations, earliest and latest departure and arrival times, the maximal detour distance and delay that are acceptable, as well as the availability of a car (possibility to drive). Note that a particular driver license owner can be unavailable for driving on a specific day of the week because the family car on that day is in use by her/his partner.

Periodic trip executions need to be matched, not people. A periodic trip on Wednesday from A to B leaving at about 08:30h needs to be matched with another one having similar characteristics. Of course, the people involved shall be mutually compatible but they are not the primary subject of matching. A particular individual can periodically carpool with several people for different trips in the week (on Monday with colleague A, on Tuesday with neighbor B who differs from A). Periodic trip execution is abbreviated by *periodicTripEx* in the remainder of the text.

After having found a good match (details on how to do so will be explained below) the matcher conveys its advice to the candidates involved (the owners of the matched *periodicTripEx*); they evaluate the proposal, negotiate about carpooling and possibly agree to cooperate. Note that this negotiation is not guaranteed to succeed. One of the reasons is that the individuals dispose of more information during the *negotiation process* than the service does during the *matching process*. Therefore, the candidates convey the negotiation result back to the matcher service. This paper assumes that sufficient (financial) incentives are in place in order to make this happen. The feedback is used by a learning mechanism incorporated in the matching service. After receiving the feedback, the matching service disposes of the *periodicTripEx* and the individuals

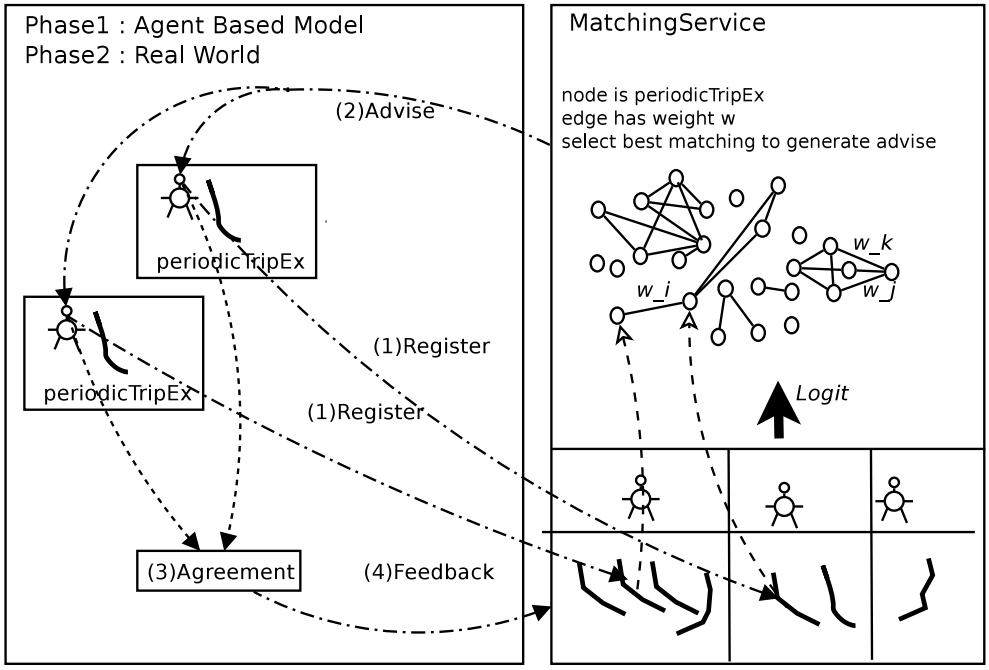


Figure 3.1: Application context: the right hand side shows the *matcher service*. People register some descriptive data about themselves and trips to be executed periodically (*periodicTripEx*). Those constitute a graph : the edges are labeled with the probability that negotiation will succeed when the trip owners are advised to carpool. Negotiation result is fed back to train the *logit* predictor. The left hand side shows the entities exercising the *matcher service* in consecutive phases.

characteristics as well as of the negotiation result; those are used to train a predictor. Please refer to Fig. 3.1 for a high level overview of data flows, relations and method activation.

It is important to note that the first implementation focuses on *pairs* of commuters carpooling. This is essential to the problem of edge weight determination and it allows the advice to be based on *binary matching*.

The model used for matching consists of a directed graph; by convention, each edge points to the *periodicTripEx* whose owner will be the driver. Each vertex corresponds to a *periodicTripEx*. A vertex for which the owner is unable to become the driver, never can be a target edge (its *indegree* equals zero). Every edge is labeled with the estimated probability for the negotiation to succeed. Two vertices are connected by an

edge if and only if it is worth to advise the *periodicTripEx* owners to start negotiating: in general the graph is incomplete. The need to determine the probability threshold to include an edge in the graph is one of the main reasons to use a MAS to exercise the GCPMS. Note that

1. the set of vertices evolves over time because people register and withdraw *periodicTripEx* as time evolves and because people join and leave the carpooling candidates society (removing all their *periodicTripEx* in the latter case).
2. edges emerge as soon as the negotiation success probability exceeds a given threshold; this can be caused by changes in the *periodicTripEx* (e.g. by relaxing the time constraints) and people characteristics respectively (e.g. by reputation changes (see below)).
3. probability estimates can change over time by re-training the predictor. Note that this can cause threshold crossing and hence edge creation or deletion.

Finally the problem size can grow large when a nation-wide service is considered. Large scale deployment probably is a necessary condition for both effective operation (delivery of advice that has a high success probability) and economic viability. The matcher needs to cope with large networks whose topology and edge weights evolve in time. This represents a complex problem and hence thorough evaluation before deployment.

3.5.2 Basic Concepts - Definitions

Some definitions for concepts in the application domain are required to explain the functions used to calculate the edge weights for the *periodicTripEx* graph. Refer to Fig. 3.2 for an overview of the datasets and relations involved. Fig. 3.3 summarizes the essential application domain functions.

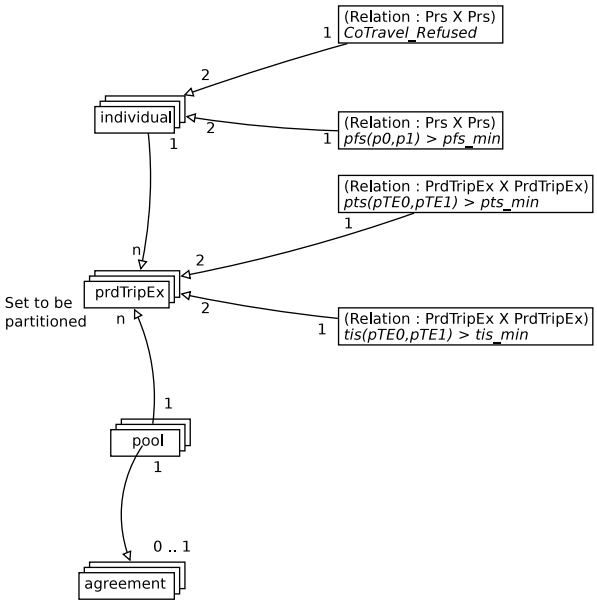


Figure 3.2: Overview of sets (*individual*, *periodicTripEx* and *agreement*) used in the model and relations those sets are involved in.

3.5.2.1 Symbols Used

$\mathcal{A}$	: the set of all *agreements* (see definition 3.5.2)
$\mathcal{I}$	: the set of all individuals
$\mathcal{P}$	: the set of all pools (see definition 3.5.3)
$range(TOD)$	: $24 * 60 * 60$ (time-of-day)
$range(TOW)$	: $7 * range(TOD)$ (time-of-week)
$\mathcal{T}$	: The set of all *periodicTripEx*'s (see definition 3.5.1)
TOD	: Time of day ; if expressed in seconds, cardinal $\in [0, range(TOD) - 1]$
TOW	: Time of week ; if expressed in seconds, cardinal $\in [0, range(TOW) - 1]$
$t_{\cdot,early}, t_{\cdot,late}$	: Earliest resp. latest time
$t_{d,\cdot}, t_{a,\cdot}$	: Departure resp. arrival time

3.5.2.2 Definitions

Definition 3.5.1 (periodicTripEx)**.** *A periodicTripEx is a tuple* $(i, O, D, w, t_{d,early}, t_{d,late}, t_{a,early}, t_{a,late})$ *where* $i \in \mathcal{I}$*,* O *and* D *denote the origin and*

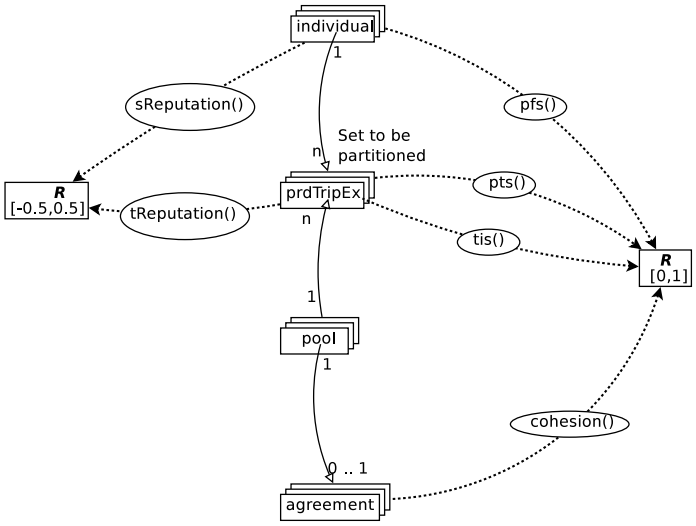


Figure 3.3: Overview of functions defined on the sets (*individual*, *periodicTripEx* and *agreement*) used in the model and functions those sets are involved in. Continuous lines represent references, dashed lines represent functions.

destination locations respectively, $t_{.,.} \in TOW$ *and* w *denotes a start-of-week moment in time so that the first trip execution for the given periodicTripEx starts in* $w, w+1$.

Notes:

1. A *periodicTripEx* denotes the weekly execution of a trip with given characteristics by a specific individual. Individual i is called the owner of the *periodicTripEx*.
2. Examples:
   (a) $t_{d,late}$ is: *Wednesday at 08:20h*
   (b) w = 2012-jun-04 00:00:00
   (c) Refer to Fig. 3.4 to see intervals overlap.

Definition 3.5.2 (agreement)**.** *An* agreement *specifies operational details about the collaborative execution of all elements in the list of* periodicTripEx *to which the* agreement *applies. An agreement specifies the moment in time at which it starts to hold.*

Notes:

1. An *agreement* has no termination time
2. A *periodicTripEx* is referred to by (belongs to) at most one *agreement*
3. *Agreement* details cover: timing, routing and driver selection.

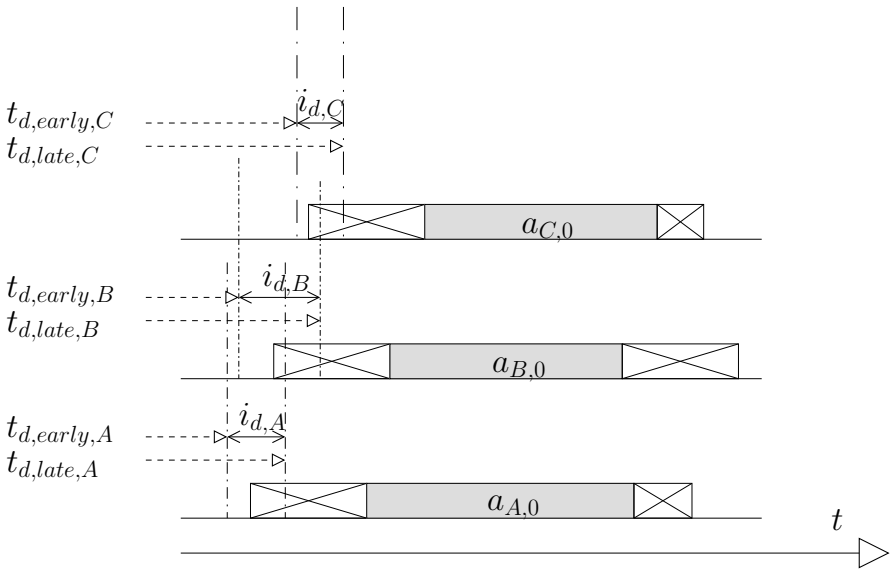


Figure 3.4: Activities $(a_{A,0}, a_{B,0}, a_{C,0})$ for individuals A, B and C and the associated trips. The valid departure intervals $i_{d,A}, i_{d,B}, i_{d,C}$ are shown. Note that B can choose to co-travel with A or C but A and C cannot co-travel.

Definition 3.5.3 (pool)**.** *A* pool *is a tuple* (A, T) *where* A *denotes a set of* agreements *negotiated by the cooperating partners and* T *is a nonempty set of* periodicTripEx *so that* $(A = \emptyset \wedge \|T\| = 1) \vee (\|A\| = 1 \wedge \|T\| > 1)$

Notes:

1. The condition states that there is either a single individual without any *agreement* or multiple cooperating individuals sharing a single *agreement*.
2. Each individual occurs in at most one *periodicTripEx* in a specific *pool*:

$$\forall t_0, t_1 \in T : (t_0 \neq t_1) \Rightarrow (t_0.owner \neq t_1.owner) \tag{3.1}$$

where $t.owner$ denotes the individual owning the *periodicTripEx*.

Definition 3.5.4 (profile similarity)**.** Profile similarity *is a value in* $[0, 1]$ *assigned to a pair of individuals that indicates to what extent the individuals are compatible for carpooling (homophily concept described in McPherson et al. (2001)).*

Definition 3.5.5 (pooled trip execution)**.** *A* pooled trip execution *(abbreviated by* pooledTripEx*) is the cooperative execution of a set of trips using a single car and a single driver.*

Note: In this context, it is assumed that each pooledTripEx is driven by exactly one driver. More complex cases are covered by Knapen et al. (2012a).

Definition 3.5.6 (path similarity)**.** *Path similarity is a value in* $[0, 1]$ *assigned to an* ordered *pair* (pte_0, pte_1) *of* periodicTripEx *that indicates to what extent the OD (Origin, Destination) pairs involved in the respective trips, are compatible for carpooling in case the owner of* pte_0 *is assigned to be the driver.*

Notes:

1. Path similarity defines a function of *periodicTripEx* that is not symmetric in its arguments. This is easily seen because the distance driven depends on the driver selection; the driver needs a detour to pick up passengers.
2. When the *single driver* constraint in *pooledTripEx* is dropped, paths driven no longer are strings of path segments but consist of *join* and *fork* trees which requires a more advanced concept of path similarity which has been discussed in Knapen et al. (2012a).

Definition 3.5.7 (tiSim)**.** *Time interval similarity (*tiSim*) is a value in the range [0,1] assigned to a pair of time intervals specified by different people, that indicates to what measure the intersection of the intervals can be used for a specific act of cooperation.*

Definition 3.5.8 (depArr_tiSim)**.** *Departure/arrival time interval similarity (*depArr_tiSim*) is a value in* $[0, 1]$ *assigned to an ordered pair* (pte_0, pte_1) *of* periodicTripEx *having identical origins and identical destinations; it indicates to what extent the time intervals involved are compatible for carpooling.*

Notes:

1. Compatibility for car pooling requires a minimal amount of intervals overlap (see Fig. 3.4).
2. Due to the *single driver* constraint (see definition 3.5.5, the route for each passenger's trip shall be included in the route for the *pooledTripEx* which is the route for the driver.
3. Time interval similarity can be calculated only for a pair consisting of the passenger trip and the part of the driver's trip for which the route coincides with the passenger trip route (because identical origins and destinations are required).

Definition 3.5.9 (sReputation)**.** *Safety reputation is a value in the range* $[-0.5, 0.5]$ *assigned to an* individual *(by the passengers) to qualify the individual as a safe driver.*

Notes:

1. *sReputation* is a characteristic of an individual because it is assumed that safe driving does not depend on the periodic trip driven.
2. The initial value for individual i is $i.sReputation = 0.0$ (which means *neutral*).

Definition 3.5.10 (tReputation)**.** *Timeliness reputation (or accuracy reputation) is a value in the range* $[-0.5, 0.5]$ *assigned (by the co-travelers) to a* periodicTripEx *in an* agreement*: it indicates to what measure the owning individual respects the timing when executing the periodic trip in the* agreement*.*

Notes:

1. *tReputation* is defined for both drivers and passengers.
2. *tReputation* has been defined as a characteristic of a tuple *(periodicTripEx,agreement)* and not as a characteristic of an *individual* or of a *periodicTripEx* because an individual can behave differently on a specific *periodicTripEx* pte_0 in different agreement contexts (pools). Example: the interval between a pick-drop activity to be executed by individual i_0 and the start of the *periodicTripEx* in a given agreement a_0 is too short so that it is difficult for i_0 to meet the timing requirements of a_0. Within a different agreement a_1 timing constraints for pte_0 can be less severe so that the owner can meet them easily.
3. The initial value for periodicTripEx pte in a is given by $(pte, a).tReputation = 0.0$ (which means *neutral*)

Definition 3.5.11 (cohesion)**.** *Cohesion qualifies the strength of an* agreement *using a value in* $[0, 1]$ *that is a function of attributes of the* agreement *only.*

Notes:

1. An *agreement* with a high *cohesion* value is less likely to be broken whenever some of its *periodicTripEx* get a proposal to set up a new cooperation. Cohesion determines the resistance to breakdown in case opportunities for recombination come available.
2. *Cohesion* does not depend on *sReputation* since that is an attribute of an *individual* and not of an *agreement*.
3. The matcher shall derive *cohesion* from individual's negotiation feedback since individuals are assumed not to be prepared to specify and maintain *cohesion* values; furthermore, they are unable to do so since no universally valid scale or method is available.

Note: In order for carpooling (co-traveling) to be suitable, both the collective departure and arrival time intervals shall be suitable for each participant.

3.5.3 Exploration/advisory and Negotiation Phases

Matching is applied in both *local* and *global* exploration phases. In both cases, matching precedes the negotiation phase where final decisions to carpool are taken. Mechanisms used in the exploration/advisory phases shall be consistent with mechanisms in the negotiation phase. It is not possible to predict the negotiation phase with certainty; reasons are:

1. Negotiation covers driver selection, co-route determination and re-scheduling (daily planning adaptation) for the cooperators. Schedule adaptation makes use of *VOT* (individual specific *Value Of Time*). An advisory mechanism does not have all required data available nor has any knowledge of the private goals (and in general the *Beliefs, Desires, Intentions (BDI)*) of the individuals (agents) involved in a negotiation. Individuals need to adapt their daily agenda because the decision to carpool introduces mutual dependency and hence additional constraints. On one hand, those constraints induce computational complexity at two levels: (i) agenda adaptation for a particular individual which includes re-timing, re-location, re-sequencing (combinatorial optimization) of activities along with activity dropping and replacement (Joh, 2002), (Joh, 2004),(Knapen et al., 2012d),(Arentze et al., 2005) and (ii) networked individuals need to take care of the agreements they are involved in. On the other hand, constraints induced by mutual agreements cannot be considered to be fixed; under certain circumstances they evolve over time. This induces the need for replanning. Relevant literature has been mentioned in 3.4. Since planning is involved, agents need to predict the near future. Knapen et al. (2012d) predicts future travel times using perception filters that actually implement part of the belief of an agent.
2. The total distance driven cannot be predicted by the *matcher* when carpool parkings are involved because in such cases the co-route can be tree structured: see note 2 for definition 3.5.6. Hence the path similarity function delivers only an approximation of the one involved in negotiation.
3. People are assumed to be prepared posting a minimal amount of data about the time intervals that suit them for departure an arrival respectively; candidates are supposed to specify just the interval boundaries. However, during negotiation, they can make use of *preferences* to state that one of a set of proposed intervals suits better than another one. Hence, the *trip times interval similarity* function available to the matcher is only an approximation for the one used during negotiation (see Fig. 3.5).

4. The *Cotravel_Refused* relation shown in Fig. 3.2 allows individuals to unconditionally avoid any advise to carpool with specific people. For privacy reasons, it is not possible for a refused individual to know the refusing party.

3.5.4 Principle of Operation of the Carpooling Model

1. An individual looks for other individuals to cooperate while executing *periodicTripEx*'s: this is called *exploration*.
2. *Local* exploration within the private social network (*PrivNet*) is applied before *global* exploration. If *carpool candidates* can be found within an individual's *PrivNet*, they will be contacted first (as preferred candidates).
3. *Global* exploration is applied only in a second stage when no suitable *pool* was found in the *PrivNet*. In the *Global* exploration phase, the *matcher* provides advice about which *pool*s an individual should negotiate with. This corresponds to the use of an online service by a candidate exploring the set of formerly unknown carpooling candidates.
4. If an individual joins a *pool*, (s)he is added to the *PrivNet* for all other participants in the *pool* (if still required) so that if i_0 and i_1 ever cooperated in a *pool*, they belong to each others *PrivNet*. Because links never are removed from the *PrivNet*, if i_0 and i_1 ever carpooled, $(i_1) \in PrivNet(i_0) \wedge (i_0) \in PrivNet(i_1)$.
5. Candidates register, join and leave *pool*s at random moments in time. As a consequence the main data structures dynamically change due to events external to the matching process.

3.6 Functions related to Domain Concepts

A specific concept can be implemented by different functions in MAS and GCPMS matcher.

3.6.1 Time Interval Based Functions

Table 3.1 contains a summary of the functions presented in this section. Two kinds of function are used. The *tiSim* and *tiSuitFunc* functions apply to a set of two intervals of the same kind (i.e. both are departure or both are arrival intervals) that correspond to two individuals considering to cooperate. The *depArr_tiSim* and *depArr_tiSuitFunc* are defined over 4 time intervals (i.e. both departure and arrival intervals for two people considering to cooperate).

	Context	
Function type	MAS (function of time)	GCPMS (constant)
2 time intervals of same kind (either *departure* or *arrival* time intervals for 2 agents)	tiSuitFunc	tiSim
2 pairs (*departure* and *arrival*) of time intervals for 2 agents are considered	depArr_tiSuitFunc	depArr_tiSim

Table 3.1: Summary of the time interval functions used.

For the reason mentioned in Section 3.5.3 item 3, different time interval similarity functions are used in respectively the agent-based exerciser and the matching operational service.

Remember that the trips considered shall have identical origins an destinations respectively (hence the time intervals stated by the participants shall apply to a single origin-destination pair which implies that the passenger trip embedded in the driver's trip is to be considered: see note 2 with definition 3.5.8).

3.6.1.1 Time Interval Based Functions for Negotiation

This section defines the *tiSuitFunc* (time interval suitability) and *depArr_tiSuitFunc* (departure/arrival time interval suitability) functions.

1. The departure (arrival) interval for a trip (*periodicTripEx*) is the time interval that suits the traveler to start (end) the trip. Let $pte.i_d()$ and $pte.i_a()$ denote respectively the departure and arrival intervals of the *periodicTripEx pte*.
2. Individual p_0's *preference* for a given moment in time is given by the function $f_{p_0} : \mathbb{R} \Rightarrow \mathbb{R} : t \mapsto f_{p_0}(t) \in [0, 1]$. The function is not required to be differentiable or continuous but the product of two such functions shall be integrable. For each moment in time belonging to the departure and arrival intervals, the *preference* value needs to be specified.
3. The *combined preference* function is the product of the preference functions associated with two *periodicTripEx*'s. It is essential to the negotiation process.
4. The *time interval suitability* is the integral of the combined preference over a fixed time interval. The length of the interval has a pre-specified constant C value; a suitable choice is the expected duration of the trip interruption to get someone on/off board of the vehicle. Let $t_{i_X,0}$ and $t_{i_X,1}$ denote the begin and

end times for a time interval specified by agent X. The *time interval suitability* is denoted by $S(C, i_A, f_A, i_B, f_B)$, where $i_A = [t_{i_A,0}, t_{i_A,1}]$ and $i_B = [t_{i_B,0}, t_{i_B,1}]$ are intervals specified by individuals A and B; f_A and f_B are the associated preference functions. The suitability function is given by

$$t_0 = \max(t_{i_A,0}, t_{i_B,0}) \tag{3.2}$$

$$t_1 = \min(t_{i_A,1}, t_{i_B,1}) \tag{3.3}$$

$$S(t; C, i_A, i_B, f_A, f_B) = \begin{cases} \frac{1}{C}\int_t^{t+C} f_A(x) \cdot f_B(x)dx & if \quad t \in [t_0, t_1 - C] \\ 0 & otherwise \end{cases} \tag{3.4}$$

where t denotes the start of the boarding/alighting operation. The dimension of the *time interval suitability* value is $[prefUnit^2]$. In this context, preference is assumed to be dimensionless, hence the suitability is dimensionless. During negotiation, $S(t; C, i_A, i_B, f_A, f_B)$ is used to find a suitable time to board/alight.

5. Piecewise linear functions are used because they are flexible, they can easily be specified by the user (in charge for the configuration of the agent-based model) and integration is computationally cheap. An example is shown in Fig. 3.5. The left hand part shows piecewise linear *preference* functions, their product and the associated time interval suitability (proportional to the crosshatched area under the product function).
6. Time interval *suitability* is a value in $[0, 1]$.
7. The departure/arrival time interval suitability *depArr_tiSuitFunc* is defined as

$$\begin{aligned} f(t) = & S_{dep}(t; C, i_{A,dep}, i_{B,dep}, f_{A,dep}, f_{B,dep}) \cdot \\ & S_{arr}(t + d; C, i_{A,arr}, i_{B,arr}, f_{A,arr}, f_{B,arr}) \end{aligned} \tag{3.5}$$

where d is the expected trip duration.

3.6.2 Time Interval Similarity Evaluation for Matching

1. It is not feasible to ask the individuals to register the piecewise linear preference function mentioned in Section 3.6.1.1. People are assumed to be prepared to register simply a time interval only. Hence the *preference* value is assumed to be a constant f over the time interval specified.
   The right hand part in Fig. 3.5 shows case for the same intervals where the preference function is assumed to equal one everywhere: this is the assumption

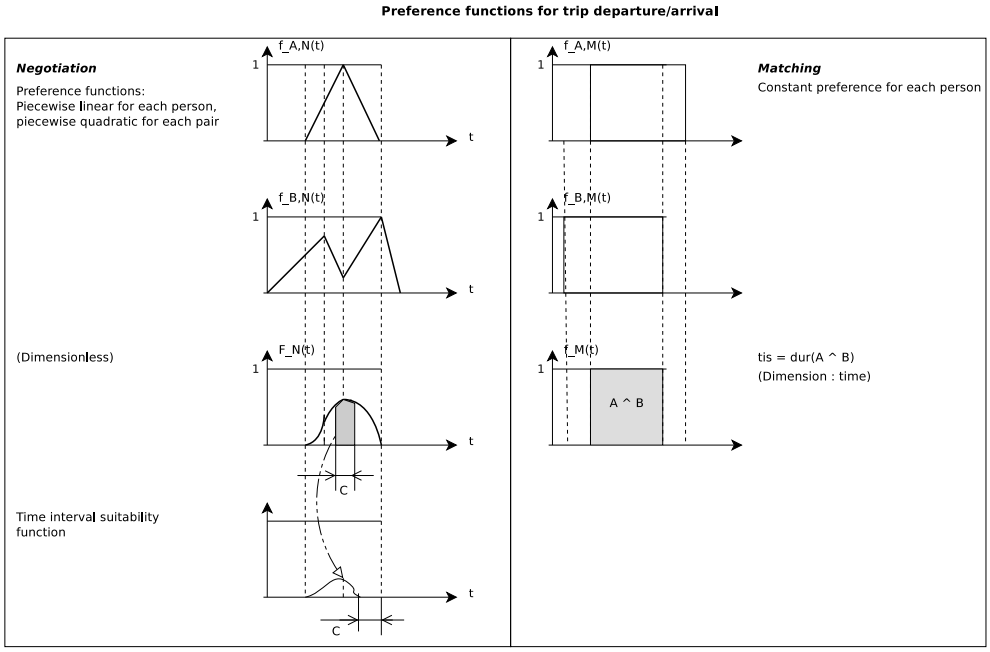


Figure 3.5: (Left) Time similarity used while negotiating: $f_{A,N}(t)$ and $f_{B,N}(t)$ are time preference functions for specific intervals. $F_N(t)$ is the *combined preference* and the size of the cross-hatched area is the resulting *time interval suitability function*. (Right) Time similarity used by the matcher: all preference functions equal 1 because users are expected to only submit feasible time intervals.

made by the matching service due to lack of information: the user only specifies the boundaries for the departure and arrival intervals.

2. The negotiation outcome is assumed to be positively correlated with the length of the intersection of the intervals associated with the *periodicTripEx*'s to compare. The value is not compared to the constant C mentioned above because this comparison would only imply a linear scaling of an independent variable which has no effect on the *logit* estimator. The time interval similarity $tiSim$ is given by

$$t_0 = \max(t_{i_A,0}, t_{i_B,0}) \tag{3.6}$$

$$t_1 = \min(t_{i_A,1}, t_{i_B,1}) \tag{3.7}$$

$$tiSim(i_A, i_B) = t_1 - t_0 \tag{3.8}$$

3. For a given pair of *periodicTripEx*'s (home-work, work-home), $tiSim$ values are fed into the *logit* estimator as two independent variables; combining them into a single value would cause a loss of information.

3.6.3 Path Similarity

3.6.3.1 Path Similarity in MAS

The first version of the agent-base exerciser does not take carpool parkings into account. As a consequence, path similarity is calculated in the same way as for the GCPMS.

3.6.3.2 Path Similarity in GCPMS

1. The GCPMS per hypothesis has no information about carpool parkings potentially being used (because that is not specified by the candidates). Therefore, it is assumed that people board and alight at home and work locations only.
2. The owner of the first *periodicTripEx* is the driver.

Table 3.2: Symbols Used (alphabetical order)

Symbol	Meaning
O_i, D_i	denote respectively the *origin* and *destination* locations for individual i (e.g. home and work locations)

Continued on next page...

Table 3.2 – Continued

Symbol	Meaning
$r(a, b, t)$	denote the route from a to b when starting at time t that is optimal with respect to some cost function $c(r)$ based on distance and travel time
$d(r, t)$	denote the duration to travel the route r starting at time t
$l(r, t)$	denote the length of the route r starting at time t
$c(r)$	denote a cost function based on route length $l(r, t)$ and route travel duration $d(r, t)$
$p_{i,solo}(O_i, D_i, t)$	denote the optimal path from O_i to D_i when individual i drives alone (solo) and starts at time t
$\overline{p}_{i,solo}(O_i, D_i, t)$	denote the optimal path from O_i to D_i when individual i drives alone (solo) and ends at time t
$p_{i,carpool}(O_i, D_i, t)$	denote the optimal path from O_i to D_i when individual i drives the carpool trip via O_j and D_j for $i \neq j$ and starts at time t
$\overline{p}_{i,carpool}(O_i, D_i, t)$	denote the optimal path from O_i to D_i when individual i drives the carpool trip via O_j and D_j for $i \neq j$ and ends at time t
$pathSim_d()$	denote the path similarity function for the case where the earliest departure is given
$pathSim_a()$	denote the path similarity function for the case where the latest arrival is given

3. Note that the minimal cost depends on the start moment in time. Also note that the optimal path can be determined either by fixing the time for the first departure or by fixing the time for the latest arrival.
4. The ratio between the lengths of the optimal routes for the driver is used as a *path similarity function*. For the *given earliest departure* case (starting at t_0) where A is the driver and the trip is $O_A \rightarrow O_B \rightarrow D_B \rightarrow D_A$, this leads to

$$t_1 = t_0 + d(r(O_A, O_B, t_0)) \tag{3.9}$$

$$t_2 = t_1 + d(r(O_B, D_B, t_1)) \tag{3.10}$$

$$pathSim_d(pte_A, pte_B, c()) = \frac{c(O_A, D_A, t_0)}{c(O_A, O_B, t_0) + c(O_B, D_B, t_1) + c(D_B, D_A, t_2)} \tag{3.11}$$

Note that t_1 denotes the time at which the carpool trip leaves O_B and t_2 denotes

the time at which the carpool trip leaves D_B. Also note that in general

$$pathSim_d(pte_A, pte_B, c()) \neq pathSim_a(pte_A, pte_B, c()) \tag{3.12}$$

since the departure times can differ. Finally note that in general

$$pathSim_a(pte_A, pte_B, c()) \neq pathSim_a(pte_B, pte_A, c()) \tag{3.13}$$

since the routes differ.

5. The departure time can have a large effect on the trip duration. In the first GCPMS this dependency is ignored due to lack of data. Because of the availability of speed profiles registered using GPS navigators, it will become feasible to take the time dependency into account (which will lead to more accurate negotiation outcome prediction) in the near future although that will require a large amount of data pre-processing and data storage. By ignoring time dependency, the equation 3.11 is reduced to

$$pathSim_d(pte_A, pte_B, c()) = \frac{c(O_A, D_A)}{c(O_A, O_B) + c(O_B, D_B) + c(D_B, D_A)} \tag{3.14}$$

3.6.4 Profile Similarity

The candidate carpooler specifies the value for a set of N_A attribute values: those constitute the candidate's *profile*.

1. The attributes can be: (a) continuous variables limited to a finite interval (b) discrete quantities for which a total order relation exists, also limited to a finite interval and (c) enumerations (discrete quantities without an intrinsic order relation).
2. Attributes that are ordinal values (cases (a) and (b)) are handled in the same way; case (a) is expected not to occur in practice. The domain is mapped onto $[0, 1]$. The distance between two attribute tuples a_0 and a_1 having N_{OA} ordinal attributes, is the Euclidean distance divided by a scale factor to normalize the distance (map to interval $[0, 1]$).

$$d(a_0, a_1) = \sqrt{\frac{\sum_{i \in [1, N_{OA}]} (a_0[i] - a_1[i])^2}{N_{OA}}} \tag{3.15}$$

Continuous variables are combined into a single distance value d_C and discrete ordinal values are combined into another one d_D. The range of d_D is a finite subset of $[0, 1]$.

3. All attributes have an equal weight.
4. The distance d_E between two vectors constituting of enumeration variables, is the number of differing attributes divided by the total number of attributes for normalization.
5. Both MAS and GCPMS use the same mechanisms to calculate profile similarity. However, the MAS can make use of particular attributes that have not been registered in the GCPMS.

3.6.4.1 Profile Similarity in GCPMS

1. The first model uses the *similarity* between two profiles as a predictor (one independent variable) for the *logit* model. It is to be investigated under what conditions its is more efficient (in terms of prediction accuracy) to feed N_A variables into the *logit* independently, each one of which is the difference between the values for a given attribute in the respective profiles.
2. The similarity values $s_C = (1 - d_C)$ (for continuous attributes), $s_D = (1 - d_D)$ (for discrete valued attributes) and $s_E = (1 - d_E)$ (for enumerations) are used as independent variables for the *logit* estimator.

3.6.5 Reputation

Both *sReputation* and *tReputation* are handled in the same way.

3.6.5.1 Reputation in MAS

1. In the agent-based exerciser a gossip based mechanism is modeled. In this case the social network is considered; the set of carpooling based connections (links) is a small subset of the social network for each individual. People keep a qualification for everyone they ever carpooled with. Furthermore, an individual can keep a qualification for another one that has been derived from *gossip*; this is a transitive mechanism. The credibility of the qualification decreases with each intermediate step involved.
2. Every agent keeps a list containing a perceived safety reputation value for a limited set of other agents. In the exerciser, a specific agent can be qualified by zero or more safety reputation values (each one of which is owned by a peer).

Every agent can determine the reputation of another agent using a method that is not specified in this section and overriding the value already in place (if any). Furthermore, everyone can adjust the reputation of peers based on *gossip* as follows. At a random moment in time, an emitter agent a_e can multicast its reputation value $R^e(q)$ to qualify agent a_q to a subset agents directly connected to it in the social network. The receiver a_r

(a) retransmits the reputation message with a given probability p_r (hence simply drops it with probability $(1 - p_r)$)

(b) adjusts its own perception of a_q with a given probability p_a (hence simply ignores it with probability $(1 - p_a)$)

Consider agents a_e, a_q, a_r, a_v that are all pairwise different. a_e emitted a qualification about a_q that reaches a_r via its neighbor a_v. If a_r did not yet have registered an opinion about a_q, the value for $R^r(q) = 0$. Reputation update by receiver a_a is done by

$$\alpha = 2^{-d(e,r)} \tag{3.16}$$

$$\beta_{r,v} \in [0, 1] \tag{3.17}$$

$$R'^r_q \leftarrow \frac{R^r(q) + \alpha \cdot \beta_{r,v} \cdot R^e(q)}{1 + \alpha \cdot \beta_{r,v}} \tag{3.18}$$

where $d(e, r)$ is the distance between emitter and receiver in the network and $\beta_{r,v}$ is the strength of the link between a_r and a_v.

3.6.5.2 Reputation in GCPMS

1. The GCPMS allows for controlled mutual evaluation of individuals with respect to timeliness and safety. Only individuals cooperating in an agreement can qualify each other; this means that the reputation mechanism in the matching service is not transitive.
2. Similar mechanisms are used for *sReputation* and *tReputation*. Qualifications received are registered in a dedicated *qualifications list* assigned to the entity they apply to; for each issuer, only the most recent qualification is kept. *sReputation* qualifications are registered with individuals and *tReputation* qualifications are registered with *periodicTripEx*.
3. Each individual has reputation values that evolve over time due to qualification by cooperators (i.e. individuals who participated in an agreement with the person being evaluated). Passengers can qualify *sReputation* for drivers. Every cooperator can qualify every other cooperator's *periodicTripEx*'s with respect to *tReputation*.

4. The attributes for a qualification are: (i) the generation timestamp $q.ts()$, (ii) the issuer $q.iss()$ and (iii) the specified value $q.rep()$.
5. In both cases, the reputation is calculated as a weighted average of the values posted in the qualification list: the weight decreases with age of the qualification and increases with the duration of the cooperation (the agreement lifetime) $a.dur(q.ts())$ up to the moment of qualification. Note that the cooperation duration in the case of *sReputation* is to be summed over a set of agreements. Let $\mathcal{Q}_i$ be the qualifications list for individual i. Let a^i_j denote an agreement in which individuals i and j cooperated. Let $a^x_y.dur(t)$ denote the lifetime of agreement between the qualified agent x and the qualifying agent y a at time t. Let A^y_x denote the set of all agreements in which individual x and y cooperate. Let α_s and β_s be parameter settings (time constants) to be determined. Then following equations determine the *sReputation*:

$$age = now - q.ts() \tag{3.19}$$

$$coopDur = \sum_{a \in A^i_{q.iss()}} a.dur(q.ts()) \tag{3.20}$$

$$w^i_{q.iss()} = \exp(-\alpha_s \cdot age) \cdot (1 - \exp(-\beta_s \cdot coopDur)) \tag{3.21}$$

$$sReputation_i = \frac{\sum\limits_{n \in \mathcal{Q}_i} q.sRep() \cdot w^i_{q.iss()}}{\sum\limits_{n \in \mathcal{Q}_i} w^i_{q.iss()}} \tag{3.22}$$

3.6.6 Cohesion of an Agreement

1. *Cohesion* is supposed to be a monotonically decreasing function of the time t elapsed since the creation of the *agreement*. *Cohesion* is a monotonically decreasing function of *pool* size s (large *pool*s are more likely to disintegrate). Note that cohesion does not depend on mutual evaluation of carpoolers; cohesion and reputation shall be independent concepts because all of them are fed into a probability estimator. Let α_c and β_c be parameter settings to be determined. The *cohesion* <u>*value*</u> is given by:

$$c = e^{\alpha_c \cdot t} \cdot e^{\beta_c \cdot (s-1)} \tag{3.23}$$

2. In the pairwise case, when considering a specific edge, exactly two *cohesion values* apply (one for each of the vertices (*periodicTripEx*'s)). Each of the cohesion values possibly applies to an *agreement*. For a given *periodicTripEx* pair (pte_0, pte_1), the c_0 and c_1 can relate to either different *agreements* or to a single

one. The meaning of the tuple (c_0, c_1) depends on the number of *agreement*s involved. Therefore, *cohesion values* are combined into a single *cohesion based indicator* using the function given in equation 3.26; the second case in equation 3.26 corresponds to the case where both *periodicTripEx* are members of an *agreement* (but not necessarily to the same one). Let $pte_0, pte_1 \in \mathcal{T}$ the *periodicTripEx*s involved. Let c_0 and c_1 denote the respective corresponding *cohesion values* and $p.T()$ denote the list of *periodicTripEx* involved in pool p. Let $pte.a()$ denote the *agreement* covering pte when it belongs to a *pool*. Let $\mathcal{P}$ denote the set of all pools. The *cohesion* <u>*indicator*</u> $\overline{c}(pte_0, pte_1)$ is a measure for the cohesion between two *periodicTripEx*'s when they already form a pair and for the feasibility to get them released when they are bound in pairs with others.

$$pte_x.a() = \begin{cases} nil & if \quad \nexists p \in \mathcal{P} | pte_x \in p.T() \\ p.a() & if \quad \exists p \in \mathcal{P} | pte_x \in p.T() \end{cases} \tag{3.24}$$

$$c_x = \begin{cases} 0 & if \quad pte_x.a() = nil \\ pte_x.a().c() & else \end{cases} \tag{3.25}$$

$$\overline{c} = \begin{cases} (1 - c_0) * (1 - c_1) & if \quad pte_0.a() \neq pte_1.a() \\ c_0 * c_1 & if \quad pte_0.a() = \\ & \quad pte_1.a() \neq nil \end{cases} \tag{3.26}$$

In case both *periodicTripEx* belong to the same *agreement*, the cohesion values are taken from that *agreement* and in fact $c_0 = c_1$. In the other case (which also covers the case where at least one of the *periodicTripEx* is not covered by an *agreement*), the complement of the *cohesion* values is used. When neither of the *periodicTripEx* belongs to an *agreement*, $\overline{c} = 1$.

3.7 Weights Determination

The weights used to label the edges in the graph, are probability values associated with the success of the negotiation process between individuals. Those probabilities are calculated by means of *logistic regression* (logit) fed by results of negotiations who have been advised by the *carpoolMatcher*.

Edges are not removed from the graph when one of the involved *periodicTripEx*'s becomes member of a pool (agreement). They are labeled with a weight in exactly the same way as the edges connecting non-bound *periodicTripEx*'s. As soon as the

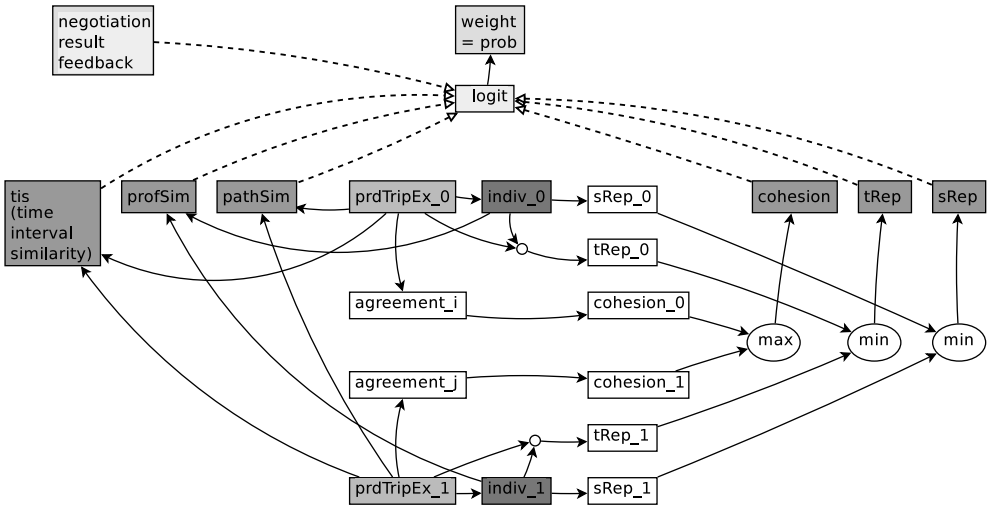


Figure 3.6: Dependencies between concepts used to calculated the weight for a edge connecting two *periodicTripEx*'s

weight of an edge e_0 linking two *periodicTripEx*'s involved in the same agreement, becomes lower than the weight of other edge e_1 sharing a vertex with e_0, the carpool members for which better opportunities occurred will get an advice to negotiate with non-pool-members to setup a new agreement.

3.7.1 Negotiation Outcome Prediction

Fig. 3.6 summarizes the data dependencies relevant to edge weight determination. From the point of view of the matcher service, the outcome of a negotiation process is a discrete variable with values : *success* (yes) and *failure* (no). Independent variables influencing the negotiation are continuous : *profSim*, *pathSim*, *tiSim*, *cohesion* and *sReputation*. A *logit* model will be used to predict the negotiation outcome. Negotiation results fed back to the *Global CarPooling Matching Service (GCPMS)* are used to determine the coefficients for the *logit* model.

3.7.2 Dynamic Actor and Agreement Attributes

While evaluating the success probability for a *pool*, exactly one *sReputation* value applies since only the *sReputation* for the driver is relevant.

The *tReputation* of an individual i_0 applies to an existing *agreement* and only

exists as long as the agreement holds. It can only be affected by the partners in the agreement different from i_0 (in the pairwise case, there is only one such partner). The *tReputation* is an evaluation score assigned by the partners; the default value equals the neutral value : see definition 3.5.10.

3.8 Matching using dynamically updated Edge Weights

We now show how to apply graph matching techniques to find optimal carpooling (see also Agatz et al. (2011)). We are also interested in quickly updating the result, when the weights of the edges (which correspond to the result of negotiation between the users) slightly change (as opposed to running the entire matching from scratch whenever such changes occur). We proceed to introduce the graph structure. In this paper, we deal with the case in which the *periodicTripEx*'s belong to either *drivers* or *non-driving passengers*, and we try to optimally match each driver trip with a passenger trip; in the future we plan to tackle more general problems. Note that this case requires the set of candidate carpoolers to be partitioned a priori.

We quickly introduce some notations: a graph G consists of a set of nodes, V, and a set of edges, E, such that each edge is associated with a pair of nodes. Denote $G = (V, E)$. A directed graph is the same as above, but where each edge is associated with an ordered pair of nodes. In a weighted graph, each edge has an associated non-negative real number with it, defined as its weight.

Given a graph $G = (V, E)$, a matching M in G is a set of pairwise non-adjacent (disjoint) edges. That is, no two edges share a common node. The weight of a matching M is the sum of the weights of the edges in M. A maximum (optimal) matching is a matching such that it obtains the maximum weight of all matchings. It does not have to be unique.

A special case of the matching problem is when the graph is *bipartite*, that is, its nodes can be partitioned into two disjoint sets, L and R, such that all edges are between a node in L and a node in R. For the carpooling problem, this may correspond to the case in which the *periodicTripEx* set is composed of trips owned by drivers (L) and trips owned by non-drivers (R). For a schematic example, see Fig. 3.7.

In order to solve the carpooling problem when represented by a bipartite graph, denote the weight of the edge between $i \in L$ and $j \in R$ by c_{ij}, and define variables x_{ij}. To find an optimal matching, solve the following optimization problem: maximize $\sum_{ij} c_{ij}x_{ij}$, subject to the constraints $x_{ij} \geq 0$, $\forall j \sum_i x_{ij} \leq 1$, $\forall i \sum_j x_{ij} \leq 1$. According

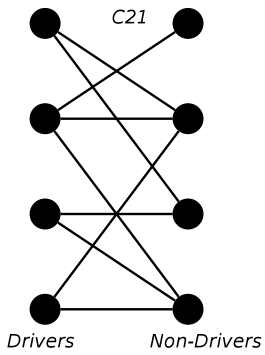


Figure 3.7: A schematic example of a graph used for the carpooling problem. c_{21} stands for the weight of the match between trip owned by driver number 2 and trip owned by passenger number 1.

to Section 3.7, the c_{ij} values are the probabilities for the negotiation to succeed when i and j are advised to carpool.

To test the algorithm, data was simulated as follows. Weights were chosen as the absolute values drawn at random from a normal distribution with mean $\mu = 0$ and given standard deviation σ, and then those smaller than a certain threshold (0.2 for the experiments reported here) were thresholded to 0. Fig. 3.8 shows run times for different numbers of nodes and values of σ.

As discussed in the introduction, the input to the carpooling problem is highly dynamic. This necessitates developing, in addition to the well-studied batch solution described above, algorithms which are *incremental*. An incremental algorithm assumes that the optimal solution of the carpooling for some input was computed already, and then it attempts to solve (either accurately or approximately) the problem for the same input but under a small perturbation.

Here we present an analysis which, given a solution to the optimization problem, allows to determine how far this solution is from the optimal one to the perturbed problem. Assume that the solution for the original (unperturbed) problem, with weights c_{ij}, is x^0_{ij}, and that x^1_{ij} is the solution for the perturbed weights $c_{ij} + \epsilon_{ij}$. Then the following holds:

$$\sum_{i,j}(c_{ij} + \epsilon_{ij})x^1_{ij} = \sum_{i,j} c_{ij}x^1_{ij} + \sum_{i,j} \epsilon_{ij}x^1_{ij} \leq \tag{3.27}$$

$$\sum_{i,j} c_{ij}x^0_{ij} + \sum_{i,j} \epsilon_{ij}x^1_{ij} \tag{3.28}$$

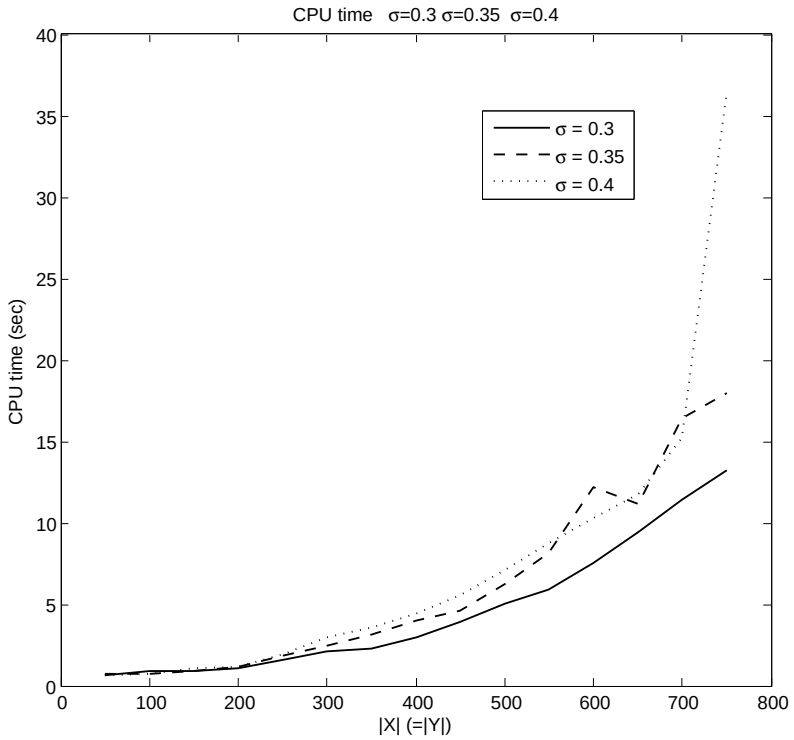


Figure 3.8: Running times [sec] for various values of σ and numbers of nodes.

where the inequality follows from the optimality of x^0_{ij} for the unperturbed weights c_{ij}. Hence, the solution for the perturbed weights is better than the old one by an error term of at most $\sum_{i,j} \epsilon_{ij} x^1_{ij}$. Since $\forall j \sum_i x^1_{ij} \leq 1$ and $\forall i \sum_j x^1_{ij} \leq 1$, this term can be bounded by $\sum_i \max_j |\epsilon_{ij}|$.

3.9 Future Research Directions

3.9.1 Ongoing research

Network characteristics are studied for the graph connecting the trips that have been posted for carpooling in order to find out how partitioning (number of independent components) depends on the probability threshold used.

The carpooling problem has been formulated in a general way. A weighted directed graph is used (in general not a bipartite graph). More than two *periodicTripEx* can be part of an agreement. Research now focuses on the complexity of the problem. First results have been presented in Knapen et al. (2013b). Although a straightforward formulation as an integer programming problem has been given, graph theoretical methods are focused in order to find out whether results from graph theory can provide solutions to mitigate the problem size explosion.

A Janus (Gaud et al., 2009) based experiment using 10000 agents has produced preliminary results. It includes negotiation and trip execution simulation in order to generate reputation feedback.

3.9.2 Research for the longer term

Negotiation for agreements involving more than two people (some of who can already be involved in an existing agreement) requires cooperative adjustment of more than two schedules and needs additional attention.

Determination of agent profiles needs improvement; in particular semantic-based models will be evaluated to replace the basic method now used in the model.

The cohesion function used in the MAS needs to account explicitly for *private social network membership* of participants in an agreement.

Finally, in order to achieve the ultimate goal of replacing a real testers community by a community of agents in a MAS, several coefficients used in the behavioral models, need to be quantified by means of surveys and appropriate statistical methods.

3.10 Conclusions

In order to evaluate a carpooling advisory service, a MAS is used as an exerciser. Large scale simulation is required in order to analyze the advisor behavior. Agents are required to accurately mimic real individuals. Thereto the process has been analyzed in depth and model components have been proposed. Preliminary experiments show that computational problems are to be expected due to combinatorial explosion. Several research paths to explore possible solutions to handle that problem have been identified.

Chapter 4

Agent-based Models in Carpooling: Estimating Scalability Issues

This chapter consists of paper

Knapen et al. (2013b) *Estimating Scalability Issues while Finding an Optimal Assignment for Carpooling*

4.1 Research Context

The previous chapter covered the problem of the edge weight calculation for the graph used to optimally advise carpooling candidates. The case of a perturbation of the weights after having found an optimal solution, was considered. If the solution for the unperturbed case continues to be used, a sub-optimal result is found. An upper bound for the difference between the sub-optimal solution and the optimal one was given.

However, the previous chapter considered the problem to find a maximal matching in a bipartite graph. This problem can be solved in polynomial time but it is not useful in practice since no information is available to partition the candidates in advance into sets of drivers and non-drivers.

In this chapter the general problem is formulated again as a MILP and a graph theoretical equivalent problem is formulated. This turns out to be a *star cover* prob-

lem. The problem is proved to be NP-hard and hence heuristics will be required to solve the practical cases.

The aim is to estimate the graph properties for several negotiation success probability thresholds in order to support the process of designing the matcher. This research is required since it cannot be determined in advance which kind of graph (random, small world, ...) is to be expected. Neither is it possible to predict how the structure of the graph evolves as a function of the probability threshold.

FEATHERS predicted schedules for the Flemish population are used in order to start the investigation from realistic data. Home-work commuting is considered. For the experiments, it is assumed that the share of carpooling does not affect the required travel times between Traffic Analysis Zones (TAZs).

In order to determine edge weights, feedback from the negotiation process is required. This is not available. Therefore, two alternative functions are proposed to calculate an estimated weight from the *time interval similarity* and the *path similarity* that is calculated for pairs of FEATHERS schedules.

For both cases, properties of the resulting graphs are determined for several probability thresholds. Results are mutually compared.

It is to be noted that the weights are estimated for pairs of PTE. Those weights are used also while advising people to negotiate about carpools of size larger than two.

1. Knapen et al. (2013d) already mention importance of the *probability threshold*
2. Knapen et al. (2013d) cover the problem of edge weight calculation
3. Knapen et al. (2013b) cover the problem to estimate the size of the resulting graph used in the matching advisor because the matching problem in general is NP complete (Ben-Arroyo Hartman et al. (2014))

4.2 Abstract

Carpooling for commuting can save cost and helps in reducing pollution. An automatic Web based *Global CarPooling Matching Service (GCPMS)* for matching commuting trips has been designed. The service supports carpooling candidates by supplying advice during their exploration for potential partners. Such services collect data about the candidates, and base their advice for each pair of trips to be combined, on an estimate of the probability for successful negotiation between the candidates to carpool. The probability values are calculated by a learning mechanism using, on one hand, the registered person and trip characteristics, and on the other hand, the negotiation feedback. The problem of maximizing the expected value of carpooling negotiation success was formulated and was proved to be NP-hard. In addition, the network characteristics for a realistic case have been analyzed. The carpooling network was established using results predicted by the operational FEATHERS activity-based model for Flanders (Belgium).

4.3 Introduction - Problem Context

We describe an advisory service for carpooling while commuting. People register their *periodic commuting trips*: the base period typically is one week i.e. a specific pattern valid for working days is repeated after every seventh day. Considering one week periods accommodates for most situations (including part-time workers). People who are able to fulfill all their carpooling needs within their own social network (*local exploration*) of acquaintances, are assumed not to need the advisor service. Others will need to explore the set of yet unknown carpooling candidates (*global exploration*). The matching service determines which trips are best suited to be combined for carpooling and provides advice by suggesting candidates to start a negotiation with respect to a specific periodically executed trip.

After negotiation candidates feed the resulting decision about the proposed carpool back to the automated advisor so that it can learn values for parameters used to prepare the advice. Candidates register periodic commuting trips with the advisor service. Some of the trips can be combined for carpooling. We show how to estimate the probability that a given pair of trips will lead to successful negotiation. The advisor service keeps track of promising pairs (i.e. the ones having sufficiently high probability). The information available to the advisor software can be represented as a digraph in which each vertex represents a periodic trip and where the edges are labeled using the negotiation success probability estimated from the feedback supplied

by the customers. The strategy used by the advisor is to maximize the expected value for the number of successful negotiations (carpool formations).

Advisory systems that make use of feedback from their customers need to solve the *cold start* problem. In the carpooling case successful cold start involving only real commuters, is expected to be infeasible. Therefore, an Agent-Based Model (AgnBM) of commuters in Flanders is being built. This model will take data generated by the FEATHERS Activity-Based Model (ActBM) as input. The FEATHERS model predicts the daily schedule for each member in a synthetic population that mimics the Flemish one with respect to several statistical distributions. The main type of agent in the AgnBM is the commuter who is able to negotiate about plans to carpool. The agent model is similar to the model described in Kamar and Horvitz (2009). This AgnBM will interact with the advisor for initial training of the logit based predictor as well as for studying the transient phenomena during the startup process. It is to be emphasized that the information supplied by the agent simulating the commuter, to the advisory system, is only a subset of the information used in the negotiation process. This is due to privacy reasons and to the limited effort people are prepared to spend to maintain their profile on a website; e.g. candidates supply time windows for trip departure but not a preference value for each moment in such time window.

The remainder of the paper is structured as follows: Section 4.4 gives a literature overview of similar related research. Section 4.5 explains the principle of operation of the advisor. In section 4.6 we give a precise mathematical formulation of the optimization problem. We also prove that the problem is NP-hard. Section 4.7 describes how a synthetic carpooling network has been built using the daily agendas predicted by the FEATHERS activity-based model Bellemans et al. (2010). Finally, in section 4.8 we demonstrate some of the network characteristics.

4.4 Related Work

In order to be effective, the carpooling advisor requires a critical mass of users. Systems of this kind need thorough testing and evaluation before being exposed to the general public. In Kamar and Horvitz (2009) the authors discuss the analysis for constructing collaborative plans for ride-sharing, including the acquisition of changing costs and preferences for ridesharing, the solution of the associated optimization problem and the use of VCG (Vickrey-Groves-Clarke) based payment systems in a dynamic setting. An agent-based model prototype consisting of self-interested individuals is considered. Rideshare plans are generated from GPS traces. The prototype

creates personalized rideshare plans while minimizing the cumulative cost of transportation. The user-modeling component captures preferences about trips and passes those to the optimization and payment components. The model takes as input time of day, day of week and attributes about the agent's commitments from an online appointment book. From those data, the system constructs a time cost function for a trip based on the nearest deadlines drawn from the agent's calendar. The net value for each participant in a pool is calculated from the sum of the costs for trip duration increase, departure and arrival time shifts and from the fuel and cognitive cost savings. The shared transportation plan generating the highest cumulative value is searched for. An optimal set cover is to be found and a greedy algorithm is used.

Agatz et al. (2010) focus on dynamic ride-sharing which is characterized by non-recurring trips that can be established on short-notice but are pre-arranged and make use of independent drivers. This problem is related to long-term planned commuting carpooling. Participants (both drivers and passengers) aim to minimize the cost for travel. Part of the cost saving is paid to the operator of the matching service. Maximizing the benefit of all parties involved is greatly in line with the societal objectives to minimize congestion and pollution. The authors show by example that the system-wide and user optima not necessarily coincide. Discrete time is used and a space-time network is considered. An edge in the network joining two vertices having identical spatial value, represents a waiting period at a given location; the remaining edges represent movements between locations. The use of each specific edge in the space-time network by a specific participant (driver or rider) is modeled by a Boolean variable. The proposed formulation thus calculates the optimal route (pick-up/drop-off sequence). Constraints ensuring that all required movements are accomplished, are formulated and the objective function to minimize the total system-wide travel time is formulated. Single-driver-single-rider, single-driver-multiple-rider and multiple-driver-single-rider arrangements are briefly discussed. It is shown that scheduled public transportation can be described using the space-time network too and the authors show how integration leads to a model for multi-modal ridesharing. The paper concludes by listing challenges (i) with respect to acceptance of the service by customers, (ii) with respect to the application design (determination of the optimization frequency under dynamic arrival of requests) and (iii) with respect to the computational effort required to solve the integer linear programming problem. It suggests that integer programming optimizers can turn out to be insufficiently performant to solve practical problems. Agatz et al. (2012) extend this work by providing a literature overview for the problem of dynamic arrival of riders and drivers. One of the concluding remarks is that "*it may be the case realistic-size instances of the*

model cannot be solved fast enough to be of use in a matching engine of an actual, sustainable ride-share system".

In Trasarti et al. (2011) the authors derive *travel routine* for individual travelers from sets of GPS traces. A routine consists of both a spatial (sequence of locations) and a timing (time windows to visit the locations) component. Two routes are combined by assuming that the common partial route is driven by carpooling. The part of the passenger trip not covered by the carpool trip, is assumed to be covered by walking. An upper bound for possible carpooling in the study area under the assumption of a maximum walking distance, is found by matching the routines.

Buliung et al. (2009) investigate the driving factors behind carpooling in the region of Toronto-Hamilton (Canada): age, gender and income category are the only relevant factors. Recent carpool advisors (like http://www.zimride.com/) take additional factors into account for matching candidates (interests, music tastes) and allow for feedback to be posted.

A comprehensive survey is given in Furuhata et al. (2013) that provides basic definitions and a classification of current matching agencies in terms of the matching search method and the target demand segment. Definitions are given for (i) spatial characteristics by means of which routes for driver and rider are compared, (ii) temporal elements and (iii) the strategic consolidations performed by the matching agency to combine incoming offers and requests. The survey covers 39 matching agencies and classifies them into 6 categories based on (i) the primary search criteria and (ii) the target markets. The problem dealt with in this paper falls into category 2. The pricing (payment) problem is discussed extensively because according to the authors it got less attention than the matching problem. The availability of information about complementary travel modes is discussed and relates to the multi-modal ride-sharing concept mentioned in Agatz et al. (2010).

4.5 Advisor Model

4.5.1 Principle of Operation

In order to find carpooling companions, people who did not find a suitable partner by exploring their private network, register themselves with the *GCPMS*. Registration implies first posting some descriptive characteristics like age, gender, education level, special interests (like music style preferences), job category, driver license availability, etc. Those qualifiers are used because it is known that continued successful cooperation between people requires a minimal level of similarity.

Secondly, people post information about each trip they periodically plan to execute: those data consist of origin and destination locations, earliest and latest departure and arrival times, the maximal detour distance that is acceptable, and the availability of a car (possibility to drive). Note that a particular driver license owner can be unavailable for driving on a specific day of the week because the family car on that day is in use by her/his partner.

Periodic Trip Execution (PTE) need to be matched, not people. For example, a periodic trip on Wednesday from A to B leaving at about 08:30h needs to be matched with another one having similar characteristics. Of course, the people involved shall be mutually compatible but they are not the primary subject of matching. A particular individual can periodically carpool with several people for different trips in the week (on Monday with colleague A, on Tuesday with neighbor B who differs from A). In the remainder of the text, Periodic trip execution will be abbreviated by PTE.

A *pooled trip execution* is the cooperative execution of a set of trips using a single car and a single driver. As a consequence, the route for each passenger shall be embedded in the route of the driver (*single driver constraint*).

After having found a good match (details on how to do so will be explained in Section 4.5.4) the matcher conveys its advice to the candidates involved (the owners of the matched PTE); they evaluate the proposal, negotiate about carpooling, and possibly agree to cooperate. Note that this negotiation is not guaranteed to succeed. One of the reasons is that the individuals share more information between them during the *negotiation process* than they share with the automated service to support the *matching process*. Therefore, the candidates convey the negotiation result back to the matcher service. This paper assumes that sufficient (financial) incentives are in place in order to make this feedback happen.

After trip execution, users can evaluate each other. The *GCPMS* (*Global Car-Pooling Matching Service*) allows for mutual evaluation of individuals, with respect to timeliness and safety. Only individuals cooperating in an agreement can qualify each other. The negotiation and qualification feedback is used by the learning mechanism incorporated in the matching service. After receiving the feedback, the matching service disposes of (i) data describing the PTE and their owners (individuals) as well as of (ii) the negotiation result and (iii) the mutual evaluations. All those data are used to train a logistic regression based predictor. Please refer to fig. 4.1 for an high level overview of data flows, relations and method activation. The relation that qualifies the PTE as *suitable for negotiation* constitutes a graph (shown in the rightmost part of the diagram). The leftmost part represents the community of carpooling candidates: during cold start and quality assessment of the advisor service (Phase 1) this

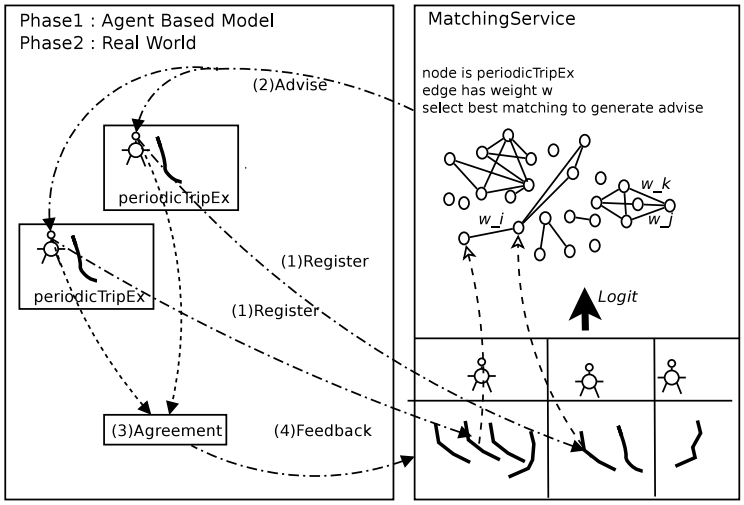


Figure 4.1: Application context: the right hand side shows the *matcher service*. People register some descriptive data (lower part) about themselves and their trips to be executed periodically (PTE). Those constitute a graph (upper part): the edges are labeled with the probability that negotiation will succeed when the trip owners are advised to carpool. Negotiation result is fed back to train the *logit* predictor. The left hand side shows the entities exercising the *matcher service* in consecutive phases.

community consists of an agent-based model. After deployment of the advisor (Phase 2), the community consists of real world commuters.

4.5.2 Graph-theoretical model

The model used for matching consists of a directed graph $G = (V, E)$ (see Figure 4.2). Each vertex in V corresponds to a PTE. A directed edge (u, v) corresponds to the feasibility to combine the PTE; the presence of an edge indicates that it is worth to advise the owners to start a negotiation. The weight of an edge, denoted by w, is the estimated probability for the negotiation to succeed for the particular pair of PTE. A loop (u, u) indicates that the owner of PTE u owns a car and can take his/her own car. Note that a vertex for which the owner is unable to become the driver, can never be a target vertex (its *indegree* equals zero). Vertices drawn using a shaded circle represent a PTE whose owner is prepared to drive. In addition, every vertex is labeled by *capacity*, which indicates the capacity of the car (including the driver) owned by the PTE owner.

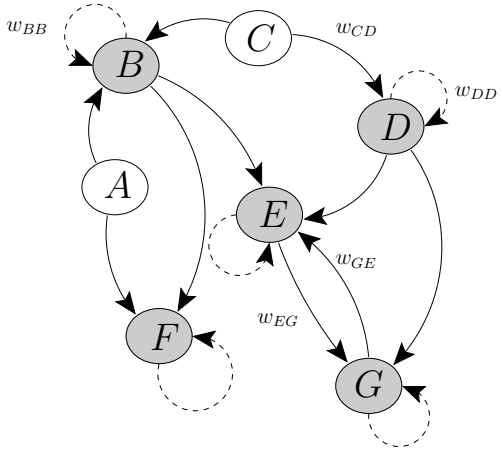


Figure 4.2: The diagram shows the graph where vertices correspond to PTE (*periodic trip execution*) and edges are labeled with the success probability for the negotiation (if that is sufficiently large). Grey vertices correspond to PTE where the owner is prepared to drive; this is also indicated by the *loops*. Each loop has zero weight. Some, but not all of the edges, have been labeled with their weights.

Note that

1. The set of vertices evolves over time since (i) people register and withdraw PTE as time evolves and since (ii) people join and leave the carpooling candidates society(removing all their PTE's in the latter case).
2. Edges emerge as soon as the estimated negotiation success probability $w(u, v)$, exceeds a given threshold; this can be caused by changes in the PTE (e.g. by relaxing the time constraints) or changes in characteristics of people (e.g. by reputation changes (see below)).
3. Probability estimates can change over time by re-training the predictor. Note that this can cause threshold crossing and hence edge creation or deletion.
4. Multiple vertices can be associated to a single car because the corresponding PTE are executed at different times. Vertices associated to a unique car cannot be linked, of course, by an edge.

Finally, the problem size can grow large when a nation-wide service and a one week period are considered. Large scale deployment probably is a necessary condition for both effective operation (delivery of advice that has a high success probability) and economic viability. The matcher needs to cope with large networks whose topology and edge weights evolve in time.

This represents a complex problem and hence a thorough evaluation before deployment. An agent-based model simulating the actual population behavior, will be used to exercise the matching service for several reasons. First, performance and effectiveness need to be evaluated on a running system since they are very difficult to predict from design data only. Second, deploying such system should go flawlessly because lost customers will be reluctant to return. Finally, the system transient behavior during the start up when only few customers are registered, is difficult to predict and hence observations made are difficult to interpret; simulation can support learning about the overall system behavior.

4.5.3 Similarity and Reputation

In this section we explore the functions used to determine the input variables for the *logit* based negotiation success probability estimator. Note that the functions specified for *path* and *time interval similarity* apply to carpools of arbitrary size, although in this paper they are applied to pairs of PTE's.

4.5.3.1 Path Similarity

Given a pair of PTE and the selected driver, path similarity indicates to what measure the travel duration for the solo-driving path for the driver differs from the one for the carpooling-path. Hence it is a measure for the time loss induced by the detour imposed to the driver. Path similarity `pathSim(pte0,pte1)` is a value in the range $[0, 1]$ assigned to an *ordered* pair (pte_0, pte_1) of PTE in which the owner of pte_1 is the driver. Path similarity is specified by equation (4.1)

$$pathSim(B, A) = \frac{d(O_A, D_A)}{d(O_A, O_B) + d(O_B, D_B) + d(D_B, D_A)} \tag{4.1}$$

where O_x and D_x denote respectively the origin and destination for individual x and $d(O, D)$ denotes the duration to travel from O to D.

The function is not symmetric in its arguments. This is easily seen because the trips driven depend on the driver selection; the driver needs a detour to pick up passengers. Equation (4.1) applies to the driver: the numerator corresponds to the solo-trip, the denominator corresponds to the carpooled trip. The solo-driving paths for driver (O_A, D_A) and passenger (O_B, D_B) in the road network do not need to be identical to deliver maximal similarity. The case where the solo-path for the passenger is embedded in the solo-path for the driver, leads to the maximal similarity value of one (when the time required to stop and pick up a passenger is ignored).

Note that the travel duration for a given origin-destination pair depends on the departure time due to time dependent congestion. The additional model complication is limited since time dependent travel times need to be computed anyway in order to evaluate time interval similarity (see Section 4.5.3.2 item 2).

4.5.3.2 Time Interval Similarity

Time interval similarity `tis()` is a value in $[0, 1]$ assigned to an ordered pair $\langle pte_0, pte_1 \rangle$ of PTE. It is a measure for the appropriateness to *start/stop* a cooperative trip at a given location. More specifically, it measures the probability that the time interval, during which all carpool members can be simultaneously present at a specific location, is sufficiently long to comfortably start or stop a trip. A low *tis* value means that the available time for people to synchronize, does incur high time pressure for at least one of the carpool members. Time interval similarity applies to the period in time required to *start/stop* an activity (but not to *perform* a cooperative action). Passengers shall get on/off board of the vehicle at an agreed location and moment in time; there shall be a tolerance period that applies to the agreed moment in time. The length of the period available to synchronize, defines the time interval similarity. Time interval similarity is calculated for locations on the shared part of the route driven by the carpool. Let $W(i, L)$ denote the time window available to individual i at location L. $W(i, L_1)$ is calculated from $W(i, L_0)$ using the expected duration to travel from L_0 to L_1.

1. Time windows are used to calculate *time interval similarity*. Let $t_b(W)$ and $t_e(W)$ denote respectively the begin and end of a time window W. Each individual supplies the time window for the trip start to the *GCPMS*. The expected travel duration is used to calculate the time window for each visited location.
2. For every pair of PTE's, we consider every candidate driver. Stops in the considered routes are origin and destination locations for participants in the carpool. Each individual visits a subsequence of the sequence of stops on the route. For a given individual i, the time window for trip departure W_i^d applies to the first stop that belongs to the individual's subsequence. The expected duration for each trip (part of the route between consecutive stops) needs to be determined. In general, accurate prediction of travel duration for a specific trip is not possible. The notion of traffic analysis zone (TAZ) is used. Travel times between TAZ centers are registered or calculated for several times of the day and taking normal congestion into account. The results are kept in (time dependent, non-symmetric) *impedance matrices*: each element in such matrix

specifies the travel duration between two TAZ. The global advisor is assumed to have the required impedance matrices available. Impedance matrix entries are used to calculate, for each individual, a feasible time window at each stop.

3. In order to calculate *time interval similarity* between two periodic trips T_A and T_B, consider a stop S_0 and the associated time windows at that stop $W_A^{S_0} = \langle t_A^{b,S_0}, t_B^{e,S_0} \rangle$ and $W_B^{S_0} = \langle t_B^{b,S_0}, t_B^{e,S_0} \rangle$ respectively. Time similarity is calculated as follows:

$$t^b = \max(t_A^{b,S_0}, t_B^{b,S_0}) \tag{4.2}$$

$$t^e = \min(t_A^{e,S_0}, t_B^{e,S_0}) \tag{4.3}$$

$$tis(T_A, T_B) = 1 - \exp(-\beta_{TS} \cdot \max(0, t^e - t^b)) \tag{4.4}$$

Equations (4.2) and (4.3) specify the earliest and latest departure time at S_0 respectively.

4. It is assumed that a period lasting for Δt_f suffices in fraction f of the cases to ergonomically start cooperation. This assumption is used to determine β_{TS}. A fixed value for $\Delta t_f = 15[min]$ for $f = 0.95$ has been used in the experiment described below.

$$f = 1 - \exp(-\beta_{TS} \cdot \Delta t_f) \Rightarrow \beta_{TS} = -\frac{\ln(1-f)}{\Delta t_f} \tag{4.5}$$

5. Note that the travel duration between stops depends on time. In this context, the travel time to move from stop S_0 to stop S_1 is the one for the earliest moment in time that belongs to the time windows for both trips (if any). If no such moment exists, $tis(T_A, T_B) = 0$. Using this convention, travel time from a stop to the next one is identical for both trips considered and hence time interval similarity calculation delivers the same value in each stop that belongs to both trips: it can be calculated for the first stop that is contained in both trips.

4.5.3.3 Profile Similarity

Profile similarity `profSim()` is a value in $[0, 1]$ assigned to a pair of individuals that indicates to what extent the individuals are compatible for carpooling (homophily concept). The profile of a participant is specified by a vector of socio-economic and preference attributes like age, gender, income category, job type, music preference etc. Most of the attributes take discrete values; some are enumerations, for others a total order is defined over the attribute range. The distance between two values $a_0, a_1 \in A$

of an attribute, is defined by equation (4.6) for the enumeration case, and by (4.7) for the ordinal case. For the ordinal case r_A denotes the range of the value.

$$d(a_0, a_1) = \begin{cases} 1 & if \quad a_0 \neq a_1 \\ 0 & if \quad a_0 = a_1 \end{cases} \tag{4.6}$$

$$d(a_0, a_1) = a_1 - a_0 \tag{4.7}$$

In a former design Knapen et al. (2013d), profile similarity `profSim()` has been defined by the Euclidean distance between two attribute vectors by

$$profsim(\mathbf{A}_0, \mathbf{A}_1) = 1 - \sqrt{\frac{1}{N_A} \sum_{i \in [1, N_A]} (\frac{d(\mathbf{A}_0[i], \mathbf{A}_1[i])}{r_{A_i}})^2} \tag{4.8}$$

This method is problematic since a value change for an attribute having a small value range such as *gender*, has a large effect on the similarity level. This effect has been observed in early experiments where profile similarity was included based on unweighted *ageCategory*, *incomeCategory* and *gender* attributes. Suppose that only three attributes are used, one of which is *gender* (having a range of 1). The similarity between two attribute vectors that differ in the *gender* attribute only, is given by the evaluation of equation (4.8): the value $1 - \sqrt{\frac{1}{3}} = 0.42$ is less than the threshold (0.75) used to determine whether or not to insert an edge in the graph (see Section 4.5.4). Hence, it causes the PTE graph to breakdown into two giant components, one for each gender. An appropriate weight needs to be assigned to each attribute but there is no way to determine it. Therefore, the distance value for each attribute is used as an independent variable for the negotiation success predictor. The weight coefficient then is determined by training the predictor.

4.5.3.4 Reputation

1. Safety reputation `sRep()` of a driver is a value in $[0, 1]$. Each individual has a safety reputation value that evolves over time due to evaluation by passengers (i.e. individuals who participated in an agreement where the person being evaluated was the driver). Evaluations received are registered in a personal *evaluations list* with the individual they apply to; for each issuer, only the most recent evaluation is kept. The safety reputation is calculated as a weighted average of the values posted in the evaluation list: the weight decreases with the age of the evaluation and increases with the duration of the cooperation.
2. Timeliness reputation `tRep()` (or accuracy reputation) is a value in the range $[-0.5, 0.5]$ assigned (by the co-travelers) to a PTE in an *agreement*: it indi-

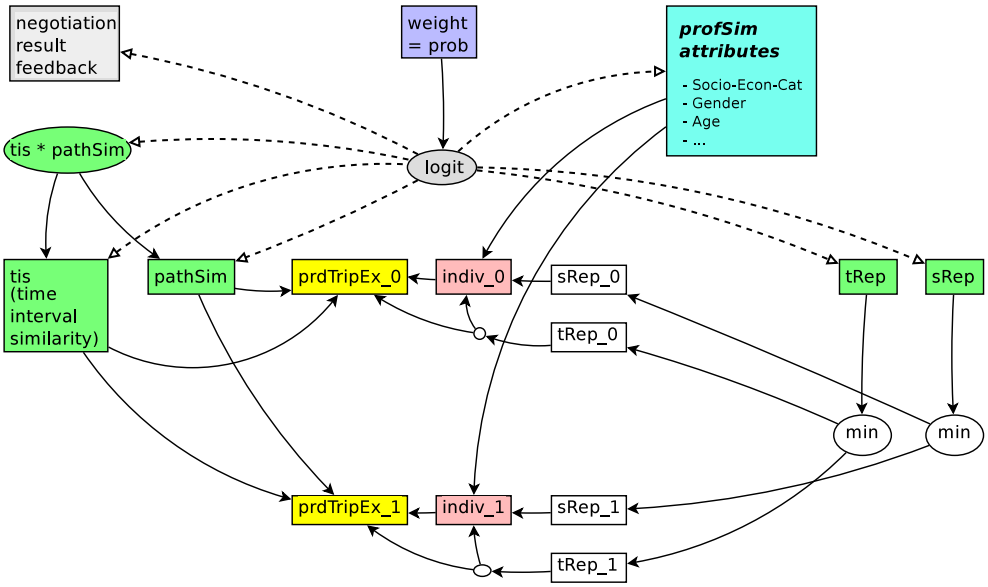


Figure 4.3: Data dependencies related to the negotiation success probability estimation. Each arrow denotes a dependency. Ovals denote functions and rectangles denote data (attributes). A *logit* predictor is fed with similarity and reputation values that act as statistical explanatory variables. Previous negotiation results are fed back and used in the training phase to determine the logit parameters (β vector).

cates to what measure the owner of the PTE respects the time schedule when executing the periodic trip in the *agreement*. Timeliness reputation is defined for both drivers and passengers. Note that it is defined as a characteristic of a tuple $\langle PTE, agreement \rangle$ and not as a characteristic of an individual or of a PTE because an individual may behave differently on a specific PTE in different agreement contexts (pools).

4.5.4 Weights Determination - Learning Mechanism

The edge weights (negotiation success probabilities) are determined by means of logistic regression (logit). Figure 4.1 shows where the logit fits in the loop. Logistic regression is a generalized linear model (GLM) Wood (2012), Tsai and Gill (2013) using the logistic function $\frac{e^x}{1+e^x}$ as the *mean function* (inverse of the *link function*). Logistic regression predicts a vector of expected values for dependent variables $\mathbf{Y}$. It

is notes as

$$E(\mathbf{Y}) = \mu = g^{-1}(\mathbf{X}^T\beta) \tag{4.9}$$

where $\mathbf{X}$ denotes the observed values for the independent variables, β is a vector of unknown parameters to be estimated, g^{-1} is the mean function (in this case the logistic function). The parameters vector β is determined by maximum likelihood estimation.

When new negotiation result values are fed back to the advisor, a new set of parameters β can be determined (learned) by maximum likelihood estimation. New values will be processed in batches in order to limit resource usage. The batch size still needs to be defined. As soon as weights have been recalculated, an additional optimal matching involving all candidates who are not yet carpool members, needs to be calculated. The recalculation scheduling strategy is related to the optimal batch size selection and is out of the scope of this paper.

The dependent variable (a scalar in this case) is the probability for successful negotiation between two candidates. The explanatory variables are

sim_{path}	: Path similarity
sim_{time}	: Time interval similarity
$sim_{time} * sim_{path}$	: Product, used to model interaction
$sim_{prof,gender}$	: Profile similarity: gender
$sim_{prof,age}$	: Profile similarity: age
$sim_{prof,SEC}$	: Profile similarity: socio-economic category
$repu_{safety}$	: Safety reputation
$repu_{timeliness}$	: Timeliness reputation

Figure 4.3 shows the data used for logistic regression. Arrows denote dependencies. Example: `tRep_i` depends on `prdTripEx_i` (the periodic trip) and on `indiv_i` (the individual); `tRep` in turn is the minimum of the timeliness reputation of the individuals involved in the negotiation. The `logit` estimator needs the `tRep` value to produce the `weight`.

A data set that applies to previous negotiations (for which feedback is available) is used in the training phase to determine the logit parameters (β vector). Profile similarity attributes are fed into the logit as separate explanatory variables so that their relative weight is determined by the maximum likelihood estimation of the logit parameters (see also Section 4.7).

The *product* explanatory variable ($sim_{time} * sim_{path}$) is used because the nego-

tiation success probability is expected to be high *only* when both *path* and *time* similarity values are sufficiently large and low when either one of them is small. In $\mathbf{Y}_i = g^{-1}(\mathbf{X}_i^T \cdot \beta)$ the predictor is a linear combination of the input variables. If the product variable is not included, the model does not reflect reality. This can be seen as follows. Assume that path similarity corresponds to index value 1 and time similarity corresponds to index value 2: consider two vectors $\mathbf{X}_i$ and $\mathbf{X}_j$ of independent variables values so that

$$X_i[1] = 0 \wedge X_j[1] > 0 \tag{4.10}$$

$$X_i[2] > 0 \wedge X_j[2] = 0 \tag{4.11}$$

Then consider $\mathbf{X}_i^T$, $\mathbf{X}_j^T$ and their convex combinations $c \cdot \mathbf{X}_i^T + (1-c) \cdot \mathbf{X}_j^T$ where $c \in [0, 1]$. The scalar products of those vectors with the β parameters are the inputs for the logistic function. For the scalar products either (4.12) or (4.13) holds

$$\mathbf{X}_i^T \cdot \beta \leq c \cdot \mathbf{X}_i^T \cdot \beta + (1-c) \cdot \mathbf{X}_j^T \cdot \beta \leq \mathbf{X}_j^T \cdot \beta \tag{4.12}$$

$$\mathbf{X}_i^T \cdot \beta \geq c \cdot \mathbf{X}_i^T \cdot \beta + (1-c) \cdot \mathbf{X}_j^T \cdot \beta \geq \mathbf{X}_j^T \cdot \beta \tag{4.13}$$

Since the logistic function is monotonically increasing either (4.14) or (4.15) holds for the corresponding probabilities, which is not what is wanted.

$$p(\mathbf{X}_i^T \cdot \beta) \leq p(c \cdot \mathbf{X}_i^T \cdot \beta + (1-c) \cdot \mathbf{X}_j^T \cdot \beta) \leq p(\mathbf{X}_j^T \cdot \beta) \tag{4.14}$$

$$p(\mathbf{X}_i^T \cdot \beta) \geq p(c \cdot \mathbf{X}_i^T \cdot \beta + (1-c) \cdot \mathbf{X}_j^T \cdot \beta) \geq p(\mathbf{X}_j^T \cdot \beta) \tag{4.15}$$

4.6 Optimization problem

The *GCPMS* aim is to maximize the effectiveness of the advice with respect to carpooling negotiation. Hence, the advice is based on maximizing the expected value for the negotiation outcomes.

4.6.1 Formulation as linear programming problem

The carpooling problem can be described by an integer linear programming problem. Let $V = \{1, 2, \ldots, N\}$ be the set of vertices, which correspond to PTE's. See Section 4.5.2. We remind that a directed edge $(i, j) \in E$ of weight $w_{i,j}$ indicates the compatibility of the owner of PTE i to get a ride with the owner of PTE j (in the latter's vehicle). Note that $w_{i,i} = 0$. Each vertex j also has maximum capacity c_j of people that he/she can take in the vehicle (including the driver). In the LP we have

a variable $x_{i,j}$ for each pair of vertices i and j, which has value one if i has a ride in j's vehicle, and zero otherwise. Variable $x_{i,i}$ exists only if i has a vehicle that he/she can drive, and $x_{i,i} = 1$ if and only if i will take his/her vehicle in the carpool solution. The LP problem is stated below:

$$maximize \sum_{i,j \in [1,N]} w_{i,j} \cdot x_{i,j} \tag{4.16}$$

subject to

$$\forall i \in [1,N] : \sum_{j \in [1,N]} x_{i,j} = 1 \tag{4.17}$$

$$\forall j \in [1,N] : \sum_{i \in [1,N]} x_{i,j} \leq c_j \tag{4.18}$$

$$\forall i,j \in [1,N] : x_{i,j} \leq x_{j,j} \tag{4.19}$$

$$\forall i,j \in [1,N] : x_{i,j} \in \{0,1\} \tag{4.20}$$

Constraint (4.17) requires that each PTE shall be assigned to exactly one vehicle (i.e. the trip shall be executed). Constraint (4.18) bounds the number of people (including the driver) in each vehicle. Constraint (4.19) makes sure that j is marked as a driver if there exists any passenger i who gets a ride with him. Constraint (4.20) limits the range of the (Boolean) variables, it is the integrality constraint.

4.6.2 Graph theory formulation

Consider a weighted directed graph $G(V, E)$ where each vertex corresponds to a PTE (periodic trip execution). There is a directed edge (u, v) if and only if the probability of negotiation success between the owners of u and v is above a certain threshold, where the owner of u will ride in the vehicle of the owner of v. The weight w of an edge (u, v) is the probability for a successful negotiation between the owners of u and v, respectively. If the owner of a PTE v has a car, then the corresponding vertex has a loop (v, v) of weight zero. Each vertex $v \in V$ has a capacity $c(v)$ which denotes the maximum number of people the vehicle of v can contain, including the driver. If the owner of PTE v does not have a car, then $c(v) = 0$.

A *star* is a graph $K_{1,t}$ consisting of a center vertex called *root* and vertices connected to it called leaves. A *directed star* is a star whose edges are all directed towards the root of the star. A *directed star partition* is a collection of vertex disjoint directed stars that cover $V(G)$. A directed star partition is *feasible* if every root r in the star partition has a loop, and its star in-degree (not including the loop) is at most $c(r) - 1$.

The *weight* of a directed star partition is the total weight of the edges of the stars in the star partition. The LP in Section 4.6.1 is equivalent to the following problem:

Problem 4.6.1. *Let $G = (V, E)$ be a directed graph with edge weights $w : E \to \mathbb{R}$, and let $c : V \to \mathbb{N}$. Find a feasible star partition of maximum weight.*

Solution to problem 4.6.1 guarantees maximum compatibility between people and their priorities for "taking rides." It does not guarantee, however, that the number of vehicles used is as small as possible. However, when $w_{ij} = 0$ or 1 these problems are equivalent, as is seen in the claim below:

Claim 4.6.2. *If the edge weight function w_{ij} is either 0 or 1, then Problem 4.6.1 is equivalent to the problem of finding a star partition with minimum number of stars (i.e. a minimum number of vehicles).*

Proof 4.6.3. *If the weight function w_{ij} is either 0 or 1 then an optimal solution to Problem 4.6.1 (or the LP above) finds the maximum number of edges in a feasible star partition of G. For any covering of $V(G)$ with d disjoint directed stars, the total number of edges in the stars is $|V| - d$, since each star contains one less edge than the number of vertices covered by it. This implies that the minimum number of stars covering the graph equals $|V|$ - Max $\sum_{ij} w_{ij}x_{ij}$. Hence, an optimal solution to the LP corresponds to a minimum feasible star partition, and vice versa.*

From the claim above, it follows that when w_{ij} is either 0 or 1 then Problem 4.6.1 is equivalent to the following problem:

Problem 4.6.4. *Let $G = (V, E)$ be a directed graph, and let $c : V \to \mathbb{N}$. Find a feasible star partition $\Gamma = \{S_r : r \in R\}$ containing a minimum number of stars.*

We will show that Problem 4.6.4 is NP-hard. This implies that Problem 4.6.1 is also NP-hard since Problem 4.6.4 is a special case of it. We will reduce the Minimum Dominating Set Problem (MDSP) to Problem 4.6.4. A *dominating set* for a graph $G = (V, E)$ is a subset S of V such that every vertex not in S is adjacent to at least one member of S. The domination number $\gamma(G)$ is the number of vertices in a smallest dominating set for G. The dominating set problem concerns testing whether $\gamma(G) \leq K$ for a given graph G and input K. It was shown to be NP-complete in Garey and Johnson (1979).

Theorem 4.6.5. *Problem 4.6.4 is NP-hard.*

Proof 4.6.6. *Given an instance of MDSP consisting of a graph G and input K, we transform it to an instance of Problem 4.6.4. We create from G a symmetric*

directed graph G' where for each $(u, v) \in E(G)$ we have (u, v) and (v, u) in $E(G')$. Let $c : V \to \mathbb{N}$ be defined by $c(v) = Maxdeg + 1$ where Maxdeg is the maximum degree in G. It is not difficult to see that every dominating set S for G corresponds to a star cover in G' where the centers of the stars are in S. By removing some edges of the stars it is possible to get a star partition which is feasible by the choice of the function c. Conversely, every feasible star partition in G' corresponds to a dominating set S in G, where S corresponds to the centers of the stars. Hence $\gamma(G) \leq K$ if and only if G' has a feasible star partition with at most K stars.

In the proof above we transformed the Minimum Dominating Set Problem to a special case of Problem 4.6.4 where the capacity function c is, $c(v) = \text{Maxdeg}$ for all $v \in V$, meaning that, in fact, there is no bound on the degrees of the stars. We can show that even if the degree of every star is at most two (i.e. every vehicle can hold at most two passengers, not including the driver), the problem is still NP-hard. This can be done by reduction from the 3-dimensional matching problem (see Garey and Johnson (1979)). We omit the details of the proof.

4.7 Data characteristics - Problem size estimation

Since the problem is NP-hard as was shown above, even in the simplified cases where all weights are equal to one, it is important to find heuristic solutions and to gain insight into the structure of the data. Therefore, simulated carpooling candidate networks have been built; their characteristics have been analyzed. Since no negotiation feedback data are available yet, the β coefficients for the logistic regression cannot be determined by a maximum likelihood procedure. This section describes some experiments conducted. Several models to determine the negotiation success probability have been investigated. All experiments have been carried out using a single set of predicted schedules in order to make them comparable; this is required because the FEATHERS simulator constitutes a stochastic model (see Section 4.7.1). Results for two of the experiments have been reported in detail below.

In Section 4.5.3.3 it has been shown that multiple giant components occur when the weights for specific attributes used in the profile similarity, are too large. Furthermore, there is no evidence to determine the weight coefficients for those attributes without measuring the negotiation feedback. Therefore, it has been decided not to include profile similarity in the simulation. This is equivalent to assuming that the weight for profile similarity equals zero.

4.7.1 Data used

Daily schedules (agendas) were generated by the FEATHERS operational *activity-based model* for the region of Flanders (Belgium). The model input consists of (i) a synthetic population for the study area, (ii) an area subdivision into TAZ, (iii) land-use data for each TAZ and (iv) a set of *impedance matrices* specifying the travel time and distance between TAZ for off-peak, morning-peak and evening-peak periods and for several transportation modes (i.e. car, slow, public transport).

Decision trees trained on survey data, are applied in a predefined fixed order that models the decision making process. The schedule (agenda) is constructed using several stages; these results in a chained decision process where each stage further completes the partially constructed agenda. FEATHERS output consists of a travel schedule for each member of the synthetic population.

The set of all schedules (agendas) allows to calculate expected mode-specific traffic flows in time and space; those flows have been validated using traffic counts made available by public traffic management services. FEATHERS is a Monte Carlo micro-simulator; the outcome for each decision is sampled using the probabilities corresponding to the decision tree leaves. Hence, sets of agendas for the synthetic population generated by different runs in general differ. The Flemish model is characterized by

Synthetic population size	: 6 million people
Number of TAZ	: 2368
TAZ area (average value)	: approximately 5 km^2
Number of diaries in survey	: approximately 8000

A simulation for a Monday was used. Agendas containing at least one work activity have been considered (since the problem of commuting is investigated). People performing a HOME-WORK trip starting between 06:00h and 09:00h are considered. It was assumed that 20% percent of them are interested to start a negotiation to carpool but cannot find a partner in their local network; hence they decide to register with the global advisor. From this set of commuters, people were selected only if their first HOME-WORK trip could be combined with a corresponding WORK-HOME trip. Note that, as a consequence, people who perform another activity immediately after each work activity (e.g. those having a WORK-SHOP trip instead of a WORK-HOME trip) are considered not to be interested in carpooling.

4.7.2 Similarity values used

Following similarity values have been calculated:

1. Path similarity for the HOME-WORK s^p_{HW} trip which is assumed to equal the value for the return trip. Note that the equality holds in case the distances are considered but that differences can occur in case travel time is used; this is due to the time dependency of travel duration (even on the same route both the forward and return trips can require a different amount of time).
2. *Schedule time similarity* is determined by two *time interval similarity* values s^t_{HW} and s^t_{WH} for the HOME-WORK (HW) and the WORK-HOME (WH) commuting trips respectively. Both values do not necessarily have equal effect on the negotiation success probability. This is accounted for only in the agent-based simulator. Some of the information used during negotiation is available exclusively to the participants in the negotiation (e.g. particular details of the schedule).

   Time similarity between two pairs of PTE (for two individuals both the HW and WH trips are considered) is assumed to be given by $s^t = s^t_{HW} \cdot s^t_{WH}$. This means that both have equal weight in the probability estimation. This reflects the fact that the advisor service needs to predict the negotiation outcome using less information than is available to the negotiators.

Finally, the logit model makes use of predictors s^p_{HW} and s^t as well as of the product $s^p_{HW} \cdot s^t$ for the reason mentioned in Section 4.5.4.

4.7.3 Experiment 1 : Estimate probability as *product* of similarity values.

Negotiation success probabilities have been estimated by the product $P(s^p, s^t) = s^t \cdot s^p$ of the similarity values. The resulting function is shown in Figure 4.4. Contour lines have been drawn on the horizontal plane.

4.7.4 Experiment 2 : Estimate probability using a *logistic* function

Approximating the probability with the product of the similarities is suspected to over-estimate the probability for low similarity values because $\frac{\partial P(s^p,s^t)}{\partial s^p}$ and $\frac{\partial P(s^p,s^t)}{\partial s^t}$ are constants. Therefore, a second experiment has been conducted where the probability (s^p, s^t) is given by a logit estimator. Since no feedback data are available yet, it was assumed that $P(s^p, s^t) = s^p \cdot s^t$ in a discrete set of 4 points. This results in 4

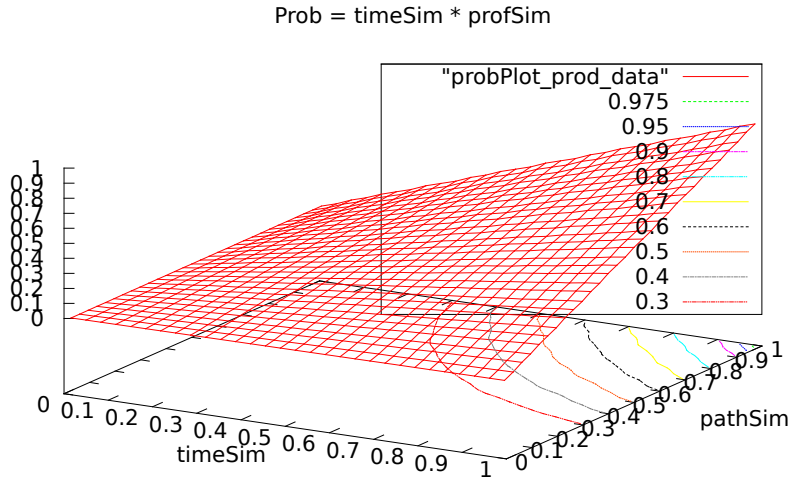


Figure 4.4: Negotiation success probability as a function of path and time similarity: $P(s^p, s^t) = s^p \cdot s^t$.

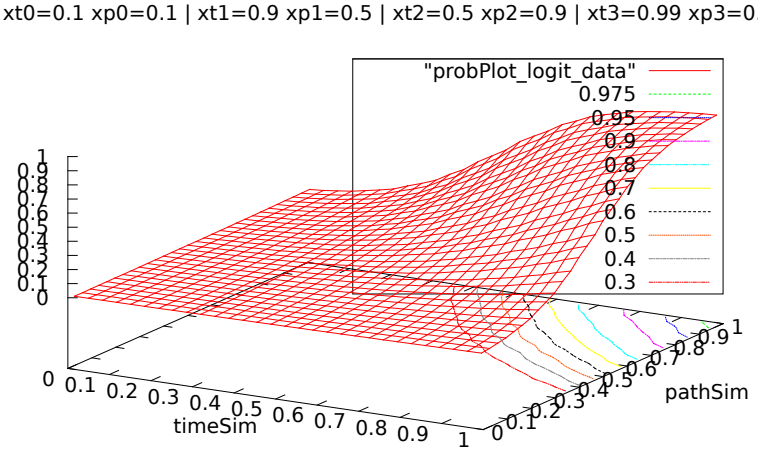


Figure 4.5: Negotiation success probability as a function of path and time similarity: $P(s^p, s^t) = \frac{\exp(z)}{1+\exp(z)}$ with $z = \beta_0 + \beta_t \cdot s^t + \beta_p \cdot s^t + \beta_{t,p} \cdot s^t \cdot s^p$.

equations like

$$P(s^{p_i}, s^{t_i}) = \frac{\exp(z(t_i))}{1 + \exp(z(p_i))} \tag{4.21}$$

with

$$z_i = \beta_0 + \beta_t \cdot s^{t_i} + \beta_p \cdot s^{p_i} + \beta_{t,p} \cdot s^{t_i} \cdot s^{p_i} \tag{4.22}$$

The chosen (t_i, p_i) points were : (0.1, 0.1), (0.5, 0.9), (0.9. 0.5) and (0.99, 0.99). The resulting function is shown in Figure 4.5. Contour lines have been drawn on the horizontal plane.

4.8 Discussion

Experiments have been executed for several *participation levels* (the fraction of commuters who make use of the advisor system while exploring for partners).

Results are presented in pairs. For each trait or metric both the `product` and `logit` cases are compared. In the `product` case the negotiation success probability was estimated by the product of the time and path similarity values. In the `logit` case a logistic predictor was used for which the parameters were determined by requiring the `logit` function value to equal the `product` function value in 4 specific points (see Section 4.7.4).

Tables 4.1 and 4.2 summarize the network traits for the probability thresholds (see 4.5.2 item 2) considered in the analysis. As was expected from the difference between the functions shown in Figures 4.4 and 4.5, the `logit` case retains more large components for higher probability values. However, the absolute numbers in the tables are relevant. Clearly, even for a probability threshold of 90%, connected components can contain 150k vertices. This figure together with the number of edges, is relevant as a problem size estimation. Note also that for both the `product` and `logit` cases, the edge density (actual number of edges divided by the maximum number of edges in the undirected graph having the same amount of vertices) has the same order of magnitude for all networks having a probability threshold up to 90%. The number of components is to be considered in conjunction with the size of the eight largest components shown in tables 4.3 and 4.4. It turns out that one giant component occurs and that all other components are small.

Figure 4.6 shows the number of *non-trivial* connected components (i.e. components having more than one vertex) as a function of the probability threshold. Note

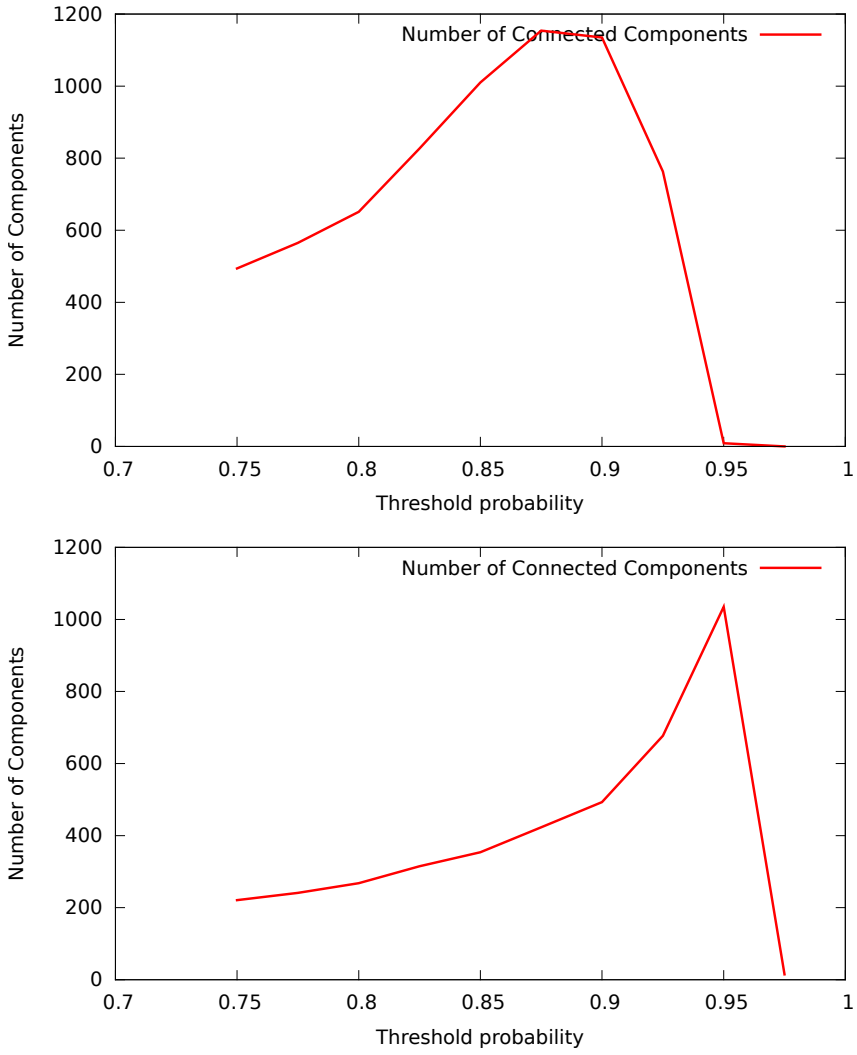


Figure 4.6: Number of *non-trivial* connected components as function of the probability threshold. Singletons are trivial components and hence are excluded. For high probability values the number of dropped singletons grows. Top: `product` case. Bottom: `logit` case.

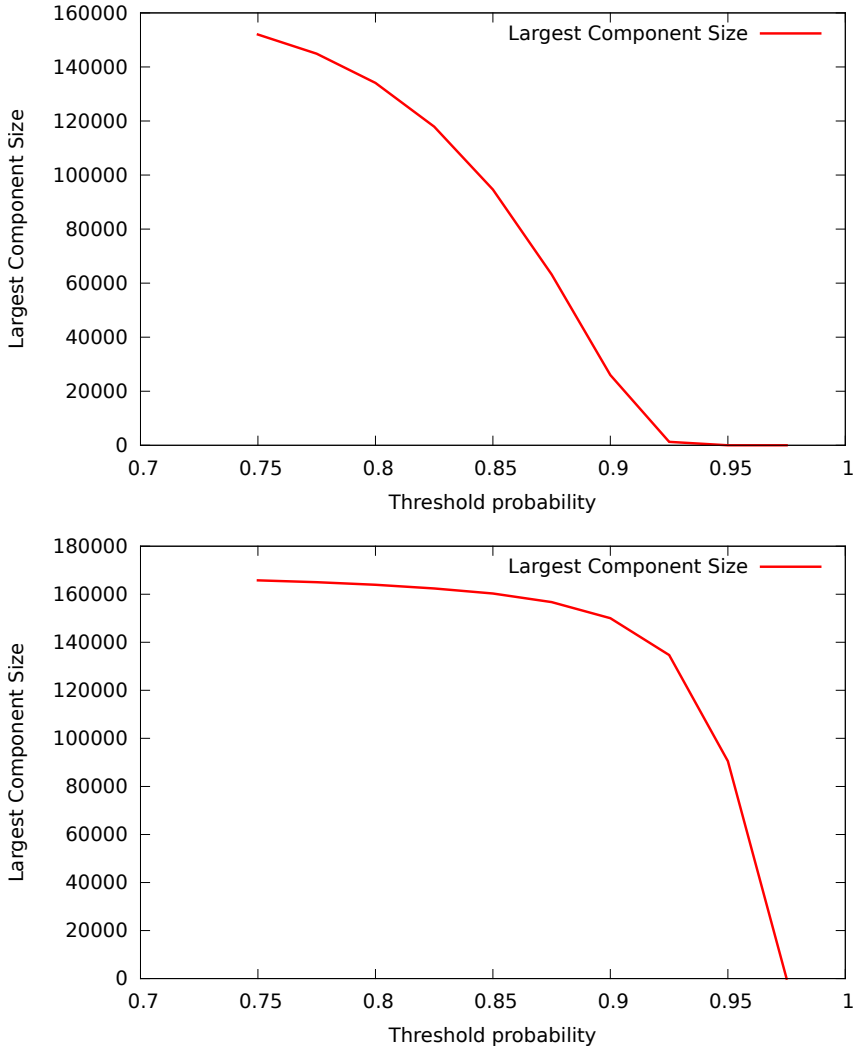


Figure 4.7: Number of vertices in the largest connected component as a function of the probability threshold. Top: `product` case. Bottom: `logit` case.

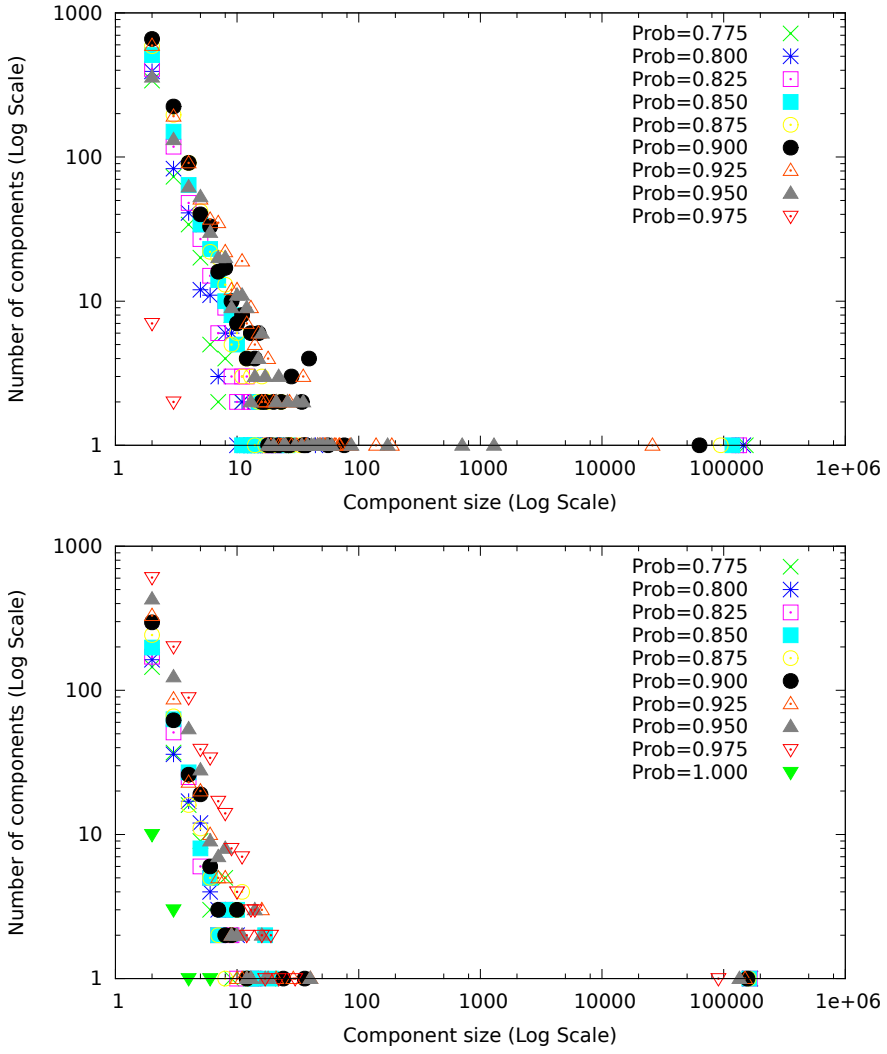


Figure 4.8: Frequency distribution for the number of vertices in the connected components. Top: `product` case. Bottom: `logit` case.

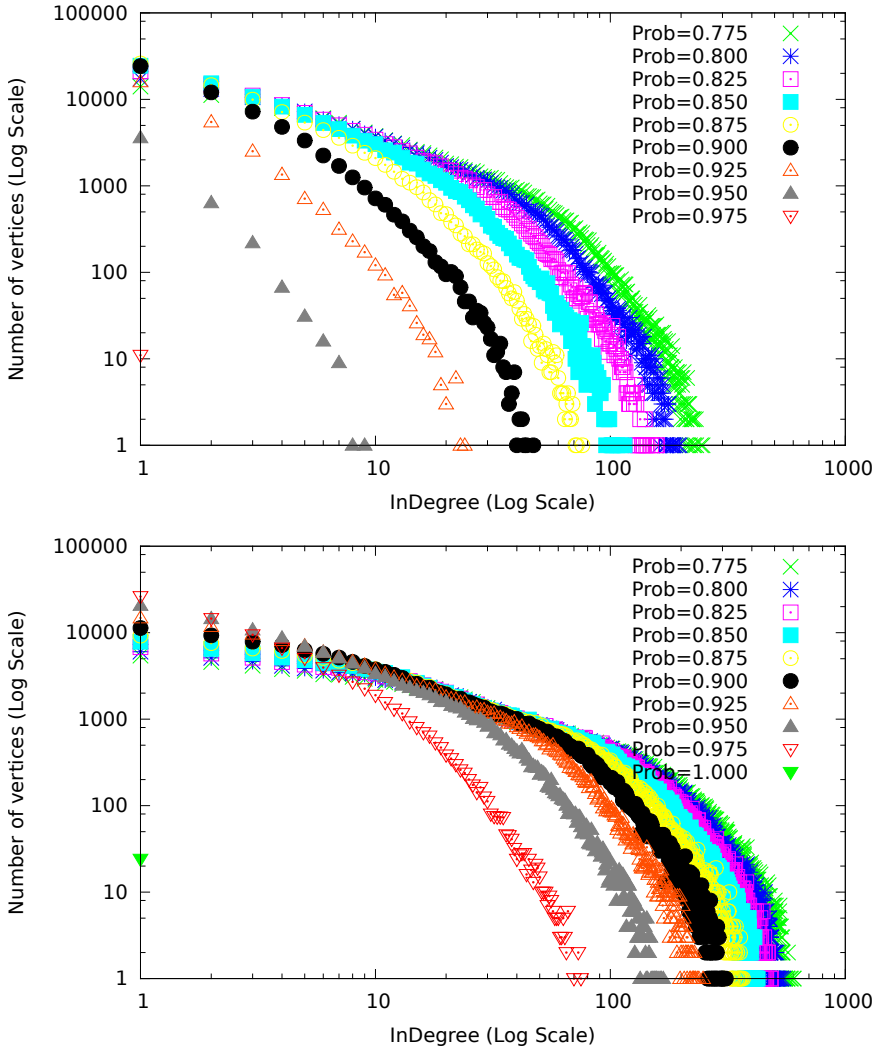


Figure 4.9: Frequency distribution for the vertex *inDegree*. Top: `product` case. Bottom: `logit` case.

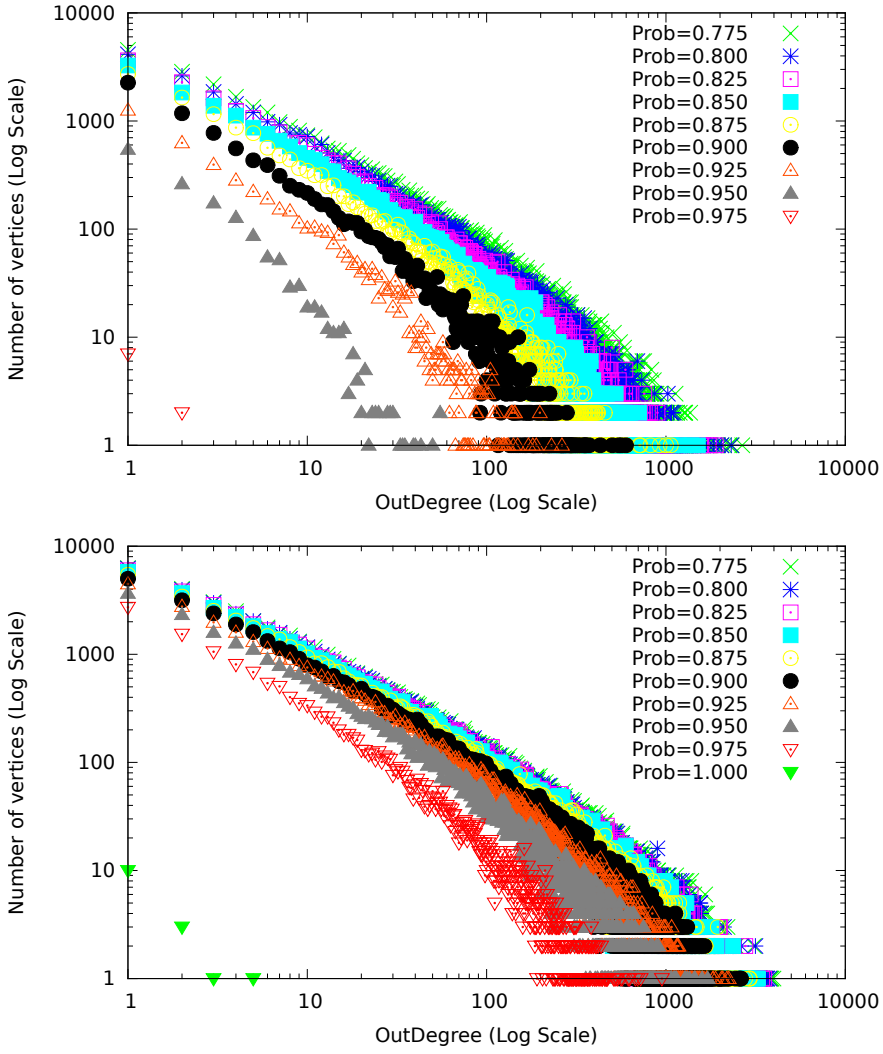


Figure 4.10: Frequency distribution for the vertex *outDegree*. Top: `product` case. Bottom: `logit` case.

probab	nVertices	nEdges	$\frac{nEdges}{nVertices}$	edgeDensity	nComponents
0.750	153407	3162041	20.612	1.343E-4	494
0.775	146532	2254394	15.384	1.049E-4	565
0.800	136012	1511902	11.115	8.172E-5	651
0.825	120373	928590	7.714	6.408E-5	827
0.850	97917	496709	5.072	5.180E-5	1010
0.875	67569	210328	3.112	4.606E-5	1154
0.900	31485	57974	1.841	5.848E-5	1135
0.925	5944	6127	1.030	1.734E-4	763
0.950	20	11	0.550	2.894E-2	9
0.975	0	0			

Table 4.1: Network characteristics for the `product` case, participation level 20%

probab	nVertices	nEdges	$\frac{nEdges}{nVertices}$	edgeDensity	nComponents
0.750	166394	11093195	66.668	4.006E-4	221
0.775	165677	9757662	58.895	3.554E-4	241
0.800	164688	8415148	51.097	3.102E-4	268
0.825	163343	7062333	43.236	2.646E-4	315
0.850	161257	5691034	35.291	2.188E-4	354
0.875	157886	4310196	27.299	1.729E-4	423
0.900	151493	2929452	19.337	1.276E-4	493
0.925	136675	1594380	11.665	8.535E-5	677
0.950	93740	449311	4.793	5.113E-5	1035
0.975	39	24	0.615	1.619E-2	15

Table 4.2: Network characteristics for the `logit` case, participation level 20%

Prob. Threshold	$Size_1$	$Size_2$	$Size_3$	$Size_4$	$Size_5$	$Size_6$	$Size_7$	$Size_8$
0.750	152000	23	22	18	15	14	12	11
0.775	144923	44	23	18	17	14	13	11
0.800	134080	22	20	17	14	14	13	13
0.825	117912	26	21	15	15	14	13	12
0.850	94653	33	28	27	26	23	23	20
0.875	63235	76	56	39	39	39	39	37
0.900	26004	187	139	85	70	69	65	63
0.925	1294	709	173	87	60	56	55	51
0.950	3	3	2	2	2	2	2	2
0.975	0							

Table 4.3: Sizes for the eight largest connected components as a function of the probability threshold. Case: `product`, participation level 20%

Prob. Threshold	$Size_1$	$Size_2$	$Size_3$	$Size_4$	$Size_5$	$Size_6$	$Size_7$	$Size_8$
0.750	165772	18	13	9	8	8	8	8
0.775	165026	10	10	8	8	8	7	7
0.800	163953	14	10	9	9	8	8	8
0.825	162432	19	17	17	14	13	10	10
0.850	160300	12	11	11	11	11	10	10
0.875	156716	36	24	12	10	10	10	9
0.900	150043	40	29	21	16	16	16	13
0.925	134686	40	18	17	16	16	14	14
0.950	90506	30	22	19	19	17	16	16
0.975	6	4	3	3	3	2	2	2

Table 4.4: Sizes for the eight largest connected components as a function of the probability threshold. Case: `logit`, participation level 20%

probab	nVertices	nEdges	$\frac{nEdges}{nVertices}$	edgeDensity	nComponents
0.750	124329	6222163	50.045	4.025E-4	210
0.775	123622	5469193	44.241	3.578E-4	222
0.800	122683	4712628	38.413	3.131E-4	234
0.825	121395	3950660	32.543	2.680E-4	294
0.850	119465	3180767	26.625	2.228E-4	328
0.875	116236	2403988	20.681	1.779E-4	363
0.900	110659	1630452	14.734	1.331E-4	460
0.925	98329	882112	8.971	9.123E-5	651
0.950	64392	242630	3.768	5.851E-5	924
0.975	14	7	0.500	3.846E-2	7

Table 4.5: Network characteristics for the `logit` case, participation level 15%

probab	nVertices	nEdges	$\frac{nEdges}{nVertices}$	edgeDensity	nComponents
0.750	81629	2758538	33.793	4.139E-4	210
0.775	81000	2426100	29.951	3.697E-4	230
0.800	80191	2092183	26.089	3.253E-4	255
0.825	79096	1754861	22.186	2.805E-4	293
0.850	77501	1412608	18.226	2.351E-4	328
0.875	74969	1069807	14.269	1.903E-4	375
0.900	70300	726894	10.339	1.470E-4	462
0.925	60805	394135	6.481	1.066E-4	568
0.950	37650	109725	2.914	7.740E-5	821
0.975	0	0			

Table 4.6: Network characteristics for the `logit` case, participation level 10%

Prob. Threshold	$Size_1$	$Size_2$	$Size_3$	$Size_4$	$Size_5$	$Size_6$	$Size_7$	$Size_8$
0.750	123779	16	13	8	8	7	7	7
0.775	123000	34	19	13	10	7	7	7
0.800	122005	19	18	17	13	10	9	8
0.825	120557	19	16	15	15	11	10	10
0.850	118525	18	15	12	12	12	11	11
0.875	115208	23	19	15	12	11	11	11
0.900	109353	25	12	11	11	11	10	9
0.925	96328	37	30	25	25	19	16	16
0.950	61090	73	37	36	35	33	32	32
0.975	2	2	2	2	2	2	2	

Table 4.7: Sizes for the eight largest connected components as a function of the probability threshold. Case: `logit`, participation level 15%

Prob. Threshold	$Size_1$	$Size_2$	$Size_3$	$Size_4$	$Size_5$	$Size_6$	$Size_7$	$Size_8$
0.750	81009	14	13	13	12	12	11	8
0.775	80318	17	13	13	12	11	11	11
0.800	79453	14	14	14	13	13	12	11
0.825	78308	13	11	11	9	8	8	8
0.850	76622	11	11	10	10	9	9	8
0.875	73896	32	20	14	12	11	10	10
0.900	68877	43	26	21	20	18	14	12
0.925	59059	36	22	14	14	14	13	12
0.950	34491	64	54	41	36	32	30	28
0.975	0							

Table 4.8: Sizes for the eight largest connected components as a function of the probability threshold. Case: `logit`, participation level 10%

that the number drops to zero since trivial components consisting of a single individual are not considered (they cannot carpool).

Figure 4.7 is more relevant since it shows the size of the largest (number of vertices) connected component as a function of the probability threshold. As can be expected from the functions shown in Figures 4.4 and 4.5, large components continue to exist for large probability thresholds in the `logit` case.

Figure 4.8 shows that in both cases the large majority of connected components is quite small. However, the distributions are fat tailed and in both cases very large components occur. Tables 4.3 and 4.4 show the size for the largest eight components. All networks (both cases for all probability threshold values) consist of exactly one giant component. It is not known whether or not the introduction of profile similarity attributes will disintegrate the giant component. No evidence for this has been found in the literature but the phenomenon cannot be excluded either. Also note the narrow range for the sizes of the second, third and fourth largest components.

Figure 4.9 shows that the *inDegree* does not largely differ between the `product` and `logit` cases. The same observation is made for the *outDegree* in Figure 4.10. Note however the difference between the distributions for *inDegree* and *outDegree* in both the `product` and `logit` cases. Clearly, the *outDegree* can grow to very large values for a small number of vertices.

The experiment described in 4.7.4, first was executed for the case where 20% of the commuters make use of the *GCPMS* (see Section 4.7.1). It has been repeated for the cases where 15% and 10% participation levels also. Network characteristics results have been summarized in tables 4.5 and 4.6 respectively. Those tables shall be compared to table 4.2 that applies to the 20% participation level. Note that the size of the largest component and the average vertex degree drop with decreasing participation level; note also that the edge density seems to remain more or less unchanged. From tables 4.7 and 4.8 follows that the networks still constitute of a single giant component and hence do not yet disintegrate at the 10% and 15% assumed participation levels investigated.

4.9 Conclusion

Giving optimal advice in a global carpooling advisor requires the solution of an NP-hard optimization problem. In order to support the search for feasible solutions, we estimated the characteristic features of the carpooling candidates network. Daily agendas for a synthetic population were predicted and the resulting network was

constructed. It was assumed that both time and path (route) similarity are relevant factors for negotiation success. No feedback data about negotiations is already available. Therefore, β parameters for a logistic regression have been estimated by assuming that the resulting probability is known for specific time and path similarity values. The resulting logit predictor has been applied to the commuting trips found in the agendas. The carpooling candidates network was built and analyzed. It turns out to consist of a single giant component and a large number of small ones. The size of the giant component and inDegree/outDegree distributions have been calculated. Even for fairly small level of carpooling interest, the size of the giant component grows large. When the carpooling participation level drops, the average vertex degree drops but the size of the giant component hardly does.

For various algorithms and heuristics for the carpooling advisor see Ben-Arroyo Hartman et al. (2014).

Acknowledgments

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement nr 270833.

4.10 Conclusion to Chapters on Carpooling

The main limitations and challenges for the models described in chapters 3 and 4 are listed below. Those lead to open interesting research paths. Extensions to the ongoing carpooling research are suggested. In the remainder of the section $\overline{n}$ denotes the size of the largest group of candidates considered for carpooling together and $\overline{c}$ denotes the maximal car capacity (driver included) registered with the advisor. Let $n_P \leq \overline{n}$ denote the number of candidate participants for a carpool.

4.10.1 Agent-based Models

The Agent-Based Models (AgnBMs) described in Hussain et al. (2014, 2015a,b,d,c) decently handle groups having $2 < \overline{n} \leq \overline{c}$ for *time interval similarity* calculation. The reported models however consider carpooling for commuting and assume that all cooperating individuals share the *home* and *work* locations respectively. Neither do those models consider the use of Carpool Parkings (CPPs). Hence the *path similarity* requirement is fulfilled automatically. At the time of writing this thesis, the first model supporting multiple home locations, is ready for testing. In this model we do not (yet) consider carpool parkings and everyone is picked up at home. This is done by considering every permutation of the n_P participants, hence $n_P!$ cases for which to calculate distances and evaluate time windows.

4.10.2 Co-routing Problem Size

Huijbregts (2015) handles the co-routing problem in a realistic scale (Flanders region) but for the case $\overline{n} = 2$ only.

The required computational effort for the co-routing problem analyzed in Knapen et al. (2012a) is estimated as follows.

1. The number of candidate participants in each case to evaluate for the co-routing problem, is restricted to the car size ($n_P \leq \overline{c}$).

2. Let n_C denote the number of carpool parkings feasible for at least two of the candidates in the considered set.

3. The number of *layers* in the proposed algorithm, equals n_P. The size of the *transferium* set $n_T = n_C + n_P$. The number of partitions of the layer set is given by the Bell number $B(n_P)$. For a partition of size x the number of possible assignments of transferia is $\binom{n_T}{x}$. An upper bound for the number of

possible assignments N then is given by

$$N \leq B(n_P) \cdot \max_{x \in [1,n_P]} \binom{n_T}{x} \tag{4.23}$$

$$N \leq B(n_P) \cdot \binom{n_T}{n_T/2} \tag{4.24}$$

4. The number of times to execute the route determination algorithm in an AgnBM is the same in cases with and without the use of carpool parkings. The algorithm is executed every time an *exploring agent* contacts another one with a request to build or join a carpool. The required computational effort is compared to the one for the currently used model (section 4.10.2) by the ratio R:

$$R = \frac{N}{n_P!} \leq B(n_P) \cdot \binom{n_T}{n_T/2} \cdot \frac{1}{n_P!} \tag{4.25}$$

5. n_C is expected to be small because of the maximal detour factors considered: e.g. between 2 and 5. For $n_P = 4, n_C = 3$ one has $N \leq 15 \cdot \binom{7}{3} = 525$ cases and $R \leq 21.9$. It can be provisionally concluded that for realistic numbers, the upper bound N can be estimated to be two orders of magnitude larger than $n_p!$. More research is required to find a smaller upper bound.

4.10.3 Global CarPooling Matching Service (GCPMS)

The Global CarPooling Matching Service (GCPMS) needs more attention.

4.10.3.1 Objective Function

It can be argued that the objective function in the optimization problem shall contain the logarithm of the weights. The probability to find a particular constellation of carpoolers is given by

$$maximize \prod_{i,j \in [1,N]} w_{i,j}^{x_{i,j}} \tag{4.26}$$

which leads to

$$maximize \sum_{i,j \in [1,N]} \ln(w_{i,j}) \cdot x_{i,j} \tag{4.27}$$

because $ln(x)$ is monotonically increasing.

4.10.3.2 Probability Estimation for Larger Carpools

The case described in chapter 4 estimates the negotiation success probability values for pairs of carpooling candidates ($\overline{n} = 2$). An operational GCPMS shall be equipped with multiple logit estimators to capture the negotiation feedback for candidate sets of size $2 \leq \overline{n} \leq \overline{c}$. This is estimated not to be problematic since $\overline{n}$ is small (typically $\overline{n} = 5$). However, estimating the β coefficients for the cases having (near) full cars could be problematic due to their infrequent occurrence and hence the lack of negotiation result feedback for such cases.

4.10.3.3 Path Similarity

The effect of the location choice problem mentioned in section 1.4 on the average value for *path similarity* used in this chapter, is yet unknown and shall be covered by future research.

4.10.3.4 Cohesion

The *cohesion* concept is not adequate: it shall not apply to an agreement but to a pair $\langle \text{agreement}, \text{PTE} \rangle$ because the set of participants in a pool can evolve and not every participant belongs to the pool for the same period of time. Definitions for *agreement* and *cohesion* need to be revised.

4.10.3.5 Star Cover Model and Carpools of $size > 2$

The MILP and equivalent *star cover* problem formulations make use of the probability values for pairs only. They assume that the probability estimated for the negotiation success between individuals i_A and i_B is a constant irrespective of the fact that either of them already belongs to a carpool. This design decision was motivated by practical concerns. Apart from the difficulties to estimate the required probabilities, brief analysis of the problem suggests that, taking into account the estimated success probability for negotiation in groups larger than two, might lead to an intractable problem.

Suppose that sufficient negotiation result feedback data are available. Note that again no carpool parkings are used. The method then can be extended as follows.

1. For path similarity, the method used in (4.1) can easily be extended but requires the passenger pick-up order to be known. Hence, for each possible driver, a Vehicle Routing Problem (VRP) needs to be solved. This is expensive but not problematic since the number of pool members is limited by $\overline{n}$.

2. The time interval similarity function (4.4) can be used. Time intervals follow from the selected path and hence also depend on the selected driver.

3. Then for a given set of candidate PTEs, the time and path similarities can be calculated for each potential driver. Those are used by the logit estimator that is applicable for the size of the candidates group to predict the negotiation result. The case delivering the maximum value is retained and determines the driver. The corresponding probability is assigned to the group as a weight. In practical cases a probability threshold p_{min} shall be used so that individuals corresponding to a set of PTE shall be advised to negotiate only if the probability for success is sufficiently large (strictly larger than p_{min} : the reason is given in item 5). If the calculated probability is below or equal to the threshold, the set of PTEs is considered not to be useful and it gets zero-weight.

4. A graph G similar to the one mentioned in 4.5.2 is build. The same weights for PTE pairs are considered *to decide* which PTEs shall be connected by an edge. Let $Prob(s)$ denote the negotiation success probability for a set s of individuals. Then $Prob(s_0) \geq Prob(s_0 \cup s_1)$. Hence by using the probability for pairwise negotiation success as a criterion to add an edge, we will not exclude any pair that is member of a larger successfully negotiated group. Note that no weight is assigned to the edges in this graph.

5. The GCPMS then needs to find a *maximum weight clique cover* of the graph G, subject to the *car capacity constraint*. Instead of assigning a weight to each edge, a weight is assigned to each clique. Cliques having size $> \overline{n}$ get zero weight which will exclude them from the solution since each such clique can be split into smaller cliques having a weight that is larger; in the ultimate case, the clique is split into a collection of singletons. Cliques having insufficient negotiation success probability also get zero weight (see item 3). Each singleton is assigned a non-zero weight w_s such that $w_s < \underline{w}/\overline{n}$ where $\underline{w}$ denotes the smallest among the strictly positive non-singleton clique weights. This is done to avoid a trend to generate solutions consisting of singletons. Then $\underline{w} = p_{min}$ and the weight for a singleton is $p_{min}/\overline{n}$.

6. The *star cover* for G is no longer a suitable model because of the additional requirement that every *star* shall consist of vertices constituting a clique. In the algebraic formulation this results in an additional set of non-linear equations

(the notation of section 4.6.1 is used: j denotes the driver)

$$\forall j \forall i \forall k : x_{i,j} \cdot x_{k,j} \leq x_{i,k} \tag{4.28}$$

Equation (4.28) is equivalent to an implication.

7. The solution is re-formulated as follows:
   (a) determine all cliques in G for which the size is not larger than the maximal car capacity
   (b) create a graph $G^C(V^C, E^C)$ where each clique found in the previous step corresponds to a vertex and vertices are connected by an edge if and only if the corresponding cliques are not disjoint (G^C is a subgraph of the clique intersection graph for G because the clique size is limited by the car capacity).
   (c) observe that overlapping cliques C_A and C_B cannot both be part of the solution since each clique represents a potential carpool
   (d) each clique has a weight (calculated by the appropriate logit estimator) and hence each vertex in G^C is weighted
   (e) then find a *maximum weight independent set* in graph G^C

   This is an NP hard problem.

8. At this moment the only numerical evidence available is the frequency distribution for the size of the carpools that emerge in our agent based model (Hussain et al. (2015d))

Size	Share	Symbol
2	0.70	α_2
3	0.22	α_3
4	0.08	α_4

   One can expect that the number of cliques of size 3 in the graph G that represent potential co-travelers, would be $s_3 \approx \frac{\alpha_3}{\alpha_2} \cdot N_E = \frac{0.22}{0.70} \cdot N_E$ where N_E is the number of edges found in graph G. The expression for s_3 holds because (i) an edge in G corresponds to a clique of size 2 and because (ii) the cliques are based on (nearly) the same similarity conditions as the ones used in the agent based model. This gives an idea about the number of logit predictions to compute. However, all those 3-cliques consist of PTE that are mutually compatible in pairs (otherwise they would not constitute a clique); hence they are all embedded in the graph

used in the study and therefore the size of the connected components will not change. Note that the number of vertices in G^C is given by

$$N_{V^C} = N_V + N_E \cdot \Big(\sum_{i=2}^{i=maxCarCap} \frac{\alpha_i}{\alpha_2} \Big) \tag{4.29}$$

where N_V is the number of vertices in the original PTE graph (since those are trivial cliques). For the number of edges

$$N_{E^C} \geq N_E \cdot 2 \tag{4.30}$$

since each pair in G induces at least two edges in G^C because two singleton cliques are contained in the pair-clique. The weights can be assumed to equal one (since sufficient negotiation success probability is required); this reduces the problem to a maximum (cardinality) independent set problem which is NP hard.

Because the problem is intractable and the required data collection cannot be expected to be successful, this solution is not considered to be feasible in practice.

4.10.3.6 Evolution

The size of the problem found in this chapter, is an upper bound because incoming requests arrive at a specific rate and not all at once. Furthermore, carpools disintegrate after some time, possibly leading to new requests and possibly triggered by new proposals being posted. Hence, the set of requests to handle evolves over time. The number of unfulfilled proposals at any time is expected to be smaller than in the case considered in this chapter and hence the average negotiation success probability will be lower. On the other hand, some effective carpoolers might consider to look for new opportunities. The set of available candidates depends on time and on behavioral phenomena. A difficult question is *when* to answer a request and to provide an advice; the longer the customer is prepared to wait, the higher the probability to find opportunities having a high negotiation success probability. To solve the mentioned time dependency, a new business model is to be designed. The problem size is expected to decrease but this could render the system operationally infeasible due to lack of pairs that can be combined.

Chapter 5

Within Day Rescheduling

This chapter consists of	
Knapen et al. (2013c)	*Within Day Rescheduling Micro-simulation combined with Macrosimulated Traffic*
which is based on	
Knapen et al. (2012d)	*Framework to Evaluate Rescheduling due to Unexpected Events in an Activity-Based Model*

5.1 Research Context

In section 1.4 the difference between *planning* (building a list of activities to be executed) and *scheduling* (completing all activity and trip attributes, included timing) for a given period, was explained. In this chapter, *rescheduling* is studied. The need for rescheduling models is argued as follows.

In the activity-based modeling context, travel demand is predicted by generating daily agendas for all members in a population and aggregating the demand from the individual schedules. Schedule predictors are based on data acquired by surveys which collect travel demand and in some cases time-use information. The resulting data and model predictions apply to the situation for which they were collected.

In many cases it is important to know how the demand changes when the context changes. The context for each individual consists of the environment, the personal attributes and objectives as well as the current behavior. Knowledge about changing demand is relevant to evaluate the effect of (i) unexpected incidents (environment changes) and of (ii) Travel Demand Measure (TDM) measures (changes in environ-

ment or in personal attributes).

TDM measures aim to optimize the use of the transportation infrastructure and services e.g. at the level of efficiency (congestion reduction), energy consumption. The intended change in travel demand is to be realized by change in travel behavior.

When the current and the expected situations are known, it generally is very difficult to find out how to control the evolution to the expected situation. The set of parameters and the required changes cannot be observed directly because, due to the complexity of the problem, it is in general not possible to identify a case having sufficiently similar initial en final situations respectively for which the evolution can be analyzed.

The analyst might be interested in the final situation only (e.g. congestion decrease in peak periods, electric power demand peak shaving, increasing the share of public or other collective transportation in the total set of trips driven). However, in any case the new situation is to be reached from the current one and hence requires *particular changes*. This suggests the importance of modeling *change* and hence schedule adaptation.

Some objectives for research on schedule adaptation due to TDM measures, are: (i) to evaluate the effect of stimuli (ii) to find out whether or not the expected final state can be realized, (iii) to study the evolution of the travel demand over time.

Micro-models are useful tools to solve the questions raised. They allow to take into account both the individual's socio-demographic properties and the schedule that is currently used by the individual. This is important since adaptation to unexpected environment changes and to TDM measures, requires modeling the change of the schedule relative to the current situation.

When collecting survey data, it is essential to record data about both the initial and final (either revealed or stated preference) states as well as data about factors triggering the change. In the state change prediction model, data about the initial state and the change need to be kept together since they essentially constitute a single piece of information. This requirement is based on the assumption that the scheduling and rescheduling behavior of an individual are closely related.

Micro-simulation allows to relate the scheduling and rescheduling behavior for a given individual. For this reason we argue that using micro-simulation for both scheduling and rescheduling constitutes a solution that is expected to generate more accurate predictions than tools operating at an aggregated level. The latter methods would first predict the current behavior at an aggregated level and then predict either the changed behavior or the changes relative to the current behavior.

The *framework* described in this chapter is aimed at evaluation of *rescheduling*

without replanning. Actors can change trip and activity attributes but not drop or add activities. Without re-planning, a schedule can be adapted by (i) *re-timing* of trips and activities, (ii) *re-location* of activities and (iii) changing the selected *mode*.

Individuals are assumed to act independently. No individual can be made aware of the schedule of another one. Constraints resulting from coordination need to be introduced by deriving them from the socio-demographic properties and from the schedule. For example, sampling based on survey results can be used (i) to determine the allowed change in the start time of a social activity (which is by definition a cooperation) and (ii) to find out whether a bring-get (pick-drop) activity induces a hard time constraint (e.g. bring someone to the train station or pick up a child from school).

The proposed framework is a hybrid model because it combines microscopic schedule adaptation with aggregated traffic assignment based on the use of Traffic Analysis Zones (TAZs). The model keeps track of a schedule for each individual. Rescheduling decisions are micro-modeled at the individual level; the resulting travel demand is aggregated for a period of one hour and loaded onto the network; this is done for every time-of-day that is an integer multiple of 15[min] by using a sliding window to determine the demand for a one hour period. The resulting interzonal travel times are skimmed from the result of the traffic assignment and made available to the individuals. Hence, the framework supports rescheduling models that do not require feedback of detailed route information. Advantages and drawbacks of technique are discussed in section 5.12.

Following components of the rescheduling behavior model are covered by the research reported in this chapter:

1. the mechanism of schedule adaptation
2. the degrees of freedom in a schedule
3. the method to evaluate (marginal) utility
4. the criteria used to compare the result with the original schedule that is used as the reference. The original schedule is assumed to be optimal.
5. the effect of the (delayed) perception of changes in the environment on the rescheduling decisions. What does the individual perceive ? How is the observation interpreted ?
6. the beliefs of the individual that constitute the base for the prediction of the near future (e.g. expected travel times)
7. time constraints in schedules

At the technical level, special attention was paid to the adjustment of the initial schedules and initial travel time matrices in the hybrid system. This is a non-trivial and critical problem. It is *required* because the schedule predictor (FEATHERS) uses off-peak and peak period travel time matrices while the schedule adapter (WIDRS) requires a travel time matrix for every 15[min] in order to create a feedback loop. The initial adjustment is *critical* because it essentially involves schedule adaptation for technical reasons. The schedules resulting from the adjustment stage are used as a reference when evaluating adaptations induced by the phenomenon being studied.

The rescheduling model used as a *first research* case and described in this chapter, allows for *re-timing* only. As a consequence, the total demand for every OD pair for the simulated period does not change; only the distribution of trips over time changes. Therefore, the application domain for the implemented case is restricted to one-time short-term rescheduling i.e. cases where no replanning is required and locations as well as mode changes are not possible. This covers unexpected event occurrence which leaves the goals for every individual unchanged (emergency or evacuation situations are excluded because they change the short term objectives of people and hence require replanning).

Research questions covered in this chapter are:

1. How to model schedule adaptation for the cases where the planning remains unchanged ? How can the rescheduling decision model be decomposed into submodels ? How to determine Degrees of Freedom (DOFs) in the schedule ?

2. Can aggregated Traffic Assignment (TA) be used to evaluate rescheduling behavior models ?

5.2 Abstract

The concept of rescheduling is essential to activity-based modeling in order to calculate effects of both unexpected incidents and adaptation of individuals to traffic demand management measures. When collaboration between individuals is involved or timetable based public transportation modes are chosen, rescheduling becomes complex. This paper describes a new framework to investigate algorithms for rescheduling at a large scale. The framework allows to explicitly model the information flow between traffic information services and travelers. It combines macroscopic traffic assignment with microscopic simulation of agents adapting their schedules. Perception filtering is introduced to allow for traveler specific interpretation of perceived macroscopic data and for information going unnoticed; perception filters feed person specific short term predictions about the environment required for schedule adaptation. Individuals are assumed to maximize schedule utility. Initial agendas are created by the FEATHERS activity-based schedule generator for mutually independent individuals using an undisturbed loaded transportation network. The new framework allows both actor behavior and external phenomena to influence the transportation network state; individuals interpret the state changes via perception filtering and start adapting their schedules, again affecting the network via updated traffic demand. The first rescheduling mechanism that has been investigated uses marginal utility that monotonically decreases with activity duration and a monotonically converging relaxation algorithm to efficiently determine the new activity timing. The current framework implementation is aimed to support re-timing, re-location and activity re-sequencing; re-routing at the level of the individual however, requires microscopic travel simulation.

5.3 Introduction

Nowadays, activity-based models are used to generate daily schedules for members of synthetic populations in order to estimate time dependent travel demand. Microsimulation allows to take into account specific traits for each individual so that sensitivity to travel demand management (TDM) measures can be modeled at the level of the individual actors. The overall effect of those measures then emerges from the simulation. Many models assume that daily planning decisions are taken one day ahead and predict schedules that are assumed to be immutable (i.e. executed exactly according to the plan). In actual practice, most people adjust their daily schedule during execution, either because of unexpected event occurrence or because the

individual discovers new opportunities or acquires more complete information. Accounting for this phenomenon, requires schedule (daily agenda) *execution* simulation. While executing the predicted schedule, individuals make use of the shared transportation network facilities. Each individual has incomplete or biased knowledge of the environment; this phenomenon is modeled via *perception filters*. The environment can change due to exogenous phenomena that affect the shared resources (e.g. a traffic incident or adverse weather conditions can decrease the network capacity in some region). As a consequence, some individuals adapt their planning which affects the load on the network which in turn affects the available capacity as a function of time and space. In the WithIn Day Re-Scheduling (WIDRS) model, information about the environment is fed back to the individual. Actor behavior is determined by the perceived state of the network and by expectations about its short term evolution. The schedule execution simulator does not strive to an equilibrium state because that would require information about the future; the system never is assumed to be in a steady state.

5.3.1 Aim of the paper

Both the basic concepts and model details for the WIDRS project are described. The first part of the text discusses related research and shows the principle of operation; it gives an overview of building blocks involved and highlights their interactions. The second part starting at section 5.6 explains details of several essential components. Section 5.7 discusses the results for an experiment involving the evaluation of the effect of a large scale road capacity reduction. Sections 5.9 and 5.8 present conclusions drawn from the experiment and plans for the future.

5.3.2 Project Objectives

The WIDRS framework is a software tool to evaluate schedule adaptation by individuals as a response to changed conditions in the environment. This project is part of our research efforts concerning dynamic activity-based simulation. Mutual dependency of individuals only is caused by sharing limited capacity resources. Direct interactions between individuals are not considered. The WIDRS project is aimed at large scale simulations used to investigate traffic demand management (TDM) measures. The experiment described here aims at quantifying the effect of an unexpected incident on both the schedules and on the time-dependent road network conditions. The incident (traffic accident, non-predicted meteo condition) is modeled by local capacity reduction on the road network.

5.4 Related Work - Context

5.4.1 Research on Schedule Adaptation

In a first category of research efforts, mechanisms underlying the schedule construction and schedule adaptation processes are investigated. The *Aurora* model developed in Joh (2004) provides for schedule generation and dynamic activity-travel rescheduling decisions. *Aurora* is based on S-shaped utility functions. The maximal utility value attainable for an activity, is given by the product of functions modeling the attenuation by start time, location, position in agenda and time gap since last execution of the activity. Bounded rationality individuals are assumed. Arentze et al. (2005) present a comprehensive description of *Aurora*. People are simulated as individual agents. A comprehensive model has been specified describing the insertion, re-positioning, deletion and substitution of activities as well as changing locations, trip chaining options and transport modes. Models of this level of detail are required to integrate cooperation concepts in a simulator (e.g. joint activity execution or carpooling). The paper describes the use of *Aurora* in an experimental setup to study schedules consisting of work activities and green recreation activities in several scenarios.

Recker (1994) and Gan and Recker (2008) present a mixed integer programming formulation of the HARP problem (Household Activity Rescheduling Problem). Both papers report on an extensively elaborated rescheduling model that has been applied to a small amount of individuals suffering from a pre-specified loss of time. The idea is that, while planning, people solve a Mixed Integer Linear Program (MILP). The examples given show that realistic schedules are produced. However, the number of constraints required in the model is large. The level of detail does not allow for large scale deployment.

Jang and Chiu (2010) describe a model that uses a quadratic utility function and integrates the scheduler with a dynamic traffic assignment tool DynusT. A similar approach has been taken by Bekhor et al. (2011) who integrated an activity-based model for Tel-Aviv with the MATSim toolkit allowing for re-timing and re-routing.

Jenelius et al. (2011) analytically derive the optimal timing in a schedule composed of three activities and two trips. The authors analyze a model using marginal utility functions that are linear combinations of time-of-day based and duration based components.

Pendyala et al. (2012a) present the integration of the open source travel demand model *OpenAMOS* with the traffic micro-simulator *MALTA* (Multi Resolution Assignment and Loading of Traffic Activities) in SimTRAVEL. That is used to de-

termine the effect of unexpected network disruption. The impact of the disruption and the network congestion dissipation dynamics are calculated for several information provision scenarios. For each minute of the day, the demand model simulates activity-travel engagement decisions for all individuals. The results are fed into the traffic assignment model that routes the trips from origin to destination and simulates car movements. Network skims by time of day are used to feed back expected travel times to the activity-based model. Unfortunately, the paper does not explain the schedule adaptation model. Three simulations were conducted: (i) no information provision to the travelers, (ii) notification of people who are about to embark on a trip without *en-route* route switching by travelers and (iii) information diversion combined with en-route route adaptation. Pendyala et al. (2012b) discuss the same framework. In this case several iterations are run over a single day mimicking the fact that travelers learn from experience (like in MATSim). This however shall not be done while simulating unexpected events. Konduri et al. (2014) show how the same framework is used to evaluate the effect of road pricing. Value of time (VOT) and household income are used to calculate an extra virtual trip delay from the trip cost. This required time expansion reduces the area covered by the time-space prisms and hence reduces the activity location choice sets.

Zhao and Sadek (2014) describe the creation process of the TRANSIMS based agent-based regional model for Buffalo-Niagara area. The model is calibrated for both normal and inclement weather situations. For the latter, probe vehicles were used on several routes. The model is used to simulate effects of incidents, inclement weather and the combination of those.

In a second category of research, factors influencing rescheduling characteristics under given circumstances and for specific activities, are being determined from surveys. Nijland et al. (2009) estimate parameters for the *Aurora* model. Flexible activities only are covered. The authors conclude that *activity relocation* and *mode change* rarely occur; they report that 55% of rescheduling is by duration reduction and 35% by activity dropping.

van Bladel et al. (2006) point out the difficulties to estimate the utility function parameters and show the S-shaped dependency of the utility on the time gap since the preceding execution of a same activity (*needs based* model). Roorda and Andre (2007) use an Multinomial Logit (MNL) model to uncover the factors that determine the choice between several rescheduling options after a well-defined unexpected delay has been presented to the respondent. van Bladel et al. (2009) use mixed logit models with random effects to estimate the effect of several factors on rescheduling. Guo et al. (2012) state that econometric models are not suited to model within-day

rescheduling. The authors list the main characteristics of Computational Process Models (CPM) and their assumed shortcomings (dichotomous classification of activities: fixed/flexible, dichotomous classification of decision making process: pre-day-planning/on-day-rescheduling, plan completion and certainty (as opposed to partial activity attribute planning)). They describe a data collection method to acquire data to uncover the (re)planning process. It consists of (i) schedule data entry (ii) GPS recording (iii) comparison of planned and recorded activities including arguments for rescheduling (iv) automatically (artificial intelligence) generated 'what-if' questions about hypothetical rescheduling if some of the planned activities would have been delayed/canceled (stated adaptation survey). The method used is similar to the one described in Weis et al. (2010) who report the frequency of activity compression (due to increased travel duration) but not the amount of compression for several activity types.

5.4.2 Positioning and context for the WIDRS project

This paper belongs to the first category mentioned above; similar to Joh (2004), it explicitly models a set of hypothesized mechanisms. The overall structure of the simulation setup is similar to the one reported in Pendyala et al. (2012a). In this paper however, the network state perception and schedule adaptation models have been elaborated in detail because our main goal is to provide a framework to evaluate travel decision adaptation models. Also, the work described in this paper currently uses SUE (stochastic user equilibrium) traffic assignment. Microscopic routing can be integrated but was not used for computational efficiency reasons. Making use of SUE means that individuals who experience increased travel times, possibly change their route. However, no route information can be extracted from the SUE process to be fed back to the schedule adapter. Since our model does not support en-route route adaptation, it is less suited to investigate dynamic effects on the road network near to the location of disruption.

WIDRS compares to the model described by Konduri et al. (2014) as follows: (i) OpenAMOS feeds the network state back to the activity scheduler for every minute while WIDRS uses a 15[min] period, (ii) WIDRS uses perception filtering so that each individual has a personal interpretation of expected excess travel time after an incident, (iii) WIDRS keeps a fixed activity sequence and maximizes utility by activity duration adaptation whereas OpenAMOS can drop and relocate activities, (iv) WIDRS as well as the SimTRAVEL model without single day iteration (learning) can handle unexpected events and (v) running WIDRS using a 50% fraction of the

population is computationally feasible for regions having 6 million inhabitants such as Flanders.

5.5 Principle of Operation

5.5.1 General Overview

The initial schedule (agenda) for every inhabitant of Flanders (Belgium) is generated by the FEATHERS activity-based model described in Bellemans et al. (2010). In the FEATHERS context a schedule is a sequence of episodes for a period of 24 hours. Each episode consists of a *journey* (trip) and an *activity*.

$$\langle schedule \rangle := \langle episode \rangle * \tag{5.1}$$

$$\langle episode \rangle := [\langle trip \rangle] \langle activity \rangle \tag{5.2}$$

The trip is characterized by a tuple *(origin, destination, startTime, duration, mode)*. The activity is characterized by a tuple *(activityType, location, duration)*. All trip attributes except for the *mode* can be derived from the consecutive activities in between which it is enclosed. For each member of the synthetic population (also called an *actor*), a schedule is predicted.

WIDRS consists of two main interwoven components: *schedule adaptation* (rescheduling) and *schedule execution*. During schedule execution, an individual can detect that the time to travel a specific non-finished trip no longer equals the originally planned trip duration; in such case, the individual estimates the new trip duration and adapts her/his schedule. In general, rescheduling can be done by (i) adapting activity execution start time and/or duration (*re-timing, time period (de-)compression*), (ii) by choosing an alternative location (*relocation*), (iii) by selecting a new activity order (*re-sequencing*), (iv) by trip mode change and finally (v) by dropping or inserting activities. This paper documents the first completed stage in the ongoing WIDRS project. It describes the framework built and the *utility-based (de-)compressor type* rescheduler used in the first experiments.

The research described in this paper focuses on modeling the rescheduling process, making it feasible for large scale application and integrating it with macroscopic traffic assignment in order to investigate the mutual influence of rescheduling and network performance; this focus is shared with the research reported by Pendyala et al. (2012b) (OpenAMOS, MALTA based), Zhao and Sadek (2014) (TRANSIMS based) and Bekhor et al. (2011) (MATSim based). The latter two focus on the effect of unexpected events. This is the focus of the first application of the WIDRS framework.

5.5.2 Routing and Travel Time Estimation

The WIDRS approach differs from the other projects in that it makes use of macroscopic traffic assignment. Rerouting is an essential feature in all MATSim based research. MATSim agents evaluate the executed agenda at the end of each iteration by calculating a score; a fraction of the agents then selects new travel times and routes attempting to find a better solution. The system essentially assumes the existence of an equilibrium state.

In contrast, WIDRS simulates schedule execution whereby each individual interprets the available data in its own specific way and needs to predict the future travel times i.e. the evolution of the *transient* traffic network state after incident occurrence. WIDRS does not assume an equilibrium to exist. Instead, it explicitly models the rescheduling mechanism for each individual without the need for microscopic rerouting. During schedule execution simulation, for each 15[min] period of time, WIDRS computes the actual network load and the associated expected travel times between TAZ. Those results are fed back to the (candidate) travelers who individually estimate the near future travel times and adapt their schedules. Modified expected trip durations result in new estimations for the arrival times for ongoing trips and in reconsideration of departure and arrival times for planned trips.

5.5.3 Network state perception

Individuals can perceive the network state changes at discrete moments in time only. *Network State Evaluation* by individuals is limited to those moments (called *NSE_moments*). The interval between them (15[min] in the current experiments) is called the *NSE_period*. *NSE_moments* define the time resolution for individuals to experience modified congestion effects. This makes it possible to integrate macroscopic network state modeling with microscopic actor behavior modeling. Note that other time related phenomena (activity/trip start times, duration values, notification times) all are modeled by WIDRS using a finer grained time resolution (as continuous variables). Individual actors can decide to reschedule at any moment in time e.g. at the end of a trip or when notified about an incident via a traffic information service (TIS).

5.5.4 Operation

WIDRS operation consists of three steps. The data flow is shown in Figure 5.1, the control flow chart in 5.2. Details can be found in 5.10.

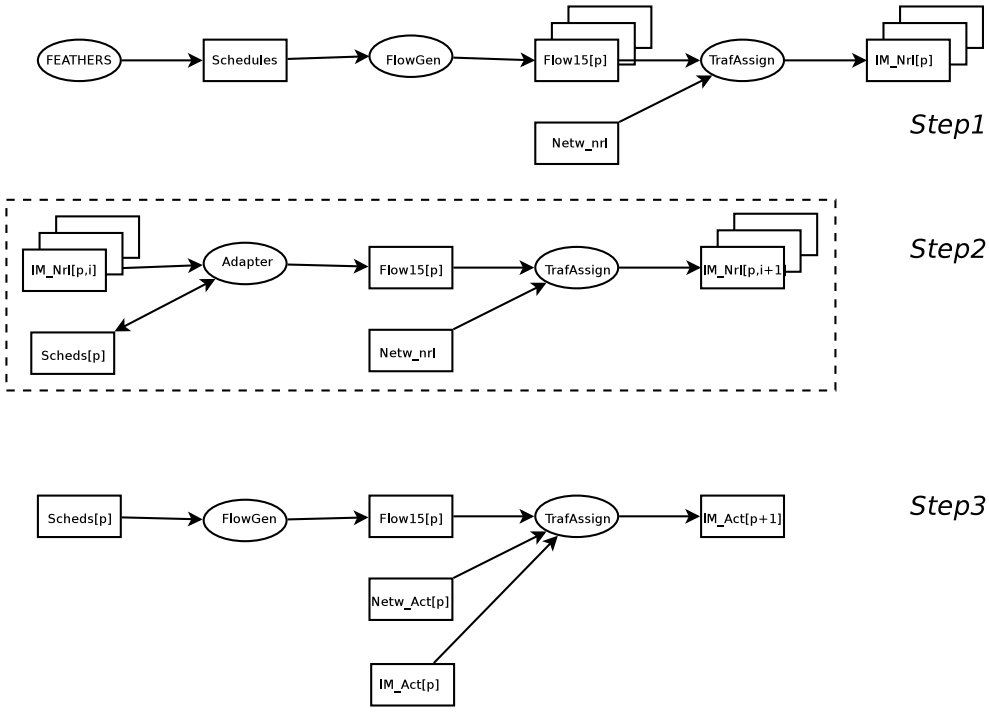


Figure 5.1: WIDRS data flow diagram. Ovals represent algorithms, rectangles represent datasets. **Step1** creates an impedance matrix for every quarter of an hour period p under normal network conditions. **Step2** iteration i is run for each 15[min] period p of the day. It adapts the travel times in the schedules to the travel times for period p resulting in Scheds_p, calculates new flows (Flow15[p]) and impedance matrices IMP_Nrl[p,i+1]) until the flow matrices IM_Nrl[p,i] and IM_Nrl[p,i+1] are sufficiently close to each other. At the end of this step, Scheds_p are consistent with IM_Nrl[p]. **Step3** calculates the flows for period p using trip durations estimated by each individual and assigns those to the actual network (having normal or reduced capacity). This results in the impedance matrix for the next quarter of an hour which, as a consequence, is based on the individual traveler's expectations and plans.

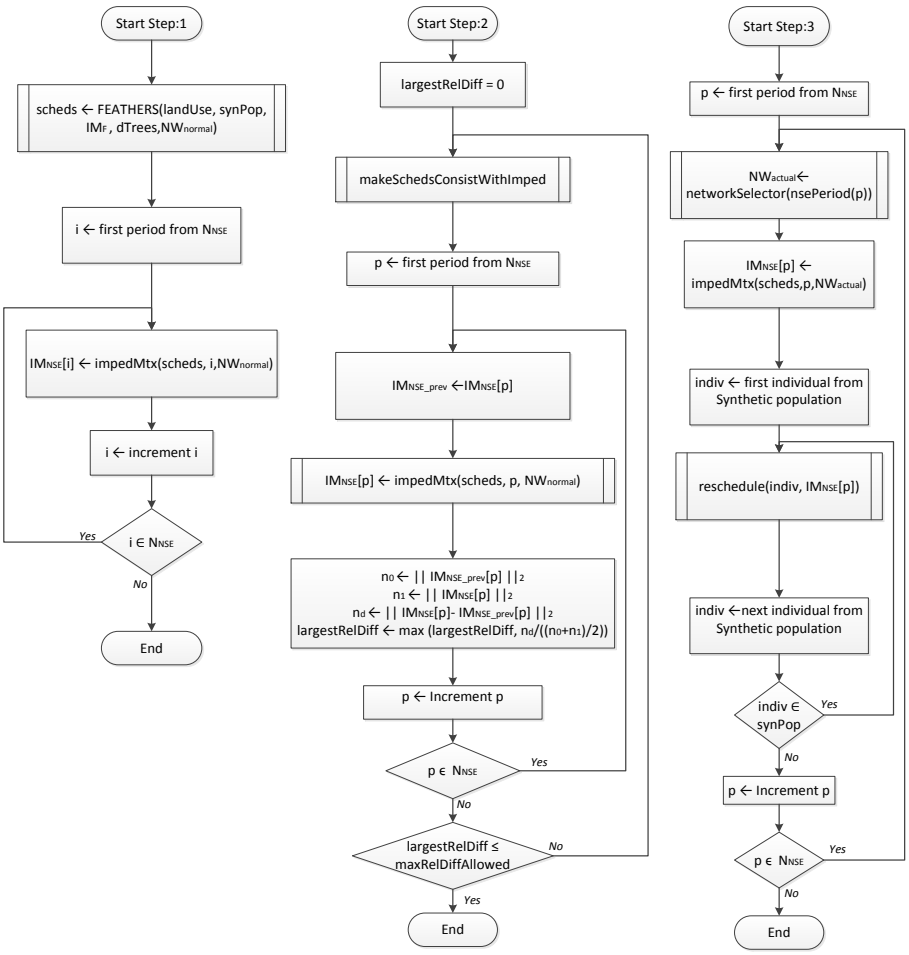


Figure 5.2: Flowchart presenting an overview of the WIDRS procedure. STEP1 prepares initial schedules and creates an OD (origin-destination) travel time matrix

- STEP1 calculates the *initial* impedance matrix for each *NSE_period* under normal network conditions from the schedules generated by FEATHERS using SUE traffic assignment.

- STEP2 repeatedly adjusts the travel times in the schedules and recalculates the OD-travel-time matrices until they are mutually consistent for each *NSE_period*. This is required since the time resolution used by FEATHERS and TransCAD is one hour whereas WIDRS uses 15[min] periods. The travel times in the schedules are weighted averages of travel times for NSE_periods where the overlap between the trip period and the respective NSE_period is used as a weight.

- STEP3 is the actual schedule execution simulator. STEP3 starts from mutually consistent schedules and impedance matrices for the *normal* case. The network characteristics are assumed to suddenly change due to an incident. The capacity for a given set of links is reduced to a pre-specified level for a given *RCP* (Reduced Capacity Period) representing *incident conditions*. Schedule execution then is simulated over a single day. Some travelers will get stuck in congestions additional to the normally expected ones, others will know about the reduced network performance before starting trip execution (see section 5.5.7). All of them need to revise their schedule and to adapt to modified travel times.

5.5.5 Environment Model

The framework is based on traffic flows between traffic analysis zones (TAZ). Macroscopic SUE (Stochastic User Equilibrium) traffic assignment is used to apply the traffic demand derived from the micro-simulated schedules to the transportation network; to that end we run TransCAD as a sub-process of WIDRS and calculate the SUE by means of the Method of Successive Averages (MSA). Microscopic routing is not supported (hence no microscopic *re-routing*). This decision is motivated by the desire to limit the simulation runtime. Travel times between TAZ centroids are calculated using the *minimal duration path tree* and made available in *impedance matrices* (travel duration matrices).

WIDRS uses a specific impedance matrix for each *NSE_period* of 15[min] while FEATHERS only distinguishes between off-peak, morning-peak and evening-peak travel duration values. Therefore, impedance matrices for the *normal* road network situation need to be calculated as reference values from the FEATHERS predicted schedules for each *NSE_period*. Furthermore, the OD-flow matrices derived from the schedules shall be consistent with the impedance matrices. STEP2 in the algorithm

fulfills this requirement. If this step were omitted, we would find schedule adaptations originating not only from network incidents but also from the difference in time resolution used in FEATHERS and in WIDRS respectively.

5.5.6 Actor Model - Behavioral Characteristics

Individuals are assumed to behave in a rational way and to try to maximize their utility by executing activities. As a consequence, in case of modified predicted travel times, they will adapt their schedules. In the first implementation, actors decide about rescheduling in a mutually independent way using information about the network state and their private agenda only: the model does not keep track of direct mutual rescheduling effects since actors are not cooperating. The only mutual influence stems from the use of the shared road network having limited capacity. Future implementations will cover rescheduling caused by negotiation (e.g. for carpooling).

5.5.7 Network State Perception

The model is aimed at simulation of large areas and large sets of individuals (complete populations). Hence, for computational reasons, the resolutions in both time and space are limited.

1. Activity locations are specified to the TAZ (Traffic Analysis Zone) level only; no detailed street addresses are used. Hence, all travel times can be summarized in *OD travel duration* matrices (impedance matrices) for several moments in time; they represent the *network state*. During the simulation, impedance matrices are derived for the *NSE_moments* using the traffic demand generated by adapted schedules and the network that applies (either the *normal* one or a reduced capacity version defined for the *incident* situation).

2. Time resolution is defined by the length of the *NSE_period*.

3. Individual behavior is modeled by *perception filtering*: this accounts for (i) *lack* of information (incomplete knowledge of the network state) and for (ii) personal *interpretation* of the information that becomes available from TIS. Travel duration matrices (impedance matrices) are calculated for both the *normal* and the *incident* situation. The impedance matrices for the normal situation are considered to represent common knowledge about the expected travel times. The excess travel time calculated for the incident situation is considered to be the expected delay made available by the traffic information service. It is interpreted (biased) by each individual in a specific way.

5.5.8 Incidents - Notification

1. In order to apply an incident, the capacity on a given set of network links is reduced by a given factor for a given period of time: this is called *network disturbance*.

2. A *reduced capacity road network* (RCN) inherits the topology from the network it is derived from while the capacity for each element of a given subset of links has been reduced with a given link-specific value. An incident is modeled by its local effect on link capacities: e.g. a traffic incident can reduce the capacity of some links near to the place of the accident, a meteo phenomenon can reduce the link capacities in a large region.

3. In WIDRS, the local effects of an incident are modeled by a given set of tuples $\mathcal{T} = \{(RCN_i, p_i)\}$ where RCN is a reduced capacity network and p_i is a period of time. Each *NSE_period* is assigned either the *normal* network or a *specific* RCN. During the simulation, a new impedance matrix is calculated using the actual traffic load at each NSE_moment taking the time dependent network link capacities into account.

4. Actors can get notified at any moment in time after the incident start time. As a consequence, an actor can get warned before starting a trip for which the duration is affected by the incident: such individual is called a *notified individual*. Those persons become aware of the network travel times disturbance in time in order to reschedule affected trips. Others only become aware after having suffered from unexpected delay (too late to avoid the resulting congestion). Those are called *experiencing individuals*. Each individual can decide to adapt her/his schedule immediately after becoming aware of additional delay (either by notification or by experiencing). Finally, for a given incident, every individual using affected OD-pairs, becomes *experiencing* unless (s)he manages to adapt her/his schedule so as to avoid the extra delay induced.

5. *Experiencing travelers* become aware of being delayed, at *NSE_moments* only. At those moments in time, the affected individuals estimate the actual distance driven and the remaining distance and duration to drive. A new estimate for the total travel time is calculated using data from the impedance matrix holding for the *NSE_moment* at which the evaluation takes place: at this point, the modeled actors compare the most recent estimate of the effective travel duration to the previous one. This is where the modeled *experiencing individual* senses

the positive or negative difference in travel duration and, where appropriate, decides to reschedule.

5.5.9 Schedule Execution Simulation

Schedule execution is simulated by recalculating the travel duration for all non-finished trips for each actor at each *NSE_moment*. There is no iteration towards some equilibrium over a single day because no information about the future shall be made available to the individual as a source for learning. Each individual makes her/his own prediction (interpretation) about the network state in the near future. Schedule execution simulation consists of 96 time-steps (15[min] periods). At the end of every *NSE_period*, the new traffic state on the network is calculated. Persons are processed as follows:

1. people traveling at that moment, become aware of congestion by *experience*, estimate new travel time as described in section 5.6.1.6 and decide to reschedule if the travel time is adapted by more than a specified threshold

2. people who are not traveling but receive a notification during the next *NSE-period*, re-evaluate the travel time for their future trips using the estimates described in 5.6.1.5. If the new estimate differs more than a given threshold from the previous one, they decide to reschedule.

Rescheduling consists of calculating new activity start times by optimizing the total utility. Note that for notified individuals, rescheduling occurs during activity execution; in some cases the individual shall stop the running activity immediately in order to move to the next activity location.

5.6 Component Details

This section explains in more detail some *modeling mechanisms* and algorithm *design decisions* that are of fundamental importance to (i) the WIDRS framework and (ii) the first application that simulates the effect of an unexpected situation on the both daily agendas and on the network status. *Delay estimation* constitutes the major part of this section. Then the *actor behavioral model* is explained (section 5.6.2); this followed by a section on the *compatibility between schedules and impedance matrices* (section 5.6.3) and a note on *network load calculation (section 5.6.4)*.

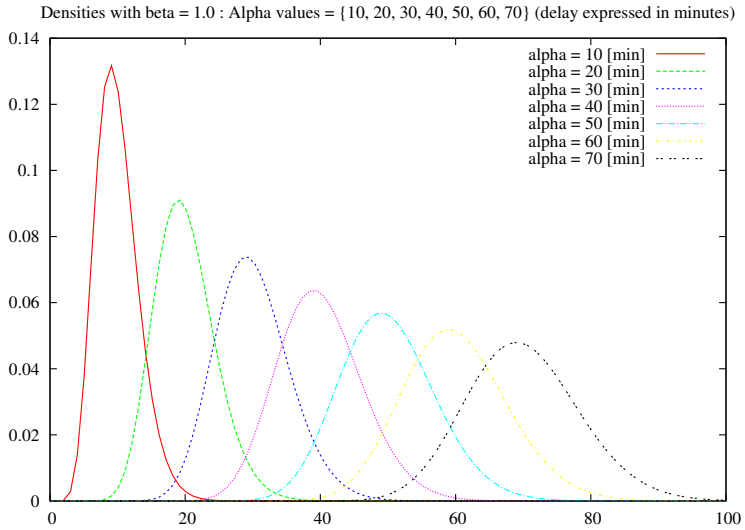


Figure 5.3: Gamma probability densities for delay values estimated by individuals when traffic information service predicts an expected value of 10[min] ... 70[min] for congestion duration. The rate factor $\beta = 1.0$ (scale factor $\theta = \frac{1}{\beta} = 1.0$) for each case.

5.6.1 Delay Estimation

Activity duration selection and expected travel time estimation are essential components for agenda adaptation. First we present the selected distribution for stochastic durations; then several fundamental durations and delays are explained.

5.6.1.1 Modeling Delays

Gamma distributions using scale factor $\beta = 1.0$ are used to model delays and durations. Both the expected value (mean) and variance are given by α.

$$f(x;\alpha,\beta) = \begin{cases} \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} & x > 0; \alpha > 0; \beta > 0 \\ 0 & x \leq 0 \end{cases} \tag{5.3}$$

Sample density functions are shown in Figure 5.3. Gamma distributions have been chosen because of the *reproductive property* (the sum of independent gamma distributed variables with α_1, β and α_2, β is gamma distributed with $(\alpha_1 + \alpha_2), \beta$ which is useful when processing accumulating delays).

5.6.1.2 Incident awareness offset

Both the information *dissemination* mechanism (traffic information service) and the probability for *assimilation* by the individuals are essential model components.

1. Two dissemination models can be considered. The first is the *broadcast* model which is a volatile push mechanism which means that the information sender is the initiator and the message can get lost; radio broadcast information is an example. The second is the *publish* model where the information consumer either subscribes and receives a non-volatile message or decides to consult a (web)service; in this case the information can be consulted multiple times at arbitrary moments in time. Both the time at which an individual gets notified and the levels of information loss and distortion, depend on the mechanism used.

2. No evidence about individual behavior in this respect was available while implementing the initial model. Hence for the experiments, the *broadcast* model is assumed and assimilation probability equals one for each individual notified in time and zero for everyone else. The delay between incident occurrence and broadcast (notification delay) is assumed to be gamma distributed $\omega_{not} \sim gamma(\alpha_{\omega_{not}}, \beta)$ with an expected value of $\alpha_{\omega_{not}} = 30[min]$ (*deliveryDelay* in 5.11). A single gamma density function is used to sample the value for the notification delay. As a consequence, every individual gets informed but many of them too late (those are not informed in time to be able to use the information).

5.6.1.3 Expected incident local effect duration

Early notifications (both by *broadcast* or *publish* mechanisms) can come available before the capacity reduction at the incident location ends. Hence, the *Reduced Capacity Period* (RCP) length is not necessarily known by the receiving actor at notification time and each individual needs to estimate it along with the level and duration of its effect on travel times. It is assumed that the TIS provides in a direct or indirect way some data about the kind of the incident which is used by the individual to estimate the RCP duration. In the current model, the duration of the specified network disturbance is used as the expected value for a gamma distributed stochastic from which each individual samples to estimate the RCP duration.

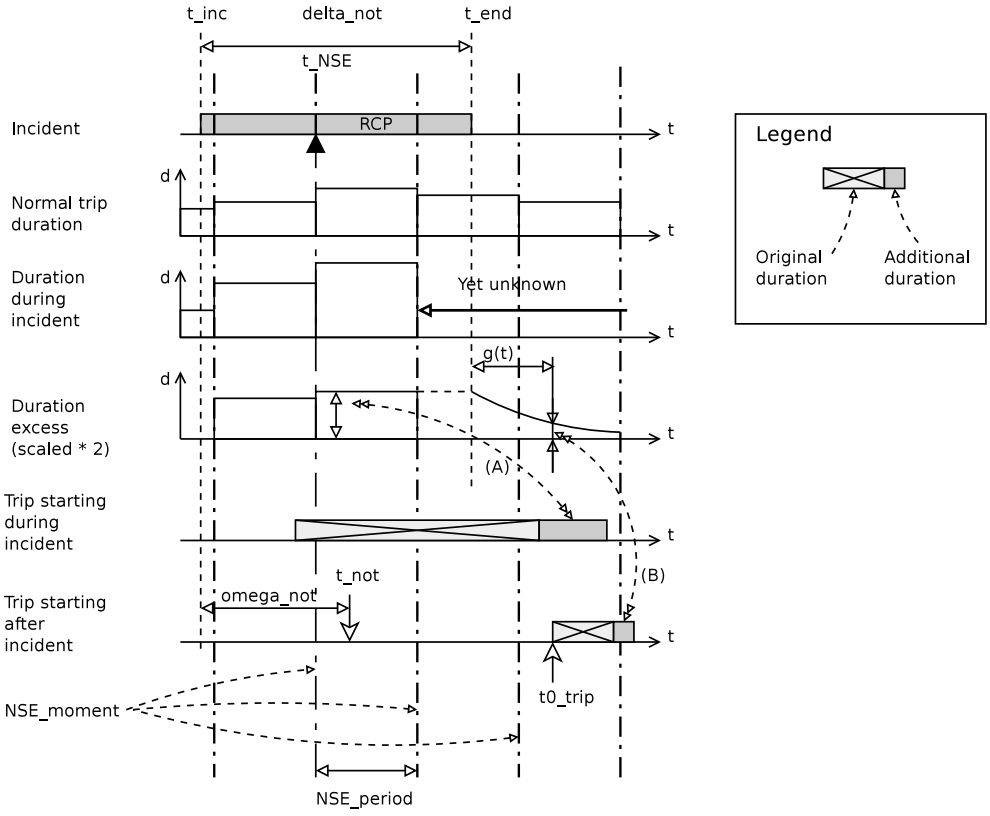


Figure 5.4: Incident occurrence, normal and adapted trip durations as perceived by a specific individual. NSE_moments correspond to vertical dash-dot lines. Schedule execution is simulated for the NSE_period starting at the black triangle. *Normal duration* is known for all NSE_periods, *duration during incident* and *duration excess* are known only up to the simulated NSE_period. The duration extension for an ongoing trip is the one for the NSE_period being simulated (A). For a later trip, exponential decay is applied (B).

5.6.1.4 Travel time adaptation for non-completed trips

Figure 5.4 shows the timing related phenomena for a single individual at a given *NSE_moment*. Let

t_0 be start of simulated time

t_{inc} be the incident start time

t_{not} be the time at which the individual gets notified about the incident occurrence

t_{NSE} be an NSE_moment

δ_{not} be the duration of the local reduced capacity period (RCP) associated with the incident estimated by the traveler who got notified by the TIS

ω_{not} be the delay between the incident start time and the notification of the individual

Δ_{NSE} be the length of an NSE_period

We assume that the person knows the incident start time as soon as (s)he gets informed: in reality, the incident occurrence time is not always contained in the traffic info conveyed but the estimate for the end of the local capacity reducing effects can have been mentioned. The diagram shows the travel time expansion as a consequence of the incident; note that for some OD-pairs travel time can decrease due to an incident because of some part of the transportation network no longer being fed with traffic flows as usual.
The diagram applies to the NSE_period starting at t_{NSE}. The effect of the incident during this NSE_period is calculated; beyond this period, travel time excess is unknown and hence has not been calculated using the TransCAD traffic assignment procedure. Point estimates for *excess travel time* are used. Those values are indicated in part *Duration excess* in the diagram; the graph shows the value for the additional travel time as a consequence of the incident; a scale factor of 2 has been applied to increase readability of the diagram.

5.6.1.5 Travel time estimation by notified individuals

People becoming notified of an incident need to determine the expected value for the duration of non-finished trips. Perception of (the effect of) incidents by individuals is influenced by (see also Figure 5.4)

1. the time lag ω_{not} between the incident occurrence and the individual becoming aware of it by traffic information

2. the duration δ_{not} for the period of capacity reduction (RCP) near the incident

location as expected by the individual. It is modeled by a gamma distributed stochastic $\delta_{not} \sim gamma(\alpha_{\delta_{not}}, 1)$ with $\alpha_{\delta_{not}}$ equals the duration of the incident (see section 5.5.8 item 3). The delay δ_{not} models the RCP duration expected by each individual aware of the incident and is based on the individual's personal conviction: as a consequence, a new value is sampled for each individual (who is aware of the incident by notification)[1]

3. the *re-normalization* function that specifies how the travel time approaches (decreasing or increasing) the normal value again: this is a decay function specifying how the (positive or negative) travel delay difference evolves back to zero after the RCP ends. Exponential decay is used. The effect is assumed to have attenuated to the relative reference level l_r after a reference time gap g_r.

$$e^{-\alpha \cdot g(t)} = l(t) \tag{5.4}$$

$$(g(t) = g_r) \Rightarrow (l(t) = l_r) \tag{5.5}$$

$$\alpha = \frac{-\ln(l_r)}{g_r} \tag{5.6}$$

Values for l_r and g_r are given as configuration parameters (*tisInfo_level* and *tisInfo_refGap* respectively in 5.11).

The individual gets informed before experiencing the incident effect. The excess trip duration is estimated by assuming that the trip will start at the planned moment in time. The excess travel time is determined for the moment of evaluation t_{NSE}. It is assumed that this travel time will still apply at the end of the *RCP* and that traffic re-normalization causes the excess duration to go to zero by exponential decay. The expected travel time is calculated as follows: let, for a given trip

t_{NSE}	be the moment at which evaluation is performed
t_{inc}	be the incident effect start time
t_{end}	be the incident's effect end time (end of the reduced capacity period RCP: $t_{end} = t_{inc} + dur(RCP) = t_{inc} + \delta_{not}$) as estimated by the individual. Note that $t_{end} < t_{not}$ is possible (i.e. the person gets informed after the local incident effect is expected (by this person) to have terminated).
t_{pts}	be the planned trip start moment; this can be an already revised version. Note that $t_{pts} > t_{NSE}$

[1] The current implementation uses $\delta_{not} = \alpha_{\delta_{not}}$.

$D_{norm,t_{NSE}}$	be the trip travel time for normal conditions valid at the moment of estimation t_{NSE} (available from impedance matrix)
$D_{norm,t_{pts}}$	be the trip travel time for normal conditions valid at t_{pts} (available from impedance matrix)
$D_{inc,t_{NSE}}$	be the trip travel time for incident conditions valid at t_{NSE} (available from impedance matrix that gives the actual traffic state during simulation)
$D_{inc,t_{pts}}$	be the trip travel time for incident conditions valid at t_{pts} (*not* available from any impedance matrix). This duration is to be estimated by the individual after having been notified about the incident start time and *RCP duration*
d_{RCP}	be the most recent known *excess* trip duration during the incident *RCP* period: $d_{RCP} = D_{inc,t_{NSE}} - D_{norm,t_{NSE}}$
$gamma(x, y)$	be a value sampled from a gamma distributed stochastic with parameters x and y. The excess travel time estimation is modeled by a stochastic because it is unknown and each actor uses a private estimate.
$g(t)$	be the time *gap* between the end of the incident induced *RCP* period and time $t > t_{end}$ for which one wants to determine the travel time

Then the trip travel time for a moment t in the re-normalization period is given by

$$d_{RCP} = D_{inc,t_{NSE}} - D_{norm,t_{NSE}} \tag{5.7}$$

$$D_{inc,t} = D_{norm,t} + gamma(d_{RCP}, 1) \cdot e^{-\alpha \cdot g(t)} \tag{5.8}$$

$$D_{inc,t} = D_{norm,t} + gamma((D_{inc,t_{NSE}} - D_{norm,t_{NSE}}), 1) \cdot \exp(\frac{\ln(l_r)}{g_r} \cdot g(t)) \tag{5.9}$$

5.6.1.6 Travel time estimation by experiencing individuals

The traveling person becomes aware of a new value for the delay at equidistant discrete times (NSE_moments) because only at those moments the network state is recalculated. The new trip duration estimate is calculated at t_{NSE}

$$t_{NSE} = t_0 + k \cdot \Delta_{NSE}, k \in \mathbb{N} \tag{5.10}$$

The individual believes that the remainder of the trip will be driven at *congested speed* because that is, at that moment, the best estimate for the duration to travel from origin to destination. Note that the same belief holds at the next NSE_moment and can result in revised remaining travel time. The *congested speed* is taken from the actual state impedance matrix for t_{NSE} (the last one calculated). The distance already

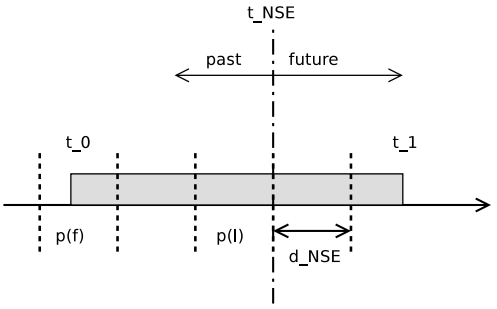


Figure 5.5: Travel time determination by *experiencing* individuals: symbols used. The gray block represents the travel period (t_0 is the departure time, t_1 is the arrival time).

driven between the trip start moment t_0 and the *NSE_moment* t_{NSE} is calculated by considering the *NSE_period*s identified by the indices f (first) and l (last) respectively (refer to Figure 5.5). *NSE_period* $p[f]$ contains t_0 and $p[l]$ ends at t_{NSE}. The length of the part of the interval $[t_0, t_{NSE}]$ overlapped by *NSE_period* $p[i]$ is denoted by $o(p[i], [t_0, t_{NSE}])$. Note that

$$i > f \Rightarrow o(p[i], [t_0, t_{NSE}]) = dur_{NSE} \tag{5.11}$$

The expected travel duration from origin to destination during period $p[i]$ is denoted by $d(i)$ and taken from the impedance matrix $Act[i]$ specifying the actual travel duration values. The distance from origin to destination is denoted by $s(O, D)$ and the average speed in period $p[i]$ is $\frac{s(O,D)}{d(i)}$. Then the distance driven up to time t_{NSE} is given by

$$\overline{d_{NSE}} = \sum_{i=f}^{i=l} \frac{s(O, D)}{d(i)} \cdot o(p[i], [t_0, t_{NSE}]) \tag{5.12}$$

The expected value for the remaining travel duration δ_{rem} is given by

$$\delta_{rem} = \frac{s(O, D) - \overline{d_{NSE}}}{d_{Act}(i)} \tag{5.13}$$

5.6.2 Actor Behavior

The first experiment makes use of a simple activity duration (de-)compressor to model rescheduling which means that the individual adapts activity start/end times in the agenda but does not modify it in any other way (by relocation, activity dropping

etc). The individual actor is assumed to maximize the agenda utility. Hence, there is a need to distribute lost or gained time over the non-completed activities. This section explains the requirements for utility functions in general and discusses schedule optimality. Concrete functions selected for the first experiments are presented. The method used to derive the required coefficients for the selected functions from the given FEATHERS schedules, is shown.

5.6.2.1 Utility Functions

1. Marginal utility (the amount of utility produced per time unit) is denoted by $v(d)$ where d is the activity duration. $v(d)$ assumed to be continuous and integrable. The corresponding utility is denoted by $u(d)$. Utility $u(d)$ is determined by integration of $v(d)$ and by requiring zero utility for zero duration.

2. A Maximal Activity Sequence WithOut Internal Constraints (MASWOIC) is a largest subset of consecutive activities in a schedule so that only the first (last) one starts (ends) at an externally stated time limit (e.g. shop closing time, time specified in public transportation timetable).

3. In a MASWOIC in an *optimal* schedule, the marginal utility is the same for each activity period. This can be seen as follows. Let t_{min} and t_{max} be the times delimiting the interval available for activity a_0 execution. Let t_0 denote the activity start time. Let $V : \mathbb{R}^+ \Rightarrow \mathbb{R}^+ : d \mapsto v(d)$ denote the marginal utility (utility generated per time unit) at $t = t_0 + d$, then the utility generated up to t is given by

$$u(t) = \int_{\max(t_0, t_{min})}^{\min(t, t_{max})} v(t)dt \tag{5.14}$$

With explicitly stated constraints, this is written as

$$t_{min} \leq t_0 \leq t_1 \leq t_{max} \tag{5.15}$$

$$d \geq 0 \Longrightarrow v(d) > 0 \tag{5.16}$$

$$u(t_0, t_1) = \int_{t_0}^{t_1} v(t - t_0)dt \tag{5.17}$$

Now consider two consecutive activities a_a (predecessor) and a_b (successor) with start/stop times respectively (t_0, t_1) and (t_1, t_2) and with marginal utility functions $v_a(d)$ and $v_b(d)$ respectively in a schedule that maximizes utility; then the

value for t_1 is determined by

$$\frac{\partial}{\partial t_1}\left(\int_{t_0}^{t_1} v_a(t-t_0)dt + \int_{t_1}^{t_2} v_b(t-t_1)dt\right) = 0 \tag{5.18}$$

hence

$$v_a(t_1-t_0) - v_b(t_2-t_1) = 0 \tag{5.19}$$

4. The first experiment assumes marginal utility $v(d)$ to monotonically decrease exponentially with activity duration d and to be independent of absolute time. It is positive everywhere, hence utility $u(d)$ is monotonically increasing. Subscript i identifies the activity.

$$v_i(d) = k_i \cdot e^{-\alpha_i \cdot d} \tag{5.20}$$

$$u_i(d) = (1 - e^{-\alpha_i \cdot d})\frac{k_i}{\alpha_i} \tag{5.21}$$

5.6.2.2 α-Value Determination from the Initial Schedule

1. Consider a time interval with given duration $D = \sum_{i=1}^{i=N} d_i$ that contains N episodes having duration $d_i = t_i - t_{i-1}$. The values t_0 and t_N are fixed. Then

$$\forall k \in [1, N-1] : \frac{\partial}{\partial d_k}[\sum_{i=1}^{i=N} u_i(d_i)] = 0 \tag{5.22}$$

$$\forall k \in [1, N-1] : \frac{\partial}{\partial t_k}[u_k(t_k - t_{k-1}) + u_{k+1}(t_{k+1} - t_k)] = 0 \tag{5.23}$$

which leads to $N-1$ equations from which $N-1$ of the α values can be determined.

2. One of the α_i in a MASWOIC shall be determined by other means. Per hypothesis, the activity of longest duration in each MASWOIC reaches a given relative utility saturation level f_U. In the experiments $f_U = 0.95$ was used. Let a_L be the activity of longest duration. Then

$$1 - e^{-\alpha_L \cdot d_L} = f_U \Rightarrow \alpha_L = \frac{-\ln(1-f_U)}{d_L} \tag{5.24}$$

3. The equality of all marginal utility values in the MASWOIC then leads to expressions giving α_i as a function of α_L, known durations and the ratio between k values (which reduces to a ratio between d values)

$$k_i \cdot e^{-\alpha_i \cdot d_i} = k_L \cdot e^{-\alpha_L \cdot d_L} \Rightarrow \alpha_i = \frac{\alpha_L \cdot d_L - \ln(\frac{k_L}{k_i})}{d_i} \tag{5.25}$$

5.6.2.3 K-Value Determination from the Initial Schedule

1. The k value for activity i is calculated using the given relationship

$$k_i = \frac{C}{d_i^m} \tag{5.26}$$

where C and m are constants and d_i is the activity duration (expressed in minutes). The k values depend on the specific *activity* and not on the *activity type*.

2. Calculation of new activity durations when the total amount of available time changed, is based on the equality of the marginal utility values for the activities involved. Consider two consecutive activities in a MASWOIC a_0 and a_1 for which the new total duration is given by $D = d_0 + d_1 + \delta_T$ where δ_T is the duration of the trip that separates the activities.

$$k_0 \cdot e^{-\alpha_0 \cdot d_0} = k_1 \cdot e^{-\alpha_1 \cdot (D - d_0 - \delta_T)} \Rightarrow d_0 = \frac{\alpha_1 \cdot (D - \delta_T) \ln(\frac{k_1}{k_0})}{\alpha_0 + \alpha_1} \tag{5.27}$$

After substituting the k values using equation ((5.26)) we find

$$d_0 = \frac{\alpha_1 \cdot (D - \delta_T) - m \cdot \ln(\frac{d_0}{d_1})}{\alpha_0 + \alpha_1} = \frac{\alpha_1 \cdot (D - \delta_T) - m \cdot \ln(\frac{d_0}{D - \delta_T - d_0})}{\alpha_0 + \alpha_1} \tag{5.28}$$

Note that the constant C has no effect on the d_i values since C does not occur in equation ((5.28)). Therefore, C can be chosen freely. The C value is determined for each MASWOIC by stating that the total utility of the optimal original schedule equals 1 (which comes down to actually make use of *relative utility* values) within a MASWOIC. In the current project, utility values are compared only when they belong to the same MASWOIC: hence, arbitrarily selecting the value for C does not induce an additional assumption or constraint.

3. In the first experiments $m = 1$ is used.

5.6.2.4 Schedule (De)Compression

The case of a single MASWOIC in a 24-hour schedule is considered to explain the schedule (de-)compression concept. At specific moments during schedule execution, new travel duration estimates for the current and/or planned trips come available. The amount of time to be spent to the (partial) activities and trips that have not yet finished at time t_0, will change due to modified travel duration predicted from the network state. As a result, the total schedule duration no longer equals 24 hours and

(de)compression of the non-finished part of the schedule is required. This is done by solving the set of equations derived from (5.28). Note that in the derivation of (5.28), the *travel* duration δ_T is assumed not to depend on the trip start time. This of course is an approximation. In general, trip duration is a yet unknown non-linear function of the trip start-time. Activity start-times are calculated by an iterative relaxation solver.

The *compressible schedule part* (CSP) (see Figure 5.6) is initialized to contain each activity that has not yet completed (at most one activity can be partially completed). Let t_{base} be the start of the first activity in CSP. If $t_{base} < t_0$

1. the optimal activity duration for the CSP is calculated by (de-) compression and relaxation of all activities in $[t_{base}, t_{end}]$ where t_{end} is the end of the schedule. After (de)compression $t_{end} = 1440[min]$.

2. if the end of the first activity in the CSP comes before t_0 then, in the optimal schedule, the running activity should have been stopped before the time of notification (t_0). It is stopped immediately and removed from the CSP. The CSP now starts at t_0: the marginal utility values for the future differ from those for the past because the future activities are subject to more time pressure than the ones in the past.

(De)Compression is done by a relaxation algorithm based solving the set of equations derived from (5.28). This relaxation can be proven analytically to converge monotonically when a monotonically decreasing marginal utility is used.

5.6.3 Initial Schedule Adaptation

For reasons mentioned in 5.5.5, the schedules predicted by FEATHERS are used to calculate an impedance matrix for each *NSE_moment*. The schedules and impedance matrices need to be made mutually consistent before the start of the schedule execution simulation. Following symbols are used:

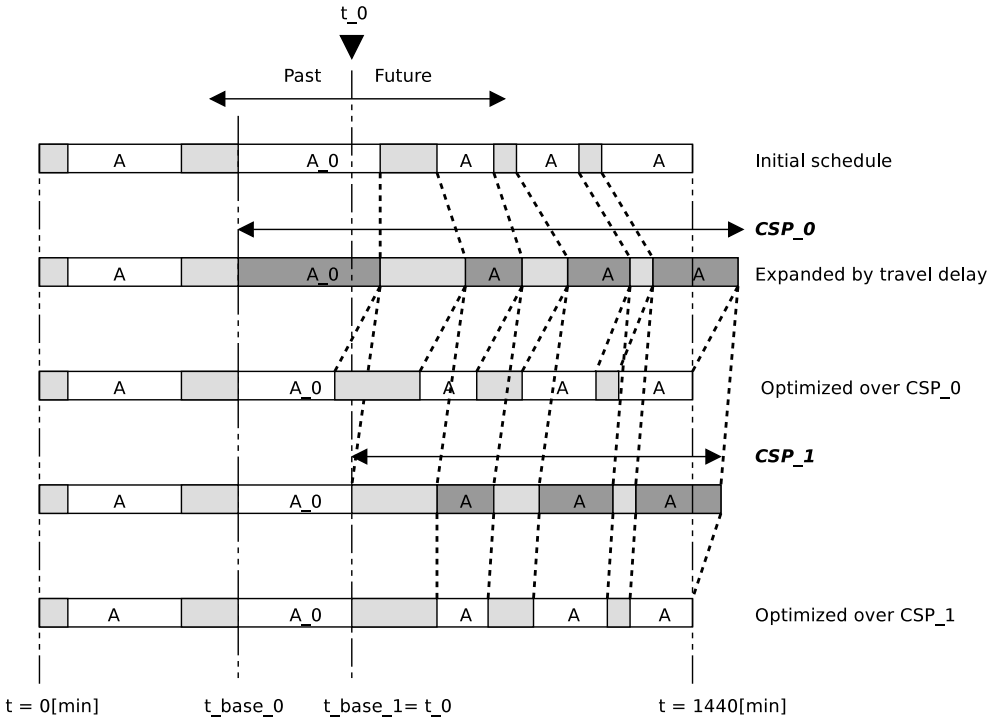


Figure 5.6: Time runs from left to right. The individual is notified at time t_0. Gray blocks represent travel periods. Dark gray blocks represent activities in a Compressible Schedule Part (CSP). The CSP_0 contains the running activity A_0. Optimization over CSP_0 shows that A_0 already should have terminated at t_0 so it is terminated immediately and the compressible schedule part is reduced to CSP_1. Optimization over CSP_1 results in the final schedule. $t_{base,0}$ and $t_{base,1}$ are start times for the respective schedule parts CSP_0 and CSP_1.

$\mathcal{P}$	denotes the set of all *NSE_periods*
$pathCost(netw, odFlow)$	determines the minimal travel duration for all OD-pairs after loading *netw* with the travel demand specified by the *odFlow* matrix.
$flow(episodes, p)$	determines from a set of episodes a flow matrix giving the number of trips for each OD-pair going on during *NSE_period* p
$epi(S)$	delivers the set of episodes for a set of schedules S
$update(S, IM)$	modifies the travel times in each schedule $s \in S$ according to the values specified in impedance matrix IM

Assignments (5.29) to (5.31) all are equivalent and show how an impedance matrix $IM_{i+1}[p]$ for period $p \in \mathcal{P}$ is derived from its predecessor $IM_i[p]$.

$$IM_{i+1}[p] \leftarrow pathCost(netw, odFlow_{i+1}[p])) \tag{5.29}$$

$$\leftarrow pathCost(netw, flow(epi(S_{i+1}), p)) \tag{5.30}$$

$$\leftarrow pathCost(netw, flow(epi(update(S_i, IM_i[p])), p)) \tag{5.31}$$

Then consistency between schedules S_{i+1} and impedance matrices IM_{i+1} is defined by

$$consistent(IM_{i+1}, S_{i+1}) \Leftrightarrow (\forall p \in \mathcal{P} : \frac{2 \cdot \|IM_{i+1}[p] - IM_i[p]\|_2}{\|IM_{i+1}[p]\|_2 + \|IM_i[p]\|_2} < \overline{M}) \tag{5.32}$$

where $\overline{M}$ is allowed relative difference upper bound.

5.6.3.1 Travel Time Updating in Schedules

The travel time for each trip is derived as a weighted average of the travel times for the OD pair in question during the *NSE_periods* overlapped by the trip period (see section 5.6.1.6). The value found in the schedule does not necessarily equal a value found in any of the impedance matrices.

5.6.3.2 Time Reference Problem

Travel time adaptation by means of schedule (de)compression makes use of a particular point in time as a reference. When optimizing *during schedule execution*, the reference point is the time at which the individual determines a new schedule. The reference time is the boundary between the past (immutable) and the future (mutable). When trip travel times initially are adapted to the impedance matrices, nothing of the schedule already has been executed and the reference point is the conventional

start of the day. FEATHERS predicted schedules all start at the arbitrarily chosen 03:00h which has no physical meaning and hence shall not have any effect on the results.

The value of the reference time determines the schedule because it fixes one specific arbitrarily chosen moment in absolute time: if the conventional start of the day is chosen as a reference, the rescheduling period lasts for 24 hours but the chosen conventional start is immutable. This is unrealistic since any other schedule that is shifted over some time interval, is an optimal schedule too because the optimality depends on the relative lengths of activity durations (at least if no exogenous time limits apply). Hence, while adapting schedules to the impedance matrices, no such absolute reference point determined a priori, does exist. As a result, the reference point is a Degree Of Freedom (DOF).

The value for the DOF is determined by shifting the schedule in time so that some criterion about the difference between the original and the adapted schedule, is optimized. Since the FEATHERS output has been validated using traffic counts, the criterion is to approximate the total car flow as a function of time, as good as possible.

Multiple criteria can be conceived (all minimizing the sum of the squared differences between corresponding quantities selected from the original predicted schedule and the adapted schedule respectively). WIDRS uses weighted trip period begin/end times using parameters specified in the configuration as follows:

1. deviations for the *trip end* time for *non-home* destinations and the *trip begin* time for *home* destinations are considered

2. the weights are derived from the activity durations (shorter activities have a higher weight). The weight function is : $w(a) = k/(k + a.dur())$ where k is a constant defined in the configuration (see $k\_schedAdaptDOF$ in 5.11) and $a.dur()$ denotes the activity duration. Large k values result in nearly identical weights for all activities.

5.6.4 Network Load Calculation

The TransCAD tool used for traffic assignment requires hourly trip amounts as input. WIDRS operates on a *NSE_period* basis; an integral number k of *NSE_period*s are contained in each hour. A one-hour period $p_{1h}(t^i_{NSE})$, consisting of k consecutive *NSE_period*s, is associated with each *NSE_moment* t^i_{NSE} in order to calculate the traffic flows for t^i_{NSE}. Let M_{NSE} denote the set of *NSE_moment*s and $t_0 \in M_{NSE}$

denote the start of the one-hour associated period and d_{NSE} denote the duration of an *NSE*_period. Then $t_0 \in M_{NSE}$ is determined so that $t_0 + \frac{k}{2} \cdot d_{NSE} - t^i_{NSE}$ is non-negative and minimal. Note that in order to calculate the traffic flows for the $k-1$ one-hour periods that contain the boundary between consecutive days, we assume that history is periodic.

For each *NSE_moment* t^i_{NSE} the set of trips whose execution period overlaps with $p_{1h}(t^i_{NSE})$ is determined. The set is used to calculate the traffic flows for t^i_{NSE}. The flows are used to calculate a SUE traffic assignment by means of TransCAD. Travel times between TAZ then are calculated for the loaded network (skim). For $k = 4$ this results in a set of 96 impedance matrices for the traffic situation considered (*normal* or *incident*).

5.7 Implementation - First Results

An OSGi framework has been used because it allows for clean structuring of services specified by their interfaces and easy runtime management that allows to activate different implementations for such services. This allows to achieve the main objective of building a flexible framework to evaluate rescheduling strategies.

Configuration parameters have been summarized in 5.11.

5.7.1 Case Study

The study area covers Flanders (Belgium). It is modeled by 2386 traffic analysis zones (TAZ) with an average area of about 5$[km^2]$. TAZ are bundled into 319 municipalities. The population consists of 5.8 million individuals.

- Schedules for half of the population (2.9 million individuals, 9 million activities) are processed. The resulting values OD-flow matrices then are doubled to get the travel demand for the full population. This technique is used on 4GB machines due to memory requirements. The code is sufficiently efficient so that processing time is not a bottle-neck for problems of the dimension of the case study.

- Two simulations have been run for a large scale incident during the morning and evening peaks respectively. The reduced capacity network used in both cases is shown in Figure 5.7. Capacity for all marked links was reduced to 50% of the normal one. The respective incident periods are [07:30h,09:30h] and [16:00h,19:00h]. Due to lack of space, morning peak incident results only are presented.

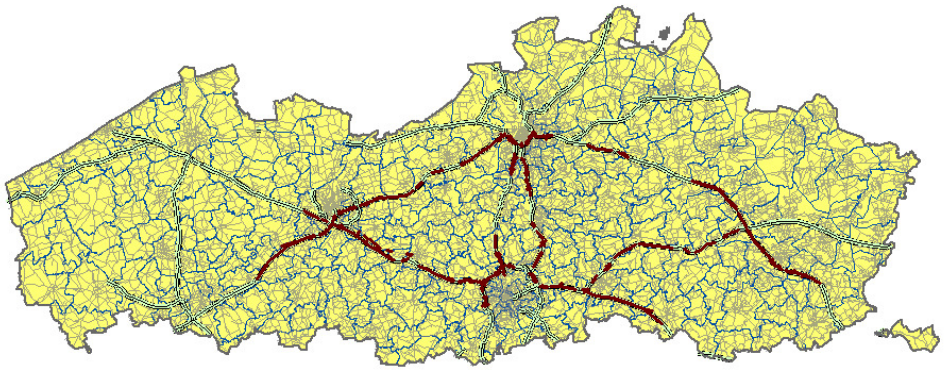


Figure 5.7: Reduced capacity network: road segments drawn using thick lines are parts of highways. Their capacity has been reduced to 50% of the normal value to simulate unforeseen events affecting network level of service.

5.7.2 Travel and Activity Duration Distributions

Figure 5.8 shows the densities for the total amount of travel duration in a schedule. Values larger than 6[hours] have not been shown in order to make the graph sufficiently readable. A very small amount of people experience larger delays.

Figure 5.9 shows the probability density and cumulative distribution for the total difference in daily travel time for each schedule (person), caused by the road incident. Figure 5.10 shows two frequency distributions for the schedules: the first one classifies the schedules according to the type of the activity for which the duration compression was maximal. The second one classifies the schedules based on the type of the activity that suffered from maximal time shift.

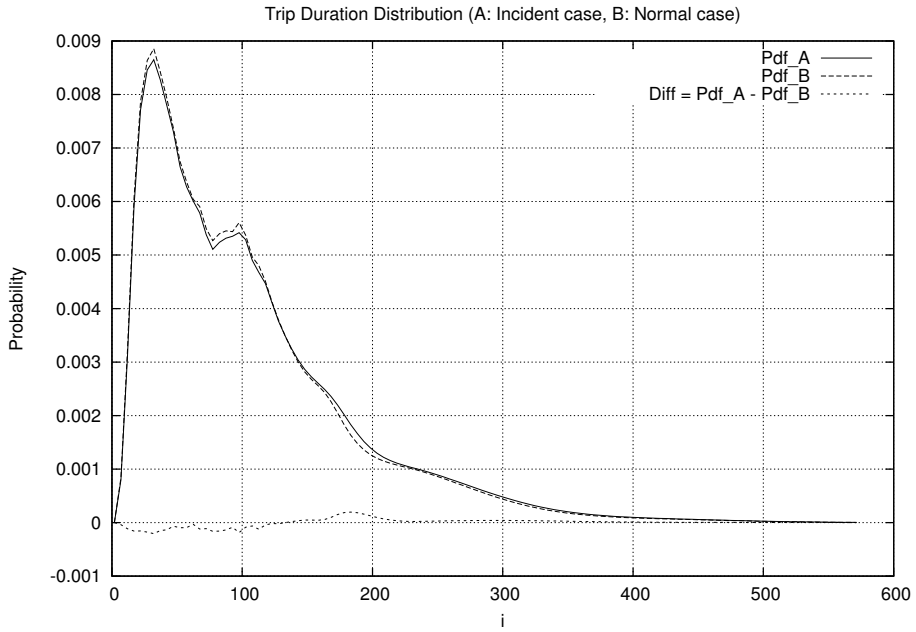


Figure 5.8: Probability distributions for the total travel duration in the affected schedules for the incident case (A) and for the normal case (B). The curve showing negative values represents the difference.

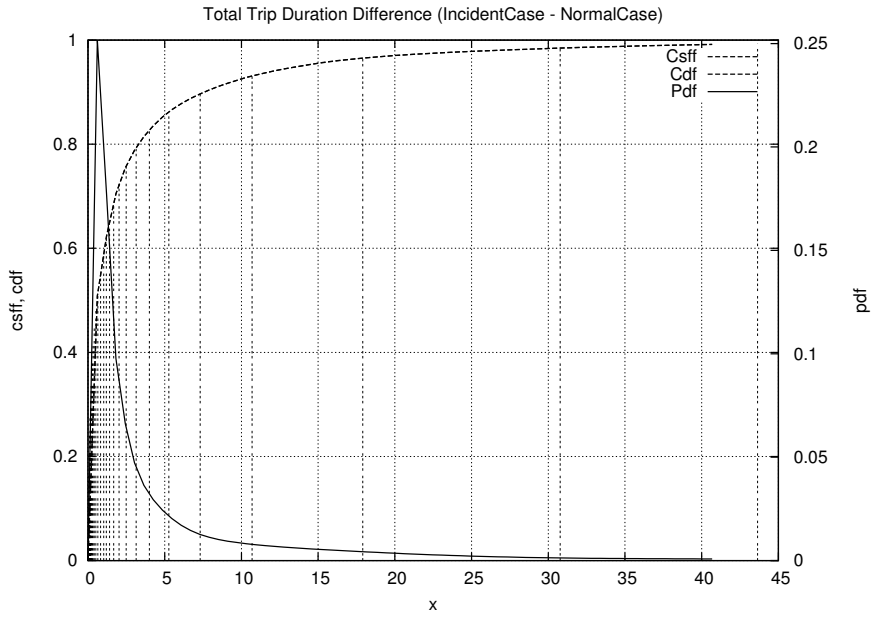


Figure 5.9: Probability density and distribution for the difference in total daily travel time per person (caused by the incident).

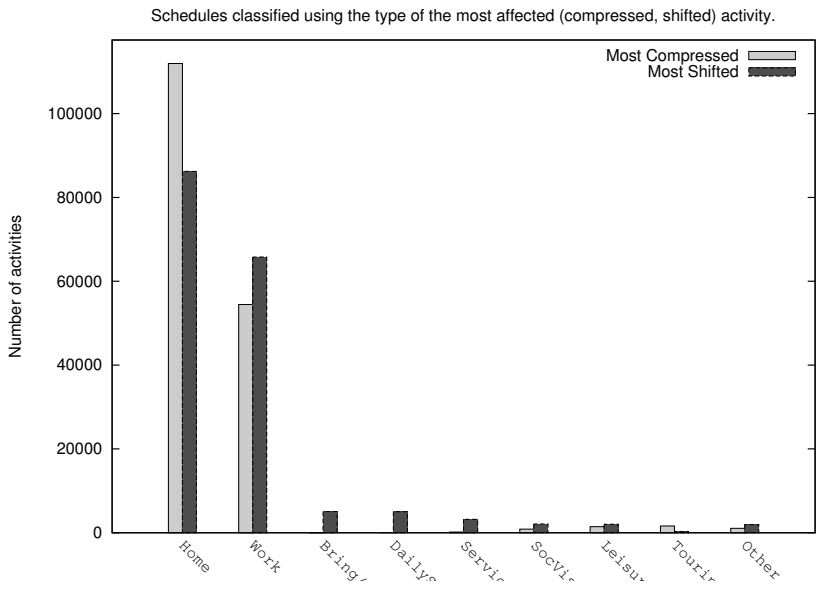


Figure 5.10: Classification of schedules according to the type of the activity that was most affected (either compressed of shifted in time).

5.7.3 Performance

1. WIDRS has been run on two machines having following specifications:

	M1	M2
Processor Type	Intel Xeon X5670	i5-2410M
Num. processors	2	1
Cores/processor	6	2
Threads/processor	12	4
Clock frequency	2.95 GHz	2.3 GHz
Memory	48 GB	4 GB
TransCAD	6.0	4.7

2. Characteristic values for a run:

Item	Value
Population size (= number of schedules)	2395513
Number of affected schedules (schedules for which at least one trip duration changed with more than one minute)	171632
Number of episodes	9139002

3. Performance characteristics have been summarized by the upper bounds mentioned in following table:

Item	Duration on M1	Duration on M2
WIDRS STEP 2	16[hours]	26[hours]
WIDRS STEP 3 (similar to STEP 2)		
MAX duration of schedules adaptation for complete population in STEP 2	49.4[sec]	89.1[sec]
MAX duration of schedules adaptation for complete population for one *NSE_period*	5.8[sec]	34.2[sec]

4. The relaxation algorithm to find new optimal schedules when timing constraints changed, handles 21000 schedules per second.

5. Since individuals act independently while rescheduling, parts of the WIDRS code have been written for multi-threading. TransCAD6.0 code also uses all available threads to execute the MSA algorithm. TransCAD4.7 uses at most

a single thread. It turns out that both TransCAD6.0 and WIDRS code are efficient but that most time is lost exchanging data between both tools because that is done via ASCII files.

6. STEP 2 never requires more than 4 passes when a tolerance of 0.01 is used. About 20% of the time is spent executing WIDRS Java code. The remainder of the time is spent in data transfer and TransCAD runs. An alternative solution to calculate the traffic assignment is to be found in order to make the framework more efficient.

5.8 Conclusions

A framework to investigate rescheduling daily activities with feedback from traffic network loading in a large area has been built by combining microscopic schedule execution simulation with macroscopic time dependent traffic network performance modeling. The microscopic component covers large amounts of actors re-optimizing their daily agenda making use of network information via perception filtering as time evolves. Both the framework and a rescheduler using monotonically decreasing marginal utility have been evaluated. The framework proved to be able to evaluate agenda adaptation by the complete Flemish population taking traffic network load feedback into account, within a feasible runtime. Initial consistency between the schedules and the travel time matrices is crucial because the unbalance shall be much smaller than the effect of schedule adaptation decisions. The inconsistency results from the difference in time resolution used in the schedule predictor on one hand and the schedule adapter/executor on the other hand. Resolving this inconsistency proved to constitute a non-trivial problem. The framework now is ready for evaluation of alternative (marginal) utility functions, traffic information conveying models and perception filters.

5.9 Future Work - Required Extensions

1. In the short term, the framework will first be extended with intermediate deadlines induced by bring/get activities. Utility optimization then applies to periods of the day delimited by those deadlines. In a second step, the trip estimated duration will be made depending on the trip start time. The travel duration available in impedance matrices, is a piece-wise constant function of start time. It will be approximated by a spline which is assumed to be sufficiently accurate

and is continuous and differentiable which is important for the optimization algorithm. Unfortunately, the relaxation process no longer will converge monotonically.

2. Parameters used in (marginal) utility functions shall be estimated from survey data. A daily schedule can have intermediate hard constraints for specific activities (e.g. bringing someone to a railway station to catch a train, picking up small children at school etc). Such constraints introduce periods with different time pressure values. First, such intermediate constraints shall be introduced in the model and secondly, they will affect the values for the parameters used in the utility functions. The model extension is planned and intermediate deadlines shall be derived from the available recorded diaries and time use survey responses.

3. Bell-shaped marginal utility functions (leading to S-shaped utility functions) will be introduced. Joh (2004) has shown that they provide a more realistic model of reality.

4. More elaborated models for the traffic information conveying model (broadcast, publish) need to be incorporated; the framework now is ready to do so. The sensitivity of the simulator to the notification model used, is to be investigated.

5. The *incident effect duration estimation* can be made dependent on the drivers history (experience) which in turn can be assumed to grow with age.

6. Activity dropping and insertion, activity re-sequencing and activity relocation will lead to challenging combinatorial optimization problems. Cooperation between individuals will add another magnitude of complexity as is suggested by preliminary investigations in Knapen et al. (2012a).

7. Currently the framework is also used to simulate electric vehicle battery charging when energy rates are time dependent due to expected availability of wind and solar power.

5.10 Appendix A: WIDRS main algorithm details

The algorithm shows an overview of the main steps in the WIDRS software operation. It is equivalent with the flowchart in Figure 5.2. Symbols used in the algorithm have been defined in the table below.

$FM[p]$	: OD flow matrix for NSE period p
IM_F	: OD impedance (travel duration) matrix used by FEATHERS
$IM_{NSE}[p]$	: OD impedance (travel duration) matrix used for NSE period p
N_{NSE}	: Number of NSE periods in day
$NW_{incident}$	: Network to evaluate traffic under incident conditions
NW_{normal}	: Network to evaluate traffic under normal conditions
scheds	: Set of schedules, one for each individual (synPop member)
synPop	: Synthetic population

- Lines 1-6 specify the function to calculate an impedance matrix for *NSE_period* p from a set of schedules for which the trips are applied to the given network.

- STEP1 (lines 8-11) calculates the initial impedance matrix for each *NSE_period* under normal network conditions. FEATHERS is used to predict a schedule (daily agenda) for each individual in the synthetic population. WIDRS assumes that the generated schedule for each individual is the optimal one.

- In STEP2 (lines 13-24) the schedules are used to determine origin-destination traffic flows (OD-flows) between TAZ. The OD-flows for *car mode* are applied to the road network that is assumed to operate under *normal conditions*: this *traffic assignment* results in expected car *flow* and travel *duration* values for each link in the road network. From those link results, new travel times between TAZ are calculated and summarized in an OD impedance (travel duration) matrix. Since on one hand, OD impedance matrices are used to predict schedules and on the other hand schedules produce OD impedance matrices, it is easily seen that schedules and impedances need to be mutually consistent. This consistency is not a trivial concept since its definition depends on the purpose for which schedules and impedances are calculated; this is explained in section 5.6.3. STEP2 makes the impedance matrices consistent with the list of schedules. The inner loop in lines 16-23 makes the impedance matrices consistent with the schedules. The limit value *maxRelDiffAllowed* has been specified in the configuration (see 5.11).

- STEP3 (lines 26-32) starts from mutually consistent schedules and impedance matrices for the *normal* case. The network characteristics are assumed to suddenly change due to an incident for a given *RCP* (Reduced Capacity Period) representing *incident conditions*. STEP3 is the actual schedule execution simulator. Line 27 determines which network to use. All networks share the same

Algorithm 5.10.1 WIDRS algorithm overview.

function IMPEDMTX($scheds, p, network$)
  $FM[p] \leftarrow flowFromSchedules(scheds, p)$
  $netwState \leftarrow traffic Assignment(FM[p], network)$
  $IM \leftarrow impedMatrix(netwState)$
  return IM
end function
▷ STEP 1: Initial impedance matrices for normal network
$scheds \leftarrow FEATHERS(landUse, synPop, IM_F, dTrees, NW_{normal})$
for all $i \in N_{NSE}$ **do**
  $IM_{NSE}[i] \leftarrow$ IMPEDMTX($scheds, i, NW_{normal}$)
end for
▷ STEP 2: Reference impedance matrices for normal network
repeat
  $largestRelDiff \leftarrow 0$
  $scheds \leftarrow makeSchedsConsistWithImped(scheds, IM_{NSE})$
  for all $p \in N_{NSE}$ **do**
    $IM^{prev}_{NSE} \leftarrow IM_{NSE}[p]$
    $IM_{NSE}[p] \leftarrow$ IMPEDMTX($scheds, p, NW_{normal}$)
    $n_0 \leftarrow \|IM^{prev}_{NSE}[p]\|_2$
    $n_1 \leftarrow \|IM_{NSE}[p]\|_2$
    $n_d \leftarrow \|IM_{NSE}[p] - IM^{prev}_{NSE}[p]\|_2$
    $largestRelDiff \leftarrow \max(largestRelDiff, \frac{n_d}{(n_0+n_1)/2})$
  end for
until $largestRelDiff \leq maxRelDiffAllowed$
▷ STEP 3: Schedule execution
for all $p \in N_{NSE}$ **do**
  $actualNetwork \leftarrow networkSelector(nsePeriod(p))$
  $IM_{NSE}[p] \leftarrow$ IMPEDMTX($scheds, p, NW_act$)
  for all $indiv \in synPop$ **do**
    $indiv.reschedule(IM_{NSE}[p])$
  end for
end for

topology but differ in link capacity values that model the local time dependent effect of the incident (see section 5.5.7).

5.11 Appendix B: Configuration

The table below summarizes parameter values that apply to the results presented in the paper.

Item	Value	Unit	Description
deliveryDelay	30	min	Notification delay
f_U	0.95	-	Utility saturation level for the activity having the longest duration in a MASWOIC
k_schedAdaptDOF	1	-	Factor determining weight used to derive the value for the schedule adaptation time-shift DOF
maxRelDiffAllowed	0.01	-	Maximum relative error between the impedance matrix used to determine travel duration in schedules and the one resulting by applying the travel demand of same the schedules to the network
tisInfo_level	0.05	-	Reference level for travel time re-normalization
tisInfo_refGap	60	min	Reference gap for travel time re-normalization

In order to determine whether or not an activity is *affected* by re-scheduling, its start time is compared with the activity type specific threshold given in the following table. Please note that those values are used only to produce statistics.

Activity Type	Start time shift threshold [min]
Home	30
Work	10
Bring/Get	2
DailyShopping	10
Services	3
SocialVisit	15
Leisure	2
Touring	15
Other	10

5.12 Critical Reflection

5.12.1 Discussion

1. The calculation of α and k values differs between the papers Knapen et al. (2012d) and Knapen et al. (2013c). In the former the k values were assumed to be activity type specific constants. In the latter, $k_i = \frac{C}{d_i}$ is taken (i.e. the k value for the i-th activity is proportional to the inverse of the duration which renders the k values individual specific (which is assumed to be more realistic). In both cases, all but one of the α values can be determined from the optimality criterion and needs to be chosen as a reference value. In the former paper, the first activity of the schedule was used as a reference, in the latter paper the activity having the longest duration is chosen. Since for the reference activity a *utility saturation level* is specified, the latter option is more realistic.

2. The C constant in equation (5.26) is not required when only a single Maximal Activity Sequence WithOut Internal Constraints (MASWOIC) is considered. This is explained in section 5.6.2.3 item 2. In the cases described in both papers, the value for C is chosen arbitrarily. However, in a case consisting of several MASWOICs, activity re-sequencing can move activities between MASWOICs. In order to compare the utility values between different MASWOICs, a C value for each of them is required. This corresponds to the introduction of a *time pressure* concept.

3. The *time reference problem* solved in section 5.6.3.2 disappears when at least one activity or trip start is assigned a fixed time-of-day (e.g. by fixing *bring-get* activities in time assuming that they are the result of a negotiated appointment). As soon as more than one moment gets fixed in time, multiple MASWOICs occur.

5.12.2 Rescheduling *with* Travel Time Recalculation

1. The current version of WIDRS considers a single *user class* with respect to TA. As a consequence, in the network loading stage, no information about individual preferences is available. In order to make WIDRS applicable to problems like *congestion charging*, it shall be extended to transfer user class information to the TA software.

   Aggregated Traffic Assignment (TA) causes a lot of information to be lost. On the other hand, the use of TAZ level travel time OD matrices to model com-

mon knowledge, makes sense. WIDRS is suited to (i) evaluate computational performance and to (ii) evaluate rescheduling models and build applications for cases where the rescheduling component does not need any information about the chosen route and the routing can be performed without any information about the individual.

2. Traffic is not modeled completely correctly since the TA tool assumes that the route for each trip can be freely chosen (the trip is assumed to *start* in the 1-hour period for which TA is computed) which is not true for all trips (some are already going on). On one hand, route information or link loads are not fed back to the rescheduler; on the other hand, the effect via travel time feedback is limited. This technical phenomenon has no effect on *notified* individuals (i.e. the ones who become aware of adjusted travel times) because they are not traveling. The *experiencing* individuals (i.e. the ones suffering from unexpected congestion) recalculate the travel time for the unfinished part of the trip; hence, only that part suffers from the phenomenon.
3. WIDRS runtimes are long (almost 24[h] to simulate a single day). Long runtimes stem from data transfer between WIDRS code and TA tool `TransCAD` (see items 5 and 6 in section 5.7.3). The integration of both tools needs to be enhanced.

5.12.3 Rescheduling *without* Travel Time Recalculation

In some cases, the effect of rescheduling has a minor effect on the global interzonal travel times which constitute the only feedback from Traffic Assignment (TA). In such cases, travel duration matrices can be pre-computed (see section 5.12.5.3 item 3 for details). When using the pre-computed trip duration, evaluation of the rescheduling models still is done with time dependent travel time. This allows for efficient evaluation of activity re-sequencing, adaptation of schedules with internal deadlines (multiple MASWOIC), alternative travel mode selection, etc. Such rescheduling models can be evaluated without travel time recalculation and the travel demand generated by the final schedules can be assigned to the network to verify the hypothesis of minor effect on travel time was correct. If the hypothesis holds, the rescheduling model evaluation is correct.

5.12.4 Provided Facilities

WIDRS keeps track of the schedules for the complete population and can adapt them very fast. It provides facilities to analyze and report differences between the original

and adapted version of a schedule. It can present results as distributions for several schedule properties e.g. time spent in activities, difference in travel time, etc. Hence it provides a test-bed to analyze the effects of schedule adaptation at the population (segment) level. Since WIDRS can produce frequency distributions as well as provide every detail for outlier analysis, it is an efficient tool to evaluate rescheduling models.

5.12.5 Possible Extensions

Topics related to (i) the individual behavioral model and (ii) the framework are handled separately. The original aim was to provide a framework to evaluate schedule adaptation models before integrating them into research projects. This idea remains valid although WIDRS can be used as a stand-alone tool to answer research questions as indicated in section 5.12.3.

5.12.5.1 Behavioral Model Extensions

The behavioral model definitely needs to be extended. Each extension needs two evaluation steps: (i) assessment of the extended *rescheduling model* quality using the WIDRS framework and (ii) assessment of feasibility to use the extended *WIDRS as a stand-alone too*l to solve a practical problem. If stand-alone deployment turns out not to be useful, the evaluated rescheduling model needs to be integrated in another framework in order to solve specific concrete problems.

Suggestions for extensions are:

1. Implementation of internal deadlines: This includes reporting of missed deadlines. Appropriate *reaction* to missed deadlines in general is not possible due to lack of information about the missed deadline consequences and the possible corrective actions.

2. Although the utility only depends on activity duration and not on the absolute time-of-day, activity re-sequencing within a given MASWOIC makes sense because travel time depends on the time-of-day. Activity re-sequencing might affect required travel time and hence total utility for the MASWOIC, even before implementing internal deadlines.

   Note however that this can break the hypothesis that the original schedule was optimal. If re-sequencing activities increases the utility of the schedule beyond its original value, one shall decide that the method to calculate utility differs among the schedule generator and the schedule adapter. The ability to increase the utility of the original schedule implies that the schedule generator, while de-

ciding optimality, used information that cannot be retrieved from the predicted schedule.

3. Extension of the utility concept: Utility shall be made time-of-day dependent and not only depend on activity duration. A concept similar to the effectiveness function defined for C-TAP can be used (Maerki et al. (2014)). Utility can turn out to be a complicated function of (i) activity type, (ii) activity duration, (iii) time-of-day i.e. the amount of overlap between the activity period and given globally defined (as opposed to *emerging from coordination*) other periods (iv) length of the period since the end of the last execution of a similar activity (defined by a *needs based model*)

4. Introduction of the VOT concept: In some applications the concept of Value Of Time (VOT) needs to be integrated. Examples are: evaluation of congestion charging or the currently considered proposal for price reduction offered to clients shopping in calm periods of the day. Monetary value for utility needs to be defined (this is equivalent to the VOT problem) for each activity.

5. Support for uncertainty: Some applications require the introduction of facilities for which the availability is not deterministic. This is the case for Electric Vehicle (EV) related applications where some tours cannot be completed without intermediate charging. The minimum required charging period at each location can be calculated but the available amount of electric energy during a given period at a specific location is stochastic, even if the maximum power supply is deterministic (this is because the demand is stochastic). Handling such cases requires the development of new behavior models able to cope with uncertainty.

6. Mode change to evaluate multi-modal tours (carpooling excluded because that requires coordination among actors).

5.12.5.2 Use of the Behavioral Model in other Frameworks

The individual behavioral model extensions discussed in section 5.12.5.1 can be used in WIDRS as well as in other frameworks. For example, the behavioral model can be integrated in the carpooling Agent-Based Model (AgnBM) discussed in previous chapters to enhance the negotiation components. This requires the integration of a predictor module for time dependent travel time in the carpooling model but that is technically less complex than introducing a generic cooperation concept (and the inevitable computational complexity) in WIDRS.

A similar reasoning holds for EV related applications which also require some form of coordination.

The individual behavioral model can be integrated in MATSim as a *replanning module*. Note that the concept of *internal deadlines* needs to be integrated in the WIDRS behavioral model first since MATSim makes use of *authority constraints* (shop opening times).

5.12.5.3 Framework Extensions

Following framework upgrades are useful when the use of aggregated TA is continued:

1. Addition of a facility to transfer *user class* information from WIDRS to the TA software tool.

2. Finding a technique to speed up the data transfer between WIDRS code and TA code, possibly by choosing an other TA tool to solve the data transfer (see items 5 and 6 in section 5.7.3).

3. Using approximated travel times: in some cases the rescheduling decisions may be assumed to have little or no effect on aggregated travel times (which is the only information fed back from the TA tool). In such cases, travel times can be computed in advance. Currently 96 travel time matrices (one for each 15[min] period) each containing about 6 million cells (for nearly 2400 TAZ) are required. By using an appropriate family of polynomials, travel duration can be modeled as a continuous function of trip start time. It is expected that the set of coefficients to represent 6 million such approximations can be kept in memory in current servers. This allows for very fast travel time evaluation that replaces the current look-up in travel time matrices which requires repeated data loading from disk storage. However, the use is limited to the cases described in section 5.12.3.

Extensions involving actor cooperation are discouraged because they introduce computational complexity that is very hard to cope with in a general way. A better solution is to handle cooperation and coordination in frameworks that provide heuristics tuned to the specific problem.

Part II

Extracting Route Structure Information From GPS Traces

Chapter 6

Introduction to Route Decomposition

6.1 Research Objective and Relevance

The research started from the hypothesis that *for their utilitarian trips* (trips having the purpose to perform an activity at a particular location) *people tend to compose their routes from a small number of least cost components.*

If this hypothesis turns out to hold, the goal is to collect quantitative data about route composition from big data. The objective is to use the resulting information to enhance the choice set used in the route selection problem either by integrating it in a constructive procedure or by using it to filter improbable candidates from the choice set in the filtering stage. The minimum number of least cost paths required to reconstruct a proposed route can be used in the *elimination by aspects stage* mentioned in Bovy (2009) where a candidate is accepted or rejected for inclusion in the *consideration set*, based on the values for a set of attributes. To the best of our knowledge, the proposed *structural* path attribute was not used for route quality assessment before.

Each path in a network leads from an origin to a destination. Non-least-cost paths can be split into parts using *intermediate destinations* so that the traveler moves to each (intermediate) destination along a least cost path. The minimum number of intermediate destinations, required to construct the path, determines the structural complexity of the path. The lower this number, the simpler the structure is.

Analysis of GPS traces allows to determine the structural complexity of paths used in practice; it seems that actually used routes have a fairly simple structure. In

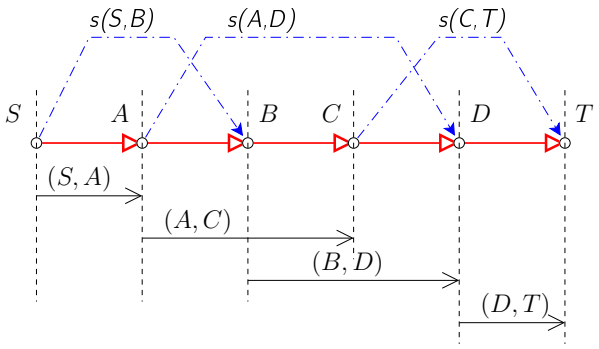


Figure 6.1: Someone moves from S to T via A, B, C, D. The longest least cost subpaths are $(S \to A), (A \to C), (B \to D)$ and $(D \to T)$. Examples of non-least-cost subpaths are: $(S \to B)$ and $(C \to T)$. The resulting splitVertexSuites are: $\{A\}, \{B, C\}, \{D\}$. The lines labeled $s(S, B)$, $s(A, D)$ and $s(C, T)$ represent shortcuts (not used by the traveler who moved along the red line).

order to be realistic, the generated choice sets shall reflect the structural properties revealed by observed trips.

When the selected path in the network is revealed by the traveler (by enumeration of all links and nodes), not all information concerning the route choice decision is known to the analyst neither is the complete information about the path structure, that could be relevant for the decision, uncovered. The reason for this is explained by following observations. The *minimum number* of least cost components in a revealed route can be determined but in general there are multiple different minimum decompositions of a given path into least cost subpaths: the minimum decompositions are not unique. Hence, the analyst can derive how many components the traveler had in mind but does not know which ones. An example is shown in Figure 6.1. Someone moving from a source S to a destination T passing via nodes A, B, C and D could have taken either the least cost paths sequence $(S \to A), (A \to B), (B \to D), (D \to T)$ or the sequence $(S \to A), (A \to C), (C \to D), (D \to T)$ both of which result in exactly the same revealed sequence of nodes in the network. This happens when A, B, C, D all are on the revealed route, the minimum decomposition consists of four components and there are two different ways to split the path (using either of the combinations $\langle A, B, D\rangle$ or $\langle A, C, D\rangle$). In such case, the analyst cannot decide what the traveler had in mind. Several possible decompositions are concealed in the revealed route. This is important to know while looking for the motivation for the route selection.

The revealed route can be described by a vector of *splitVertices*. A *splitVertex* is

a vertex that belongs to two consecutive components in a minimum path splitting. In the example given, A, B, C and D are *splitVertices*. The size of the vector is known because the size of the minimum decomposition can be determined unambiguously. However, for a single path, several such vectors do exist. Each element in the vector is to be chosen from a specific subset of the vertices in the path: i.e. the i-th element is to be chosen from the i-th set. In the example given

1. the size of the minimum decompositions is four so that three *splitVertices* are required
2. the sets from which to select are $\{A\}$, $\{B, C\}$ and $\{D\}$
3. and two vectors generating valid splittings were identified : $\langle A, B, D\rangle$ and $\langle A, C, D\rangle$

In chapter 8 an efficient way to determine the size of the minimum decomposition and the sets from which to select split vertices, is presented. The sets turn out to be disjoint but not every combination of vertices selected from them, constitutes a valid decomposition.

This is a fortunate observation since it means that the data contain additional information. Chapter 9 defines *vertex importance relative to a set of paths* S as the occurrence frequency of the vertex in the set of *splitVertex* vectors for all possible minimum decompositions for the paths in S. A method is provided to enumerate all possible decompositions for each path found in the set of GPS traces for a given region, in polynomial time. Hence vertex importance calculation is feasible. Furthermore, it can be interpreted as a measure for the probability of a network node (junction) to be used as an intermediate destination. This in turn supports route candidate construction.

The usefulness of an efficient trip decomposer is shown as follows. Specific sets of recorded routes can be analyzed (e.g. (i) all routes for a specific person, (ii) the routes starting in a specific time period, (iii) routes linking given areas or (iv) routes using a given vehicle type). After determining all possible minimum decompositions for each route, the occurrence frequency of each vertex as a *splitVertex* can be determined and reveals information that is useful for route choice set generation and for trip annotation.

The following examples illustrate some applications of vertex importance and of path complexity (the size of minimum path decomposition):

1. *route choice set generation*: (i) trips can be analyzed with respect to road category use in order to find out whether *splitVertices* correspond to road category changes (e.g. local road to motorway) aiming to find out to what measure

people use *hierarchical routing* and (ii) a *splitVertex* can identify way points corresponding to a safe junction crossing or can reflect the effect of specific road signalization.

2. *trip annotation*: stop and trip detection algorithms typically make use of time and distance thresholds (e.g. 180[sec] and 100[m] were found to be optimal settings to reproduce the trips in a set of diaries from the set of associated GPS traces in a study reported in Cich et al. (2015)). The detection algorithm detects a stop as soon as the traveler stays within a small area for sufficiently long time. This technique can conceal short stops on purpose (e.g. pick-drop activities). Such stops are expected to generate a *splitVertex*.
3. *mode detection*: particular sequences of split vertices might correspond to sequences of mode transfer locations in multi-modal trips (train stations, bus stops, carpool parkings, etc).
4. *travel behavior properties* might be revealed by the distribution for the size of the minimum path decompositions for the trips in particular segments: e.g. it is worth to find out whether traces for electric vehicles differ from traces for combustion engine vehicles due to range anxiety

All aspects mentioned in the examples are relevant for travel demand prediction and network loading in activity-based micro-simulators.

6.2 Components of the Research

Trips are extracted from GPS traces. The resulting sequences of recordings are transformed to *walks* in the road network graph by means of map matching. Walks that constitute a *path* (i.e. in which no vertices are visited more than once) are considered to represent utilitarian trips and the size of the minimum decomposition into least cost paths is determined for each of them. Finally, all minimum decompositions are enumerated. Those components are briefly introduced in the following subsections.

The chapters covering route splitting derive properties of the observed route exclusively by applying graph theoretical results, not requiring any tuning nor any operator judgment or interaction.

6.2.1 Trip Detection

A trip detector software was written to partition a sequence of GPS recordings into contiguous subsequences each of which constitutes a trip. This work was not reported

in a paper but is mentioned in Cich et al. (2015) where *trip* detection is compared to *stop* detection in GPS traces.

In *stop* detection, the individual is assumed to reside at a location when all recordings for a period longer that Δt are within a circle having radius R. All recordings between *stops* are assumed to constitute trips.

In *trip* detection, the longest possible subsequence for which (i) the moved distance is sufficiently large (similar condition as for *stop* detection), (ii) the speed values start from near zero, grow and drop again to near zero, and (iii) which does not have large gaps caused by missed recordings, is considered to constitute a *trip*. Sequences without gaps and consisting of recordings having almost zero speed are considered to be *stops* and the remainder of the recordings is considered to be junk.

The trip detector scans the GPS records and maintains a variable size sliding window containing the last records seen. Those records have not yet been finally qualified as *stop*, *trip* or *junk*. Each time a record is read, several quantities are evaluated: instantaneous and smoothed speed and acceleration, window size and period etc. Specific changes in the evaluated quantities lead to event firing. The events are fed to the state machine shown in Figure 6.2 that controls the qualification of the subsequence contained in the window. Definitely qualified records are dropped from the window.

The speed condition is required because part of the processed traces were recorded using devices that can be turned on/off by the user while driving. Plausible evolution of speed and lack of large gaps caused by missed recordings, were required to assure the extraction of complete trips (as opposed to junk parts).

6.2.2 Map Matching

Chapter 7 describes a new method for offline map matching (i.e. batch processing of GPS traces). It is as efficient as the state of the art methods that keep track of a limited set of candidate solutions; those methods need to make crucial decisions about link use and about dropping solution candidates, for every GPS recording being processed. As a consequence the decisions are based on information about the point being processed and its predecessors. On the other hand the new method is a *global* one in the sense that all recordings in the GPS trace are involved in all link selection decisions but it is faster than other published global methods.

While processing the GPS recordings sequentially, a graph is built that contains all topological and temporal information that is available from (i) the road network, (ii) chronology of the recorded coordinates and (iii) the distance between each GPS

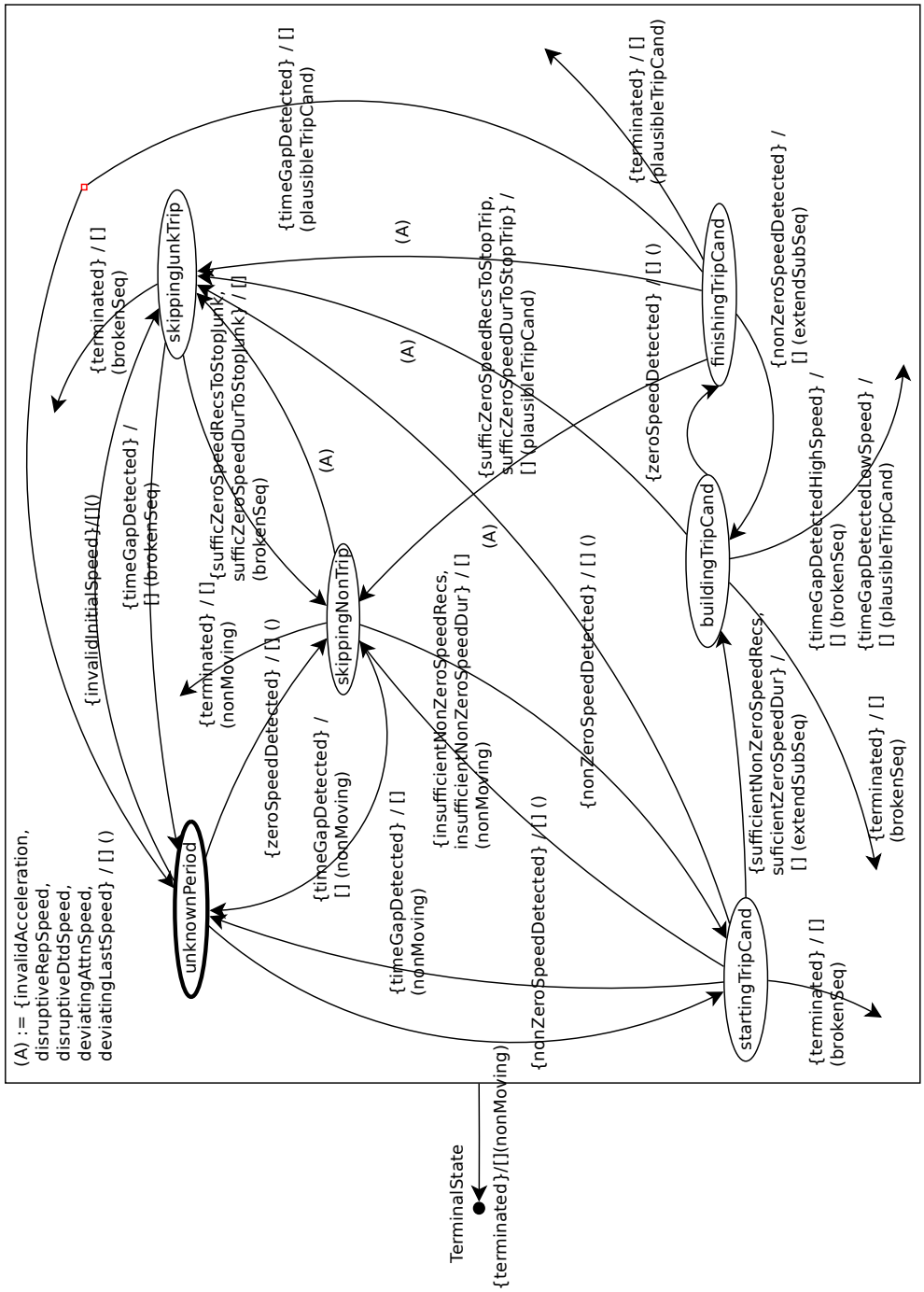


Figure 6.2: Finite State Machine (FSM) controlling the trip detector software while scanning a stream of GPS recordings.

point and each network link geometry which results is a weight for each *link touching (matching)*. After processing the complete sequence of GPS recordings, the maximal weight path in the graph generates the most likely walk in the road network.

The current implementation of the map matcher requires that each link on the route used by the traveler is *touched* by at least one GPS recording. The reason for this decision is that, in order to verify the hypothesis formulated in section 6.1, sequences of effectively used network links are required. Many map matching tools close detected gaps by inserting shortest paths. This is unwanted in the route decomposition research context.

6.2.3 Minimum Path Decomposition Size

The paths in the network resulting from map matching are analyzed. Chapter 8 describes an efficient algorithm to determine the *size of the minimum splitting* of a given path into least cost components. The algorithm is applied to process two sets of GPS traces.

GPS traces owned by IMOB (Bellemans et al. (2008)) were processed using the map matching tool described in chapter 7. A second, much larger, dataset was map matched using a tool written by Fraunhofer IAIS, Bonn (IAIS). The reason for this decision is data ownership. For the latter dataset, the raw GPS traces could not be disclosed; instead, link sequences generated by the IAIS map matching tool along with link lengths but without vertex coordinates were made available. Chapter 8 compares distributions for the size of the minimum decompositions for both datasets and map matching tools.

A third dataset recorded for Electric Vehicles (EVs) was acquired but it turned out to be unusable for the planned research because of problems during GPS recording. The distribution for the trip length was unrealistic and the technical recording problem was confirmed by the people who collected and cleaned the data.

The procedure to determine the minimum path decomposition ignores trivial nodes (pass-through nodes). Trivial nodes are introduced in maps to describe changes in road characteristics or administrative borders. The traveler does not have to make a link selection decision in trivial nodes. They are excluded from the analysis so that a *normalized* network is used.

6.2.4 Enumeration of all Minimum Path Decompositions

Chapter 9 specifies a five-stage procedure to enumerate all minimum decompositions for a given path in a graph. Graph theory is used to prove the correctness of the

procedure. It is also shown that the procedure can be executed in polynomial time which makes it suitable for practical use.

Finally the notion of *vertex importance* with respect to a given set of paths, is defined. The chapter suggests to correlate vertex importance with network attributes of the vertex in order to predict importance values for vertices that were not visited by the revealed paths. Those values might uncover (i) the use of road hierarchy while routing and (ii) the transportation relevance of particular nodes or their incident links.

Implementation of the proposed five-stage procedure is not part of this thesis.

6.3 Kind of Reported Results

In this research, several new methods were developed and each of them is described in a separate chapter. At the time of writing, chapters 7, 8 and 9 each correspond to a paper submitted for peer review. The methods described in chapters 7 and 8 have been implemented and the reported results consist of algorithm explanation, algorithm proof (for chapter 8) as well as application results. For chapter 9 algorithms and mathematical proofs are given but implementation and application results are not yet available.

Chapter 7

Map Matching

This chapter consists of	
Knapen et al. (2015a)	*Efficient Offline Map Matching of GPS Recordings Using Global Trace Information*
Related co-authored papers	
Cich et al. (2015)	TRIP/STOP Detection in GPS Traces to Feed Prompted Recall Survey

7.1 Abstract

A new procedure for map matching of GPS recordings by batch processing is presented. Such methods are used in transportation science in order to extract patterns from large datasets. The proposed technique is said to be *global* because it processes the complete sequence of GPS recordings before making any decision about the sequence of links used by the traveler. Most recent offline map matching techniques keep track of a limited set of candidate sequences while sequentially processing GPS recordings. Since in general, more candidates are produced than can be kept in the candidate set, non-promising candidates need to be discarded. This is done based on the information contained in the partial GPS recordings stream processed up to the moment the pruning decision is taken and hence the decision is said to be based on *local* information. In order to make use of all information contained in the trace, the proposed method makes use of *link-use-period assumptions*. They assign a weight (likelihood) to the assumption that a given road network link is used during a given period of time. At the conceptual level two processing stages are distinguished (although both are executed in a single pass over the data). In the first stage, links matched by GPS recordings are identified, the GPS trace is partitioned in contiguous sub-traces so that recordings in each part touch the same set of links. Link use period assumptions are extracted from the partitioned GPS trace and organized in a graph based on the chronological and topological relations among them. The vertices in the graph represent link-use-periods. Each of them gets assigned a weight based on the distance between the GPS points and the geometry of the touched (matched) road network link. In the second stage, this graph is used to find the sequence of link-use-periods for which the corresponding walk in the road network has the highest likelihood to have generated the trace of GPS recordings observed. The proposed technique is computationally as efficient as the current methods based on candidate sets of limited size but it takes all GPS points in the trace into account for each link selection decision.

7.2 Introduction

Map matching combines a road transport network description consisting of nodes and directed links with a time series of coordinate tuples that describes the movement of a traveler. The purpose is to reconstruct the sequence of links crossed by the traveler in chronological order. In this section a short overview of existing map matching techniques and their respective fields of application is given in order to sketch the

state of the art. The technique proposed in this paper is aimed at offline (batch) map matching of GPS traces. Two main classes of map matchers are distinguished.

7.2.1 Online Map Matching

Online near real time map matching processes coordinate pairs as soon as they come available and aim to determine the network link that is actually being traveled. Map matchers in this class are deployed in navigation aids. Their software operates on dedicated microprocessors and typically data sampling is in the order of 1 to 100 Hz. In many cases data from several sensors (odometer, gyroscope, accelerometer, etc) are available for data fusing along with GPS coordinates. Quddus et al. (2007) provide a comprehensive overview of online map matchers. Greenfeld (2002), Ochieng et al. (2010), Li et al. (2013), Abdallah et al. (2011) discuss the data fusion techniques and inference methods. The aim of online map matching is to determine the link on which the vehicle is moving and to calculate the position of the vehicle on the link as accurately as possible (e.g. for traffic signal influencing by buses (Quddus et al. (2007))). The latter is essential in Intelligent Transportation Systems (ITS) applications and in Advanced Driver Assistance Systems. Nowadays map matchers are based on the Multi-Hypothesis Technique (MHT) about the position of the vehicle. Such methods are called Multi-Hypothesis Map Matching (MHMM) in Bonnifait et al. (2009). In many cases, MHT and sensor data fusing feed maximum likelihood Bayesian inference engines and often Kalman filtering is used. Most of current online map matchers, starting with Greenfeld (2002), incorporate topology constraints.

7.2.2 Offline Map Matching

Offline or batch map matchers aim to process previously recorded sequences of coordinate pairs in order to extract travel behavior information either for a single moving object (either person or vehicle) over a long period or for a large set of moving objects. GPS recordings are either *vehicle traces* produced by dedicated devices mounted in a vehicle or *person traces* recorded by smartphones carried by individuals. The aim is to determine the sequence of links used by the moving object. Schüssler and Axhausen (2009) state that map matching of person traces requires high resolution network information. Available data consist of time series of GPS recordings and in some cases from other sources (Bluetooth, Wifi and mobile phone related events). Large datasets are available and need to be processed efficiently. In Quddus et al. (2007), Schüssler and Axhausen (2009), map matching techniques (both online and offline) are classified as (i) pure geometry based methods, (ii) topological methods, (iii) probabilistic

methods and (iv) advanced procedures. Pure geometric methods are further classified by Quddus et al. (2007) as point to point matching (finding the nearest node or shape point), point to curve matching (finding the polyline to which the distance is minimal) and curve to curve matching (matching the vehicle trajectory against known roads). Those methods can deliver link sequences that represent non-connected walks in the network.

The technique proposed by Marchal et al. (2005) solves the latter problem by adding topological constraints. It starts by determining which links are identified by the first few GPS recordings. Each of those constitutes the first link in a candidate path. When the next GPS coordinate pair is processed, for each route candidate being built, only the last link in the sequence and the links that can be reached from that link (*forward star*) are investigated when looking for links matched by the new coordinate pair. Since each candidate shall consist of a linear sequence of links, candidates are cloned and each clone is extended by exactly one member of the forward star. The candidates then are assigned a score and in order to avoid huge sets of candidates, only the N candidates having the best scores are kept ($N = 30$). Scoring is done as follows. Each GPS point can match at most one link in each candidate. The distance between the point and the link is a measure for the quality of the selection (the lower the better). The score for a candidate takes the sum over all points of the distance between the point and its corresponding matched link. If there are too many candidates, the ones having the highest scores are discarded. The computational effort and memory requirements grow with N. Making N too small, can cause promising candidates to be removed prematurely and hence can decrease the average quality of the final candidates. Schüssler and Axhausen (2009) evaluate this technique by comparing the quality (score) of the best solutions found and the corresponding computational effort for several values for the maximal candidate set size N. The paper concludes that the value reported in Marchal et al. (2005) is a valid one; the average score per GPS point does not significantly decrease with the candidate set size for $N > 30$. It also reports that the processing time per point is between 10[ms] (for $N = 20$) and 75[ms] (for $N = 100$).

Zhou and Golledge (2006) use a similar procedure implemented in ArcGIS. GPS recordings are processed sequentially and a pool of candidate solutions is kept. In a preprocessing stage, they first replace clusters of GPS points by their centroid (*cluster reduction*) but also add interpolated GPS points when the distance between two consecutive points is larger than half of the minimum length for the links in the buffer defined by the two GPS points. Then a 2-norm (distance) and a rotation measure are used to determine the weight for each point in the preprocessed dataset.

A set of candidate partial paths is kept and extended so that a connected walk results from the method. In the link selection phase, a Dempster belief function is used to determine the plausibility of the selected link. However, the authors do not explain what criteria were used.

Feng and Timmermans (2013) use a Bayesian Belief Network (BBN) to replace the ad hoc rules used in map matchers not making use of the Multi-Hypothesis Technique (MHT), to select the next road segment in a route. The input for the method consists of (i) Positional Dilution Of Precision (PDOP) (ii) the difference in direction between the road segment and the line segment defined by the last two GPS points, (iii) the distance from the GPS point to the line segment, (iv) the connectivity between road segments and (v) azimuth information. For a set of routes the effectively used line segments have been recorded by the traveler. This dataset serves as the truth value which is used for training the BBN. While processing a new sequence of GPS recordings, the BBN is used to determine the probability for a candidate link to become the next one in the route. The link having the highest probability is selected. In this procedure, the topological constraint is not forced. Connectivity information is used as an input variable and the resulting sequence of selected road segments is not necessarily a connected one.

Chen et al. (2011) propose a probabilistic method to simultaneously detect the road segment sequence and the transportation modes used. The likelihood that a given multi-modal path in a network generates the observed sequence of smartphone data is estimated. The measurement equations establish the probability that a given path generates a given time series of measurements. The travel model consists of frequency distributions for the speed estimated for six different modes. The phone measurement model involves GPS coordinates, speed, acceleration and Bluetooth events.

Bierlaire et al. (2013) further elaborate the proposed probabilistic measurement model introduced by Chen et al. (2011) and show how to compute the integrals. The path is decomposed into arcs and integrals are evaluated over each arc and summed. The concept of *Domain of Data Relevance (DDR)* is used to limit the computational requirements; e.g. the difference between the arc direction and the reported heading (in points where the speed is sufficiently high) are used to discard candidate links. The procedure explicitly takes the map inaccuracy into account and rigorously elaborates the measurement equations and the traffic model. Network topology is taken into account during the path generation phase which is similar to the one used in Marchal et al. (2005) and in Schüssler and Axhausen (2009) but allows to look ahead over multiple links in order to relax the requirement that each link needs to be matched

by at least one GPS recording.

The methods mentioned above process the GPS points in chronological order. For each point, they decide whether or not to accept a link as the next one in a candidate sequence based on scoring or rigorous stochastic likelihood calculations respectively. Each procedure keeps track of a limited set of candidate paths.

Brakatsoulas et al. (2005) propose three algorithms: (i) a greedy algorithm processing one point at a time using a distance and an angular criterion to select the next edge, (ii) a *recursive local look-ahead* method (inspecting up to 4 network links and GPS points ahead) and (iii) a *global method* that minimizes the Fréchet distance between curves. An overview of the global method is given in order to allow comparison with the method proposed in this paper. First the concept of *free space* is introduced. This is the set of points on two curves for which the distance is less than a given ϵ. The curves of finite length are defined by $[0,1] \rightarrow \mathcal{R}^2$ so that the free space is a subset of $[0,1]^2$. Then it is observed that if and only if a (monotone) continuous curve from (0,0) to (1,1) does exist in the free space, the (strong) Fréchet distance between the curves is less than ϵ. The free space concept then is extended to *free space surface* in order to compare a curve C to a graph (each edge combined with C generates a free space and those are combined into a *free space surface*). The sequence of GPS coordinates constitutes a piecewise linear curve. For a given ϵ, the free space surface for such curve and each path in the graph is computed. Finally the minimum value for ϵ for which a (monotonic) curve can be found in the free space surface, is determined by parametric search. This results in the *globally optimal* sequence of links (i.e. the one that delivers the minimum ϵ value). This method delivers topologically valid sequences and does not require each traversed link to be matched by a GPS point. The complexity of the method using weak Fréchet distance is $O(mn \cdot log(mn))$ where m is the number of vertices and edges and n is the number of GPS points. Processing time is not given: the paper only states that the runtime for the global methods was much longer than the one for the incremental methods.

The method proposed in this paper takes topological constraints into account in order to deliver valid paths and in order to limit the computational effort by reducing the search space where to apply geometric verification. Furthermore, it constitutes a *global procedure* similar to the one described in Brakatsoulas et al. (2005) in the sense that the decision to select a given path is postponed until all GPS recordings have been processed. The evaluation method however is not global because the weights of individual GPS points (point to curve distance) are accumulated as opposed to the global Fréchet distance method for curve to curve comparison. The performance of the proposed method is similar to the one reported in Schüssler and Axhausen (2009).

7.3 Application Domain - Design Decisions

This section briefly discusses the intended use of the map matcher and some of the design decisions emerging from the related requirements.

1. The map matcher is used in a research context. Its results serve as input to multiple research efforts each of which has specific objectives. Hence, the assumptions underlying the map matcher algorithms need to be made clear and the software shall not make any assumptions other than the explicitly formulated requirements about the input data.

2. The purpose of the research projects using the map matching results is the analysis of revealed travel behavior. In particular, researchers aim to extract properties of routes revealed by GPS traces in order to support route choice set generation (i.e. the same purpose as the one mentioned in Bierlaire et al. (2013)). This leads to the requirement to efficiently derive the walk in the road network that has the highest probability to have generated the time series of GPS recordings.

3. The map matcher is aimed at processing large sets of GPS recordings and hence shall be efficient.

4. Trace analysis requires the use of high resolution road network maps (as opposed to the coarser networks mostly used in traffic assignment procedures fed by zone based origin destination demand matrices).

5. The map matcher requires high frequency recording which means that each link in the path used by the traveler is touched by at least one GPS point. Although it is technically feasible to extend the proposed algorithm using a look-ahead technique similar to the one used in Brakatsoulas et al. (2005) or using a method that determines the look-ahead horizon from the speed in the last processed GPS point, it was deliberately decided not to do so. The proposed look-ahead aims to skip links not touched by any GPS recording. This requires the introduction of an hypothesis with respect to gap filling. In the current implementation this was unwanted because of the *route splitting* research reported in Knapen et al. (2015c). This research investigates how revealed routes can be decomposed into a minimum set of least cost sub-routes. The size of such minimum decompositions is expected to deliver relevant information to support the route choice set generation. For this reason gap filling by means of shortest route segments in the case of missing recordings, is not allowed. This motivates the requirement

that each link is touched by at least one GPS point. This behavior is identical with the one reported by Marchal et al. (2005) and by Schüssler and Axhausen (2009).

6. Trip detection and map matching are separated. Sequences of GPS points have been broken down into subsequences each of which corresponds to a particular trip. Several procedures (e.g. Marchal et al. (2005)) perform trip detection and map matching in one step. In our case trip detection has been performed using acceleration and speed criteria. The reason for this decision was that the resulting map matched sequences served as input for a study described in Knapen et al. (2015c) that verifies properties of complete utilitarian trips. GPS sequences in which people switched the device on or off during the trip had to be excluded.

7.4 Principle of Operation

In a first step, links touched (matched) by GPS recordings are selected for processing. A distance threshold is used and no weighting is applied yet. This is similar to what is done on other methods described in the literature. It corresponds to what is called *Domain of Data Relevance (DDR)* by Bierlaire et al. (2013). The chronologically ordered sequence of GPS recordings is partitioned into contiguous subsequences so that each recording in a part touches the same set of links (see Figure 7.1). Each such part corresponds to a period in time p, defined by the first and last recordings in the part. Using the information contained in this partition, a graph $\mathcal{G}$ is constructed in which each vertex represents the assumption that a specific link l is used during a specific period p. Hence, each vertex can be identified by a *link-use-period* pair $\langle l, p \rangle$.

In a second step a weight is calculated for each *(link, GPS-recording)* pair. The weight value decreases with the distance between the location specified by the GPS recording and the link geometry. If the GPS coordinate is exactly on the link, the weight equals one. For each link l, the weight values are accumulated over the link-use-period p and the sum is assigned to the vertex in G that is identified by $\langle l, p \rangle$. Finally a maximal weight path in the graph $\mathcal{G}$ is found. This allows to reconstruct the walk in transportation network that accumulated the largest *link touch weight* and hence is assumed to have the largest probability to have produced the observed sequence of GPS recordings.

Note that the *touching* and *weighting* steps are conceptually distinguished in order to describe the method. However, both are integrated in the implementation and the

data stream is scanned only once. Details are described in the following subsections.

7.4.1 Link Touching - Sub-network to Search

A link in the road network is *touched* (matched, selected) by a GPS recording (x, y, t) if and only if the minimum distance between the point (x, y) and the link geometry is not larger than the accuracy threshold $\overline{d}$. This is the positional error that is not exceeded with a probability $\overline{p}$.

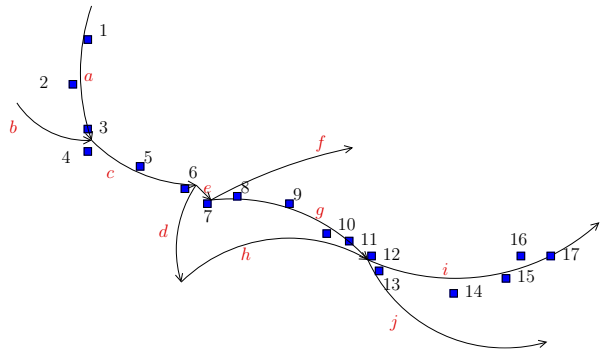
Given the accuracy threshold $\overline{d}$ and the associated probability $\overline{p}$, we derive that the probability to find N_r consecutive erroneous recordings is given by $(1-\overline{p})^{N_r}$. Let p_a denote the acceptable probability to experience a matching failure (i.e. a missed matching) due to GPS errors causing outlier recordings. The minimum value for N_r to avoid trouble caused by consecutive erroneous recordings is given by $N_r \geq \frac{\ln(p_a)}{\ln(1-\overline{p})}$. Several values for N_r were used to produce the results reported in Table 7.3. The N_r is used in the algorithm in places where at least one good recording is required to be able to proceed.

For each trip (GPS sequence to be map matched) the complete road network is searched for links touched by the first N_r GPS recordings. At least one correct link match is required to start the algorithm (hence N_r points are used). This is a very heavy operation. The required time per GPS point is two orders of magnitude larger than the time required for all remaining operations.

The map matcher keeps track of a *sub-network to search (SNTS)* for each trip being processed. Only this network is searched in order to find touched links for each GPS point. The SNTS is not necessarily a connected network. The initial version of every specific SNTS is a sub-network G_0 of the road network $G_R(V_R, E_R)$ constituted by the links touched by the first N_r GPS points. In order to find the links touched by the i-th recording ($i > N_r$) the SNTS $G_i(V_i, E_i)$ is derived as follows. The set of edges in SNTS $G_{i-1}(V_{i-1}, E_{i-1})$ touched by the N_r most recent GPS recordings $\underline{E} \subseteq E_{i-1}$ is extended with all edges $e \in E_R$ that have at least one vertex in common with end edge in $\underline{E}$. The result is G_i. This incremental procedure shows the importance of the N_r value. It is determined by the quality of the dataset to be processed. The probability to have N_r consecutive outliers (i.e. no good measurement among the last N_r recordings) shall be near to zero.

7.4.2 Touched Link Sets and GPS Trace Partitioning

Similar to other authors, we assume that the timestamp in the GPS records is correct. GPS recordings are processed in chronological order. For each recording, the *touched*



GPS Points	→ (TLS	,Period)
(1,2)	→ ({a}	,P_1)
(3,4)	→ ({a,b,c}	,P_2)
(5)	→ ({c}	,P_3)
(6)	→ ({c,d,e}	,P_4)
(7)	→ ({d,e,f,g}	,P_5)
(8)	→ ({e,f,g}	,P_6)
(9)	→ ({g}	,P_7)
(10,11)	→ ({g,h}	,P_8)
(12,13)	→ ({g,h,i,j}	,P_9)
(14,15,16,17)	→ ({i}	,P_{10})

Figure 7.1: Mapping chronologically contiguous GPS subsequences to *(Touched Links Set (TLS),Period)* pairs. Each subsequence is a CTLS-MP (Complete Touched Link Set for Maximal Period).

links set (TLS) is determined.

Chronologically consecutive recordings can share the TLS. This allows to partition the sequence of recordings into contiguous subsets such that two consecutive recordings belong to the same part if and only if they share the TLS. Since the GPS sequence is chronologically ordered, each part corresponds with a time period and the time periods are disjoint. The partitioning is illustrated by Figure 7.1. The left side shows a part of the road network along with some locations determined by GPS recordings. The right side shows contiguous subsequences of the GPS trace and their mapping onto tuples consisting of a TLS and a period. Each such subsequence is a *complete touched link set for a maximal period* (CTLS-MP). It is called *complete* since the link set contains all links touched by each GPS point in the subsequence. It is *maximal* since it cannot be extended in the time dimension (due to the construction rule). The CTLS-MP is described by a tuple $\langle\langle t_f, t_l\rangle, L \subseteq E_R\rangle$ where L is the touched link set and t_f and t_l are the timestamps for the first and last GPS recording (the tuple $\langle t_f, t_l\rangle$ constitutes the period identifier shown in Figure 7.1).

Note that an outlier GPS recording creates a part containing the outlier recording as the only element. Assume three consecutive parts p_{i-1}, p_i, p_{i+1} in the GPS trace where p_i is generated by an erroneous (outlier) recording. Then it is possible that the

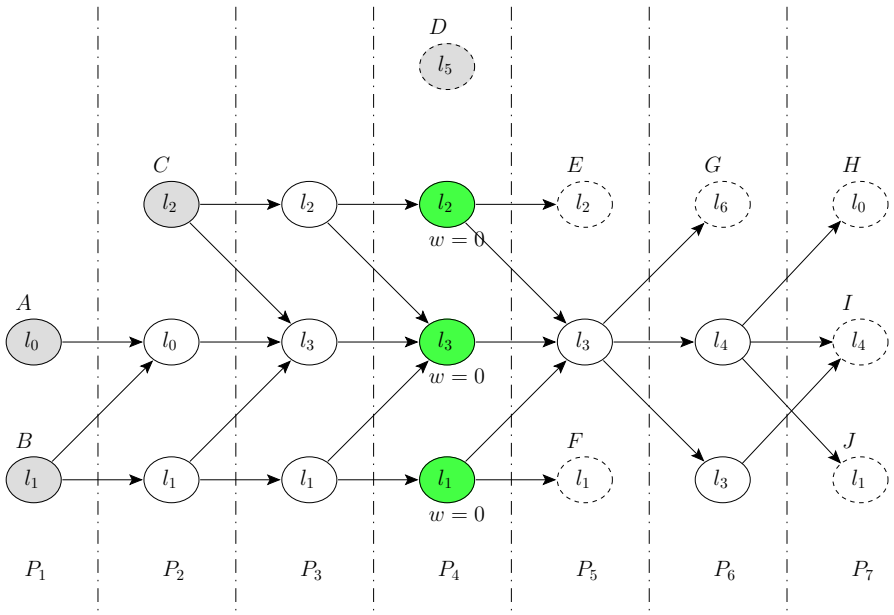


Figure 7.2: Sample ChronoLinkTouchGraph: P_i indicate periods, l_j denote used links, $V_I^L = \{A, B, C, D\}$ and $V_T^L = \{D, E, F, G, H, I, J\}$ denote the sets of *initial* and *terminal* vertices respectively. Vertex D represents a spatial outlier, the other vertices for P_4 (green) have zero-weight because they are inherited. It is possible that the route was a tour since l_0 and l_1 appear in the first and last periods.

TLS associated with p_{i-1} contains some links that have a node in common with links in the TLS for p_{i+1} while none of the links in the TLS for p_i has any node in common with links in the TLS for $_{i-1}$ and p_{i+1} respectively. This phenomenon is shown in Figure 7.2 (which does *not* apply to the example given in Figure 7.1).

7.4.3 Chronologically and Topologically Consistent Link Touch Graph

While generating the CTLS-MP for period P_i, a tuple $\langle l_j, P_i, w(l_j, P_i)\rangle$ is created for each link in the link set associated with period P_i. Here $l_j \in TLS(P_i)$ denotes the link and $w(l_j, P_i)$ denotes the weight accumulated by l_j during P_i. Details about the weight calculation are given in section 7.4.6 since they are not relevant here. Consider each period P_i and the corresponding touched links set $TLS(P_i)$. In case Pi contains less than N_r recordings, the minimum number of preceding periods $P_{i-1}, \ldots, P_{i-m}$

are considered so that

$$\bigcup_{x\in[i-m,i]} |P_x| \geq N_r \tag{7.1}$$

(if sufficient recordings do exist). $|P_x|$ denotes the number of recordings in period P_x. The inherited links set consists of all links contained in the selected predecessors link sets that are not contained in $TLS(P_i)$. The inherited link set is specified by

$$ILS(P_i) = \Big(\bigcup_{x\in[i-m,i-1]} TLS(P_x) \Big) \setminus TLS(P_i) \tag{7.2}$$

Then a zero-weight $\langle l_j, P_i, 0\rangle$ tuple is created for each $l_j \in ILS(P_i)$. The $\langle l_j, P_i, w(P_i)\rangle$ tuples for the touched and inherited links are used as vertices in a newly constructed graph. Vertices $v_a = \langle l_{j_a}, P_{i_a}, w(l_{j_a}, P_{i_a})\rangle$ and $v_b = \langle l_{j_b}, P_{i_b}, w(l_{j_b}, P_{i_b})\rangle$ are connected by an edge if $P_{i_b} = P_{i_a+1}$ (*chronological* constraint) and either $l_{j_a} = l_{j_b}$ or l_{j_a} and l_{j_b} share a node (*topological* constraint). The resulting graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is called the *ChronoLinkTouchGraph (CLTG)*. Note that P_{i_a+1} denotes the immediate successor period of P_{i_a}. The resulting graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is a acyclic digraph because $\forall\langle v_a, v_b\rangle \in \mathcal{E}$: $P_{i_b} = P_{i_a+1}$ while the periods are disjoint and hence ordered by a total order relation. A vertex in the ChronoLinkTouchGraph is called *initial (terminal)* if and only if it has no predecessors (successors). The sets of initial and terminal vertices in the LinkTouchGraph are denoted by V_I^L and V_T^L respectively.
A sample ChronoLinkTouchGraph is shown in Figure 7.2. The figure shows inherited zero-weight links only for period P_4 which is generated by an outlier GPS recording touching only l_5. Inherited links appear in all periods that contain less than N_r GPS recordings and not only in periods generated by outliers. This illustrates the problem described at the end of section 7.4.2.

In order to avoid confusion between graphs in the remainder of the text, symbols denoting vertices and edges in the ChronoLinkTouchGraph will bear a superscript L; for the transportation network a superscript T is used.

7.4.4 Uninterrupted Link Use Periods - Chronologically Compatible Neighbors

As soon as the *ChronoLinkTouchGraph* CLTG is built, it is possible to identify uninterrupted periods of use for each road network link. This is done as follows. Let v be a vertex in the CLTG, then $l(v)$ denotes the associated road network link. Each vertex in the CLTG is inspected and each vertex v_0 having no predecessor v in the

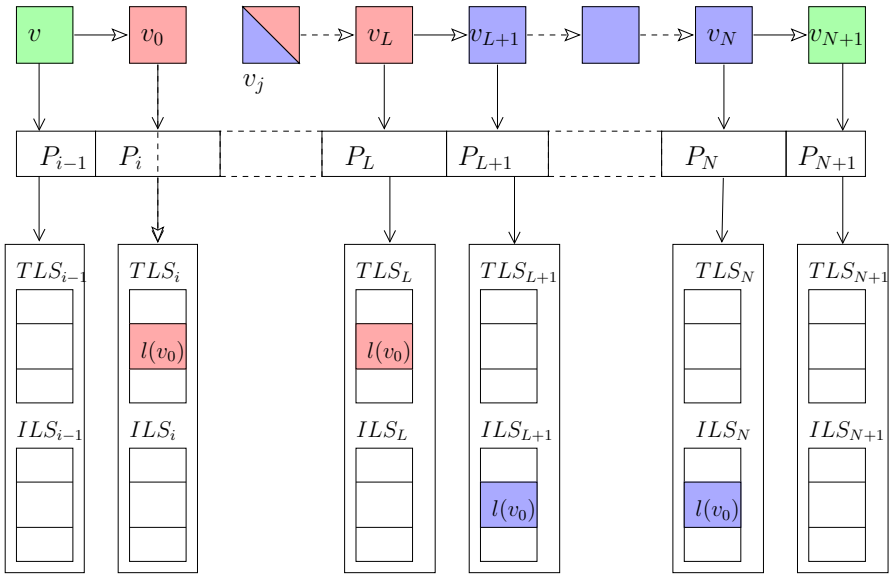


Figure 7.3: $((v, v_0, \ldots, v_L, v_{L+1}, \ldots, v_N, v_{N+1})$ represents a path in the *ChronoLink-TouchGraph*. $P_{i-1}, \ldots, P_{N+1}$ represent parts in the GPS trace. Each $\langle TLS_x, ILS_x \rangle$ pair represents the *touched* and *inherited* link sets respectively for period P_x. Green cells do not have $l(v_0)$ in either link set. For blue cells $l(v_0) \in ILS$. For red cells $l(v_0) \in TLS$. Red/blue cells represent vertices for which $l(v_0) \in TLS \cup ILS$. Red and partially red cells together constitute an uninterrupted link-use-period.

CLTG for which $l(v) = l(v_0)$ holds, is used as a start vertex to find the longest path $(v_0, \ldots, v_N)$ such that $\forall v \in \{v_0, \ldots, v_N\} : l(v) = l(v_0)$. Please refer to Figure 7.3.

If v_0 applies to period P_i, then the vertex is effectively touched by the first recording in P_i since (i) all GPS recordings in P_i share the link set TLS_i and (ii) a link that is not touched in P_i can only have been inherited from predecessors of v_0 but v_0 has no predecessor v so that $l(v_0) = l(v)$ which is equivalent to $l(v_0) \notin TLS(v) \cup ILS(v)$.

On the other hand, $l(v_N)$ can be inherited. Let $v_L \in \{v_0, \ldots, v_N\}$ denote the last vertex for which $l(v_L)$ was effectively touched i.e. for which $l(v_L) \in TLS(v_L)$. Vertices $v_{L+1}, \ldots, v_N$ (if any) inherited $l(v_L)$ as a zero-weight link (i.e. v_L is in their *ILS* not in their *TLS*). Then v_L determines the last GPS recording in the uninterrupted link-use-period (ULUP) for $l(v_0)$.

As soon as is known *that* a link was used it is important to know in which direction it was crossed. This cannot be determined with certainty by looking at the GPS recordings for a single ULUP. Therefore we determine the sets of probable *entry* and

exit nodes for the link. This is done by inspecting the first and last GPS recordings in the uninterrupted period. Their respective distances to the nodes connected to the link are compared to the Euclidean distance between those nodes. If the sets delivered by equations (7.7) and (7.8) are singleton sets, the link crossing direction is assumed to be sufficiently certain. The confidence is quantified by the parameter $0 < F < 0.5$.

Let $d(a, b)$ denote the Euclidean distance between a and b. The link entry and exit vertices for l_v are derived from the ULUP as follows:

$$F = \frac{1}{3} \tag{7.3}$$

$$\mathbf{p}_f = \mathit{firstGpsRecInULUP} \tag{7.4}$$

$$\mathbf{p}_l = \mathit{lastGpsRecInULUP} \tag{7.5}$$

$$\tag{7.6}$$

$$\mathit{entryNodes} = \begin{cases} \{n_0\} & if \quad \Big(d(\mathbf{p}_f, n_0) \leq F \cdot d(n_0, n_1)\Big) \wedge \Big(d(\mathbf{p}_l, n_1) \leq F \cdot d(n_0, n_1)\Big) \\ \{n_1\} & if \quad \Big(d(\mathbf{p}_f, n_1) \leq F \cdot d(n_0, n_1)\Big) \wedge \Big(d(\mathbf{p}_l, n_0) \leq F \cdot d(n_0, n_1)\Big) \\ \{n_0, n_1\} & else \end{cases} \tag{7.7}$$

$$\mathit{exitNodes} = \begin{cases} \{n_0\} & if \quad \Big(d(\mathbf{p}_l, n_0) \leq F \cdot d(n_0, n_1)\Big) \wedge \Big(d(\mathbf{p}_f, n_1) \leq F \cdot d(n_0, n_1)\Big) \\ \{n_1\} & if \quad \Big(d(\mathbf{p}_l, n_1) \leq F \cdot d(n_0, n_1)\Big) \wedge \Big(d(\mathbf{p}_f, n_0) \leq F \cdot d(n_0, n_1)\Big) \\ \{n_0, n_1\} & else \end{cases} \tag{7.8}$$

Link crossing direction is determined in order to eliminate U-turns in cases where those are improbable. This is required because the weight maximization technique we used tends to introduce U-turns. The value for F is not critical since in most cases the link direction crossing is clear from topological constraints only. A lower value for F results in more $\{n_0, n_1\}$ cases and hence in lower uncertainty about the link crossing direction; it also results in more edges in the *ChronoLinkTouchGraph*. Since the first and last recordings for the complete uninterrupted period of link use are considered, the result can be assumed to be more reliable than methods that use information up to a given point in time only.

Finally, consider a CLTG vertex v_0 for P_i and a vertex v_1 for P_{i+1}. Vertices v_0 and v_1 are said to be *chronologically compatible neighbors (CCN)* if and only if

$$exitNodes(l(v_0, ULUP(P_i)) \cap entryNodes(l(v_1), ULUP(P_{i+1})) \neq \emptyset \tag{7.9}$$

Chronologically compatible neighborhood of l_a and l_b for periods P_i and P_{i+1} is denoted by $CCN(l_a, l_b, i)$. It specifies that the assumption of moving on l_a in period P_i, then crossing exactly one node in the road graph G_R to continue moving on l_b in period P_{i+1} is compatible with the GPS trace.

7.4.5 ChronoLinkTouchGraph Pruning

While building the CLTG, vertices for consecutive periods were linked by an edge if they are associated with a single transport road network link or have at least one shared node. No timing or link traveling direction information was used. The CLTG was then used to determine the first and last GPS recording for uninterrupted periods of link use. This information leads to the notion of chronologically compatible neighbors which now is used to prune redundant edges from the CLTG.

Consider an edge connecting v_a (for P_i) and v_b (for P_{i+1}) in the CLTG. The edge is kept if and only if $l(v_a) = l(v_b)$ or

(i) $l(v_a)$ and $l(v_b)$ share at least one node $n^T_{a,b}$ in the transportation network digraph and

(ii) it is topologically possible to move from $l(v_a)$ to $l(v_b)$ by crossing the shared node $n^T_{a,b}$ in the road digraph and

(iii) $CCN(l(v_a), l(v_b), i)$ which means that movement from $l(v_a)$ to $l(v_b)$ can have generated the observed GPS recordings sequence for periods P_i and $Pi + 1$.

7.4.6 Link Touch Weight

The error for the GPS device is assumed to have a normal distribution with zero mean and given standard deviation σ for both longitude and latitude: $e_{lon} = e_{lat} \sim Normal(0, \sigma)$. It is assumed that the error does not exceed a given value $\overline{e}$ with probability $\overline{p}$. Then the standard deviation follows from the inverse of the cumulative distribution function for the normal distribution : [1]

$$\sigma = \frac{\overline{e}}{\sqrt{2} \cdot erf^{-1}(2 \cdot \overline{p} - 1)} \tag{7.10}$$

[1] expressions for the cumulative distribution function for the normal distribution are found in Weisstein (1999), Spiegel (1968) and others

The distance d between the true position and the measured one then is given by $d = \sqrt{e_{lon}^2 + e_{lat}^2}$ and has a Rayleigh distribution: $d \sim Rayleigh(\sigma)$. The probability that the error is larger than the observed distance between the GPS point and the network link, is used as a weight. The CDF (cumulative distribution function) for the Rayleigh distribution is given by

$$F_R(x) = 1 - exp(-\frac{x^2}{2 \cdot \sigma^2}) \tag{7.11}$$

and hence for observed distance d the weight is given by

$$w(d) = exp(-\frac{d^2}{2 \cdot \sigma^2}) \tag{7.12}$$

The scoring function only makes use of the weights discussed above. Schüssler and Axhausen (2009) add a penalty term proportional to the square of the difference between the actual speed and the free-flow speed in the scoring function. We deliberately refrain from doing this because part of the traces are produced by vehicles in congested traffic. The mentioned speed difference is not related to the positional measurement error. In general, we aimed at only combining positional and chronological information with the network topology. Incorporating the minimum number of assumptions in the map matching process allows the results to be used for several purposes (e.g. to compare the speed distribution on a link with the speed limit).

7.4.7 Weighted Walk Generation

The *ChronoLinkTouchGraph* is used to find the route that delivers the highest weight $\overline{W}$. Since the ChronoLinkTouchGraph is a cycle-free digraph, there is an efficient procedure to achieve this. Once this route is found, it can be used to deliver routes delivering an *sufficient* weight $f \cdot \overline{W}$ with $f \in [0, 1]$. The map matcher performs following steps.

1. Each vertex $v^L \in \mathcal{V}$ corresponds to a link $l(v^L)$ in the road network. A path $(v_0^L, \ldots, v_l^L)$ in the *ChronoLinkTouchGraph* $\mathcal{G}(\mathcal{V}, \mathcal{E})$ determines a walk $(l(v_0^L), \ldots, l(v_l^L))$ in the transportation graph.

2. The weight for a given path $P = (v_0, \ldots, v_l)$ in the *ChronoLinkTouchGraph* is the sum of the weights associated with the vertices on the path: $w(P) = \sum_{v \in P} w(v)$.

3. First we determine the maximum achievable weight $\overline{w}$ for each initial vertex (which is trivial in a acyclic digraph). The overall maximal achievable weight

then is given by $\overline{W} = \max_{v \in V_I^L} \overline{w}(v)$ and the path delivering the maximum weight then is easily identified. The procedure determines at the same time the maximum achievable weight for each vertex using a variant of the Dijkstra (1959) algorithm.

4. We provide a facility to enumerate a set of paths connecting an initial vertex to a terminal vertex having a weight that exceeds a given value expressed as a fraction $f \in [0, 1]$ of the maximal achievable weight $\overline{W}$. The user specifies the maximum number of solutions to be reported and the maximal number of paths to check.

   The weight for a *specific path* $(v_0, \ldots, v_N)$ in the ChronoLinkTouchGraph $\mathcal{G}$ is given by the sum of the vertex weights:

$$p = (v_0, \ldots, v_N) \tag{7.13}$$

$$w(p) = \sum_{v \in \{v_0, \ldots, v_N\}} w(v) \tag{7.14}$$

   The achievable weight aW_2 for a *specific vertex pair* $\langle v_a, v_b \rangle$ is the maximum weight that can be reached by considering every path between the vertices. Let $\mathcal{P}(v_a, v_b)$ denote the set of all possible paths between v_a and v_b, then

$$aW_2(v_a, v_b) = \max_{p \in \mathcal{P}(v_a, v_b)} w(p) \tag{7.15}$$

   The achievable weight aW_v for a *specific vertex* v is the maximum weight that can be achieved by considering every path starting in an initial vertex and ending in v.

$$aW_v(v) = \max_{v_i \in V_I^L} aW_2(v_i, v) \tag{7.16}$$

   The required weight rW_2 for a *specific vertex pair* $\langle v_a, v_b \rangle$ is the minimum weight required in v_a so that there is a path $p(v_a, v_b)$ having a given weight W.

$$rW_2(W, v_a, v_b) = W - \max_{p \in \mathcal{P}(v_a, v_b)} aW_2(succ(v_a), v_b) \tag{7.17}$$

   The required weight rW_v for a *specific vertex* v is the minimum weight required in v so that there is at least one path having weight W that starts in v and ends in a terminal vertex v_t.

$$rW_v(W, v) = \min_{v_t \in V_T^L} rW_2(W, v, v_t) \tag{7.18}$$

number of links in the network (Belgium)	=	1136101
number of nodes in the network (Belgium)	=	841810
number of GPS recordings	=	10635372
number of movingObject-trip pairs (number of unique trips)	=	10672
largest number of trips for a single moving object	=	96
number of moving objects	=	744
maximum number of sufficient weight walks to deliver	=	8
maximum number of trials allowed to find sufficient weight walk	=	16384
required weight fraction f for sufficient weight walks	=	0.97

Table 7.1: Properties of the dataset used to generate the performance figures and sample diagrams.

The overall maximum achievable weight is computed as $\overline{W} = \max_{v \in V_T^L} aW_v(v)$.

The *sufficient weight* then is given by $f \cdot \overline{W}$. This value is registered with each terminal vertex as the required weight and the required weight $rW_v(v)$ for every other vertex v is calculated recursively.

Enumerating the paths delivering *sufficient weight*, is done by successively starting in each initial vertex $v_i \in V_I^L$, recursively extending the path with a vertex v and calculating the achievable vertex pair weight $aW_2(v_i, v)$. If for a given vertex v the achievable weight is sufficient (i.e. $aW_2(v_i, v) \geq rW_v(W, v)$) then v is used to extend the path in the *ChronoLinkTouchGraph*. Every time the recursive procedure reaches a terminal vertex, a *sufficient weight walk* in the transportation network is found and given as output.

7.5 Experimental Results

7.5.1 Data

The properties of the dataset used and the configuration setting have been summarized in Table 7.1. The recorded traces are described in Bellemans et al. (2008).

The algorithm needs the GPS accuracy specification as an input. The GPS accuracy value used in the map matching process, was determined as follows. Map accuracy has been discussed in Ochieng et al. (2010) and in Bierlaire et al. (2013).

Here we assume (similar to what is done by Marchal et al. (2005) and Brakatsoulas et al. (2005)) that the map is correct. However, our algorithm accounts for the positional error for roads in the map by using an increased expected error value for the GPS measurements. This is done because (i) the errors in the map cannot be ignored and (ii) the errors for points on a single polyline cannot be expected to be mutually independent.

OpenStreetMap (OSM) was used in our experiments. Haklay et al. (2010) investigated the positional accuracy for OpenStreetMap roads in the Greater London area. 109 different roads having a total length of 328[km] were compared to their counterpart in Integrated Transport Network (ITN) maps for which it can be assumed that the error is below 1[m]. It is concluded that if 15 contributors are active in an area, the positional error for the road is well below 6[m]. In *complete areas* the average error is 9.57[m] with a standard deviation is 6.51[m]. In incomplete areas the average error is 11.72[m] and the standard deviation is 7.73[m]. Completeness is defined as '*a measure of the lack of data*' and examined for specific areas by Haklay (2010) using visual inspection of maps and by comparing (by means of GIS) the total road length found in OSM and in reference maps respectively.

For the practical calculations we assume that Belgium constitutes a *complete area*. From several non-authoritative website sources we derived that the accuracy threshold $\overline{d}$ at 95% can be assumed to be 20[m]. However we observed several cases where the distance between the recordings and the map turns out to be larger. An example is shown in Figure 7.4. This can be explained by following facts: (i) the GPS traces were recorded in the period 2006-2008 using a Personal Digital Assistant (PDA) (pre-smartphone-era hand-held device) and the accuracy can be assumed to be less than for current devices (ii) the OSM map was downloaded on 2014-Dec-05: hence the network can have changed since the position recording. Several experiments have been carried out to estimate the effect of accuracy threshold $\overline{d}$ and the expected maximum for consecutive erroneous recordings N_r. Results are reported in Table 7.3. The cases are in the header of Table 7.3 by `'C_'<NR>'_A'<d>` where `NR` stands for the expected maximum for consecutive erroneous recordings N_r and `d` stands for the accuracy threshold value $\overline{d}$ expressed in meters. The performance figures reported in section 7.5.3 apply to a dataset characterized by the figures in Table 7.1

Two machines were used: (i) an Intel(R) Xeon(R) CPU E5-2620 v2 at 2.10GHz 24 core 64GB memory machine (slightly loaded by other processes) where the Java map matching software was allowed to use at most 22 threads and (ii) an Intel(R) XEON(R) E5440 at 2.83 GHz 8 core 32GB memory machine not used for other work

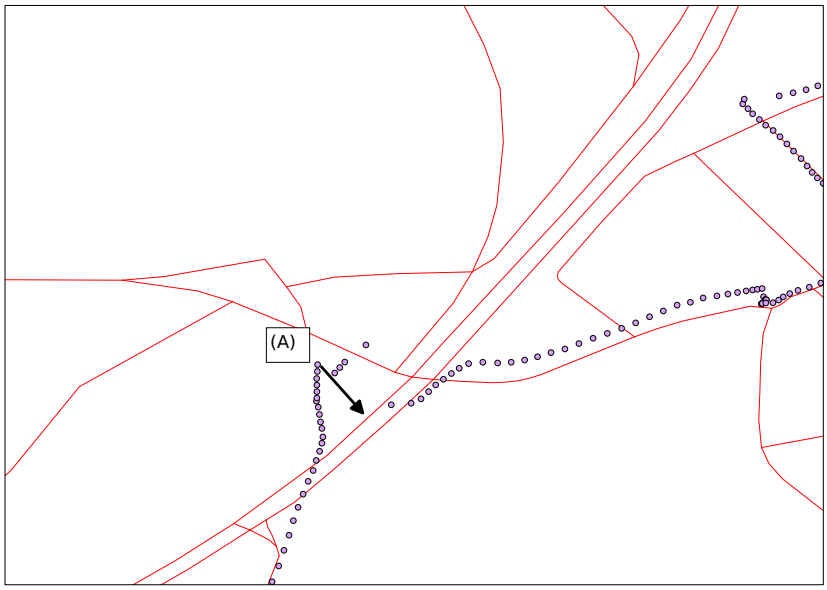


Figure 7.4: Trace for which no walk was found. The length of the segment indicated by the arrow between the point labeled (A) and the road is 77[m], well beyond the assumed accuracy of 20+10[m] at 95%.

where the Java software was allowed to use at most 6 threads.

7.5.2 Software Libraries Used

`postgis 1.5.3-2` library functions on `PostgreSQL 9.1` are used to determine the distance between a GPS point and the link geometry. The map matcher is written in `Java 7`. The experiments ran on `Debian GNU-Linux version 7.8`

7.5.3 Results Overview

The map matching application consists of two stages: (i) searching the complete road network for the links touched by the first vertices in the GPS sequence for each trip and (ii) processing all GPS recordings (including the first $N_r + 1$) using the relevant sub-network determined in the first stage

The first step is performed by calling `postgis` functions in `psql` queries issued from a `bash` script and is not (yet) multi-threaded. The Java implementation of the second stage is multi-threaded and is allowed to use the number of threads specified in the table that summarizes all results. Results have been summarized in Table 7.3.

CPU	Identifies the machine used
maxThreads	The maximum number of threads that the Java program operating on the *ChronoLinkTouchGraph* was allowed to use
real	Wall clock time used for the first stage
user	Time spent in user mode
sys	Time spent in system mode
nLinks	Number of links touched
movingObjects s	Number of moving objects for which at least one link was touched
movingObject-trip	Number of unique movingObject-trip pairs
mmRunTime	Run time for the second stage of the map matcher (wall clock time, determined from log)
nTrips	Number of trips successfully map matched
nFullCover	Number of trips for which the map matched sequence contains all recorded GPS recordings (i.e. the maximum weight path in the *ChronoLinkTouchGraph* generates a walk in the transportation graph that contains the links touched in respectively the first and the last *link use* period)
ratio	$\frac{nFullCover}{nTrips}$

Table 7.2: Definitions for the row labels used in Table 7.3.

The column headers identify the $\langle N_r, \overline{d} \rangle$ experiment settings. The table is subdivided in parts: (i) the first part identifies the machine used; (ii) the second part specifies values for the processing stage in which the complete network is searched for links touched by the first $N_r + 1$ GPS recordings in each trip; (iii) the third part shows performance results for the processing stage that constructs and exploits the *ChronoLinkTouchGraph* and (iv) the fourth part specifies the meaning of row labels used in the preceding parts.

	C07_A30	C11_A30	C15_A30	C19_A30	C07_A55	C11_A55	C15_A55	C19_A55
CPU	XEON	XEON	XEON	XEON	XEON	XEON	XEON	XEON
	8 Core	24 Core	8 Core	24 Core	8 Core	24 Core	8 Core	24 Core
	32GB	64GB	32GB	64GB	32GB	64GB	32GB	64GB
maxThreads	6	22	6	22	6	22	6	22
real	141m46.911s	199m03.149s	142m19.556s	197m23.817s	149m33.480s	198m0.871s	147m2.320s	198m19.981s
user	13m08.365s	28m13.090s	13m10.373s	27m49.872s	13m11.065s	27m58.769s	13m11.573s	28m0.893s
sys	3m22.241s	10m22.379s	3m20.453s	10m17.987s	3m21.349s	10m20.375s	3m21.157s	10m22.263s
nLinks	64353	81332	98958	117597	100671	119340	139644	161320
movingObjects	739	740	740	741	743	743	743	743
movingObject-trip	9972	10085	10177	10232	10271	10331	10364	10381
mmRunTime	5562	5380	5979	5390	5960	5424	5986	5259
nTrips	41094	41487	41931	42041	42886	43030	43174	43110
nFullCover	21777	21803	21847	21788	33408	33503	33550	33490
ratio	.5299	.5255	.5210	.5182	.7789	.7785	.7770	.7768

Table 7.3: Comparison of performance characteristics for several combinations of accuracy threshold $\overline{d}$ and expected maximum number of consecutive erroneous GPS recordings N_r. Definitions for row labels are given in Table 7.2. In the header labels `'C'x_'A'y`, the `x` stands for the N_r value and y stands for the accuracy threshold value $\overline{d}$.

A striking phenomenon observed in the table is that the runtime for the first stage depends only slightly on the number of points for which touched links need to be found. This is probably caused by a combination of disk caching and the use of `R-trees` in `postgis`. The number of recordings to process is the first stage is $(N_r + 1) \cdot nTrips$. The ratio between the number of points processed for the `C11-*` and `C19-*` cases is $\frac{12}{20} = 0.6$ whereas the processing times on a given machine seem to be nearly constant. A second observation is that the allowed number of threads has no large influence which means that memory contention probably is the limiting factor.

The total processing time is in the range of 14000 to 17000 seconds for 10 million points or about 15 [msec/point] which is in the same order as the one reported in Schüssler and Axhausen (2009) i.e. 10[msec/point] for a 20 candidates set and 75[msec/point] for 100 candidates using single-threaded code. This is explained by the fact that the part of time spent in single threaded software is in $[0.59, 0.69]$ in our case. The runtimes include the generation of *sufficient weight walks* mentioned in section 7.4.7 item 4.

7.5.4 Cover - Completeness

The map matcher determines the heaviest weight path between an initial and a terminal vertex in the LinkTouchGraph but does not force them to correspond to the first and last link-use-periods respectively. Some of those can be ignored because they are not associated with a maximum or sufficient weight path in the *Chrono-LinkTouchGraph*. This occurs because the topological constraint is not met. In order to evaluate the results, we consider two periods: (i) the *recording period* (duration d_R) determined by the first and the last GPS recording and (ii) the *matched period* (duration d_M) determined by the first and last recording in the *link-use-periods* associated with the maximal weight path in the ChronoLinkTouchGraph. The ratio $c = d_m/d_r$ is called the coverage: it is a measure of the completeness of the matching process. Table 7.3 shows that coverage is about 0.51 for the `C*_A30` cases and about 0.77 for the `C*_A55` cases. From this result we conclude that the initial estimate for the accuracy threshold $\overline{d} = 20 + 10[m] = 30[m]$ was too low.

Figure 7.5 shows the frequency distribution for the coverage value. Since the coverage equals one for the majority of the trips, the topmost diagram which contains all cases shows a sharp peak near the abscissa value of one. In the diagram at the bottom, only the cases having $coverage < 1$ are included. It shows the distribution of the coverage value for the non-complete cases only.

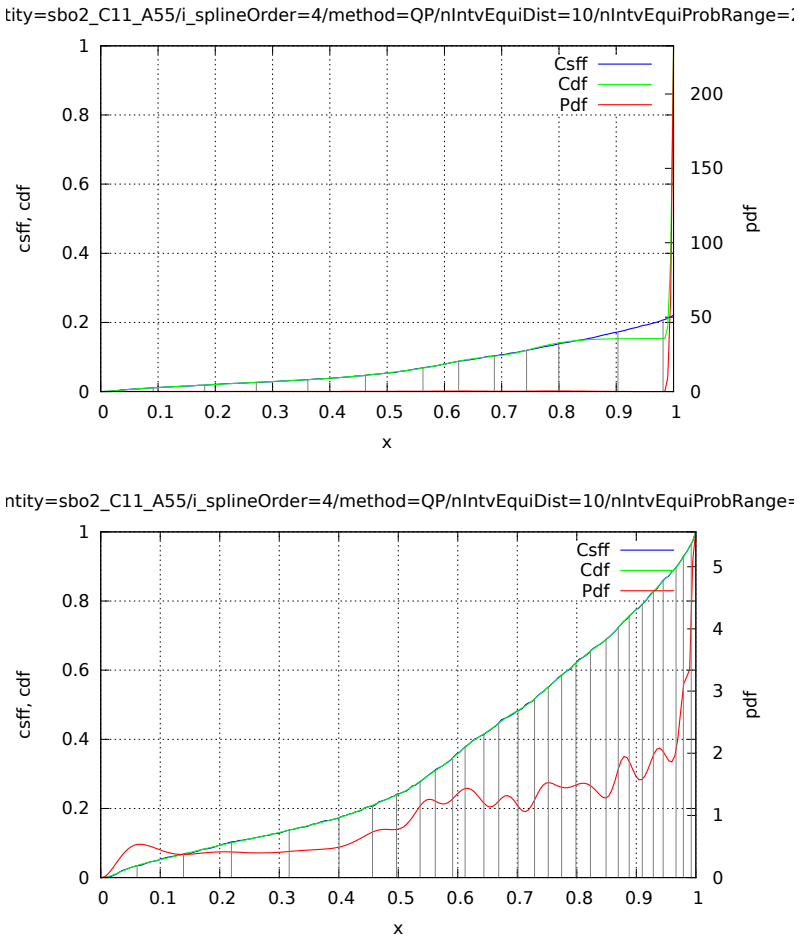


Figure 7.5: Frequency distribution for the fraction of the recording period that is covered by the map matched path. The diagrams hold for case C11_A55 where *ratio* = 0.7785. The top diagram is built using *time cover* values $\in [0, 1]$ hence the sharp peak at value 1. The bottom diagram holds for time cover values $\in [0, 1)$ (the open interval) hence $(1 - 0.7785 = 0.2215)$ of the cases. (`pdf`: probability density function, `cdf`: cumulative distribution function, `csff`: cumulative sample frequency function).

	0	1	2	3	4	5	6	7	8
C07_A30	4	1131	2087	1857	1215	816	497	408	1930
C11_A30	5	1127	2112	1897	1221	830	496	405	1950
C15_A30	9	1150	2147	1915	1231	841	500	403	1974
C19_A30	7	1152	2159	1914	1236	844	500	407	1977
C07_A55	0	706	2592	2034	1157	830	541	370	2035
C11_A55	4	719	2604	2039	1195	807	540	381	2033
C15_A55	3	726	2599	2058	1192	813	554	377	2035
C19_A55	1	727	2610	2063	1181	811	552	373	2034

Table 7.4: Frequency table for the number of heavy-weight paths found for the respective $N_r, \overline{d}$ combinations.

7.5.5 Heavy-weight Paths

Table 7.4 shows the frequency distribution for the number of heavy-weight walks found in the trip. In the reported experiment, a walk is a heavy-weight walk if and only if its weight is not less than $f = 0.97$ times the maximum weight. The cases where no maximum weight path was found, correspond to traces that go off-road. The traces have been recorded in 2006-2008, the OSM map is the version of 2014-Dec-05. An example of this phenomenon is shown in Figure 7.4. Note that the software was configured to stop searching after eight heavy-weight walks were found. Due to the algorithm properties, the maximum weight walk always is found. The other reported walks are walks having sufficient weight but they do not necessarily constitute the set of *heaviest*-weight walks.

7.5.6 Details

The maximum weight walk determined for trip 20 of individual HH10037GL23916, is shown in Figure 7.6. Since the data were recorded using a PDA as indicated in section 7.5.1, they constitute *person* traces (as opposed to *car* traces). As a consequence, each trip can be multi-modal and go off-road. However, neither the recording process nor the trip detection process identified the mode used. Hence, this information is not available to select a subset of the road network links in advance. Raw GPS points are not shown in order to avoid diagram clutter. This example shows that the overall matching is plausible. Figure 7.7 shows a *cycle path problem* which is a part of

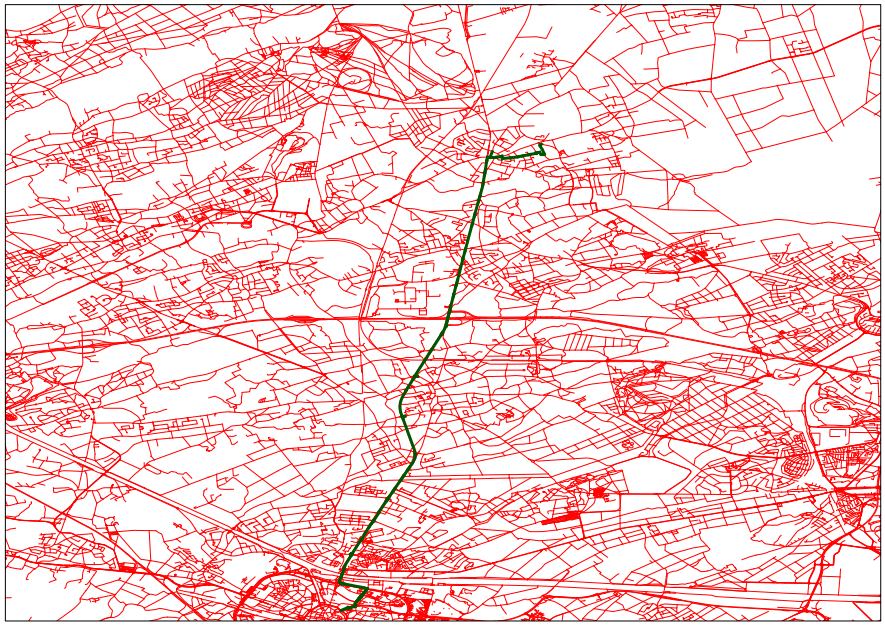


Figure 7.6: Complete walk for individual HH10037GL23916 trip 20.

the same trip for the same individual. The maximum weight walk includes some road segments that are cycle paths. This can only be avoided by performing mode detection in the trip detection stage (before map matching), provided that the road segments are correctly encoded in the map database. Additional direction or azimuth verification might help to avoid such cases. Figure 7.8 shows a map-trip discrepancy near the location labeled (A). This is caused by the long time period between data recording and map updates (see also section7.5.5).

7.6 Discussion - Comparison to other Methods

7.6.1 No Need to maintain a Candidates Set - Global evaluation

This section compares the proposed method to other procedures for batch processing of large sets of GPS recordings. The chronoLinkTouchGraph replaces the candidate sets maintained in the methods proposed by Marchal et al. (2005), Schüssler and Axhausen (2009) and Bierlaire et al. (2013). The chronoLinkTouchGraph contains all information required to enumerate all possible candidates. Furthermore, that data

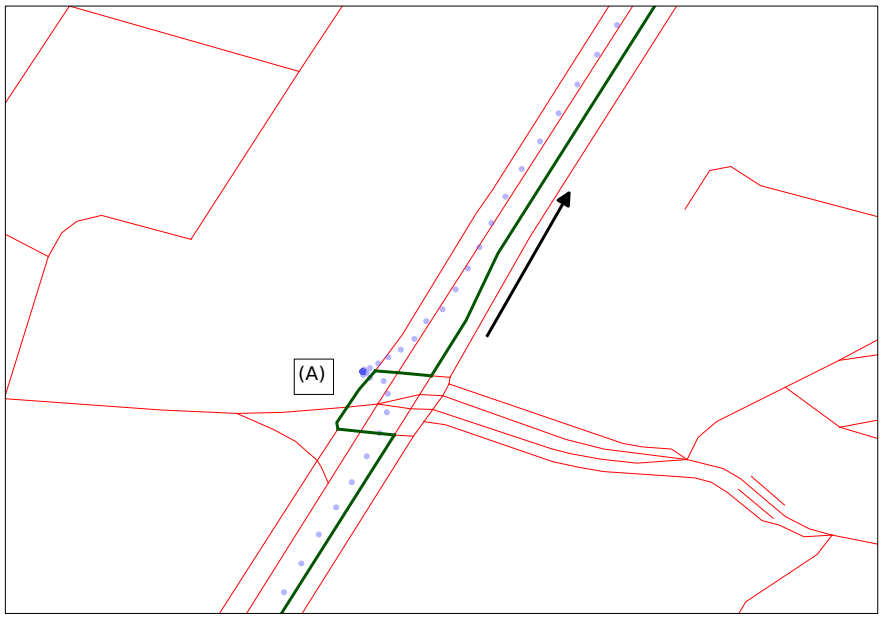


Figure 7.7: Cycle path problem for individual HH10037GL23916 in trip 20. Some links of a cycle path are included in a walk because this maximizes the matching weight. Such cases can be avoided by adding travel mode detection or direction/azimuth verification.

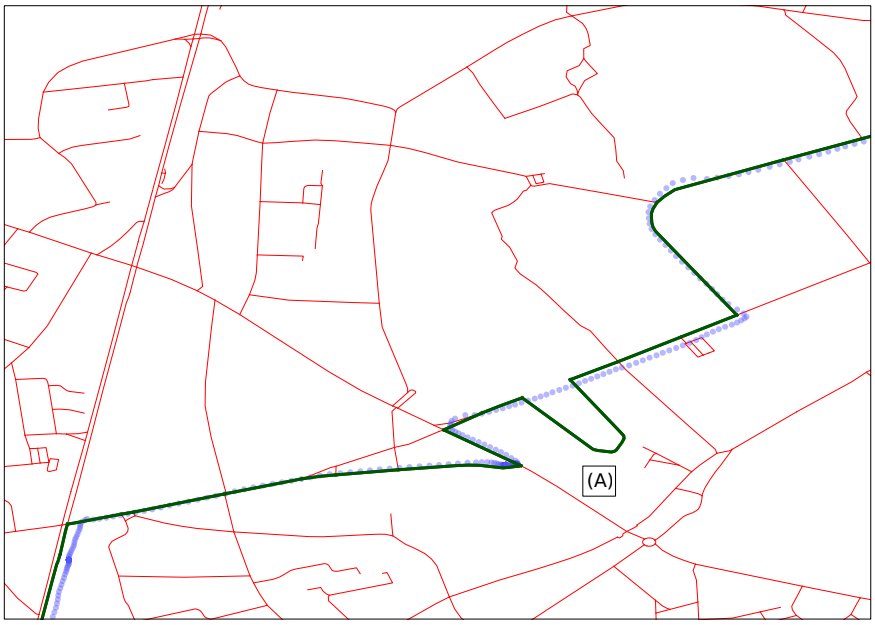


Figure 7.8: Map-trace discrepancy problem for individual HH10037GL23916 in trip 24. Clearly a link is missing in the map which causes the *detour* via (A). This example shows the need for selecting appropriate accuracy settings compatible with the map and the recording devices used.

structure allows for easy evaluation of the score of the complete walk for every possible path without regeneration of any data. The evaluation of the total score of the path can be compared to the evaluation of the Fréchet distance for each possible path in Brakatsoulas et al. (2005). In both cases all information contained in the complete sequence of GPS points is used in the selection of each link transition in the walk. This technique avoids premature dropping of promising walks.

7.6.2 Other aspects

The *U-turn* problem mentioned in Schüssler and Axhausen (2009) is solved by determining which of the nodes delimiting a link can have been used as the exit node. This is made possible because it only requires one additional edge in the LinkTouchGraph. The selection of the exit node is postponed until the maximal weight path determination as explained in section 7.4.4.

In the same way as Marchal et al. (2005) and Schüssler and Axhausen (2009), weights are used and the methods lacks the rigorous statistical foundation of the probabilistic method proposed by Bierlaire et al. (2013). On the other hand, the chronoLinkTouchGraph can be considered as DDR (Domain of Data Relevance) and the weighting using the Rayleigh CDF can easily be replaced by combination of probabilistic models (both *measurement* and *traffic* models).

7.7 Conclusion

A new map matching software for offline processing of large batches of recorded GPS traces, is presented. Topological, geometric an chronological constraints are used. Apart from the LON,LAT coordinates no other information is required. The complete information available in the GPS trace is used to evaluate every candidate route. No route can be prematurely dropped because a graph is built that contains every candidate. The technique shows similar computational performance as state-of-the art map matching tools, based on the Multi-Hypothesis Technique (MHT), that keep track of a limited number of candidate routes. The proposed tool builds a cycle-free digraph $\mathcal{G}$ containing the chronological and topological information for the complete period of recording. This graph embeds all possible candidates so that no candidates need to be dropped while processing the GPS trace. After processing the complete trace, the walk assumed to have produced the GPS trace, is produced by finding a maximum weight path in $\mathcal{G}$.

Chapter 8

Determining Structural Route Components from GPS Traces

This chapter is based on	
Knapen et al. (2014a)	*Canonic route splitting*
Knapen et al. (2015c)	*Determining Structural Route Components from GPS Traces*

8.1 Abstract

It is well known that people often do not use the least cost path through the transportation network while making trips. This leads to the question which structural path characteristics can be derived from GPS traces and be used to construct realistic route choice sets for use in traffic simulation models. In this paper, we investigate the hypothesis that, for utilitarian trips, the route between origin and destination consists of a small number of concatenated least cost paths. This hypothesis is verified by analyzing routes extracted from large sets of recorded GPS traces which constitute revealed preference information. Trips have been extracted from the traces and for each trip the path in the transportation network is determined by map matching. This is followed by a path decomposition phase. There are multiple ways to split a given path in a directed graph into a *minimal number of subpaths* of minimal cost. By calculating two specific path splittings, it is possible to identify subsets of the vertices (*splitVertexSuites*) that can be used to generate every possible minimum path splitting by taking one vertex from each such subset. The first result of this study is a frequency distribution for the minimal size of the least cost path decompositions. This constitutes input to evaluate candidates during the route choice set construction. A second result consists of the sets of vertices that can act as boundary vertices separating consecutive route parts; those vertices can be considered as *way points* having a particular meaning to their user. This allows for statistical analysis of structural route characteristics which in turn can support constrained enumeration methods for route choice set building. The paper explains theoretical aspects of route splitting as well as the process to extract *splitVertexSuite*s from big data. It also reports statistical distributions extracted from sets of GPS traces for both multimodal person movements and unimodal car trips.

8.2 Introduction - Context

Travel demand prediction by means of micro-simulation in activity-based models results in an agenda for each individual for the simulated period of time. Such agenda consists of a sequence of episodes each one of which is defined by a period of time, an activity type, a location and the modes used to reach the location. As soon as the locations are known, the traffic demand needs to be assigned to the transportation network. Thereto *route selection procedures* are required.

Such procedures are based on utility maximization where the utility depends on person and route characteristics (trip duration, route length, number of left turns

etc). In this paper it is proposed to integrate an additional (to the ones currently used) route feature in the choice process. In this paper it is proposed to integrate in the choice process an additional route feature (to extend the currently used set). The additional feature is the minimum number of intermediate destinations that is required to reconstruct the path from least cost components. The idea is that the traveler might have mentally constructed the path as a sequence of such intermediates (which *can* be landmarks) and then tries to reach each one of them as efficiently as possible.

Availability of big data sets of GPS traces allows for statistical analysis of the *structural* characteristics of large sets of routes. The minimal size of the decomposition into least cost paths was determined for routes derived from two sets of GPS traces. For both of them the traces for each participant were recorded for at least one week. No user interaction with the recording device was required. Hence this data collection can be interpreted as accurate revealed preference. The traces reflect what actually happened.

In order to extract the minimal decomposition size for each path, the recorded data are processed using the following steps:

1. **Trip detection**: the sequences of GPS recordings for each traveler are broken down into subsequences. Each such subsequence corresponds to either a trip (movement) or a stop (stay at a particular location having a non-zero finite area). Stops are not relevant in this study and hence are ignored in the remainder of the paper.

2. **Map matching**: the GPS sequence for each trip is matched to a network consisting of links (road segments) and nodes (junctions). Map matching is applied to each individual trip. It associates a sequence of visited links to the trip (in general a *walk* in a graph). From the set of walks, the subset constituting simple paths is kept. The other walks are ignored since they are assumed not to correspond to a trip for which the user aims at maximal utility (minimal generalized cost) since those walks contain at least one node that is visited multiple times without an intermediate stop to perform an activity.

3. **Route Splitting**: the map-matched route corresponds to a path in a graph in which each link is labeled with the link travel cost. The path is split into a minimal number of *Basic Path Components (BPCs)* each one of which is either a least cost path or a non-least-cost-edge (i.e. a single edge which is not the least cost connection between its vertices). Every two consecutive BPC's are

separated by a *splitVertex*.

4. **Statistical analysis** of route structural characteristics.

This paper focuses on the third and fourth steps in the process. First, the problem of route choice is sketched in order to show how the size of the minimum path decomposition can be integrated in the existing models. After this motivation, the concept of splitting routes into basic path components is introduced in section 8.5. Definitions are given, the concept is elaborated in a mathematical way, theorems on route splitting are proved and the used algorithm is explained. Section 8.6 is devoted to the interpretation of the detailed route splitting results and their relevancy in travel behavior research. It also formulates research questions that can be solved using the results generated by the algorithm.

Finally, the paper reports the results extracted from the available datasets. A conclusion is presented in section 8.9.

8.3 Route Selection Context

Route selection induces a complex discrete choice problem and in general consists of two parts: a route *choice set generator* and a route *choice model*. Prato (2009) provides a comprehensive overview of solutions to the route choice problem. The traveler is assumed to select an optimal route according to personal preferences while having limited information and limited processing capacity. When predicting routes for network loading simulation, either collective of individual choice sets need to be generated for each origin-destination pair. The set of possible routes is huge and the traveler never has a mental representation of the complete set. Furthermore, the choice sets considered by the traveler and the researcher are not necessarily identical. Hence, in the route choice problem, multiple route sets are considered. Several similar schemes have been proposed to classify those sets and the one given in Kaplan and Prato (2010) is shown below: each set is derived from the one mentioned immediately above it.

	Traveler view	Researcher view
Set of all possible routes for OD-pair	Universal set	Universal set
Set of routes feasible based on a given criterion	Master set	Awareness set
Set of routes from which the individual would select	Consideration set	Viable set

For both model estimation and route prediction purposes, the consideration set is constructed algorithmically, in general by making use of non-compensatory techniques Bovy (2009). The consideration set is used in route choice models most of which are derived from MNL (multinomial logit).

Several studies investigate the quality of the choice set generators and of the overall process as well as the mutual influence of choice set generation and choice model estimation. However the effect of including or excluding a given path characteristic is not documented.

Bekhor et al. (2006) evaluate sixteen label minimization/maximization algorithms along with K-shortest path selection, link elimination, link penalty and stochastic link impedance based methods. Choice sets are generated for data collected from MIT researchers and choice models are estimated. The models are based on distance, free-flow travel time, some landmarks, income indicators and time spent on government numbered routes.

The relevancy of the composition of the choice set is investigated by Prato and Bekhor (2007). The authors investigate the effect of the choice set generation technique on the choice model parameter estimates and on the generated predictions. The study creates two choice sets (one generated by the branch-and-bound technique and one that merges results from labeling, link elimination, link penalty and stochastic link trait adaptations). Six choice models are estimated using each choice set. All models use ten explanatory variables. The effect of combining choice set generation techniques and choice models is investigated. The branch-and-bound generator uses fixed settings for the criteria used. The effect of the criteria is not investigated, probably because the branch-and-bound technique already outperforms the other ones with respect to *coverage*.

Prato (2012) performs a meta analysis of the effect of the choice set generation technique on the accuracy of the choice model estimates and on the link flow predictions. Deterministic (K-shortest path, link penalties, branch-and-bound) and stochastic (link impedance, combination of link impedance and travel taste, random walk biased to search for the shortest path) are considered. The choice model used is the

path size correction (PSC) model. The same techniques are used to generate synthetic data and to generate objective choice sets and choice models. This is done for several parameter sets and in pairwise combinations. The choice models are restricted to account for route length, number of speed bumps, number of turns and the path size correction. The only investigated route generator that can include path attributes, is the branch-and-bound technique. Several settings for thresholds are investigated.

Since the consideration set is not observable, it can be argued that the parameters of the consideration set construction model and the choice model shall be estimated together. This is done by Kaplan and Prato (2010). The consideration set C_n for traveler n is derived from the master set using a *conjunctive heuristic semi-compensatory model*: the probability to find consideration set C_n is given by the probability than respondent n uses a specific set of thresholds for the independent values. If an independent variable is out of range w.r.t. a threshold, the corresponding route is not considered. The thresholds are unknown in advance and they are estimated using a maximum likelihood method. The authors only consider the route length and the number of turns.

Consideration set construction is the point where the research presented in this paper fits. The feasibility of a route for inclusion in the consideration set is assessed using several attributes (detour factor, number of left turns and others): we propose to additionally include the size of the minimum path decomposition in the assessment. The idea applies to both choice set generation an choice models (explanatory variables). It allows (i) to avoid overly circuitous routes and (ii) to avoid introducing unrealistic bias towards the shortest paths.

Quality assessment using the minimum path decomposition size can be used (i) in the consideration set construction stage (ii) or as a posterior assessment of the generated set. In both cases distributions for the minimum decomposition size extracted from recorded routes are used. They can be collective or individual; the latter applies to the case where sufficient longitudinal data for each participant are available.

Consideration set construction is discussed by several researchers. For the purpose of this paper the reported research efforts are subdivided in two categories according to how the proposed assessment can be integrated.

8.3.1 Assessment in Choice Set Construction Stage.

The research reported in the following papers allows to use additional route attributes in the consideration set construction stage.

Zijpp and Catalano (2005) present the Constrained K-Shortest Paths (CKSP) technique based on Lawler's algorithm. In case a constraint can be evaluated on the first part of a partially generated path only, a part of the search space can be discarded if the constraint is not met. The CKSP method is based on consecutive least cost path evaluations and the new *minimum path decomposition size* algorithm can be integrated with limited effort. If an upper bound for the minimum decomposition size is specified in advance, the CKSP method can discard subspaces that would deliver overly complicated paths.

Prato and Bekhor (2006) present a deterministic path generator using a branch-and-bound (breadth-first-search) technique that constructs a connection tree of candidate paths between a given origin and destination. Each time a link is to be added several constraints are verified. Given constant factors are used for partial path assessment: (i) a distance factor filters partial trips moving back to the origin, (ii) a time factor filters partial paths taking too long, (iii) a detour factor limiting the length of partial paths relative to the minimal distance between their endpoints and (iv) a similarity constraint that limits overlap between candidates. The generator is applied to a network for the city of Torino, Italy. Commuting trips were recorded using a web-tool. The *branching rules* account for behavioral constraints. The authors evaluate the quality of the choice set using the concept of *coverage*. The authors compare several generation techniques and reports that *coverage* levels attained by branch-and-bound techniques are much higher than for other techniques. Finally, several choice models are estimated using the generated choice sets. The results of our research can be used in branch-and-bound rules. The size of the minimum decomposition can be calculated for the head part each path being built; the complexity is of the same order as the evaluation of the *loop constraint* mentioned by the authors. The distribution for the minimum decomposition size extracted from GPS traces can be used to determine threshold values.

Schüssler et al. (2010) mention the difficulty of avoiding bias when establishing route choice sets for high resolution networks: either behaviorally advanced choice set generation procedures are required or large sets of routes need to be explored and reduced by considering attractivity, plausibility and similarity between routes. The authors present the Breadth First Search - Link Elimination (BFS-LE) algorithm suitable for route set generation in high density networks. The minimum decomposition size of the path can be used as a selection criterion in the mentioned reduction phase.

Finally, Pillat et al. (2011) investigate how path assessment can be based on thresholds acquired from GPS recordings. The authors describe an experiment where

a route choice set is generated and compared with trips recorded from GPS traces. Preferred routes were collected using a survey and saved in a database by identifying consecutive junctions on a map. From those data, the maximum detour factor as a function of the travel time is derived. In the choice set generation phase, the detour factor changes as the duration of the partially generated path grows. The resulting generated paths are compared to trips derived map-matched GPS traces.

In this paper we propose to use the information extracted from the GPS data to define the selection criteria for use during route generation instead of using it for verification only.

8.3.2 Posterior Assessment of the Generated Choice Set.

In some cases it is not possible to include minimum decomposition size verification in the route generation stage. In such cases, posterior assessment of consideration sets can be applied to the generated results. E.g. the distribution for the minimum decomposition size found in MATSim generated routes can be validated with the one found in GPS traces.

MATSim finds the optimal route for each traveler by micro-simulation. The movements of cars crossing links on the road network are simulated. The time to cross each link depends on the link characteristics and on the link occupation by other cars. Travelers execute their daily plan and derive the time to travel from the network. The plan gets an evaluation score at the end of the day and, under certain conditions, a new one is generated. The basic assumption is that individuals always try to minimize their travel cost (Balmer et al. (2009)) by finding new routes and by changing their departure times. No route choice set is to be determined a priori but built and maintained by the genetic algorithm as an integral part of the plan (agenda).

8.4 Research Objective

Each path can be split into least cost subpaths in several ways. Each path has one or more minimum decompositions i.e. decompositions consisting of the smallest set of least cost paths from which the path can be reconstructed by concatenation. In other words, each path clearly has a smallest set of intermediate destinations which were visited by the traveler and each subpath connecting two consecutive intermediate destinations, is a least cost path. The traveler is assumed to have some intermediate destinations in mind and to hop from one to the other using least cost paths. The traveler does not stop in the intermediate locations but merely uses them as anchor

points to construct a route.

The size of the minimum decomposition is a structural qualifier for the path that can be used in the consideration set generation as mentioned in section 8.3.

The research described in this paper focuses on splitting of routes into basic components (BPC's), i.e. partitioning a route into subpaths, each of which is of minimal cost. We investigate the following:

Hypothesis 8.4.1. *In utilitarian trips, individuals tend to construct their route as a concatenation of a small number of minimal cost routes i.e.* basic path components *(BPC).*

In a utilitarian trip, the destination differs from the origin and the traveler moves in order to perform a planned activity at the destination location. Utilitarian trips represent travel for a given purpose.

The individual is thought to make use of a small set of preferential locations between the origin and destination and to travel in the most efficient way between those intermediate locations. In order for route choice sets to be realistic, the distribution of the minimum decomposition size shall reflect the one found in recorded traces.

Generalized path traversal cost is approximated by considering individual link travel cost only. Node traversal cost (e.g. *left turn* cost) is associated with a pairs of links and is not covered by the algorithms presented in this research.

We aim to investigate the characteristics of a large set of GPS-recorded and map matched routes. We are interested in the distribution of the minimal number of BPC's in each route. Furthermore, it is not known in advance *why* particular intermediate locations in non-least-cost paths are chosen. Therefore, we want to analyze the use frequency of nodes as split nodes in the routes in order to verify the hypothesis that some nodes are preferential trip splitters due to traffic related characteristics of the network (like availability of traffic lights).

The research reported in this paper focuses on (i) the minimal BPC's that constitute the path chosen and (ii) the resulting sets of *splitVertex* candidates. In section 8.5.2 it is shown that these sets are subpaths of the given path: they are called *splitVertexSuite*s.

8.5 The main graph-theoretical Algorithms and Proofs

8.5.1 Minimal Splitting of Routes into Basic Path Components

We begin with some definitions from graph-theory. Let $G = (V, E)$ be a directed graph with vertex set V and edge set E. The vertices correspond to *nodes* in a road network, and the edges correspond to *links* in the network. Each edge e has a nonnegative *cost* $c(e)$ which is the effort (e.g. time or money) required to traverse the link in the network. For a subgraph $H \subseteq G$, $V(H)(E(H))$ denote the set of vertices (edges) of H.

Definition 8.5.1 (walk, initial, terminal, internal vertices, internally-disjoint)**.** *A* walk *is a sequence of vertices $P = (v_0, v_1, \ldots, v_l)$, not necessarily distinct, where $(v_i, v_{i+1}) \in E(G)$ for all $i = 0, 1, \ldots, l-1$. Vertices v_0 and v_l are called* initial *and* terminal *vertices, respectively, of P, and vertices $v_1, \ldots, v_{l-1}$ are called* internal vertices *of P. The walk P is said to be* connecting *v_0 and v_l, and it is also denoted by $P(v_0, v_l)$. A walk $Q(v_0, v_l)$, is* internally-disjoint *from P if all the internal vertices of Q, are distinct from the vertices in P.*

Definition 8.5.2 (path, subpath, size, cost, least cost distance)**.** *A* path *is a walk where all its vertices are distinct. For a path $P = (v_0, v_1, \ldots, v_l)$, any subsequence of vertices $v_i, v_{i+1}, \ldots, v_j$, where $0 \leq i, \leq j \leq l$ is a* subpath *of P, and is denoted by $P(v_i, v_j)$. The* size *of a path, denoted by $|P|$, is the number of edges in it (i.e. l), and the* cost *of a path, denoted by $c(P)$ is the sum of the costs of its edges. The* least cost distance *between u and v, denoted by $lc(u, v)$ is the cost of the least cost path connecting u and v.*

We remark that if $c(e) = 1$ for all $e \in E$ then the cost of a path coincides with its size and that vertex traversal cost is assumed to be zero.

The following lemma is easy to prove:

Lemma 8.5.3. *If $P = (v_0, v_1, \ldots, v_l)$ is a least cost path, then any subpath of P is also a least cost path.*

Proof 8.5.1. *Assume, by contradiction, that $P(v_i, v_j)$ is not a least cost path between v_i and v_j, for some $0 \leq i < j \leq l$. Let $Q(v_i, v_j)$ be a path of smaller cost. Then by replacing $P(v_i, v_j)$ by $Q(v_i, v_j)$ we get a walk connecting between v_0 and v_l of smaller cost than P. This walk contains a path connecting v_0 and v_l of smaller cost than P, since we assumed that the cost function is non-negative. This contradicts the fact that P is a least cost path.*

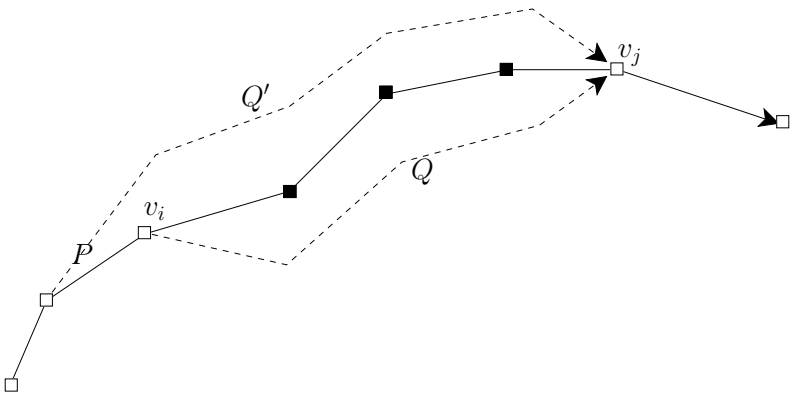


Figure 8.1: A path P with a $P(v_i, v_j)$-shortcut. v_i is a fork vertex and v_j is a join vertex

The converse of Lemma 8.5.3 is false since it is possible that all the subpaths of $P(v_0, \ldots, v_l)$ (except P itself) are least cost paths, but P is not a least cost path connecting v_0 and v_l and there is another least cost path Q connecting v_0 and v_l. This fact motivates the following definition:

Definition 8.5.4 (P- shortcut, fork and join vertices)**.** *Let $P = (v_0, v_1, \ldots, v_l)$ be a given path. A $P(v_i, v_j)$-*shortcut *(or for brevity, P -* shortcut*, or* shortcut*), is a path $Q(v_i, v_j)$, internally- disjoint from P, where $v_i, v_j \in V(P)$, such that $c(Q(v_i, v_j)) < c(P(v_i, v_j))$. The vertices v_i and v_j are called* fork *and* join *of the shortcut, respectively. (See Figure 9.1).*

By Lemma 8.5.3 a least cost path cannot have any shortcuts.

Assume $e = (u, v)$ is an edge in P whose cost is larger than the least cost path connecting u and v. Then e is called a *non-shortest-edge* .

Definition 8.5.5 (Basic Path Component (BPC))**.** *Given a path P, a subpath of P is called a* Basic Path Component*, or for short, a BPC, if it is either a least cost path connecting its endpoints, or P consists of a single non-shortest-edge.*

Claim 8.5.6 (path splitting)**.** *Let $P = (v_0, v_1, \ldots, v_l)$ be any path connecting v_0 and v_l. Then $E(P)$ can be partitioned into BPC's.*

Proof 8.5.2. *The trivial partition into edges $(v_0, v_1), (v_1, v_2), \ldots, (v_{l-1}, v_l)$ is an example of such a partition.*

There are many ways to do path splitting, the trivial partition is among them. We are interested in finding a path splitting with a *minimum number* of basic path

components. Such a path splitting is called *minimum path splitting*. Each non-shortest-edge is a part in each minimum path splitting since it constitutes a BPC. If we remove the set of non-shortest edges in a path (each of which is a BPC), we are left with a set of disjoint paths, each of which contains no non-shortest-edges.

From now on we will assume that P does not contain any non-shortest-edges.

We will address the following problem:

Problem 8.5.7 (minimum path splitting)**.** *Given a path $P = (v_0, v_1, \ldots, v_l)$ with origin v_0 and destination v_l, and assume P does not contain any non-shortest-edges. Find efficiently a minimum path splitting of P .*

A vertex separating two consecutive BPC's is called a *splitVertex*. To solve problem 8.5.7 we note that a minimum path splitting will contain a minimum number of splitVertices (since, by definition, any two consecutive basic path components are separated by a splitVertex). Hence, an equivalent formulation of Problem 8.5.7 would be to find a minimum number of splitVertices in P, denoted by $v^s_{i_1}, v^s_{i_2}, \ldots v^s_{i_k}$, such that any subpath connecting consecutive splitVertices will be *least cost*.

Lemma 8.5.8. *Let $P = (v_0, v_1, \ldots, v_l)$ be a path connecting v_0 and v_l. Assume P is not least cost, and let $Q(v_i, v_j)$ be a shortcut in P. Then any path splitting of P will contain at least one internal vertex in the path segment $P(v_i, v_j)$.*

Proof 8.5.3. *By contradiction. If no internal vertex in $P(v_i, v_j)$ is a splitVertex, then $P(v_i, v_j)$ is a least cost path, contrary to the fact that $Q(v_i, v_j)$ is a shortcut in P.*

We are now ready to describe an efficient algorithm for partitioning a given path P into a minimum number of basic path components. The algorithm begins with the initial vertex of P, v_0, and finds a maximal least cost path beginning with it. This is done using Dijkstra's least cost path algorithm (Dijkstra (1959)). Assume v_{j_1} is the first vertex on P for which $P(v_0, v_{j_1})$ is not least cost, then the algorithm marks v_{j_1} as a join vertex and continues with the subpath of P beginning from the vertex prior to v_{j_1} on P, looking for the next join vertex in $P(v_{j_1-1}, v_l)$. We continue until no more join vertices are found. The pseudo code is given below. See also Figure 8.2.

Theorem 8.5.9. *Algorithm 8.5.1 finds a partition of path P into a minimum number of BPC's.*

Proof 8.5.4. *Given the output of the algorithm, we will partition the given path P into exactly k subpaths . Define the splitVertices to be the vertices preceding the*

Algorithm 8.5.1 Algorithm for finding a partition of a path into BPC's

Input Graph G, edge costs c, $P = (v_0, v_1, \ldots, v_l)$ containing no non-shortest edges
$start \leftarrow 0$;
$k \leftarrow 1$
while ($P(v_{start}, v_l)$ is not a least cost path) **do**
  Find the first vertex v_{j_k} in $P(v_{start}, v_l)$ such that $lc(v_{start}, v_{j_k}) < c(P(v_{start}, v_{j_k}))$
  $start \leftarrow j_k - 1$
  $k \leftarrow k + 1$
end while
return join vertices $v_{j_1}, v_{j_2}, \ldots, v_{j_{k-1}}$

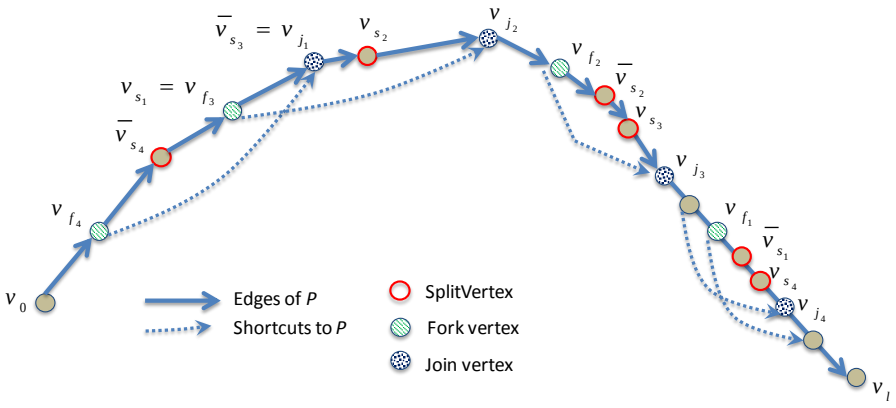


Figure 8.2: A path P with join and fork vertices and basic path components.

join vertices on P, i.e. $v_{s_i} = v_{j_i - 1}$ for $1 \leq i \leq k-1$. Each subpath begins in a splitVertex and ends in the next splitVertex, except for the first subpath which begins in v_0 and the last subpath which ends in v_l. In other words, the subpaths are: $P(v_0, v_{s_1}), P(v_{s_1}, v_{s_2}), P(v_{s_2}, v_{s_3}) \ldots, P(v_{s_{k-1}}, v_l)$. By the algorithm, it is clear that each of these subpaths are minimum cost, and hence are BPC's. We are now left to prove that no other partition exists with fewer than k BPC's. Assume, by contradiction, that such a partition exists, with $k-1$ BPC named $\overline{P_1}, \overline{P_2}, \ldots, \overline{P_{k-1}}$. Then, by the pigeon hole principle, at least one BPC, say $\overline{P_i}$ contains at least one splitVertex of P, say v_{s_t} as an internal vertex of $\overline{P_i}$ and $v_{s_{t-1}}$ (or v_0 when $t = 1$) is also included in $\overline{P_i}$. This implies that the join vertex v_{j_t} is included in $\overline{P_i}$, and hence $\overline{P_i}$ is not a least cost path, which is a contradiction.

We have described an algorithm of splitting a path into a minimum number of

BPC's, and finding splitVertices that separate between these basic path components. The algorithm is efficient since it uses Dijkstra's algorithm no more than k times, where k is the minimum number of BPC's used to partition P. Two natural questions arise here; is the partition into a minimum number of BPC's unique, and are the splitVertices unique? Since the splitVertices may denote some point of interest for the traveler (otherwise a minimum cost path would have been chosen), we are interested to find efficiently other splitVertices and partitions into BPC's.

We use Algorithm 8.5.2 to find another partition of P into BPC's, starting from the end of the path, and a collection of fork vertices in the given path P. This is done by 'going backwards' from v_l to v_{l-1} etc. and finding the first vertex v_{f_1} such that the subpath $P(v_{f_1}, v_l)$ is not a least cost path, but $P(v_{f_1+1}, v_l)$ is a least cost path. The algorithm to produce fork vertices is similar to Algorithm 8.5.1 , except that we run it on the "reverse graph" of G, obtained by reversing all the edges in G. Note that when all the edges in G are reversed, the first vertex v_0 of the " reversed path " $\overleftarrow{P}$ corresponds to the last vertex v_l of the original path P. We include the algorithm for completeness.

Algorithm 8.5.2 Algorithm for finding a partition of a path into BPC's and fork vertices.

Input Graph G, edge costs c, P containing no non-shortest edges
Reverse the edges in G; Assume the reversed path is $\overleftarrow{P} = (v_l, v_{l-1}, \ldots, v_0)$
$start \leftarrow l$;
$k \leftarrow 1$
while ($\overleftarrow{P}(v_{start}, v_0)$ is not a least cost path) **do**
  Find the first vertex v_{f_k} in $\overleftarrow{P}(v_{start}, v_0)$ such that $lc(v_{start}, v_{f_k}) < c(\overleftarrow{P}(v_{start}, v_{f_k}))$
  $start \leftarrow f_k + 1$
  $k \leftarrow k + 1$
end while
return fork vertices $v_{f_1}, v_{f_2}, \ldots, v_{f_{k-1}}$

Now define another set of splitVertices $\overline{v_{s_i}} = v_{f_i+1}$ for $1 \leq i \leq k-1$, which are the vertices following the fork vertices on P found in Algorithm 8.5.2. It is easy to see, as in the proof of Theorem 8.5.9, that the k subpaths $P(v_0, \overline{v_{s_{k-1}}}), P(\overline{v_{s_{k-1}}}, \overline{v_{s_{k-2}}}), \ldots, P(\overline{v_{s_1}}, v_l)$ are all BPC's, and hence are a minimum partition into BPC's. (See Figure 8.2).

8.5.2 Selecting SplitVertices for minimum Path Splitting

We have seen in section 8.5.1 that there are at least two sets of splitVertices which break up the given path P into k BPC's. One set was obtained by looking for maximal shortest paths starting from the beginning of P, these vertices were labeled by $v_{s_1}, v_{s_2} \ldots, v_{s_{k-1}}$, and the other was obtained by looking at maximal shortest paths starting from the end vertex of P. These were labeled as $\overline{v_{s_1}}, \overline{v_{s_2}}, \ldots, \overline{v_{s_{k-1}}}$. Each such set breaks up P into exactly k BPC's. Denote by S_i the sequence of consecutive vertices on P in which $\overline{v_{s_{k-i}}}$ is the first one and v_{s_i} is the last one, for each $1 \leq i \leq k-1$. We call each such sequence *splitVertexSuite (SVS)*. Then any partition of P into a minimum number of BPC's is obtained by choosing a unique splitVertex from each splitVertexSuite S_i. Hence the number of ways of splitting a path into a minimum number of BPC's, N_P is bounded above by

$$N_P \leq \prod_{1 \leq i \leq k-1} |S_i| \tag{8.1}$$

The following example (see figure 8.3) shows why in some cases, this is a strict upper bound, i.e. the actual number of partitioning P into a minimum number of BPC's is smaller than this bound. In figure 8.3 our algorithms discover the join vertices v_7 and v_{10} and fork vertices v_6 and v_1. The SVS's are $S_1 = \{\overline{v_{s_2}} = v_2, \ldots, v_{s_1} = v_6\}$ and $S_2 = \{\overline{v_{s_1}} = v_7, \ldots, v_{s_2} = v_9\}$. However, if for example, we choose the split vertices v_3 and v_9 then we do not break up P into 3 BPC's since the middle subpath, $P(v_3, \ldots, v_9)$ contains a shortcut between v_5 and v_8, which was not discovered by our algorithm, and hence is not a BPC. In fact, in this example any splitting of P into 3 BPC's must not avoid the vertices v_6 or v_7 . We conclude that formula 8.1 is an upper bound for the number of ways to split a path into a minimum number of BPC's. In the sequel paper we will show how to avoid this problem of an "*invisible shortcut*" and compute precisely the number of ways to split a path into a minimum number of Basic Path Components.

8.6 Interpretation of Route Decomposition

When splitting traveled routes, the *splitVertexSuites* can easily be determined. However it is not known which of the possible splits the user had in mind while traveling: i.e. the researcher cannot derive solely from the decomposition which of the *splitVertices* were relevant to the traveler. It is reasonable to assume that at least one *splitVertex* in each *splitVertexSuite* or one of its incident edges belonging to the

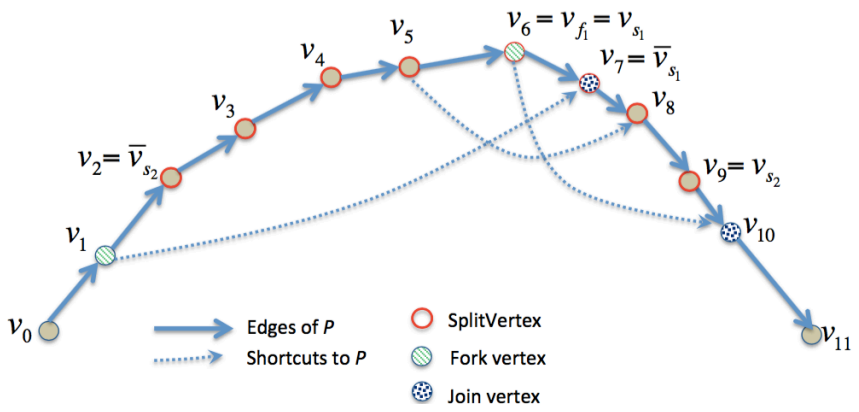


Figure 8.3: A path P and an "invisible" shortcut $(v_5 \rightarrow v_8)$ to P.

path, has a special meaning to the traveler; otherwise, there would be no reason for a rational traveler to visit the subpath determined by the *splitVertexSuite*. On the other hand, the *splitVertexSuites* deliver the sets of route nodes to be investigated for verification of transportation related characteristics.

Extracting information from large sets of recorded traces by route splitting, delivers input for the route selection models used in the network loading phase of travel demand simulators. The analyst is interested in probability distributions for (i) the minimum number of BPC's, (ii) the number of possible splittings and (iii) the visit frequency for *splitVertices*

8.6.1 *SplitVertexSuite*s and Traveler Intentions.

The path P chosen by the traveler can be expressed as a minimal concatenation of BPC's but the minimum decomposition in general is not unique. The chosen path has been recorded, so the path as well as the *splitVertexSuite*s constitute *revealed evidence*. However, the actual decomposition (i.e. the sequence of specifically chosen *splitVertices*) is not revealed by the traveler.

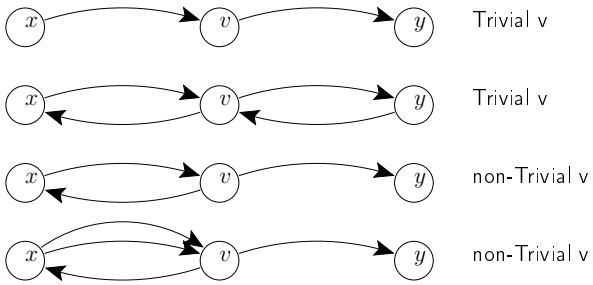


Figure 8.4: Examples of trivial and non-trivial nodes having exactly two neighbors in the road network.

8.6.1.1 *SplitVertexSuite* Size as Uncertainty Measure

For a given path P, each vertex in a *splitVertexSuite* or the edges they belong to, could have been the motivation for the selection of the path. If a *splitVertexSuite* has cardinality larger than one, we need to assume that each of its *splitVertices* can be the boundary between two BPC that constitute a least cost path. Hence, the *splitVertices* belonging to a given *splitVertexSuite* cannot be distinguished with respect to their meaning to the traveler. The size of a *splitVertexSuite* is a measure for the uncertainty about the motivation of the traveler for the detour represented by that *splitVertexSuite*.

The number of possible splittings of a path reflects the inherent lack of information uncovered by the recorded trace about the traveler's intentions. For this reason we say that the value of N_P given by inequality (8.1) is a measure for the uncertainty about the set of road network nodes motivating the selected route.

Note that a non-least-cost-edge BPC always results in two singleton *splitVertexSuite*s and hence does not induce any uncertainty.

8.6.1.2 Need for Network Normalization

In order to calculate the uncertainty value, an additional *network normalization* step is required. In road networks, both directed (one-way) and undirected (bidirectional) links co-exist. Therefore, each road network is modeled by a digraph. The digraph is verified in advance not to contain any sources or sinks (to assess data quality). Parallel edges (i.e. two different edges sharing a single ordered pair $\langle v_a, v_b \rangle$ of vertices) are allowed in graphs representing road networks.

Some vertices have exactly two neighbors. If each of the two neighbors for vertex v is connected to v by at most one edge in each direction and if $indegree(v) =$

$outdegree(v)$, then v is called a *trivial vertex* (see Figure 8.4 for examples). Formally, let $E(a,b)$ denote the set of edges associated with the vertex pair $\langle a,b\rangle$ and let $N(v)$ denote the set of neighbors for v. Then v is a trivial vertex if and only if

$$\big(|N(v)|=2\big)\wedge$$
$$\big(\forall w\in N(v):|E(v,w)|\leq 1\wedge|E(w,v)|\leq 1\wedge indegree(v)=outdegree(v)\big) \quad (8.2)$$

In digital maps, trivial vertices (nodes) are used to subdivide road segments into parts at locations where a road segment attribute value changes (e.g. speed limit, number of lanes, municipality name, road owner, pavement type ...). Those *changes* by themselves are assumed to be irrelevant to the traveler while selecting a route. It is however obvious that an *aggregated* measure (e.g. the fraction of the road length having good quality pavement) can be a reason to select a particular link.

In a chain of trivial vertices, either all of which belong to the same *splitVertexSuite* or none is a *splitVertex* since none of them can be a fork or a join vertex. This observation allows for network normalization by *link contraction* which removes all trivial vertices. Network normalization does not change the number of *splitVertexSuite*s neither affects the set of *splitVertices* since trivial vertices cannot be *splitVertices*. It leads to an unambiguous value for the uncertainty about the *splitVertex* selection. The need for normalization is made clear by Figure 8.5.

As a consequence, we add the process of network normalization after *map matching* and before *route splitting*.

8.6.2 Frequency Distributions to feed Simulators

We will extract the following distributions for various quantities used in route choice set generation:

1. *Distributions of the minimum number of basic path components* are essential to route generation. The distribution derived from the complete set of traces uncovers characteristics for the complete observed population. Similar distributions derived for specific individuals may reveal the existence of several behavioral categories when sufficient longitudinal data are available.

2. The use frequency of a vertex as a *splitVertex* at the *population* level is a measure for the attractiveness of the vertex or its incident edges. This provides additional input to the route generator when applied to the region where the traces have been captured. It allows to automatically identify specific spots on the network that serve as *way points* for route generation methods. Analyzing

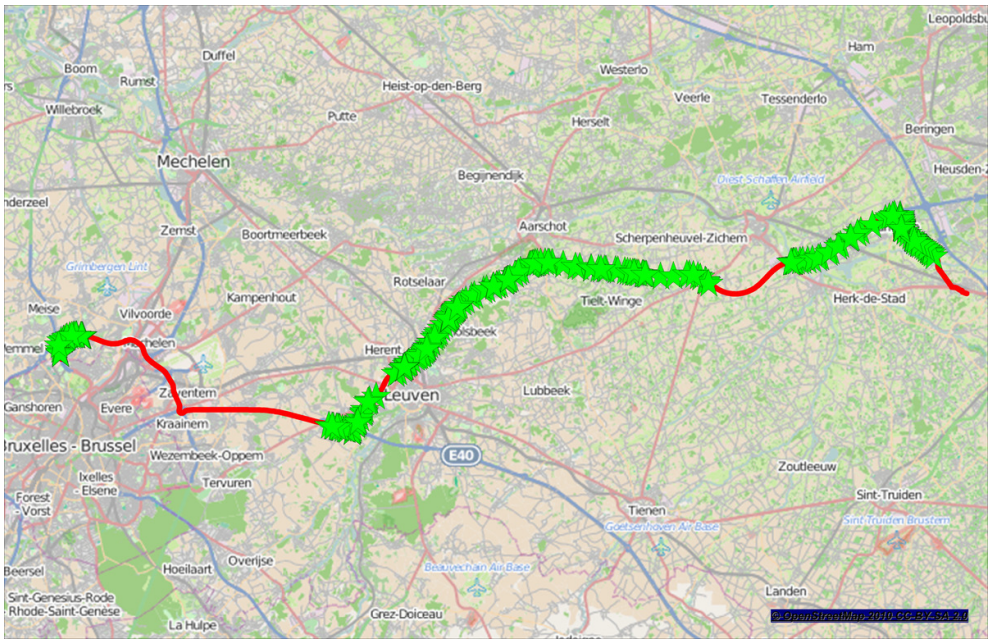


Figure 8.5: The largest part of the route (from Herk-de-Stad (right) to Kraainem (left)) makes use of the E314 and E40 highways. *SplitVertices* (represented by green star symbols) were determined using the raw (non-normalized) OSM network. The diagram shows the need for network normalization.

the use frequency of road network nodes as *splitVertex* and the links they delimit, will elicit *network* characteristics (as opposed to the minimum size of a decomposition which is a *trip* characteristic). Frequently used *splitVertices* are interpreted as *route attractors*. The question is whether they can be associated with specific elements in the road network (highway entry/exit ramps, traffic lights, tunnel) or with specific (types of) POI (point of interest) like a school, public transportation station, carpool parking etc. This analysis requires a much larger data set than the one that was available for research reported here.

3. The use frequency of a vertex as a *splitVertex* at the level of an *individual* may uncover short intentional stops (bring/get, pick/drop activities) that are not discovered by the trip detector because of their short duration. Current trajectory annotation literature (e.g. Giannotti et al. (2007), Zheng et al. (2009), Kuijpers et al. (2009), Spinsanti et al. (2010), Alvares et al. (2007), Andrienko et al. (2011), Furletti et al. (2013)) focuses on stops found in GPS traces. From mobility science point of view, it is also relevant to annotate (i.e. to attach a meaning to) *splitVertexSuites*. Annotating *splitVertices* is expected to be more complex than annotating stop locations because of the uncertainty mentioned above.

8.7 Data Preparation Steps

In order to investigate Hypothesis 8.4.1, a large set of GPS trajectories has been analyzed.

8.7.1 Belgian Person Traces

A set of 999 GPS traces recorded during the period 2006-2008 using a PDA have been analyzed. People took the PDA with them: the result is a set of *person* traces as opposed to *car* traces often used. Person traces contain more information but are more expensive to collect over a long period and are sensitive to omissions because people can forget to take the device with them. GPS recording frequency was 1[Hz].

8.7.1.1 Processing Method 1 : IMOB tools + OpenStreetMap Network

1. Trip detection was performed by finding recording gaps and by analyzing speed variations in sequences of GPS recordings. The detected trips can have several

modes (e.g. car-train-walk) but mode detection was not performed (although walking and biking are quite easily identified).

2. Some trips have been detected to start/end at a petrol station located near a highway as a result of the threshold values used for *stop detection*. Those were not altered because refueling can be considered to be a shopping activity.

3. OpenStreetMap (http://www.osm.org/) was used to extract a road network for Flanders (Belgium). The network has 479920 links and 372608 nodes. Trips have been *map-matched* onto that network. The map matching step is crucial. Some map matchers try to fill (small) gaps in the recording by assuming that the traveler moved along a least cost path (according to some criterion). This shall not be done in this research because the hypothesis to be tested shall not be influenced by hypotheses used while map matching. Because of the high recording frequency, it can be expected that every road segment used by the traveler is selected by at least one GPS point. Traces of this kind are called *high density recordings*. Use of high density recording was essential to the reported research. Furthermore, map matching high density recordings can benefit from topological information available from the network. Making use of that information makes the matching process efficient. The map matcher for high density recording described by Knapen et al. (2015a) was used.

   Trip detection and successive map matching resulted in 13098 cases.

8.7.1.2 Processing Method 2 : Fraunhofer Tools + Navteq Network

The set of recordings used in section 8.7.1.1 has been map-patched to the Navteq network which was not normalized. This network consists of 903.217 links and 748.705 nodes for Belgium. The software used to perform the map-matching was written by Fraunhofer IAIS. In general the process used within this map matching software consists of two steps.

In the first step a data preparation is done. This includes the detection of outliers and standstills. GPS points falling in at least one of both categories were excluded from the map matching. In addition the time gaps M [min] between two consecutively logged GPS points, were calculated.

In the second step the map matching process is conducted. The goal is to connect each GPS point to exactly one Navteq street segment. Here the software uses a combination of a geometrical and topological map matching. This means that the software uses both, the spatial distance of the GPS points to the Navteq street net-

work, and the information of the topological connections in combination with driving restrictions, to assign a GPS point to a street segment.

In addition small gaps within the GPS traces were closed by a shortest path routing. Here we worked with three different time gaps (M = 1.0, 2.5 and 5.0[min]). When M grows, the number of detected trips decreases and their average size (distance, number of road network links used) grows. Furthermore, the probability that a reported trip is not a simple path but a walk, grows because small movements are combined to a single trip.

8.7.2 Italian Car Traces

A set of car trajectories recorded in the region of Milano, Italy was processed using the Fraunhofer IAIS map matcher software and the Navteq network (Processing Method 2). The numbers of nodes and links for Italy are nearly ten times larger than the corresponding values for Belgium.

8.8 Analysis Results

In all experiments described below, *distance along the road* (as opposed to travel time) was used as the generalized cost value to travel a road segment.

8.8.1 Examples of Discovered *splitVertexSuites*

Sample routes extracted from the recorded datasets are shown in following figures in order to grasp the idea of the observed *splitVertexSuites*. All diagrams have been generated using the Navteq network.

Figure 8.6 shows a long trip (about 80[km]) consisting of a single least cost path (i.e. without any *splitVertexSuite*).

Figure 8.7 shows a trip of about 16[km] consisting of three basic components. The large *splitVertexSuite* near Sint-Truiden coincides with a segment of an arterial *express road* which runs parallel to the straight line north-west to it. Higher speed is allowed on the arterial road but the distance is longer. The large size of the *splitVertexSuite* again shows the need for network normalization.

Figure 8.8 shows part of a route starting at the right hand side, visiting the center of the city of Geel, then moving around the city in clockwise direction, heading to the north and finally arriving near the center of the city of Mol. The partial route shows 11 *splitVertexSuite*s and Figure 8.9 shows the lower-left part of the same route. The

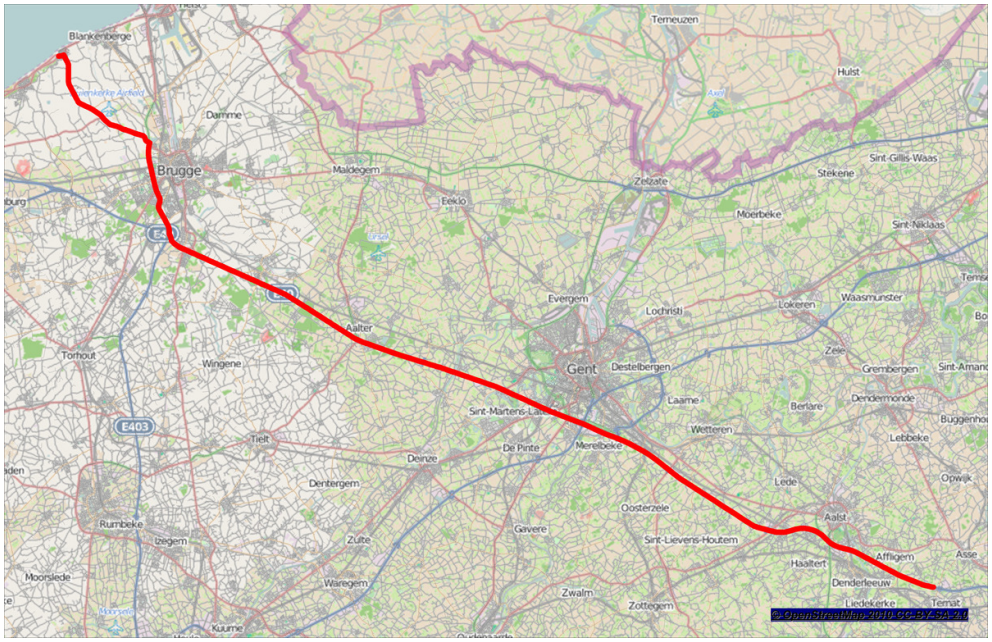


Figure 8.6: Route consisting of a single basic component (no *splitVertexSuite*s).

lower-right *splitVertexSuite* in Figure 8.9 suggests that a particular street in the city center was an intermediate destination. The lower-left *splitVertexSuite* suggests the intentional use of the ring way and/or a specific junction.

Figure 8.10 shows a route of about 4 kilometers having 7 *splitVertexSuite*s first visiting something special at the first *splitVertexSuite* and then avoiding the narrow streets in a residential area up to the 4-th *splitVertexSuite* which represents a location equipped with traffic lights. The arterial road is used up to the 5-th *splitVertexSuite* which also contains a junction equipped with traffic lights. The trip ends near the parking of a shopping center. The 7-th *splitVertexSuite* is the upper-right one in Figure 8.11. It is an artifact caused by the fact that the street labeled *Van Groesbeekstraat* (south of the *splitVertexSuite*) constructed in 2012-2013 did not exist at the time of trajectory recording (between 2006 and 2008).

8.8.2 Distributions for the Size of the Splits

Figure 8.12 shows the absolute and relative distributions for the number of *basicComponents* per trip for all cases.

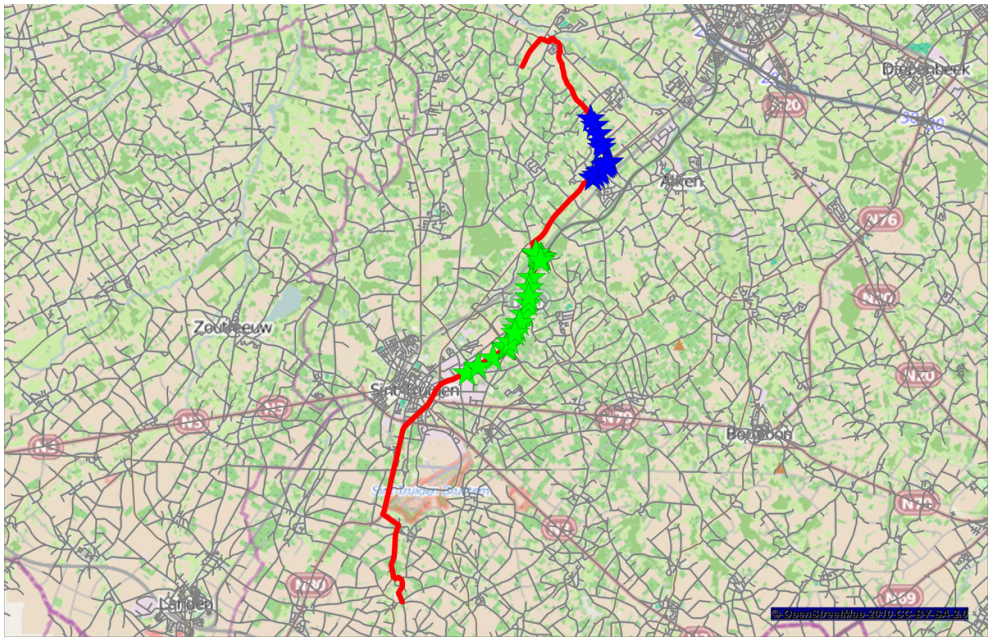


Figure 8.7: Route consisting of three basic components and showing the need for network normalization.

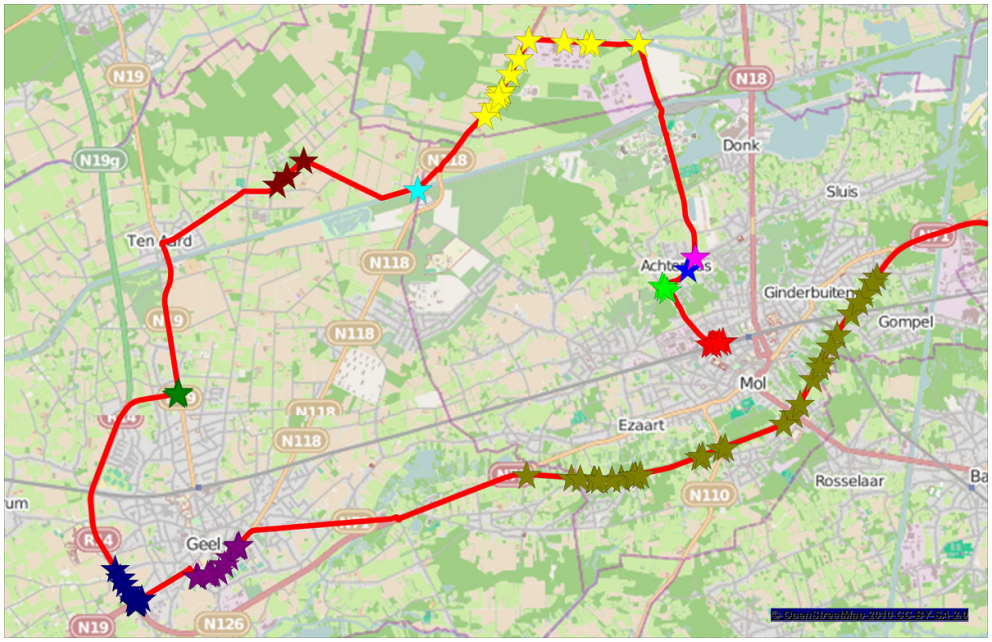


Figure 8.8: Part of a larger route showing a large number of *splitVertexSuite*s.

1. The relative frequency distributions suggest that Hypothesis 8.4.1 holds. The distributions depend on the methods used for trip detection and map matching. The distributions labeled *Belgium Navteq_x.y[min]*, differ only in the value for delay threshold parameter used in stop-detection. The smaller the threshold value, the more stops are detected and hence the smaller the size of the trips (expressed as the number of links they contain). For a given sequence of GPS recordings, the more subsequences are flagged as stops, the lower the number of detected *basicPathComponent*s; this occurs because some briefly visited locations will be flagged as a *stop* when using a small delay threshold parameter value whereas they are detected to be a *splitVertexSuite* in the opposite case. This is reflected in the relative frequency distribution diagram. The probability (relative frequency) to find routes having 1 *basicPathComponent*, decreases with increasing value of the delay parameter (1.0 , 2.5 , 5.0). For the case of 2 *basicPathComponents* the phenomenon is largely attenuated. Starting at 3 *basicPathComponents* per trip, the effect is reversed as expected.

2. For Belgium, the OSM and Navteq cases seem to slightly differ. This can

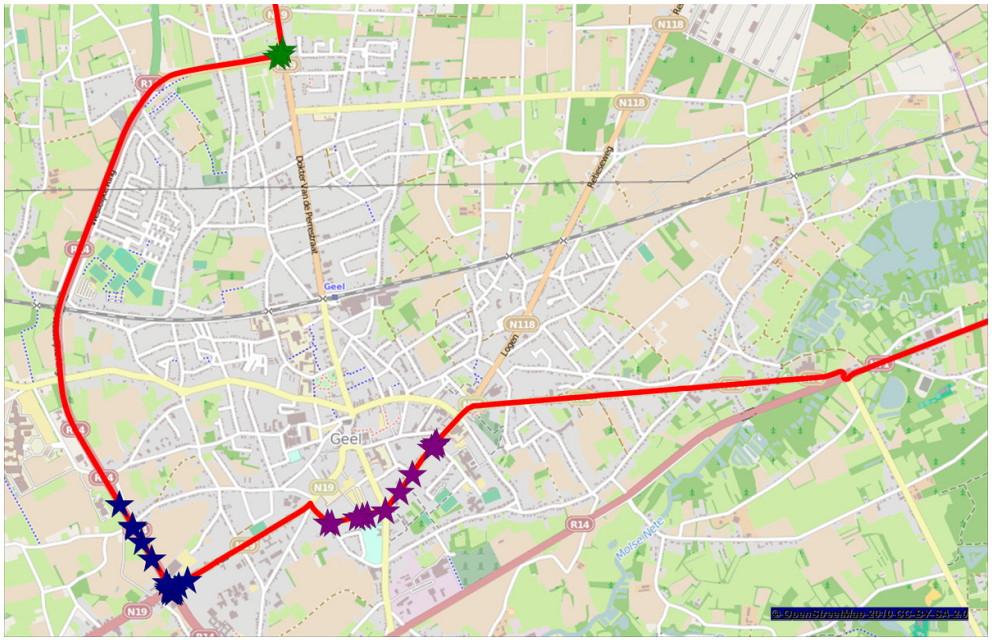


Figure 8.9: Detail view of Figure 8.8 showing splitVertexSuites in and near the city center.

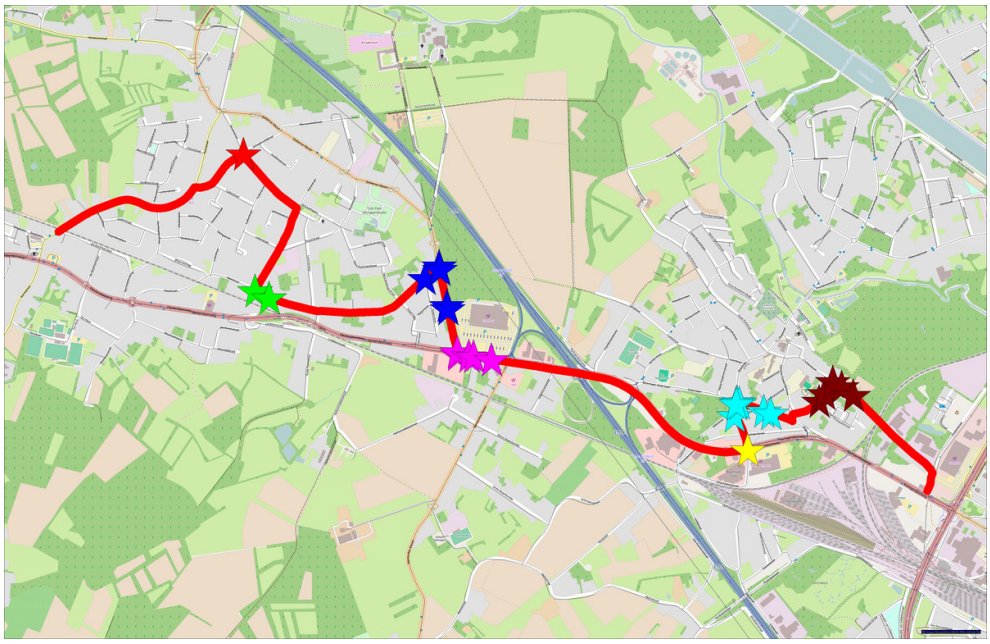


Figure 8.10: Route showing *splitVertexSuite*s that correspond to traffic lights.

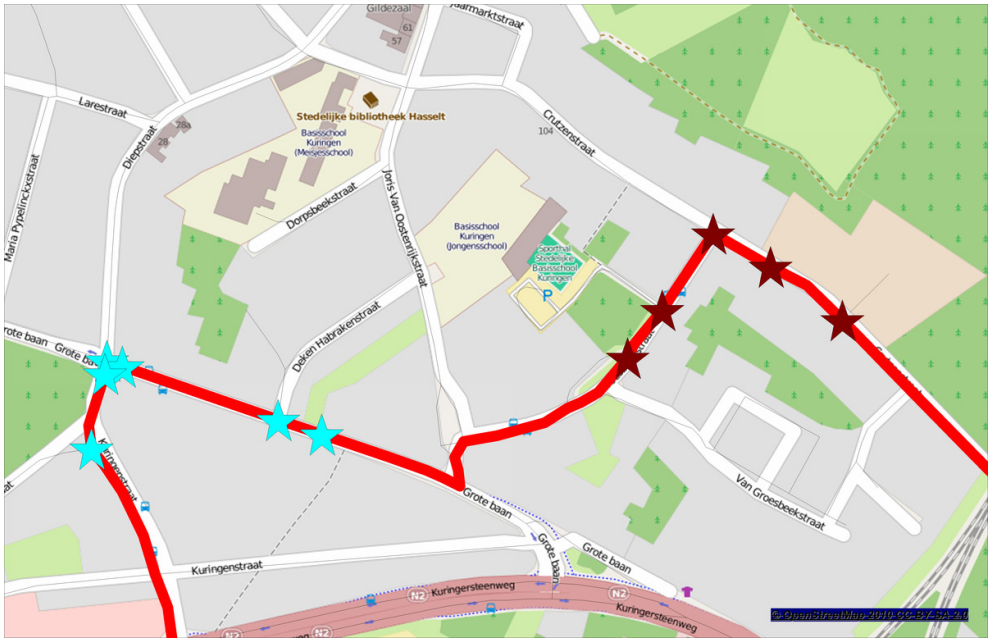
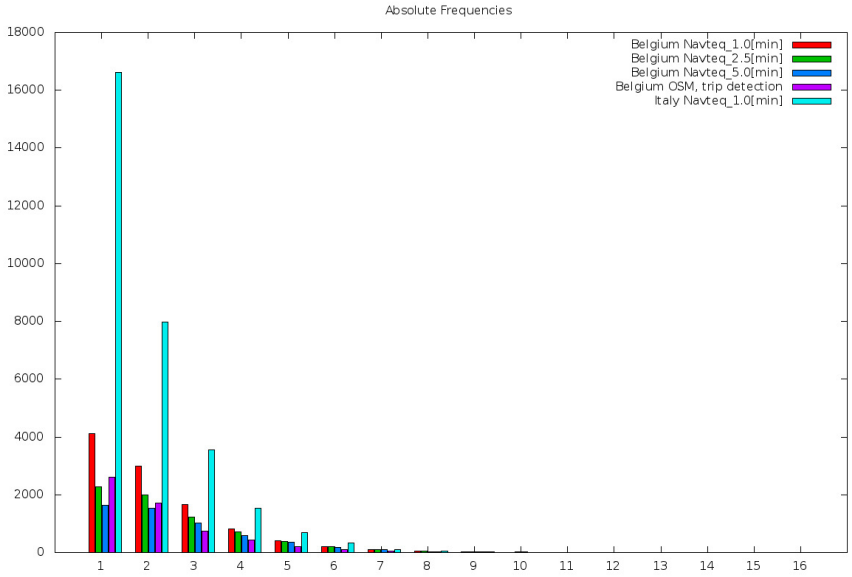
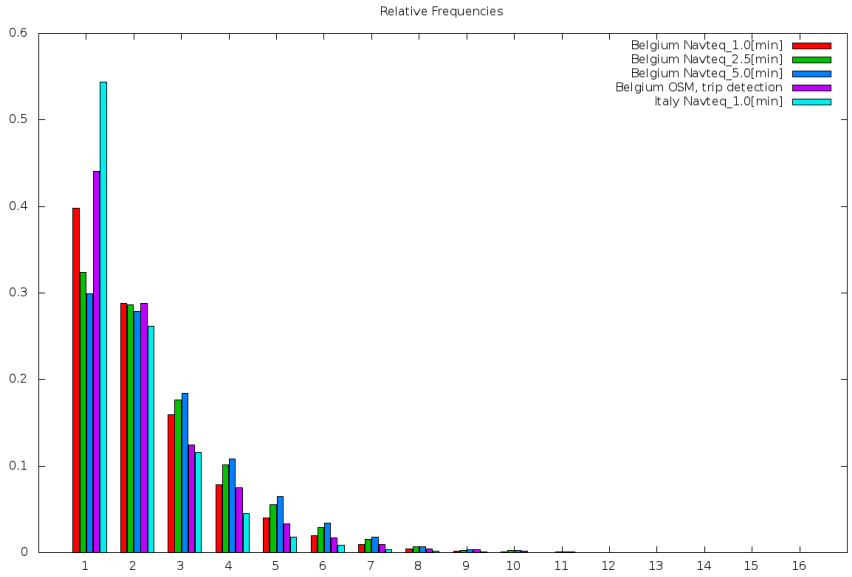


Figure 8.11: Detail of Figure 8.10 (rightmost part of the route).



Absolute frequency distribution for the number of *basicPathComponent*s.



Relative frequency distribution for the number of *basicPathComponent*s.

Figure 8.12: Frequency distributions (top:absolute, bottom:relative) for the number of *basicComponent*s per trip. The number of trips in each set depends on the map matcher used.

have been caused by the use of different map matching software tools, based on different methods and concepts. The Fraunhofer IAIS map matcher closes small gaps by assuming that the traveler used the shortest path. The IMOB map matcher does not use this procedure. This conclusion is not final because the cases differ both in the network and the map matcher used.

3. The difference between the Belgian and Italian cases, however, is much larger. The relative frequency for trips consisting of a single *basicPathComponent* is much larger than for the trajectories registered in Belgium. For routes having more than one component, the relative frequency is lower than in any Belgian case. Detail analysis is required to find out whether this phenomenon occurs because in the Italian data set, the pre- and post-car-trip components are missing (Italian traces are car traces).

8.8.3 Algorithm Execution Run Times

Table 8.1 summarizes characteristics of the runs. The results for the computed cases differ as a result of:

1. the difference in map matching methods and parameters used and
2. the difference between the networks (it is suspected but not yet verified) that the number of links between two junctions that are each others neighbors in the road network, is larger for the Navteq network than for the OSM network (due to the presence of trivial nodes mentioned in section 8.6.1).

8.9 Conclusions and future Research

A given path in a graph can be split in Basic Path Components (BPCs) that are either least-cost paths or non-least-cost edges. We developed an algorithm that computes the size of the minimum path splitting in an efficient way. A graph-theoretical correctness proof for the algorithm is given. This analysis sheds a light on the characteristics of *splitVertexSuites* (subpaths consisting of potential *splitVertices*) and their relationship to specific minimal shortcuts. It results in a set of *splitVertexSuites* from which minimum path splittings can be generated and an upper bound for the number of possible minimum path splittings.

This algorithm is used to verify the hypothesis that for utilitarian trips, people tend to compose their route from a small number of least cost paths. Both car and person GPS traces have been recorded. Individual trips have been identified in

Region	Belgium	Belgium	Belgium	Belgium	Italy
Case	OSM	Navteq 1.0[min]	Navteq 2.5[min]	Navteq 5.0[min]	Navteq 1.0[min]
Runtime[sec]	5191	17058	25822	36424	498850
Machine	calc2	calc2	lucp2364	linux1	calc4
OS	Linux Debian wheezy	Linux Debian wheezy	Linux Debian wheezy	Linux Debian wheezy	Windows Server 2008
CPU Memory	Xeon 3[GB]	Xeon 3[GB]	i5 4[GB]	Core2 2[GB]	Xeon X5670 48[GB]
ClockFreq	2.8[GHz]	2.8[GHz]	2.4[Ghz]	2.4[GHz]	2.93[GHz]
Cores used	7	7	3	2	20
Trips scanned	6632	12429	9408	8020	34308
Trips dropped	694	2066	2426	2508	3427
Net number of trips	5938	10363	6982	5512	30881
Number of basic components	12687	22921	17429	14412	56346
Number of basic components per trip	2.14	2.21	2.50	2.61	1.82
Average number of nodes per trip	21.68	82.47	75.43	62.20	55.37
Number of least cost path calculations	128736	854637	526652	342846	1772569
Least cost calculations per second	24.8	50.10	20.40	9.41	3.55
Number of trips per second	1.14	0.61	0.27	0.15	0.06

Table 8.1: Run characteristics overview.

the traces. Trips have been converted into paths in a graph by map-matching the traces to Open Streetmap and Navteq networks. The minimum number of least cost components in each path was computed and frequency distributions are presented. The paper shows the feasibility to analyze the structure of each route for large sets of collected traces. The results achieved are useful to statistically analyze factors that influence route choice for both individuals and population segments. As such they provide a foundation to generate realistic routes in transportation simulation models. The following questions need to be addressed in future research:

1. Determine an exact computation of the number of possible ways to partition a path into a minimum number of BPC's.

2. Investigate more rigorously how the distribution for the minimum number of BPC's in a path depends on the level of detail (coarseness) of the network and the map matcher used.

3. Design an efficient algorithm that generates routes from a given origin to a given destination for which the travel distance and the number of basic path components are values sampled from distributions determined from recorded trajectories.

4. Generate routes using a method such as the one described in Frejinger et al. (2009) in order to compare the resulting *splitVertexSuite* size distribution with the one extracted from GPS traces.

Chapter 9

Enumeration of Minimal Path Decompositions to Support Route Choice Set Generation

This chapter consists of

Knapen et al. (2015b)	*Enumeration of Minimal Path Decompositions to Support Route Choice Set Generation*

9.1 Abstract

A novel method is proposed to extract structural information about the routes revealed by GPS traces. This information is not available by any currently used method and is synthesized to feed the route choice set generation process.

Recorded GPS traces are converted to routes in a transportation network by map-matching. This results in large sets of paths in a graph. Each path can be decomposed into a sequence of least cost subpaths. The boundary between two successive components is a split vertex. Such split vertices constitute intermediate destinations for the traveler. We are interested in the minimum decompositions because travelers are assumed to compose their trips as simple as possible. The size of the minimum decompositions characterizes the structure of the path.

In general, multiple minimum decompositions do exist for every revealed path. Furthermore, not every vertex in the path can act as a split vertex in a decomposition. We propose an efficient method to enumerate all minimum decompositions of a path. This is used to reveal the importance of every vertex as a split vertex possibly considered by the traveler. The more possible splits contain a given vertex, the more important that vertex is.

This paper uses graph theoretical concepts to propose and prove an efficient algorithm to enumerate all possible minimum path decompositions. The method is based on the set of minimum shortcuts to the path used by the traveler. This set can be identified in polynomial time. As soon as it is available, the problem is reduced to enumerating all minimum clique covers in an indifference graph (a proper interval graph) which also can be done in polynomial time.

9.2 Introduction

Information about both the structure of the routes and about the network nodes (junctions), is extracted from map-matched GPS traces describing revealed routes recorded by travelers, and is made available to support route choice set generation.

When travel demand is generated using micro-simulation (e.g. by activity-based modeling), the individual trips need to be loaded on the network. Often this is done by aggregating the predicted demand into time dependent origin-destination (OD) matrices. This procedure causes information about the individual traveler to be lost. In case individual trips are handled, a plausible route connecting a given origin to a given destination needs to be found. This constitutes a discrete choice problem and a *route choice set* needs to be built. This set shall be populated by realistic

choice alternatives. While populating the choice set, the quality of the candidates can be assessed by means of the number of problematic left turns in the route, by the number of signalized crossings or by excess cost factors (e.g. detour factor or extra travel time factor when compared to the respective least cost routes). We propose to enhance automated quality assessment by taking into account additional information reflecting the structural complexity of the path for which an indicator can be acquired from paths revealed by GPS recording.

Each given path in a graph can be split into *basic path components (BPC)*. Such BPC is either a *least cost subpath* or a subpath consisting of a single *non-least-cost edge* (i.e. a single edge that does not constitute the lowest cost path between the vertices it connects). Two consecutive BPC's in a path share a *split vertex* (the boundary separating the BPC's). Consecutive split vertices can be seen as intermediate destinations connected by least-cost subpaths. Those can be any location having some meaning to the traveler (a school where to drop a child, a location to pick up a colleague, a safe crossing, a signalized junction providing easy left-turn, a junction where to access/leave the motorway, etc). (Formal definitions of BPC and split vertices will be given in section 9.5).

The following hypothesis was formulated and verified in Knapen et al. (2015c): *people tend to construct the routes for their utilitarian trips by concatenating a small set of basic path components (BPC)*. Utilitarian trips are trips having the purpose to achieve a specific activity at a given location (i.e. touring trips constituting a cycle are excluded). In order to verify the hypothesis, routes were decomposed into the smallest possible number of BPC's (called *minimum path decompositions*). The Minimum Path Decomposition Size (MPDS) is a measure for the structural complexity of the path. Frequency distributions for the MPDS were reported and sets of possible *splitVertices* were identified but no actual path splittings were generated in Knapen et al. (2015c).

In general, there are several ways to split a path into a minimum number of BPC's. This leads to uncertainty for the analyst about which of the split vertices were considered by the traveler as intermediate destinations. On the other hand it turns out that if N is the size of the minimum decompositions of path P, there are $N-1$ mutually disjoint sets of splitVertices, called *splitVertexSuites*, corresponding to subpaths in P so that every minimum decomposition for P is generated by taking exactly one vertex from each *splitVertexSuite*. However, not every combination of vertices selected from the *splitVertexSuites*, generates a valid decomposition.

The enumeration of all possible decompositions provides additional information because *splitVertex* candidates are not all equally probable since (i) not all *splitVertexSuites* have the same size and (ii) not every selection from the *splitVertexSuites*

generates a valid decomposition.

This paper presents an efficient technique to enumerate all possible minimum decompositions for a given route in a network. Section 9.3 sketches work in *route choice set generation* and determines the research objectives that motivated the development of the method described in this paper. Section 9.4 introduces the concept of *splitVertex* importance. Section 9.5 lists definitions and lemmas required for the development of the path splitting procedure. The developed decomposition technique then is described in detail in section 9.6. Finally section 9.7 summarizes the proposed solution and suggests follow-up research topics. Algorithms are presented in the appendices.

9.3 Context

9.3.1 Research Related to the Route Choice Problem

The choice set generation procedure and the influence of the choice set on the estimation of route choice models, are the subject of intensive research. Many choice set generators found in the literature are derivatives of or related to shortest path based methods. Link labeling finds least cost paths using several criteria. Link elimination techniques Zijpp and Catalano (2005), Schüssler et al. (2010) modify the topology. Link impedance adjustment is used in both deterministic (update link impedance on the shortest path) and stochastic methods. Doubly stochastic techniques integrate stochastic user preferences and stochastic link attributes. The constrained random walk generator proposed by Frejinger and Bierlaire (2007) also is based on path length since the link use probabilities are derived using a shortest path measure only.

Branch-and-bound techniques take more information into account. Hoogendoorn-Lanser (2006) proposes a rule based branch-and-bound algorithm to generate multimodal route choice sets. The embedded constrained path enumeration includes the number of transfers in the public transportation network as a *route-factor* (a structural path property).

The branch-and-bound technique specified by Prato and Bekhor (2006) introduces additional route quality constraints in the choice set generation phase. It is reported to generate choice sets showing high coverage and consistency values which indicates that it successfully reproduces routes collected by surveys. The constraints used in the non-compensatory decision to include a route in the choice set are: (i) a *directional* constraint excluding links that bring the traveler farther from the destination, (ii) a *temporal* constraint excluding links for which the travel time to the endpoint along the constructed path is much longer than the minimum travel time, (iii) a *loop* constraint

excluding paths that contain a subpath for which the distance detour factor is larger than the given threshold, (iv) a *similarity* constraint rejecting highly overlapping paths and (v) a *movement* constraint based on the number of left turns on the route. The values for the constraint thresholds are given constants in Prato and Bekhor (2006), Prato and Bekhor (2007) and Prato et al. (2012). Five predefined parameter sets are used in the meta-analysis reported in Prato (2012).

A similar technique is proposed by Pillat et al. (2011) who use a detour threshold that is a function of the duration of the first part of the route being constructed. The function was derived from routes collected by a survey.

Kaplan and Prato (2010) describes a branch-and-bound generator using thresholds for the detour (based on travel time) and the number of left turns. The thresholds are not pre-defined. The author constructs a *conjunctive heuristic* which integrates the decisions in the choice set generation phase and the choice model. The thresholds for the route generator and the choice model parameters are jointly estimated.

Based on this overview, one needs to conclude that route characteristics other than overall length or travel time, were demonstrated only by the branch-and-bound methods. However, it is possible to integrate them in the algorithms proposed by Zijpp and Catalano (2005) and Schüssler et al. (2010). Finally they can be used in a route filtering phase when deriving the consideration set from the master set Bovy (2009).

9.3.2 Structure of Chosen Routes

The hypothesis mentioned in the introduction was verified in Knapen et al. (2015c) by analyzing two sets of GPS recorded traces. Trips were extracted and map-matched which results in a set of walks (sequences of connected links) in the road network graph. Each walk that is a path (i.e. which visits every node at most once) is assumed to constitute the route for a utilitarian trip. Frequency distributions for the size of the minimum decompositions into BPC have been determined for 47134 trips in the Milano (Italy) region and 5876 trips in Flanders (Belgium). The results show that the hypothesis holds. In Knapen et al. (2015c) it is proposed to use Minimum Path Decomposition Size (MPDS) as an additional criterion for path plausibility assessment in route choice. The distribution for the MPDS can be used in the choice set generation stage in route choice modeling.

9.3.3 Contribution of this Paper

Route choice procedures can use information about the network other than the attributes of the paths evaluated for inclusion in the consideration set. This can be done by means of landmark information by including boolean explanatory variable (dummies) in the choice model (e.g. Prato and Bekhor (2007)). The landmarks are predefined by the analyst for use in a survey.

Kazagli and Bierlaire (2014) propose to replace paths by *abstract geo-marked items* denoted by Mental Representation Items (MRIs). An MRI is not necessarily a specific location or landmark. An area like a city center can act as an MRI. MRIs are used in the choice set generation phase and not only in the choice set model as an explanatory variable. However, practical problems can arise when MRIs need to be derived from recorded data.

The results presented in this paper might help to support this problem. A method is proposed that enumerates all possible decompositions of a given path in a graph, in polynomial time. This allows to easily calculate for every vertex the occurrence frequency as a *splitVertex* in a minimum path decomposition. This is a measure for the probability that the vertex was used as an intermediate destination by travelers. It contains information about the network (topology, path travel cost) and about the revealed routes. This information can be derived from big data (GPS traces). It quantifies the appropriateness of a vertex to act as a landmark useful for route construction. The level of abstraction is much lower than the one associated with the MRI in Kazagli and Bierlaire (2014) but the information can be derived automatically.

9.4 The Use of Route Structure Information

First, the notion of vertex importance is explained. It is a measure for the probability that the vertex is considered to be the endpoint of a least-cost subpath by the traveler. Then we propose a method to integrate Minimum Path Decomposition Size (MPDS) and vertex importance in the route choice procedure.

9.4.1 Vertex Importance

Let N denote the size of the minimum decompositions of path P into BPC. The algorithm specified in Knapen et al. (2015c) delivers $N-1$ mutually disjoint sets of splitVertices corresponding to subpaths in P called *splitVertexSuites*, so that every minimum decomposition for P is generated by taking exactly one vertex from

each *splitVertexSuite*. However, not every combination of vertices selected from the *splitVertexSuites*, generates a valid decomposition.

A *splitVertexSuite* S_A is independent of S_B if and only if the selection of a *splitVertex* from S_A is not constrained by the prior selection made from S_B. If S_A is mutually independent of every other *splitVertexSuite*, then the *splitVertex* from S_A can be freely chosen in each minimum decomposition. Every *splitVertex* in S_A then has the same probability to be the one that was considered by the traveler as an intermediate destination.

It is improbable that the decomposition of all paths in a large collection, leads to mutually independent *splitVertexSuites* for each of the paths. For this reason one shall be careful when creating statistics using information about properties of *splitVertexSuites* only.

We assume that the probability that a vertex carries a meaning relevant to the traveler (but yet unknown to the analyst), increases with its use frequency as a *splitVertex* in a decomposition. This assumption is similar to what is done in the *trajectory annotation* process (i.e. the process that tries to assign a meaning to each *stop* detected in a GPS trace). In that process, the visit frequency and the total time spent at a given location are quantities used while trying to discover the intention of a stop. In the case of route splitting, we try to find the probability for a vertex in a *splitVertexSuite* to be the *splitVertex* that the user had in mind.

Definition 9.4.1. *The* importance $i(v, \mathcal{P})$ *of a vertex* v *in a set of paths* $\mathcal{P}$ *is the relative occurrence frequency of* v *as a* splitVertex *in the set of all possible minimum decompositions of all* $P \in \mathcal{P}$.

Let $\mathcal{D}(P)$ denote the set of all minimum decompositions of P. Let $S(d)$ denote the set of splitVertices constituting the decomposition d. Let $\mathcal{D}_v(P)$ denote the set of all minimum decompositions of path P making use of *splitVertex* v i.e.: $\mathcal{D}_v(P) = \{d \in \mathcal{D}(P) | v \in S(d)\}$. Then

$$i(v, \mathcal{P}) = \frac{\sum_{P \in \mathcal{P}} |\mathcal{D}_v(P)|}{\sum_{P \in \mathcal{P}} |\mathcal{D}(P)|} \tag{9.1}$$

Specific importance values can be calculated by restricting the set $\mathcal{P}$ of paths considered (e.g. only the paths for a given individual, only the paths having a given destination etc). If a particular path P is considered and if in addition the *splitVertexSuite*s for P are mutually independent, the importance for a specific vertex v is

given by

$$i(v,\{P\}) = \frac{|\mathcal{D}_v(P)|}{|\mathcal{D}(P)|} = \frac{\prod_{s\in\overline{SVS}(P)\setminus SVS(v)} |s|}{\prod_{s\in\overline{SVS}(P)} |s|} = \frac{1}{|SVS(v)|} \tag{9.2}$$

where $\overline{SVS}(P)$ is the set of *splitVertexSuite*s that determine the minimum decompositions of path P and $SVS(v)$ is the *splitVertexSuite* containing vertex v.
We assume that the mentioned probability equals the vertex *importance* (or that the importance is a sufficiently accurate approximation). Note that for the singleton $\mathcal{P} = \{P\}$, the importance values for the vertices in a given *splitVertexSuite* sum up to one as should be; this is because every minimum splitting contains exactly one *splitVertex* from each *splitVertexSuite.*

The computation of *importance* in general requires the enumeration of all possible decompositions of a path.

9.4.2 Choice Set Filtering

We briefly indicate how the results of the proposed method support the route choice problem. Following steps need to be performed.

1. Using a set of GPS recordings, the distribution for the MPDS and the overall importance $i(v,\mathcal{P})$ of each vertex v used in a path, are determined.

2. For vertices having a sufficiently high importance to be an intended *splitVertex*, additional attributes are retrieved from a GIS. Examples are: availability of traffic lights, change in road category, train station or school neighborhood, carpool parking, etc. A model is formulated and estimated to predict the vertex importance from those attributes. The intention is to predict importance values for vertices that are not used in the recorded trips.

3. The predicted importance $\bar{i}$ for the vertices is used to evaluate and compare route candidates. Thereto, the set $\mathcal{D}(P)$ of minimum decompositions for the proposed path P is determined (by the technique proposed in this paper). For each decomposition d, the likelihood is determined as the sum of the importance values for the *splitVertices* in the set $S(d)$ constituting the decomposition: $L(P) = \sum_{v\in S(d)} \bar{i}(v)$.

4. A choice set generator algorithm $\mathcal{A}$ (of type B&B, BFS-LE or other) is designed to propose routes and to prune infeasible candidates using a filter that is based

on the size of the minimum decomposition of the candidate and the likelihood defined in 3.

5. In the model estimation phase, a choice set $\mathcal{P}_{CS}$ is built. It is initialized using the set $\mathcal{P}_{GPS}$ of paths derived from the recorded traces. After initialization it is extended by adding paths generated by means of the algorithm $\mathcal{A}$. A logit based route selection model $\mathcal{M}$ is specified that takes the size of the minimum decomposition and the likelihood defined in item 3 as independent variables along with the variables that are currently used (maximal detour factor, number of left turns, path size correction factor etc). The model parameters are estimated using $\mathcal{P}_{CS}$ and $\mathcal{P}_{GPS}$.

6. In the prediction phase, the algorithm $\mathcal{A}$ is used to generate the choice set and the model $\mathcal{M}$ is used to sample a route.

9.5 Definitions and Basics

First some definitions from graph-theory are presented. Let $G = (V, E)$ be a directed graph with vertex set V and edge set E. The vertices correspond to *nodes* in a road network, and the edges correspond to *links* in the network. Each edge e has a non-negative *cost* $c(e)$ which is the effort (e.g. time or money) required to traverse the link in the network. For a subgraph $H \subseteq G$, $V(H)(E(H))$ denote the set of vertices (edges) of H.

Definition 9.5.1 (walk, initial, terminal, internal vertices, internally-disjoint)**.** *A* walk *is a sequence of vertices $P = (v_0, v_1, \ldots, v_l)$, not necessarily distinct, where $(v_i, v_{i+1}) \in E(G)$ for all $i = 0, 1, \ldots, l-1$. Vertices v_0 and v_l are called* initial *and* terminal *vertices, respectively, of P, and vertices $v_1, \ldots, v_{l-1}$ are called* internal vertices *of P. The walk P is said to be* connecting *v_0 and v_l, and it is also denoted by $P(v_0, v_l)$. A walk $Q(v_0, v_l)$, is* internally-disjoint *from P if all the internal vertices of Q, are distinct from the vertices in P.*

Definition 9.5.2 (path, subpath, size, cost, least cost path)**.** *A* path *is a walk where all its vertices are distinct. For a path $P = (v_0, v_1, \ldots, v_l)$, any subsequence of vertices $v_i, v_{i+1}, \ldots, v_j$, where $0 \leq i \leq j \leq l$ is a* subpath *of P, and is denoted by $P(v_i, v_j)$. The* size *of a path, denoted by $|P|$, is the number of edges in it (i.e. l), and the* cost *of a path, denoted by $c(P)$ is the sum of the costs of its edges. A path $P(v_0, v_l)$ is a* least cost path *between v_0 and v_l, if there exists no other path connecting v_0 and v_l of lower cost.*

Note that if $c(e) = 1$ for all $e \in E$ then the cost of a path coincides with its size. We assume that the vertex traversal cost is zero. A single edge (u, v), being a path connecting between u and v, may be least cost, or not. If it is not a least cost path connecting between u and v, then it is called a *non-least-cost-edge*.

It is easy to see that if P is a least cost path, then any subpath of P is also a least cost path (see proof in Knapen et al. (2015c)).

The converse of this statement is false since it is possible that all the subpaths of $P(v_0, \ldots, v_l)$ (except P itself) are least cost paths, but P is not a least cost path connecting v_0 and v_l and there is another least cost path Q connecting v_0 and v_l. This fact motivated the following definition, as in Knapen et al. (2015c).

Definition 9.5.3 (P- shortcut, minimal shortcut, fork and join vertices, bypassed vertex set)**.** *Let $P = (v_0, v_1, \ldots, v_l)$ be a given path. A $P(v_i, v_j)$-*shortcut *(or for brevity, P -* shortcut*, or* shortcut*), is a path $Q(v_i, v_j)$, internally- disjoint from P, where $v_i, v_j \in V(P)$, such that $c(Q(v_i, v_j)) < c(P(v_i, v_j))$. The vertices v_i and v_j are called* fork *and* join *of the shortcut, respectively, and the internal vertices of P between the fork and the join (i.e. $v_{i+1}, \ldots, v_{j-1}$) are called Q-*bypassed vertex set*, or* bypassed vertex set *and denoted by $B(Q)$. A shortcut Q is* minimal *if $B(Q)$ does not contain $B(Q')$ where Q' is another shortcut to P. (See Figure 9.1).*

We emphasize that $B(Q)$ contains consecutive vertices on P. Therefore, it can be marked by the fork and join of a shortcut Q, which are the vertices preceding, and following the set $B(Q)$, respectively.

Clearly, a least cost path cannot have any shortcuts.

Definition 9.5.4 (Basic Path Component (BPC), path splitting, splitVertex)**.** *Given a path P, a subpath of P is called a* Basic Path Component*, or for short, a BPC, if it is either a least cost path connecting its endpoints, or P is a single non-least-cost-edge. A* path splitting *of P is a partition of P into subpaths each of which is a basic path component. A* splitVertex *is a vertex separating two consecutive BPC in a path splitting, and is denoted by v_i^s.*

Note that there may be many ways to split a path, for example, the trivial partition into edges $(v_0, v_1), (v_1, v_2), \ldots, (v_{l-1}, v_l)$ is an example of such a partition. We are interested in finding a path splitting with the *minimum number* of basic path components. Such a path splitting is called *minimum path splitting*. Each non-shortest-edge is a part in each minimum path splitting since it constitutes a BPC. If we remove the set of non-shortest edges in a path (each of which is a BPC), we are left with a set of disjoint paths, each of which contains no non-shortest-edges.

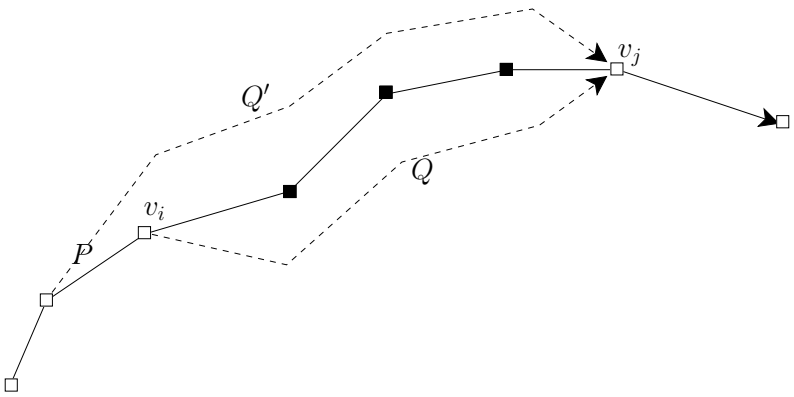


Figure 9.1: A path P with a minimal $P(v_i, v_j)$-shortcut Q. Q' is a non-minimal shortcut. Vertices v_i and v_j are fork and join vertices of Q, respectively and the marked vertices are the bypassed vertex set $B(Q)$.

From now on we will assume that P does not contain any non-shortest-edges.

In Knapen et al. (2015c) we addressed the problem of finding efficiently a minimum path splitting of a given path.

Since a minimum path splitting will contain a minimum number of splitVertices, an equivalent formulation of the problem above is to find a minimum number of splitVertices in the path, such that any subpath connecting consecutive splitVertices will be least cost.

Lemma 9.5.5. *Let $P = (v_0, v_1, \ldots, v_l)$ be a path connecting v_0 and v_l. Assume P is not least cost, and let $Q(v_i, v_j)$ be a shortcut in P. Then any path splitting of P will contain at least one vertex in $B(Q)$, the Q-bypassed vertex set, as a splitVertex.*

Proof 9.5.1. *By contradiction. If no vertex in $B(Q)$ is a splitVertex, then $P(v_i, v_j)$ is a least cost path, contrary to the fact that $Q(v_i, v_j)$ is a shortcut in P.*

Corollary 9.5.6. *A minimum path splitting of P is obtained by a minimum set of splitVertices which meets $B(Q)$ for all minimal shortcuts Q to P.*

Proof 9.5.2. *Since every two consecutive BPC are separated by a splitVertex, it is sufficient to minimize the number of splitVertices. By Lemma 9.5.5 it is necessary to meet each $B(Q)$, where Q is a shortcut. However, since every $B(Q')$ contains a $B(Q)$ where Q is a minimal shortcut, it is sufficient to meet all bypassed sets of minimal shortcuts. The converse also holds - a minimum set of vertices which meets the $B(Q)$ of all minimal shortcuts Q, defines a minimum path splitting of P.*

The algorithm described in Knapen et al. (2015c) begins with the initial vertex of P, v_0, and finds a maximal least cost path beginning with it. This is done using Dijkstra's least cost path algorithm (Dijkstra (1959)). Assume v_{j_1} is the first vertex on P for which $P(v_0, v_{j_1})$ is not least cost, then the algorithm marks v_{j_1} as a join vertex and continues with the subpath of P beginning from the vertex prior to v_{j_1} on P, looking for the next join vertex in $P(v_{j_1-1}, v_l)$. The algorithm continues until no more join vertices are found. It was proved that the vertices preceding the join vertices found on P are splitVertices, and their number is minimal.

In a similar way, a backward pass on P is done, beginning with the end vertex v_l and going backwards, to find a minimum set of fork vertices, whose successors on P are also splitVertices in a minimum path splitting. The algorithm is efficient since it uses Dijkstra's algorithm no more than N times, where N is the minimum number of BPC's used to partition P.

Since the bypassed vertex sets may contain points of interest for the traveler (otherwise a minimum cost path would have been chosen), we are interested in enumerating all splitVertices and all minimum partitions into BPC's.

Note that the algorithm in Knapen et al. (2015c), is highly efficient, but it does not find all the shortcuts to P. This fact does not allow us to enumerate all possible minimum path splittings, as was demonstrated in the example in Knapen et al. (2015c) Figure 3.

9.6 Path Decomposition Enumeration Technique

Assume that a path in a graph is given. In the context of this paper, the path is derived by applying *trip detection* and *map matching* on a GPS trace. The resulting path is broken down into a set of non-least-cost edges and subpaths that are free of non-least-cost edges. The latter subpaths are considered for decomposition into a minimum number of BPC that are least cost paths.

The enumeration of all minimum partitions of the path into BPC's is done in five stages:

1. Finding all minimal shortcuts to P.
2. Defining the intervals and the proper interval graph G^I derived from the shortcuts to P.
3. Enumerating all cliques in G^I
4. Constructing the directed-clique-graph G^C
5. Enumerating all shortest source-sink paths in G^C

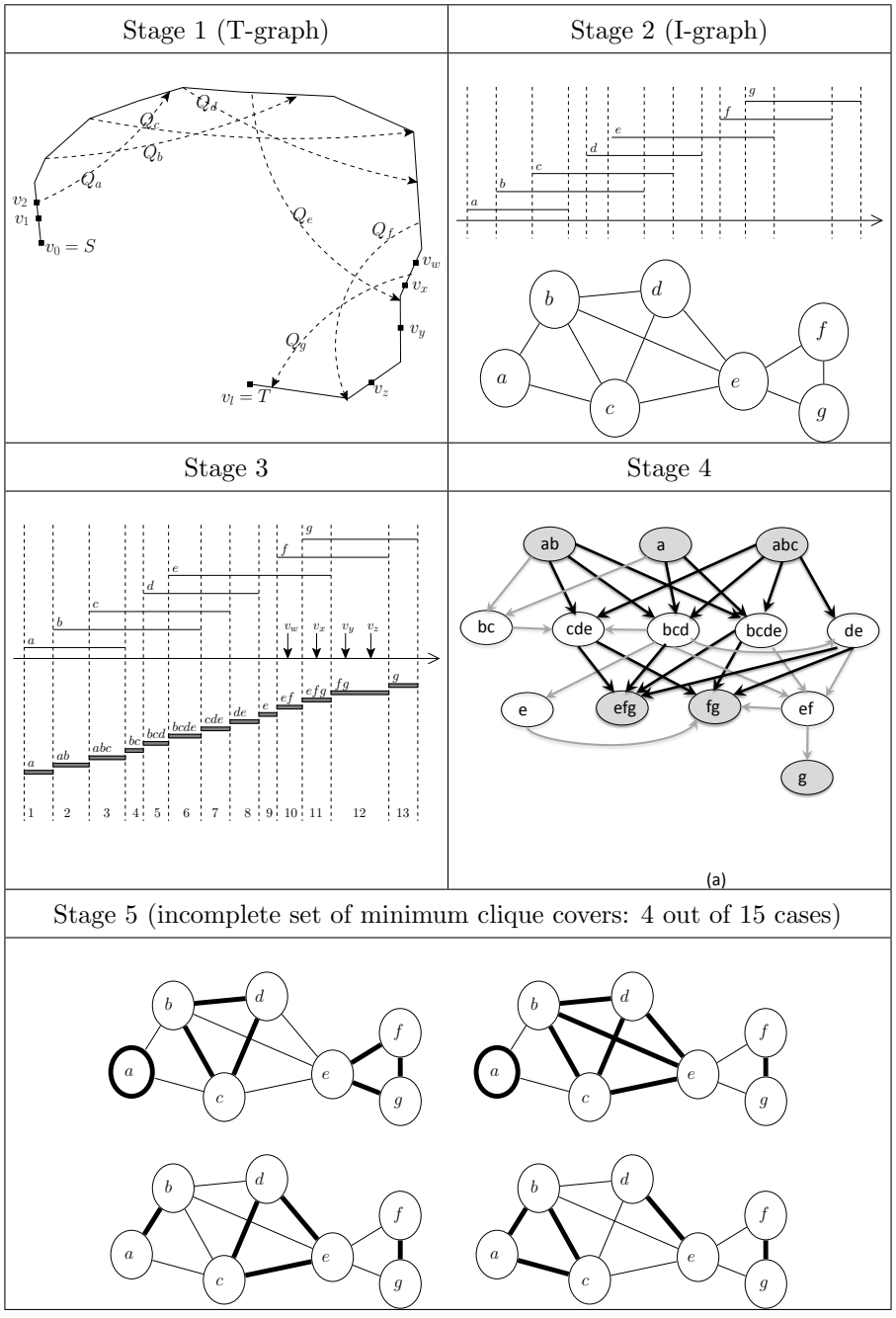


Figure 9.2: Overview of graphs used to enumerate minimum path decompositions.

9.6.1 Stage 1: Finding all minimal shortcuts to P

In this stage we find all minimal shortcuts to P. We do not need the shortcut paths to P, but rather their endpoints, the fork and join vertices of each shortcut. The output is a list of pairs $< v_f, v_j >$ corresponding to the fork and join vertices of all minimal shortcuts.

Assume a traveler moves from point v_0 to point v_l along a path $P = (v_0, v_1, \ldots, v_l)$. If this path is the least-cost path between v_0 to v_l then it has no shortcuts, it is a BPC, and nothing needs to be done. Otherwise, Dijkstra's shortest path algorithm is used to find the first vertex on P, say v_j for which $P(v_0, v_j)$ is not the shortest path connecting v_0 and v_j. We mark v_j as a join vertex, and continue by finding the *last* vertex in the subpath $P(v_0, v_j)$, say v_f for which $P(v_f, v_j)$ is not a least-cost path. We output the shortcut $< v_f, v_j >$ and continue with the subpath of P beginning with v_{f+1} . The pseudo code is given in the Algorithm 9.8.1.

9.6.2 Stage 2: Defining the intervals and the proper interval graph G^I

Once all minimal shortcuts to P are known, the corresponding bypassed vertex sets are considered. By Corollary 9.5.6, a minimum path splitting will contain a smallest set of splitVertices which meets all bypassed vertex sets. To find sets of splitVertices we construct a set of intervals corresponding to all bypassed vertex sets in the following way: If $B(Q) = \{v_i, v_{i+1}, \ldots, v_j\}$, then it is represented by the closed interval $[i, j]$ on the real line. (see Figure 9.2-Stage 2). Note that the integral points on the interval $[i, j]$ (i.e. the points $i, i+1, \ldots, j$ correspond to the vertices $v_i, v_{i+1}, \ldots, v_j$ on P). Since the shortcuts found in Stage 1 are minimal shortcuts, no two intervals contain each other. The intersection graph of this set of intervals, is by definition, a *proper interval graph* (see Looges and Olariu (1993) for definition). We denote it by $G^I = (V^I, E^I)$, where each $v \in V^I$ corresponds to $B(Q)$ of some shortcut Q, and two vertices are adjacent if and only if the corresponding bypassed vertex sets intersect.

Note that if we order all the intervals representing V^I by their left hand endpoint, in increasing order, then, being a proper interval graph, their right hand endpoints will also be in increasing order (otherwise one interval will contain another). We label the ordered set of intervals as $I_1, I_2, \ldots I_n$ or, when more convenient, as $a, b, c, \ldots$ (see Figure 9.2-Stage 2).

9.6.3 Stage 3: Enumerating relevant cliques in G^I

In Stage 2 we have defined a set of intervals on the real line whose intersection graph is a proper interval graph G^I. Each point on the real line meets a subset of these intervals which correspond to some clique in G^I. Conversely, for every set of intervals which mutually pairwise intersect, by the Helly property, there exists a point on the real line which meets all the intersecting intervals. Moreover, since the intervals begin and end in integral points, we may assume that every clique of mutually intersecting intervals is met by some integral point, i.e. a vertex on P. There are at most $2|V^I|$ endpoints of all the intervals (some right hand endpoint may coincide with a left hand endpoint). They define at most $2|V^I| - 1$ possible non-empty intersections of the intervals, ordered linearly in increasing order of the intersecting points (see Figure 9.2-Stage 3). Each interval intersection is some clique in G^I. Since the graph is a proper interval graph, *all* cliques can be ordered so that each interval belongs to consecutive cliques (Booth and Lueker (1975), Fulkerson and Gross (1965), Gardi (2007)). This is the characteristic linear clique order for the proper interval graph. For interval graphs in general, a similar property holds for the *maximal* cliques.

See Algorithm 9.9.1 for the pseudo code for enumerating all relevant cliques in G^I, denoted by $\mathcal{C}(G^I)$.

9.6.4 Stage 4: Constructing the directed-clique-graph G^C

Given the clique family found in Stage 3, $\mathcal{C}(G^I)$, we are interested in finding all possible minimum coverings of V^I by cliques from the set $\mathcal{C}(G^I)$. In order to find those coverings, we construct a directed graph $G^{\mathcal{C}} = (V^{\mathcal{C}}, E^{\mathcal{C}})$, where the vertex set corresponds to the cliques $C_i \in \mathcal{C}(G^I)$. To distinguish between the vertices in $V^{\mathcal{C}}$ and the vertices in other graphs we call these vertices *c-vertices*. There is a directed edge from a $c-$vertex C_i to a $c-$vertex C_j according to the following rule: Assume clique C_i contains the consecutive set of intervals labeled $I_i, I_{i+1}, \ldots, I_{i+k}$ and clique C_j contains the consecutive set of intervals labeled $I_j, I_{j+1}, \ldots, I_{j+t}$. There is a directed edge from C_i to C_j if and only if

$$i < j \leq i + k + 1 \text{ and } i + k < j + t \tag{9.3}$$

See example in Figure 9.2- Stage 4. By the definition above, G^C is acyclic i.e. it contains no directed cycles. All the *source* vertices (i.e. vertices with indegree zero) are the vertices in G^C whose label contains I_1 (or a as in Figure 9.2-Stage 4) and the *sink* vertices (i.e. vertices with outdegree zero) are the vertices whose label contains I_n (or g as in Figure 9.2-Stage 4). The source and sink vertices are marked in Figure 9.2-Stage

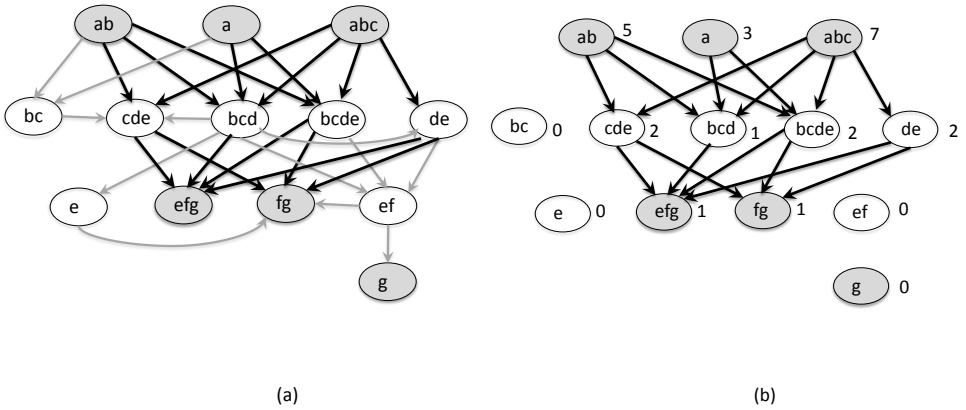


Figure 9.3: The graph G^C constructed in stage 4: diagram (a): before pruning, diagram (b) after pruning lower layer sinks and deleting edges not on a shortest source-sink path. The numbers beside each vertex represent the number of shortest paths from the vertex to a sink vertex.

4 as dark vertices. The condition in equation (9.3) guarantees that any directed path from a source vertex to a sink vertex will cover all the intervals $\{I_i; i = 1, 2, \ldots, n\}$. The first part of the condition means that $C_i \cup C_j$ contains all intervals $I_i, \ldots, I_{j+t}$; the second part implies that $C_i \not\supset C_j$. As a consequence, C_i and C_j can be members of the same minimum clique cover of G^I.

9.6.5 Stage 5: Enumerating all shortest source-sink paths in G^C

We look for all minimum length paths in G^C which connect some source vertex to some sink vertex. The easiest, most efficient way to find shortest paths is to perform a breadth-first search (BFS) on G^C starting with the source vertices. Recall that BFS search divides a graph into *layers*, starting with the source vertices, in which all the nodes in layer d have distance d from the source vertices. We say that layer d is *above* layer d' if $d < d'$. Since we are looking for the shortest paths from sources to sinks, we consider only the closest sink vertices, i.e. sink vertices with the highest layers; other sink vertices we can ignore, as well as their incoming edges. We also ignore edges which connect two vertices at the same layer of the BFS tree, and edges which do not lead to any highest-level sinks. (See Figure 9.3).

To *count* the number of shortest source-sink paths, we begin with the sinks in the highest layers, and label them 1. (since there is a unique shortest path beginning at

them and ending at them - the trivial path consisting of one vertex). As we move up through the BFS layers, we see that the number of shortest paths to sinks from each node is the *sum* of the number of shortest paths from all nodes directly below it to sinks in the BFS search. Working upwards through the layers, we get the number of shortest paths from each source vertex to sink vertices. It is quite easy to construct these paths layer-by-layer.

9.6.6 Enumerating minimum path splittings

In Section 9.6.5 we have found all shortest source-sink paths in the clique graph G^C of the interval graph G^I. We recall that each vertex in such a path corresponds to a clique in the interval graph, in other words, to a set of vertices in the original path P traveled by the user, which are bypassed by a unique set of shortcuts.
Let $C_{i,j}$ denote the clique containing intervals $I_i, I_{i+1}, \ldots, I_j$.

Definition 9.6.1 (Core of a clique)**.** *The core $\mathcal{K}(C_{i,j})$ of a clique $C_{i,j}$ is the set of bypassed vertices on P specified by $\mathcal{K}(C_{i,j}) = \bigcap_{k\in[i,j]} B(Q_k) \setminus \bigcup_{k\notin[i,j]} B(Q_k)$ i.e. the* core *is the set of vertices belonging to exactly the intersection of the bypassed vertex sets for $C_{i,j}$ and not to any other bypassed vertex set.*

For example, in Figure 9.2 the clique *ef* contains the bypassed vertex v_w, the clique *efg* contains bypassed vertex v_x and clique *fg* contains v_y, v_z. Each one of these bypassed vertices is a potential splitVertex and its choice is independent of the selection of bypassed vertices corresponding to the other cliques on the same source-sink path. (see Figure 9.2-Stage 3). The number of possible path splittings is therefore

$$\sum_{p^C\in\mathcal{P}^C}\left(\prod_{C\in p^C}|\mathcal{K}(C)|\right) \tag{9.4}$$

where $\mathcal{P}^C$ is the set of shortest source-sink paths in G^C and each $C \in p^C$ represents the clique in G^I corresponding to a vertex on p^C.

Theorem 9.6.2. *Stages 1-5 allow us to enumerate all minimum decompositions into BPC's of a path, and Equation* (9.4) *counts the total number of such decompositions.*

Proof 9.6.1. *By Corollary 9.5.6 every minimum path splitting of P is obtained by a minimum set of splitVertices which meets every $B(Q)$, for all minimal shortcuts Q to P. We have found in Stage 1 all bypassed vertex sets $B(Q)$ of minimal shortcuts, and*

represented them in Stage 2 by intervals on the real line, no two of which contain each other, and who's endpoints are integer points. In Stage 3 we have found all integer points which meet all possible cliques - these correspond to all possible splitVertices. In Stage 4 we constructed the directed clique graph G^C and in Stage 5 we found all shortest source-sink path in G^C.

Each shortest source-sink path in G^C has L vertices. We argue that $L = N - 1$, where N is the size of the minimum decomposition of the path into BPC's. Each shortest source-sink path in G^C corresponds to a collection of sets S_i of splitVertices with $i \in [1, L]$ so that every tuple $T \in S_1 \times \ldots \times S_L$ partitions the given path P into a minimum path decomposition, and vice versa - every minimum path decomposition is found in Stage 5. An outline of the proof is given below: details have been omitted for brevity.

(a) Each considered source-sink path p^C is minimal (by construction, Stage 5).

(b) Each vertex on the path in G^C corresponds to a clique in G^I (i.e. to a set of intervals). No two vertices in G^C correspond to the same clique in G^I (by construction, Stage 4).

(c) If two cliques are different, their cores are different : $C_i \neq C_j \Rightarrow \mathcal{K}(C_i) \neq \mathcal{K}(C_j)$ (by definition 9.6.1). Furthermore, their cores are disjoint $C_i \neq C_j \Rightarrow \mathcal{K}(C_i) \cap \mathcal{K}(C_j) = \emptyset$ (follows from Definition 9.6.1).

(d) From (b) and (c) follows that the cores for any pair of vertices in G^C are disjoint.

(e) The union of the cliques on each source-sink pair in G^C is the set of all intervals V^I (i.e. no interval is missing) (by construction, Stage 4: see expression (9.3)*, second condition).*

(f) If $(C_i, C_j) \in E^C$ then C_i precedes C_j in the characteristic linear clique order for the proper interval graph. Furthermore $(C_i, C_j) \in E^C \Rightarrow (C_j, C_i) \notin E^C$ (by construction of the clique enumerator and due to the G^C construction procedure (see expression (9.3)*, first condition))*

(g) Two different paths in G^C cannot correspond to the same set of cliques because all considered paths have equal length (see (a)) and because of (f) they cannot be built using the same set of vertices.

(h) From (c) and (g) follows that two different paths have different sets of cores.

(i) The splittings enumeration procedure is: for each source-sink path in G^C exactly one vertex in path P is selected from each core.

(j) Because all vertices in the core of a clique belong to exactly the same set of bypassed vertex sets (follows from Definition 9.6.1) and because selecting a vertex

corresponds to meeting *a bypassed vertex set, all vertices in the core are equivalent. They are indistinguishable with respect to the set of bypassed vertex sets they meet. Hence while selecting vertices from cores in order to meet* all *bypassed vertex sets, selection of a vertex in a core is independent of the selection made for any other core.*

(k) Creation of a tuple $T \in S_1 \times \ldots \times S_L$ *by selecting one vertex from each core for a source-sink path in* G^C *has the following properties*

(a) selection from each core can be done independently (follows from (j))

(b) the tuple of selected vertices defines a minimal decomposition (it is minimal because the number of cliques on the path is minimal according to (j) and it is a decomposition because it delivers a vertex in each bypassed vertex set (see (e))

(c) no two paths deliver the same selection sets (because of (h))

hence, selecting all possible combinations for each shortest source-sink path p^C*, delivers different solutions since no two paths deliver the same selection set (because of (h)).*

(l) Finally, all possible decompositions are enumerated because the cores for all cliques in G^I *were used (guaranteed by the clique enumerator).*

(m) From (k) and (l) follows that Equation (9.4) *counts the total number of minimum path splittings.*

This concludes the proof of the theorem.

9.7 Summary, Discussion and Future Research

Decomposition of recorded routes into a minimum number of basic path components (BPC) delivers information that is useful for route choice set formation. Knapen et al. (2015c) provide an efficient algorithm to determine for a given path the size of minimum route splittings and a set of *splitVertexSuites*. Minimum path decompositions are generated by taking a single vertex from each *splitVertexSuite*. However, not every combination generates a valid solution. As a consequence not all vertices in a *splitVertexSuite* do occur in the same number of decompositions. Therefore, the concept of *importance* is introduced. It allows to quantify the probability that a given vertex (or a small ordered set (e.g. 2-tuple)) was selected as a *splitVertex*(set) by the traveler. Importance is useful to automatically identify way-points or landmarks from big data.

Determination of importance requires the enumeration of all minimum path decompositions. Efficient enumeration is performed by the following steps: (i) determination of minimal shortcuts to the given path thereby considering bypassed vertex sets as intervals (ii) constructing the interval graph (iii) enumerating all possible minimum clique covers (since the interval graph turns out to be an indifference graph, this can be done in polynomial time) and finally (iv) enumerating the path decompositions generated by each minimum clique cover.

Subsequent research will apply the reported technique to sets of GPS traces for which the number of basic path components was determined using the algorithm described in Knapen et al. (2015c). The resulting *importance* values will be correlated with road network and land-use data available in a GIS. The aim is to create a predictor for vertex importance in order to reduce interactive work.

The final goal is to create an extended branch-and-bound (Prato and Bekhor (2007)) or BFS-LE (Rieser-Schüssler et al. (2012)) route generator algorithm that, besides the currently used route attributes, also takes the number of BPC in the minimal path decomposition into account while making use of vertex tuples of sufficient importance.

9.8 Minimum Shortcut and Non-least-cost Edge Finder

Line 27 in algorithm 9.8.1 is used to minimize the work. At that point it is known that no minimum shortcut forks in a vertex $v \in [dsucc(start), v_j]$.

9.9 Clique Enumerator.

The interval tuple $\langle n, lhe, rhe \rangle$ contains the interval identifier (n) and the integers (lhe, rhe) identifying the left and right hand endpoints of the interval on P.
The algorithm requires as input a sorted list of intervals constituting a proper interval graph. The list is sorted so that the left hand endpoints (lhe) appear in increasing order.
The algorithm does not keep track of a set of intervals. The next clique to be output is identified by a reference to the first interval in the clique and a reference to the first interval not in the clique.

Algorithm 9.8.1 Minimum shortcut finder.

Require: graph G, path P in G

```
1: function findFirstJoin(P, from)        ▷ Search first join following vertex from
2:     join ← succ(P, from)
3:     while (join ≠ nil) ∧ (shortPath(P, from, join)) do
4:         join ← succ(P, join)
5:     end while
6:     return join
7: end function

8: function findLastFork(P, from)         ▷ Search first fork preceding vertex from
9:     fork ← pred(P, from)
10:    while (fork ≠ nil) ∧ (shortPath(P, fork, from)) do
11:        fork ← pred(P, fork)
12:    end while
13:    return fork
14: end function

15: start ← v_0
16: while start ≠ nil do
17:    v_j ← findFirstJoin(P, start)
18:    if v_j = nil then
19:        start ← nil                                   ▷ Path exhausted
20:    else if v_j = succ(P, start) then                 ▷ Non-least-cost edge detected
21:        output('nonShortestEdge', ⟨start, v_j⟩)
22:        start ← v_j                                   ▷ Skip over non-least-cost edge
23:    else
24:        v_f ← findLastFork(P, v_j)                    ▷ Regular shortest
25:        if v_f = pred(P, v_j) then                    ▷ Non-least-cost edge detected
26:            output('nonShortestEdge', ⟨v_f, v_j⟩)
27:            start ← v_j                               ▷ Skip over non-least-cost edge
28:        else
29:            output('minShortCut', ⟨v_f, v_j⟩)
30:            start ← succ(P, v_f)                      ▷ start ≠ v_j
31:        end if
32:    end if
33: end while
```

Algorithm 9.9.1 Clique Enumerator.

Require: List of Interval L where Interval is tuple $\langle n, lhe, rhe \rangle$

```
function OUTPUTCLIQUE(List Of Interval L, Interval firstIn, Interval firstBehind)
    clique ← ∅
    c ← firstIn
    while c ≠ firstBehind do
        clique ← clique ∪ {c}
        c ← succ(L, c)                          ▷ Successor of c in list L
    end while
    output(clique)
end function

fIn ← head(L)                                   ▷ First interval in clique
fNotIn ← succ(L, fIn)                           ▷ First interval not in clique
while fIn ≠ nil do
    outputClique(L, fIn, fNotIn)
    if fNotIn = nil then
        fIn ← nil                                                    ▷ Done
    else
        if fIn.rhe() < fNotIn.lhe() then
            fIn ← succ(L, fIn)      ▷ End of first in clique before begin of first behind
clique
        else
            fNotIn ← succ(L, fNotIn)          ▷ Contains the single vertex overlap case
        end if
    end if
end while
```

Chapter 10

Discussion and Future Research

10.1 The Use of Activity-based Models

Activity-Based Models (ActBMs) are powerful tools and the fact that they generate detailed microscopic results is tempting. The results are useful to feed other models but this shall be done with care, especially in cases where they feed models including coordination. This is not a qualification of the results of ActBM but a warning about model compatibility. Coupling models via data requires careful thought. It is always required to keep in mind for which goal the results were generated. Results can fulfill all the accuracy requirements with respect to the goal for which they were produced and at the same time not fulfill the requirements of another model using those results as an input. Arguments to investigate potential issues by simply connecting models, are given below.

10.1.1 Aggregating Applications

The Electric Vehicle (EV) related research shows how spatio-temporal information in schedules predicted by Activity-Based Models (ActBMs) can feed research to evaluate the electric power demand generated by EV. It shows the flexibility that results from the availability of individual schedules that can be aggregated in different ways according to the scenarios defined for several hypotheses. However, such aggregated result shall be used with care.

Location choice prediction seems to be difficult because of the homogeneity in land-use data (which in some cases reflects reality and in some cases is caused by the data ignoring relevant aspects of heterogeneity). As a consequence, for a given person at a given location, multiple opportunities having similar attraction to perform a given activity do exist. The effect is that the distributions for the predicted travel time and distance can accurately reflect the distributions extracted from surveys while the distribution of trips over OD pairs might deviate from reality.

The effect on the predicted spatio-temporal energy demand is estimated to be limited because the number of trips leaving origin locations and arriving at destination locations respectively, as well as the distance distribution, seem to be accurate. In the carpooling research however, the effect of the location choice inaccuracy is less obvious.

10.1.2 Applications involving Coordination

The probability for carpooling negotiation success depends on the trip similarity and this in turn depends on the OD pairs involved (as opposed to aggregated incoming and outgoing trips in the EV case). Current research at IMOB aims to determine the number of required FEATHERS runs to find sufficiently low standard deviation for the average over the runs, for particular aggregated quantities (number of activities in a TAZ, etc). This research is carried out for mutually independent individuals.

In the short term, a sensitivity analysis shall be set up in order to find out how sets of trips feasible for carpooling change over several FEATHERS runs. This is not a *simple* variable but a value that depends on similarity relationship between particular attributes in the schedules for two individuals.

For each location, the number of both outgoing and incoming trips feasible for carpooling (based on trip and time interval similarity) can be computed. This results in two vectors. It is to be found out how the variability of those vectors relates to the variability found for the aggregated quantities for mutually independent individuals mentioned above.

Current research on the Agent-Based Model (AgnBM) for carpooling uses travel time estimates taken from a TAZ based OD distance matrix used by the ActBM that produce the schedules. This is a simplification because carpooling research considers pairs of individuals for which the travel distances (times) might differ. The accuracy can be improved by using street addresses and computing the route length from the network. However, this comes at a large cost because in addition it requires the solution of a mini-VRP for each candidate driver in the carpool to pick-up all the

passengers. A cost-benefit analysis is to be performed. A first step to estimate the possible error is by determining, for a particular OD pair, the distribution of the travel distance (time) in a set where origin and destination locations are sampled from the sets of street addresses for the respective origin and destination zones. The variance could turn out to be large due to local accessibility features. The same considerations hold for the use of carpool parkings.

10.1.3 Behavioral Models

The WithIn Day Re-Scheduling (WIDRS) framework is aimed at evaluation of rescheduling behavior models. Those models are application specific. Examples are: (i) schedule adaptation caused by a one-off incident (used as test case in WIDRS), (ii) schedule adaptation using learning in MATSim, (iii) schedule adaptation due to carpooling negotiation and (iv) schedule adaptation due to changing environment parameters (e.g. cost optimization in EV and congestion charging context). Some but not all models can be evaluated by means of WIDRS because that is hybrid model.

The major challenge however, is the determination of context based marginal utility i.e. depending on time-of-week, location, available resources, internal deadlines etc. This requires the concept of (perceived) time pressure. For particular applications (carpooling, EV charging) monetary value of time is required as well. This is challenging since a lot of detailed time recording is required to collect data. In an ongoing project focusing on flexible working (time and place) a GPS trace based *prompted recall* tool, requiring a minimum of input, has been deployed. However, it seems to be particularly challenging to find sufficient participants prepared to cooperate.

10.2 Route Properties Research

Following hypothesis was formulated and verified: *people tend to compose their routes for utilitarian trips, as a small concatenation of least-cost paths in the network. Verification of the hypothesis required the creation of several tools.*

Analyzing the structure of a route in a network might look like an obvious idea. However, to the best of our knowledge, it was not published before neither in transportation science nor in mathematics (graph theory). Possible reasons are: (i) the required datasets are becoming available only recently and (ii) a non-trivial (amount of) software is required for this research. For example, a map matching tool with known properties (as opposed to black box software) and fulfilling specific requirements was needed. This need resulted in a tool using a new map matching technique.

Both path splitting methods developed in this doctoral thesis research, contribute to enhancement of the route choice model. The method to determine the *minimum path decomposition size* allows to enhance route choice models by adding a route quality assessment based on the path structure. The method to *enumerate all minimum path splittings*, contributes to automatic way-point identification; this avoids interactive work and the need for operator judgment.

For several cases, the distribution for the size of the minimum path decomposition was determined. In one case, synthetic routes were created by sampling an origin, a direction and a trip distance. Trips were constructed by selecting at each junction the network link for which the direction approximates best the direction to the given one (while avoiding cycles). Real routes derived from both car traces (in Milano, Italy) and person traces (in Flanders, Belgium) were analyzed. In all cases the distribution for the minimum path decomposition size was determined. The Milano car traces show higher probabilities for smaller decompositions than the Flemish person traces. This can be caused by the fact that the car traces are subtraces of person traces; this however cannot be stated with certainty because of the different context of trip recording.

The distributions for the synthetic and real traces seem to heavily differ whereas the distributions for the real routes are mutually similar. This is particularly interesting because of the different properties (car traces in a large city vs. person traces in a lower density region). For the real data, less than five percent of the routes consist of more than five least-cost paths. The difference between the distributions for the synthetic and the real routes shows that the final result is not a technical-mathematical artifact of the method. Therefore, it can be concluded that the formulated hypothesis holds.

As soon as the size of the minimal decompositions is known, it is interesting to find out how to determine the minimal route splittings. In general, there are multiple minimal decompositions for a given path. However, it seems to be impossible to generate complete classes of decompositions at once. It is only possible to determine the size N of the minimum decomposition and to identify $N-1$ mutually disjoint sets of vertices. Exactly one vertex is to be chosen from each of the $N-1$ sets in order to find a vector of $N-1$ split vertices that constitute the boundaries between successive least-cost path components. Not every combination is a valid one. This means that the dataset contains information that can be useful. A method is designed to enumerate all possible minimum decompositions for a given path, in polynomial time.

The occurrence frequency of a vertex as a split vertex in the minimum decompo-

sitions for a given set of paths, is used to define the *importance* of the vertex. The concept of *importance of a vertex for use as a split vertex in a given set of paths*, was defined. Importance is defined as the relative occurrence frequency of a vertex as a split vertex in all minimum decompositions of all paths in the set.

Future research is required to find out whether correlations can be found between the vertex GIS attributes and its importance in a particular set of paths. It would be interesting when a predictor for importance can be built making use of GIS attributes. The reason is that this constitutes an automatic method for way-point identification. Note that if the correlations seem to be region specific, they are still interesting. Indeed, even a large set of GPS traces constitutes only a small sample in the context of importance determination. This is explained using numerical data. The OpenStreetMap (OSM) network for Flanders has about $6 \cdot 10^5$ links and about $4.5 \cdot 10^5$ nodes. Assume that only 10% of the links are used as origin and/or destination (hence $6 \cdot 10^4$ of them). Then there are $36 \cdot 10^8$ OD pairs. If 3 possible routes are assumed (which definitely is an under-estimation) for each OD pair, about $1 \cdot 10^{10}$ routes could be found in an infinitely large dataset of recorded trips for the region.

Now assume a dataset for one month for the Flemish population: $6 \cdot 10^6$ individuals making 3 trips per day for 31 days results in $5.4 \cdot 10^8$ trips. This only represents 5% of the $1 \cdot 10^{10}$ possible routes.

If a dataset containing one million ($1 \cdot 10^6$) trips is available, it only represents $1 \cdot 10^{-4}$ of the possible routes.

From route splitting results, it is derived that it is reasonable to assume 50 vertices per trip. Hence $5 \cdot 10^7$ vertex uses will be in the one million trips dataset. The average vertex use frequency in the dataset is $5 \cdot 10^7 / 4.5 \cdot 10^5$ which is in the order of magnitude of 100. Due to clustering of routes, it is not certain that each vertex will be used at least once. On the other hand, the ratio is large enough to expect feasibility to determine correlations suitable for prediction of the importance for the vertices not used in the dataset.

Finally, importance is a route-set specific concept. This can be interesting and challenging. It allows to investigate importance for the sets of a given traveler, a given transportation mode, specific OD pairs, period of trip start time and other sets.

Appendix A

Curriculum Vitae

ir Luk Knapen

Hasselt University Campus Diepenbeek
Transportation Research Institute (IMOB)
Wetenschapspark 5 bus 6
BE-3590 Diepenbeek Belgium

Tel: +32 (0)11 26 91 26
Fax: +32 (0)11 26 91 99
E-mail: luk.knapen@uhasselt.be

PERSONAL DETAILS

Place of birth	Sint-Truiden, Belgium
Date of birth	1951-Jul-26
Marital status	Married
Nationality	Belgian

EDUCATIONAL BACKGROUND

1978 – 1979	Civil Engineer Applied Mathematics
	Institute: KULeuven Computer Science
	Grade: All Exams (80%), no thesis
1969 – 1974	Civil Engineer (Construction)
	Institute: KULeuven Department Construction Engineering
	Grade: Onderscheiding (Distinction)

WORK EXPERIENCE

2010-Nov-10 - Today	UHasselt/IMOB
	Job title: Researcher - PhD Student (supervisor prof. Dr ir Tom Bellemans)
2014-Jan-12 - 2014-Jan-25	Université de technologie Belfort-Montbéliard
	Research visit: Janus Agent-based carpool simulation model
2013-May-20 - 2013-Jun-20	Université de technologie Belfort-Montbéliard
	Research visit: Janus Agent-based carpool simulation model
1998-Aug-14 – 2010-Nov-09	fks BVBA
	Job title: Managing Partner (co-founder, business manager)
1983-Jan-01 - 1998-Aug-13	u-Soft Information Technology PVBA / SCOPE NV (co-founder)
	Job title: Technical Director
1981-Jun-01 – 1982-Nov-15	Libost NV, Hasselt
	Job title: Software Development Engineer
1981-May-01 - 1981-May-31	Belfotop, Wemmel
	Job title: Software Development Engineer
1978-Dec-01 - 1981-Apr-30	KULeuven Department of Construction Engineering
	Job title: Assistant (Prof dr ir F. Mortelmans)
1980-May-19 - 1980-Jun-13	Royal Institute of Technology Dept. Telecommunication Computer Systems Stockholm (NFWO-IBM grant)
1977-Oct-01 – 1978-Nov-30	KULeuven Department of Construction Engineering
	Job title: 1/3 time Assistant (Prof dr ir F. Mortelmans)
	Job title: 2/3 time Student Applied Mathematics
1974-Sep-01 – 1977-Sep-30	KULeuven Department of Construction Engineering
	Job title: Assistant (Prof dr ir F. Mortelmans)

Appendix B

List of Publications

Publication Status Codes	
Code	Meaning
ACC	Peer-reviewed Accepted
PRS	Peer-reviewed Accepted and Presented
prs	Presented (no peer review)
PUB	Peer-reviewed Accepted and Published
REV	Peer-reviewed and being Revised
SUB	Submitted for peer-review

Report Kind	
Code	Meaning
DATASIM	DATASIM Research Report (Project Deliverable)
IMOB	Instituut voor Mobiliteit, Transportation Research Institute (Hasselt University) (IMOB) Internal Research Report

B.1 Peer-Reviewed Journal Publications

Knapen et al. (2012b)	PUB	Using Activity-Based Modeling to Predict Spatial and Temporal Electrical Vehicle Power demand in Flanders
Knapen et al. (2013d)	PUB	Exploiting Graph-theoretic Tools for Matching in Carpooling Applications
Knapen et al. (2013c)	PUB	Within Day Rescheduling Micro-simulation combined with Macrosimulated Traffic
Knapen et al. (2014b)	PUB	Scalability issues in optimal assignment for carpooling

B.2 Submissions to Peer-Reviewed Journals

Knapen et al. (2015a)	SUB	Efficient Offline Map Matching of GPS Recordings Using Global Trace Information (Submitted to: Transportation Research - Part C: Emerging Technologies)
Knapen et al. (2015c)	SUB	Determining Structural Route Components from GPS Traces (Submitted to: Transportation Research - Part B: Methodological)
Knapen et al. (2015b)	SUB	Enumeration of Minimal Path Decompositions to Support Route Choice Set Generation (Submitted to: Transportation Research - Part B: Methodological)

B.3 Peer Reviewed Conference Papers

Knapen et al. (2011)	PRS	Activity-based models for countrywide electric vehicle power demand calculation
Knapen et al. (2012a)	PRS	Analysis of the Co-routing Problem in Agent-based Carpooling Simulation
Knapen et al. (2012d)	PRS	Framework to Evaluate Rescheduling due to Unexpected Events in an Activity-Based Model
Knapen et al. (2012c)	PRS	Using Activity-Based Modeling to Predict Spatial and Temporal Electrical Vehicle Power demand in Flanders
Knapen et al. (2013b)	PRS	Estimating Scalability Issues while Finding an Optimal Assignment for Carpooling
Knapen et al. (2014a)	PRS	Canonic route splitting

B.4 Reports (as first Author)

Knapen (2012b)	IMOB	WIDRS, model design, software specification
Knapen (2012a)	DATASIM	Carpooling, model design
Knapen and Galland (2013)	IMOB	Carpooling agent-based, authored at UTBM
Knapen (2013)	DATASIM	Data used to feed schedule generators (*data4simulation*)
Knapen et al. (2013a)	DATASIM	D3.1 Behaviorally-sensitive simulator design ready for the calculation of Mobility-EV scenarios (*FEATHERS+*)
Knapen et al. (2014d)	DATASIM	Notes on the use of radiation laws in simulation
Knapen et al. (2014c)	DATASIM	D3.2 Prototype development of a fully integrated data-driven simulator

B.5 Peer-Reviewed Book Chapters

Knapen et al. (2014e)	PUB	Agent-based modeling for carpooling. (DATASIM)

B.6 Peer-Reviewed Conference Papers co-authored with IMOB first Author

Bellemans et al. (2012)	PRS	An Agent-Based Model to Evaluate Carpooling at Large Manufacturing Plants
Cho et al. (2012)	PRS	A Conceptual Design of an Agent-based Interaction Model for the Carpooling Application
Usman et al. (2014b)	ACC	A framework for electric vehicle charging strategy optimization tested for travel demand generated by an activity-based model (ITSC(IEEE) accepted, not presented due to visa problem)
Usman et al. (2014a)	PRS	Effect of electrical vehicles charging cost optimization over charging cost and travel timings
Hussain et al. (2014)	PRS	Organizational and Agent-based Automated Negotiation Model for Carpooling
	PRS	Usman's electric vehicle papers
Cich et al. (2015)	PRS	TRIP/STOP Detection in GPS Traces to Feed Prompted Recall Survey
Raza et al. (2015)	PRS	Diary Survey Quality Assessment Using GPS Traces
Hussain et al. (2015a)	PRS	An Agent-based Negotiation Model for Carpooling: A Case Study for Flanders (Belgium)
Hussain et al. (2015b)	PRS	Agent-based Simulation Model for Long-term Carpooling: Effect of Activity Planning Constraints

B.7 Non-Peer-Reviewed Conference Papers as co-author, IMOB first author

Vuurstaek et al. (2015)	prs	Modelling hospital visitors for the city of Leuven as input for a FEATHERS-MATSim simulation
Hussain et al. (2015d)	prs	An Agent-based Model for Carpooling: Effect of Strict Timing Constraints on Carpooling Trips
Hussain et al. (2015c)	prs	Agent-based Negotiation Model for Long-term Carpooling: A Flexible Mechanism for Trip Departure Times

B.8 Peer-Reviewed Journal Papers as co-author, external first author

Galland et al. (2013b)	PUB	Multi-Agent Simulation of Individual Mobility Behavior in Carpooling using the Janus and JaSim Platforms
Galland et al. (2014)	PUB	Simulation of Carpooling Agents with the Janus Platform

B.9 Submissions to Peer-Reviewed Journals as co-author, external first author

Alvaro-Hermana et al. (2015)	SUB	Peer to Peer Energy Trading with Electric Vehicles (Submitted to: IEEE Transactions on Intelligent Transportation Systems)

B.10 Peer-Reviewed Conference Papers as co-author, external first author

Keren et al. (2012)	PRS	Exploiting Graph-theoretic Tools for Matching and Partitioning of Agent Population in an Agent-based Model for Traffic and Transportation Applications
Ridder et al. (2013)	PRS	Applying an Activity based Model to Explore the Potential of Electrical Vehicles in the Smart Grid
Galland et al. (2013a)	PRS	Simulation Model of Carpooling with the Janus Multiagent Platform
D'hulst et al. (2012)	PRS	Decentralized Coordinated Charging of Electric Vehicles Considering Locational and Temporal Flexibility
Ben-Arroyo Hartman et al. (2014)	PRS	Theory and Practice in Large Carpooling Problems
Alvaro et al. (2014)	PRS	Vehicle to vehicle energy exchange in smart grid applications
Gonzales et al. (2014)	PRS	Determining Electric Vehicle Charging Point Locations Considering Drivers' Daily Activities

B.11 Peer-Reviewed Book Chapters as co-author, external first author

De Ridder et al. (2013)	PUB	Electric Vehicles in the Smart Grid

Appendix C

Supervised Student Work

C.1 Supervised PhD Theses

2013-today	Muhammad Usman Shaukat	Activity-based Models and EV Charging Behavior
2013-today	Iftikhar Hussain	Agent-based Models for Carpooling
2014-today	Glenn Cich	Smart Public Transportation: Provider Models
2014-today	Jan Vuurstaek	Smart Public Transportation: Customer Models

C.2 Supervised Master Theses

2012-2013	Muhammad Usman Shaukat	Framework for within-day Rescheduling due to Unexpected Incidents in Transportation Networks. (Master student at Linköping University, Sweden)
2013-2014	Abbas Golmohammadi	The impact of electric vehicles on travel behaviour. Formulate and estimate choice models for use of electric vehicles.
2014-2015	Joris Huijbregts	Voorspelling van de bezettingsgraad van carpoolplaatsen. (Huijbregts (2015))
2014-2015	Ben Ceyssens	Optimalisatie leerlingenvervoer in het buitengewoon onderwijs.
2014-2015	Muhammad Arsalan Khan	(Co-promoter): Activity-based models: agent negotiation to cooperate for carpooling. (Khan (2015))
2014-2015	Muhammad Arsalan Effendi	Multi Agent Transport Simulation (MATSim): Network loading, Traffic evolver for Flanders

C.3 Co-Supervised Master Theses

2011-2012	Wim Vanderheyden	(Co-promoter) Development of an agent-based model for electric vehicle penetration of the European fleet over the next decennium.
2014-2015	Julia Naumova	(with Jan Vuurstaek, MSc) Building Traffic Models Using Freely Available Data.
2014-2015	Syed Muhammad Noman	(with dr ir Bruno Kochan) Traffic assignment using DTALite : Model setup for Flanders.

C.4 Supervised Bachelor Theses

2011-2012	Jochen Roosen	Laadpalen op parkeerterreinen van bedrijven.
2011-2012	Peter Hendrix	Mogelijkheid tot laden van elektrische voertuigen op eigen terrein.
2012-2013	Wouter van Haperen	OPENSTREETMAP: An alternative data source for mobility studies ? (van Haperen (2013))
2013-2014	Julia Naumova, Albina Khusainova	Open Source Tools for Transportation and Mobility Management.
2013-2014	Karlien Van Aerschot	Analyse van het verband tussen de beperkte interesse in carpoolen en de inflexibiliteit van agenda's. (Van Aerschot (2014))
2013-2014	Natasha Kovac	Verwerking van GPS traces van elektrische voertuigen en bijhorende dagboeken.

C.5 Supervised Mobility Science Internships

2012-2013	Samaneh Hosseinzadeh Bahreini	Evaluation of outputs of the Feathers system in relation to electric vehicles
2013-2014	Nick Strackx	Using Open Source data and Open Source Software for traffic simulation. Open Street Map Data and NeXTA/DTALite for Flemish Cities.
2013-2014	Muhammad Arsalan Effendi	Making MATSim operational for the Flemish Region
2013-2014	Ali Raza	(co-authored Raza et al. (2015)) Prompted Recall Technique.
2014-2015	Geoffrey Kizito Dambi Filis	Statistical analysis of route splitting results.
2014-2015	Wim Casteels	Alternatieve oplossingen voor aanpassing openbaar vervoer te Neder-Over-Heembeek.

C.6 Supervised International Internships

2011-July	Saurabh Aggarwal	FEATHERS-TransCAD loop to determine OD-travel-time matrix.
2013-July	Rabab Kamal	Analysis of EV calculations for Flanders

Bibliography

Abdallah, F., Nassreddine, G., Denoeux, T., 2011. A Multiple-Hypothesis Map-Matching Method Suitable for Weighted and Box-Shaped State Estimation for Localization. IEEE Transactions on Intelligent Transportation Systems 12, 1495–1510.

Agatz, N., Erera, A., Savelsbergh, M., Wang, X., 2010. Sustainable Passenger Transportation: Dynamic Ride-Sharing. Research Paper ERIM Report Series Reference No. ERS-2010-010-LIS. Erasmus University of Rotterdam Erasmus Research Institute of Management.

Agatz, N., Erera, A., Savelsbergh, M., Wang, X., 2012. Optimization for dynamic ride-sharing: A review. European Journal of Operational Research 223, 295 – 303.

Agatz, N., Erera, A.L., Savelsbergh, M.W.P., Wang, X., 2011. Dynamic Ride-Sharing: a Simulation Study in Metro Atlanta, in: Procedia - Social and Behavioral Sciences, pp. 532–550.

Alvares, L.O., Bogorny, V., Kuijpers, B., de Macedo, J.A.F., Moelans, B., Vaisman, A., 2007. A model for enriching trajectories with semantic geographical information, in: Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems, ACM, New York, NY, USA. pp. 22:1–22:8.

Alvaro, R., Gonzales, J., Gamallo, C., Fraile-Ardanuy, J., Knapen, L., Janssens, D., 2014. Vehicle to vehicle energy exchange in smart grid applications, in: ICCVE2014, IEEE, Vienna, Austria.

Alvaro-Hermana, R., Fraile-Ardanuy, J., Zufiria, P., Knapen, L., Janssens, D., 2015. Peer to Peer Energy Trading with Electric Vehicles. IEEE Transactions on Intelligent Transportation Systems .

Andrienko, G., Andrienko, N., Hurter, C., Rinzivillo, S., Wrobel, S., 2011. From Movement Tracks through Events to Places: Extracting and Characterizing Signif-

icant Places from Mobility Data, in: IEEE Conference on Visual Analytics Science and Technology, IEEE, Providence, Rhode Island, USA.

Arentze, T., Pelizaro, C., Timmermans, H., 2005. Implementation of a model of dynamic activity-travel rescheduling decisions: an agent-based micro-simulation framework, in: Proceedings of CUPUM 05, Computers in Urban Planning and Urban Management,, London.

Balmer, M., Rieser, M., Meister, K., Charypar, D., Lefebvre, N., Nagel, K., 2009. MATSim-T: Architecture and Simulation Times., in: Multi-Agent Systems for Traffic and Transportation Engineering., Igi global edition. pp. 57–78.

Bekhor, S., Ben-Akiva, M., Ramming, M., 2006. Evaluation of choice set generation algorithms for route choice models. Annals of Operations Research 144, 235–247. 10.1007/s10479-006-0009-8.

Bekhor, S., Dobler, C., Axhausen, K.W., 2011. Integration of Activity-Based and Agent-Based Models: Case of Tel Aviv, Israel. Transportation Research Record: Journal of the Transportation Research Board , 38–47.

Bellemans, T., Bothe, S., Cho, S., Giannotti, F., Janssens, D., Knapen, L., Körner, C., May, M., Nanni, M., Pedreschi, D., Stange, H., Trasarti, R., Yasar, A.U.H., Wets, G., 2012. An Agent-Based Model to Evaluate Carpooling at Large Manufacturing Plants, in: Procedia Computer Science, pp. 1221 – 1227. ANT 2012 and MobiWIS 2012.

Bellemans, T., Kochan, B., Janssens, D., Wets, G., Arentze, T., Timmermans, H., 2010. Implementation Framework and Development Trajectory of FEATHERS Activity-Based Simulation Platform. Transportation Research Record: Journal of the Transportation Research Board Volume 2175, 111–119.

Bellemans, T., Kochan, B., Janssens, D., Wets, G., Timmermans, H.J., 2008. Field Evaluation of Personal Digital Assistant Enabled by Global Positioning System : Impact on Quality of Activity and Diary Data. TRB Research Record 2049, 136–143.

Ben-Arroyo Hartman, I., Keren, D., Abu Dbai, A., Cohen, E., Knapen, L., Yasar, A.U.H., Janssens, D., 2014. Theory and Practice in Large Carpooling Problems, in: Proceedings of the 5th International Conference on Ambient Systems, Networks and Technologies (ANT-2014), Procedia Computer Science, Elsevier, Hasselt, Belgium.

Bierlaire, M., Chen, J., Newman, J., 2013. A probabilistic map matching method for smartphone {GPS} data. Transportation Research Part C: Emerging Technologies 26, 78 – 98.

Binding, C., Sundstroem, O., 2011. A simulation environment for Vehicle-to-grid Integration Studies, in: Summer Computer Simulation Conference 2011, SCSC, Den Haag, NL. p. 248.

van Bladel, K., Bellemans, T., Janssens, D., Wets, G., 2009. Activity Travel Planning and Rescheduling Behavior: Empirical Analysis of Influencing Factors. Transportation Research Record: Journal of the Transportation Research Board , 135–142.

van Bladel, K., Bellemans, T., Wets, G., Arentze, T., Timmermans, H., 2006. Fitting S-Shaped Activity Utility Functions Based on Stated-Preference Data, in: 11th International Conference on Travel Behaviour Research, Kyoto, Kyoto.

Bliek, F., Albert van den Noort, Roossien, B., Kamphuis, R., de Wit, J., van de Velde, J., Eijgelaar, M., 2010. PowerMatching City, A living lab smart grid demonstration, in: Innovative Smart Grid Technologies Conference Europe (ISGT Europe), 2010 IEEE PES, IEEE, Goteborg, Sweden. pp. 1–8.

Bonnifait, P., Laneurit, J., Fouque, C., Dherbomez, G., 2009. Multi-hypothesis Map-Matching using Particle Filtering, in: 16th World Congress for ITS Systems and Services, HAL Id: hal-00445673, Stockholm, Sweden. pp. 1–8.

Booth, K., Lueker, G., 1975. Linear Algorithms Test to Recognize Interval Graphs and Test for the Consecutive Ones Property .

Bovy, P.H.L., 2009. On Modelling Route Choice Sets in Transportation Networks: A Synthesis. Transport Reviews 29, 43–68,.

Brakatsoulas, S., Pfoser, D., Salas, R., Wenk, C., 2005. On Map-Matching Vehicle Tracking Data, in: Proceedings of the 31st VLDB Conference, Trondheim, Norway, 2005, Trondheim, Norway.

Buliung, R., Soltys, K., Habel, C., Lanyon, R., 2009. The "Driving" Factors Behind Successful Carpool Formation and Use. Transportation Research Record .

Chen, J., Bierlaire, M., Flötteröd, G., 2011. Probabilistic multi-modal map matching with rich smartphone data, in: STRC 2011.

Cho, S., Yasar, A.U.H., Knapen, L., Bellemans, T., Janssens, D., Wets, G., 2012. A Conceptual Design of an Agent-based Interaction Model for the Carpooling Application, in: Procedia Computer Science, pp. 801 – 807. ANT 2012 and MobiWIS 2012.

Chun, H.W., Wong, R.Y., 2003. N* - an agent-based negotiation algorithm for dynamic scheduling and rescheduling. Advanced Engineering Informatics , 1–22.

Cich, G., Knapen, L., Bellemans, T., Janssens, D., Wets, G., 2015. TRIP/STOP Detection in GPS Traces to Feed Prompted Recall Survey, in: Procedia Computer Science, Elsevier, Greenwich (London). pp. 262–269.

Clement-Nyns, K., Haesen, E., Driesen, J., 2009. Analysis of the Impact of Plug-In Hybrid Electric Vehicles on Residential Distri bution Grids by using Quadratic and Dynamic Programming. World Electric Vehicle Journal 3.

Cools, M., Declercq, K., Janssens, D., Wets, G., 2011. Onderzoek Verplaatsingsgedrag Vlaanderen 4.2 (2009-2010) : Tabellenrapport. Technical Report. IMOB. Diepenbeek, Belgium.

Cornelis, E., Malchair, A., Asperges, T., Ramaekers, K., 2007. COCA : Company Cars Analysis (Rapport final).

Cui, X., Liu, C., Kim, H.K., Kao, S.C., Tuttle, M., Bhaduri, B., 2011. A Multi Agent-Based Framework for Simulating Household PHEV Distribution and Electric Distribution Network Impact, in: 2011 TRB 90st Annual Meeting Compendium of Papers, Washington, D.C.

Davies, J., Kurani, K., 2011. Estimated Marginal Impact of Workplace Charging on Electricity Demand and Charge Depleting Driving. Scenarios based on Plausible Early Market Commuters' Use of a 5kwh Conversion PHEV, in: 2011 TRB 90st Annual Meeting Compendium of Papers, Transportation Research Board, Washington, D.C.

De Ridder, F., D'Hulst, R., Knapen, L., Janssens, D., 2013. Electric Vehicles in the Smart Grid, in: Data Science and Simulation in Transportation Research. IGI Global, Hasselt. InfoSci, pp. 340–363.

D'hulst, R., De Ridder, F., Claessens, B., Knapen, L., Janssens, D., 2012. Decentralized Coordinated Charging of Electric Vehicles Considering Locational and Temporal Flexibility. IEEE TRANSACTIONS ON SMART GRID .

Dijkstra, E., 1959. A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271.

EABEV, 2010. Energy consumption, CO2 emissions and other considerations related to Battery Electric Vehicles (http://www.going-electric.org/).

Elgowainy, A., Han, J., Poch, L., Wang, M., Vyas, A., Mahalik, M., Rousseau, A., 2010. Well-to-Wheels Energy Use and Greenhouse Gas Emissions Analysis of Plug-In Hybrid Electric Vehicles. Technical Report ANL/ESD/10-1. Argonne National Laboratory.

Federal Planning Bureau, B., 2009. Transportdatabanken : Indicator PARC010 (http://www.plan.be/). Technical Report.

Feng, T., Timmermans, H.J., 2013. Map Matching of GPS Data with Bayesian Belief Networks, in: Proceedings of the Eastern Asia Society for Transportation Studies, Eastern Asia Society for Transportation Studies, Taipei. p. 13.

Frejinger, E., Bierlaire, M., 2007. Capturing correlation with subnetworks in route choice models. Transportation Research Part B: Methodological 41, 363 – 378.

Frejinger, E., Bierlaire, M., Ben-Akiva, M., 2009. Sampling of alternatives for route choice modeling. Transportation Research Part B: Methodological 43, 984 – 994.

Fulkerson, D.R., Gross, O.A., 1965. Incidence matrices and interval graphs. Pacific Journal of Mathematics 15, 835–855.

Furletti, B., Cintia, P., Renso, C., Spinsanti, L., 2013. Inferring human activities from GPS tracks, in: UrbComp 13 Proceedings of the second ACM SIGKDD International Workshop on Urban Computing, ACM, Chicago.

Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.E., Wang, X., Koenig, S., 2013. Ridesharing: The state-of-the-art and future directions. Transportation Research Part B: Methodological 57, 28 – 46.

Galland, S., Gaud, N., Yasar, A.u.h., Knapen, L., Janssens, D., Lamotte, O., 2013a. Simulation Model of Carpooling with the Janus Multiagent Platform, in: Yasar, A.u.h., Knapen, L. (Eds.), 2nd International Workshop on Agent-based Mobility, Traffic and Transportation Models, Methodologies and Applications (AMBTRANS13), Elsevier, Halifax, Nova Scotia, Canada.

Galland, S., Knapen, L., Yasar, A.U.H., Gaud, N., Janssens, D., Lamotte, O., Wets, G., Koukam, A., 2013b. Multi-Agent Simulation of Individual Mobility Behavior in Carpooling using the Janus and JaSim Platforms. Transportation Research Part C .

Galland, S., Yasar, A., Knapen, L., Gaud, N., Bellemans, T., Janssens, D., 2014. Simulation of Carpooling Agents with the Janus Platform. Journal of Ubiquitous Systems and Pervasive Networks (JUSPN) .

Gan, L.P., Recker, W., 2008. A mathematical programming formulation of the household activity rescheduling problem. Transportation Research Part B 42, 571–606.

Gardi, F., 2007. The Roberts characterization of proper and unit interval graphs. Discrete Mathematics 307, 2906 – 2908.

Garey, M.R., Johnson, D.S., 1979. Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York, NY, USA.

Gaud, N., Galland, S., Hilaire, V., Koukam, A., 2009. Programming Multi-Agent Systems, Springer-Verlag, Berlin, Heidelberg, pp. 104–119.

Giannotti, F., Nanni, M., Pinelli, F., Pedreschi, D., 2007. Trajectory pattern mining, in: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, New York, NY, USA. pp. 330–339.

Gilbert, N., Troitzsch, K.G., 2005. Simulation for the Social Scientist. Open University Press.

Gonzales, J., Alvaro, R., Gamallo, C., Fuentes, M., Fraile-Ardanuy, J., Knapen, L., Janssens, D., 2014. Determining Electric Vehicle Charging Point Locations Considering Drivers' Daily Activities, in: Procedia Computer Science, Elsevier, Hasselt, Belgium. pp. 647–654.

Greenfeld, J., 2002. Matching GPS Observations to Locations on a Digital Map, in: TRB 2002 Annual Meeting, TRB (Transportation Research Board), Washington, D.C.. p. 13.

Guo, J., Nandam, S., Adams, T., 2012. A Data Collection Framework for Exploring the Dynamic Adaptation of Activity-Travel Decisions, TRB (Transportation Research Board), Tampa, Florida.

Hadley, S., Tsvetkova, A., 2008. Potential Impacts of Plug-in Hybrid Electric Vehicles on Regional Power Generation. Technical Report ORNL/TM-2007/150. OAK RIDGE NATIONAL LABORATORY.

Haklay, M., 2010. How good is volunteered geographical information? A comparative study of OpenStreetMap and Ordnance Survey datasets. Environment and Planning B: Planning and Design 37, 682–703.

Haklay, M., Basiouka, S., Antoniou, V., Ather, A., 2010. How Many Volunteers Does it Take to Map an Area Well? The Validity of Linus' Law to Volunteered Geographic Information. Maney Publishing (c) The British Cartographic Society 47, 315–322.

van Haperen, W., 2013. OPENSTREETMAP: An alternative data source for mobility studies ? Bachelor Mobility Science. Hasselt University. Diepenbeek, Belgium.

Hoogendoorn-Lanser, S., 2006. A Rule-based Approach to Route Choice Set Generation, Amsterdam.

Huijbregts, J., 2015. Voorspelling van de bezettingsgraad van carpoolplaatsen. Master Thesis. Hasselt University. Diepenbeek, Belgium.

Hussain, I., Knapen, L., Bellemans, T., Janssens, D., Wets, G., 2015a. An Agent-based Negotiation Model for Carpooling: A Case Study for Flanders (Belgium), in: TRB 2015 Annual Meeting, TRB (Transportation Research Board), Washington, DC, USA.

Hussain, I., Knapen, L., Galland, S., Janssens, D., Bellemans, T., Yasar, A.U.H., Wets, G., 2014. Organizational and Agent-based Automated Negotiation Model for Carpooling. Procedia Computer Science 37, 396 – 403. The 5th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN-2014)/ The 4th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH 2014)/ Affiliated Workshops.

Hussain, I., Knapen, L., Galland, S., Yasar, A.U.H., Bellemans, T., Janssens, D., Wets, G., 2015b. Agent-based Simulation Model for Long-term Carpooling: Effect of Activity Planning Constraints, in: Procedia Computer Science, pp. 412 – 419. The 6th International Conference on Ambient Systems, Networks and Technologies (ANT-2015), the 5th International Conference on Sustainable Energy Information Technology (SEIT-2015).

Hussain, I., Knapen, L., Khan, M.A., Bellemans, T., Janssens, D., Wets, G., 2015c. Agent-based Negotiation Model for Long-term Carpooling: A Flexible Mechanism for Trip Departure Times, in: Urban Transport XXI, WIT Press, Valencia, Spain.

Hussain, I., Knapen, L., Kochan, B., Bellemans, T., Janssens, D., Wets, G., 2015d. An Agent-based Model for Carpooling: Effect of Strict Timing Constraints on Carpooling Trips, in: BIVEC/GIBET Transport Research Day 2015, Eindhoven.

Iwan, L., Safar, M., 2010. Pattern mining from movement of mobile users. Journal of Ambient Intelligence and Humanized Computing 1, 295–308.

Jang, Y., Chiu, Y.C., 2010. Within-Day Schedule Adjustment Decision Process in a Dynamic Traffic simulation and Assignment Framework, in: The 3rd Conference on Innovations in Travel Modeling, Tempe, Arizona.

Jenelius, E., Mattsson, L.G., Levinson, D.M., 2011. Traveler delay costs and value of time with trip chains, flexible activity scheduling and information. Transportation Research Part B: Methodological 45, 789–807.

Joh, C.H., 2002. Modeling Individuals' Activity–Travel Rescheduling Heuristics : Theory and Numerical Experiments. Transportation Research Board of the National Academies , 16 – 25.

Joh, C.H., 2004. Measuring and Predicting Adaptation in Multidimensional Activity-travel Patterns. PhD Thesis. TUE. Eindhoven.

Kamar, E., Horvitz, E., 2009. Collaboration and Shared Plans in the Open World: Studies of Ridesharing, in: Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence, IJCAI Organization, Pasadena.

Kang, J.E., Recker, W., 2009. An activity-based assessment of the potential impacts of plug-in hybrid electric vehicles on energy and emissions using 1-day travel data. Transportation Research Part D: Transport and Environment 14, 541 – 556.

Kaplan, S., Prato, C.G., 2010. Joint modeling of constrained path enumeration and path choice behavior: a semi-compensatory approach, in: Proceedings of the European Transport Conference. Association for European Transport.

Kazagli, E., Bierlaire, M., 2014. Revisiting Route Choice Modeling: A Multi-Level Modeling Framework for Route Choice Behavior, in: STRC 2014, Ascona.

Keren, D., Yasar, A.U.H., Knapen, L., Cho, S., Bellemans, T., Janssens, D., Wets, G., Schuster, A., Sharfman, I., 2012. Exploiting Graph-theoretic Tools for Matching and Partitioning of Agent Population in an Agent-based Model for Traffic and Transportation Applications, in: Procedia Computer Science, Niagara Falls, Canada. pp. 833 – 839.

Khan, M.A., 2015. Activity-based models: agent negotiation to cooperate for carpooling. Master Thesis. Hasselt University. Diepenbeek, Belgium.

Knapen, L., 2012a. Notes on Collaboration - Carpooling Model. IMOB internal report. IMOB. Diepenbeek, Belgium.

Knapen, L., 2012b. Within day rescheduling without detailed routing. IMOB internal report. IMOB. Diepenbeek, Belgium.

Knapen, L., 2013. Data used to feed schedule generators (data4simulation). IMOB DataSIM Research Report. IMOB. Diepenbeek, Belgium.

Knapen, L., Bellemans, T., Janssens, D., Wets, G., 2014a. Canonic route splitting, Diepenbeek, Belgium.

Knapen, L., Bellemans, T., Janssens, D., Wets, G., 2015a. Efficient Offline Map Matching of GPS Recordings Using Global Trace Information. Submitted to Transportation Research - Part C: Emerging Technologies .

Knapen, L., Ben-Arroyo Hartman, I., Bellemans, T., Janssens, D., Wets, G., 2015b. Enumeration of Minimal Path Decompositions to Support Route Choice Set Generation. Submitted to Transportation Research Part B: Methodological .

Knapen, L., Ben-Arroyo Hartman, I., Keren, D., Yasar, A., Cho, S., Bellemans, T., Janssens, D., Wets, G., 2014b. Scalability issues in optimal assignment for carpooling. JCSS (Journal of Computer and System Sciences) 83, 568–584.

Knapen, L., Ben-Arroyo Hartman, I., Schulz, D., Bellemans, T., Janssens, D., Wets, G., 2015c. Determining Structural Route Components from GPS Traces. Submitted to Transportation Research Part B: Methodological .

Knapen, L., Galland, S., 2013. Design Note : Agent-based Carpooling Model. IMOB internal report. Hasselt University Transportation Institute (IMOB). Diepenbeek, Belgium - Belfort, France.

Knapen, L., Janssens, D., Yasar, A.U.H., 2013a. D3.1 Behaviourally-sensitive simulator design ready for the calculation of Mobility-EV scenarios. Technical Report. Hasselt University Transportation Institute (IMOB). Diepenbeek, Belgium.

Knapen, L., Janssens, D., Yasar, A.U.H., 2014c. D3.2 Prototype development of a fully integrated data-driven simulator. Project Deliverable Deliverable: D3.2. Hasselt University Transportation Institute (IMOB). Diepenbeek, Belgium.

Knapen, L., Keren, D., Yasar, A., Cho, S., Bellemans, T., Janssens, D., Wets, G., 2013b. Estimating Scalability Issues while Finding an Optimal Assignment for Carpooling, in: Procedia Computer Science, Procedia Computer Science, Elsevier, Halifax, Nova Scotia, Canada.

Knapen, L., Keren, D., Yasar, A.U.H., Cho, S., Bellemans, T., Janssens, D., Wets, G., 2012a. Analysis of the Co-routing Problem in Agent-based Carpooling Simulation, in: Procedia Computer Science, pp. 821 – 826. ANT 2012 and MobiWIS 2012.

Knapen, L., Kochan, B., Bellemans, T., Janssens, D., 2014d. Notes on the use of radiation laws in simulation. IMOB DataSIM Research Report. Hasselt University Transportation Institute (IMOB). Diepenbeek, Belgium.

Knapen, L., Kochan, B., Bellemans, T., Janssens, D., Wets, G., 2011. Activity-based models for countrywide electric vehicle power demand calculation, in: 2011 IEEE International Conference on Smart Grid Communications (SmartGridComm 2011), 2011 IEEE First International Workshop on Smart Grid Modeling and Simulation (SGMS), Brussels, Belgium. pp. 13 – 18.

Knapen, L., Kochan, B., Bellemans, T., Janssens, D., Wets, G., 2012b. Using Activity-Based Modeling to Predict Spatial and Temporal Electrical Vehicle Power demand in Flanders. Transportation Research Record .

Knapen, L., Kochan, B., Bellemans, T., Janssens, D., Wets, G., 2012c. Using Activity-Based Modeling to Predict Spatial and Temporal Electrical Vehicle Power demand in Flanders, in: 2012 TRB 91st Annual Meeting Compendium of Papers, TRB (Transportation Research Board), Washington, D.C.

Knapen, L., Usman, M., Bellemans, T., Janssens, D., Wets, G., 2012d. Framework to Evaluate Rescheduling due to Unexpected Events in an Activity-Based Model, in: TRB 2013 Annual Meeting, Washington, D.C.

Knapen, L., Usman, M., Bellemans, T., Janssens, D., Wets, G., 2013c. Within Day Rescheduling Microsimulation combined with Macrosimulated Traffic. Transportation Research Part C .

Knapen, L., Yasar, A., Cho, S., Keren, D., Abu Dbai, A., Bellemans, T., Janssens, D., Wets, G., Schuster, A., Sharfman, I., Bhaduri, K., 2013d. Exploiting Graph-theoretic Tools for Matching in Carpooling Applications. Journal of Ambient Intelligence and Humanized Computing .

Knapen, L., Yasar, A.U.H., Cho, S., Bellemans, T., 2014e. Agent-based modeling for carpooling, in: Data Science and Simulation in Transportation Research. IGI Global, Hasselt, Datasim davy janssens, ansar-ul-haque yasar, luk knapen edition.

Konduri, K., Pendyala, R., You, D., Chiu, Y.C., Hickman, M., Noh, H., Waddell, P., Wang, L., 2014. The application of an Integrated Behavioral Activity-Travel Simulation Model for Pricing Policy Analysis, in: Data Science and Simulation in Transportation Research. IGI Global, Hasselt. Advances in Data Mining and Database Management, pp. 86–102.

Kromer, M., Heywood, J., 2007. Electric Powertrains: Opportunities and Challenges in the U.S. Light-Duty Vehicle Fleet.

Kuijpers, B., Moelans, B., Othman, W., Vaisman, A., 2009. Analyzing Trajectories Using Uncertainty and Background Information, in: Mamoulis, N., Seidl, T., Pedersen, T., Torp, K., Assent, I. (Eds.), Advances in Spatial and Temporal Databases. Springer Berlin / Heidelberg. volume 5644 of *Lecture Notes in Computer Science*, pp. 135–152. 10.1007/978-3-642-02982-0_11.

Li, L., Quddus, M., Zhao, L., 2013. High accuracy tightly-coupled integrity monitoring algorithm for map-matching. Transportation Research Part C: Emerging Technologies 36, 13 – 26.

Looges, P.J., Olariu, S., 1993. Optimal greedy algorithms for indifference graphs. Computers & Mathematics with Applications 25, 15 – 25.

Luetzenberger, M., Masuch, N., Hirsch, B., Ahrndt, S., Albayrak, S., 2011. Strategic Behaviour in Dynamic Cities, in: Weed, D. (Ed.), Proceedings of the 43-rd Summer Computer Simulation Conference, The Hague, The Netherlands, pp. 148–155.

Maerki, F., Charypar, D., Axhausen, K.W., 2014. Agent-based model for continuous activity planning with an open planning horizon. Transportation .

Manzini, R., Pareschi, A., 2012. A Decision-Support System for the Car Pooling Problem. Journal of Transportation Technologies , 85–101.

Marchal, F., Hackney, J., Axhausen, K.W., 2005. Efficient Map Matching of Large Global Positioning System Data Sets: Tests on Speed-Monitoring Experiment in Zuerich. Transportation Research Record: Journal of the Transportation Research Board 1935, 93–100.

McPherson, M., Smith-Lovin, L., Cook, J.M., 2001. Birds of a Feather: Homophily in Social Networks. Annual Review of Sociology 27, 415–444.

Nemry, F., Guillaume Leduc, Almudena, M., 2009. Plug-in Hybrid and Battery-Electric Vehicles: State of the research and development and comparative analysis of energy and cost efficiency.

Nijland, L., Arentze, T., Borgers, A., Timmermans, H.J., 2009. Individuals' activity-travel rescheduling behaviour: experiment and model-base d analysis. Environment and Planning A 41, 1511–1522.

Ochieng, W.Y., Quddus, M., Noland, R., 2010. Map-Matching in Complex Urban Road Networks. Revista da Sociedade Brasileira de Cartografia, Geodésia, Fotogrametria e Sensoriamento Remoto 55, 14.

Papandrea, M., Giordano, S., 2013. Location prediction and mobility modelling for enhanced localization solution. Journal of Ambient Intelligence and Humanized Computing , 1–17.

Pendyala, R., Konduri, K., Chiu, Y.C., Hickman, M., Noh, H., Waddell, P., Wang, L., You, D., Gardner, B., 2012a. An Integrated Land-use Transport Model Application: Simulating the Impact of Network Disruptions on Activity-travel Engagement Patterns, Transportation Research Board, Tampa, Florida.

Pendyala, R., Konduri, K., Chiu, Y.C., Hickman, M., Noh, H., Waddell, P., Wang, L., You, D., Gardner, B., 2012b. Integrated Land Use–Transport Model System with Dynamic Time-Dependent Activity–Travel Microsimulation. TRB Research Record 2303, 19–27.

Perujo, A., Ciuffo, B., 2010. The introduction of electric vehicles in the private fleet: Potential impact on the electric supply system and on the environment. A case study for the Province of Milan, Italy. Energy Policy 38, 4549 – 4561.

Perujo Mateos Del Parque, A., Ciuffo, B., 2009. Potential Impact of Electric Vehicles on the Electric Supply System. A Case Study for the Province of Milan, Italy. OPOCE .

Pillat, J., Mandir, E., Friedrich, M., 2011. Dynamic Choice Set Generation Based on Global Positioning System Trajectories and Stated Preference Data. Transportation Research Record 2231, 18–26.

Prato, C.G., 2009. Route choice modeling: past, present and future research directions. Journal of Choice Modelling 2, 65 – 100.

Prato, C.G., 2012. Meta-analysis of choice set generation effects on route choice model estimates and predictions. Transport 27, 286–298.

Prato, C.G., Bekhor, S., 2006. Applying Branch-and-Bound Technique to Route Choice Set Generation. Transportation Research Record , 19–28.

Prato, C.G., Bekhor, S., 2007. Modeling Route Choice Behavior: How Relevant Is the Composition of Choice Set? TRB Research Record 2003, 64–73.

Prato, C.G., Bekhor, S., Pronello, C., 2012. Latent variables and route choice behavior. Transportation 39, 299–319.

Qiong, B., Kochan, B., Bellemans, T., Janssens, D., Wets, G., 2015. Investigating micro-simulation error in activity-based travel demand forecasting: a case study of the FEATHERS framework. Transportation planning and technology 38, 425–441.

Quddus, M.A., Ochieng, W.Y., Noland, R.B., 2007. Current map-matching algorithms for transport applications: State-of-the art and future research directions. Transportation Research Part C: Emerging Technologies 15, 312 – 328.

Ramage, M., 2010. Transitions to Alternative Transportation Technologies–Plug-in Hybrid Electric Vehicles. National Academies Press, Committee on Assessment of Resource Needs for Fuel Cell and Hydrogen Technologies, Washington, DC 20001.

Raza, A., Knapen, L., Declercq, K., Bellemans, T., Janssens, D., Wets, G., 2015. Diary Survey Quality Assessment Using {GPS} Traces. Procedia Computer Science 52, 600 – 605. The 6th International Conference on Ambient Systems, Networks and Technologies (ANT-2015), the 5th International Conference on Sustainable Energy Information Technology (SEIT-2015).

Recker, W., 1994. The Household Activity Pattern Problem: General Formulation and Solution. Transportation Research Part B: Methodological 29, 61 77.

Recker, W.W., Kang, J.E., 2010. An Activity-Based Assessment of the Potential Impacts of Plug-In Hybrid Electric Vehicles on Energy and Emissions Using One-Day Travel Data. University of California Transportation Center, Working Papers 1593422. University of California Transportation Center.

Ridder, F.D., D'Hulst, R., Knapen, L., Janssens, D., 2013. Applying an Activity based Model to Explore the Potential of Electrical Vehicles in the Smart Grid, in: ANT/SEIT, pp. 847–853.

Rieser-Schüssler, N., Balmer, M., Axhausen, K.W., 2012. Route choice sets for very high-resolution data. Working Paper ETH Zürich eth-5386. ETH Zürich. Zürich.

Ronald, N., 2012. Modelling the effects of social networks on activity and travel behaviour. PhD Thesis. TUE. Eindhoven.

Roorda, M., Andre, B., 2007. Stated Adaptation Survey of Activity Rescheduling Empirical and Preliminary Model Results. Transportation Research Record , 45–54.

Schüssler, N., Axhausen, K.W., 2009. Map-matching of GPS traces on high-resolution navigation networks using the Multiple Hypothesis Technique (MHT). Working Paper 589. ETH Zürich. Zürich.

Schüssler, N., Balmer, M., Axhausen, K.W., 2010. Route Choice Sets for Very High-Resolution Data, in: TRB 2010 Annual Meeting, TRB (Transportation Research Board), Washington, DC, USA. p. 16.

Simini, F., Gonzales, M., Maritan, A., Barabasi, A.L., 2012. A universal model for mobility and migration patterns. Nature 484, 96–100.

Spiegel, M., 1968. Mathematical Handbook of Formulas and Tables. Schaum Outline Series.

Spinsanti, L., Celli, F., Renso, C., 2010. Where you stop is who you are: understanding people's activities by places visited, in: Proceedings of the 5th BMI, Workshop on Behaviour Monitoring and Interpretation 2010, EU-FET Coordination Action MODAP, Karlsruhe. pp. 38–52.

Trasarti, R., Pinelli, F., Nanni, M., Giannotti, F., 2011. Mining mobility user profiles for car pooling, in: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, New York, NY, USA. pp. 1190–1198.

Tsai, T.h., Gill, J., 2013. Interactions in Generalized Linear Models: Theoretical Issues and an Application to Personal Vote-Earning Attributes. Soc. Sci. 2013, 2 OPEN ACCESS, 91–113.

UO_ECI, 2011. Country pictures Belgium (http://www.eci.ox.ac.uk/research/energy/).

Usman, M., Fraile-Ardanuy, J., Knapen, L., Yasar, A.U.H., Bellemans, T., Janssens, D., Wets, G., 2015. Relationship Between Spatio-temporal Electricity Cost Variability and E-mobility. Procedia Computer Science 52, 772 – 779. The 6th International Conference on Ambient Systems, Networks and Technologies (ANT-2015), the 5th International Conference on Sustainable Energy Information Technology (SEIT-2015).

Usman, M., Knapen, L., Bellemans, T., Janssens, D., Wets, G., 2014a. Effect of electrical vehicles charging cost optimization over charging cost and travel timings, in: ICCVE (IEEE), IMOB, Vienna, Austria.

Usman, M., Knapen, L., Yasar, A., Bellemans, T., Janssens, D., Wets, G., 2014b. A framework for electric vehicle charging strategy optimization tested for travel demand generated by an activity-based model, in: IEEE ITSC, Qingdao, China.

Van Aerschot, K., 2014. Analyse van het verband tussen de beperkte interesse in carpoolen en de inflexibiliteit van agenda's. Bachelor Verkeerskunde. Hasselt University. Diepenbeek, Belgium.

Varrentrapp, K., Maniezzo, V., Stützle, T., 2002. The Long Term Car Pooling Problem On the soundness of the problem formulation an d proof of NP-completeness. Technical Report AIDA-02-03. Fachgebiet Intellektik, Fachbereich Informatik, TU Darmstadt. Darmstadt, Germany.

Vuurstaek, J., Knapen, L., Kochan, B., Bellemans, T., Janssens, D., Wets, G., 2015. Modelling hospital visitors for the city of Leuven as input for a FEATHERS-MATSim simulation, in: BIVEC/GIBET Transport Research Day 2015, Eindhoven.

Waraich, R.A., Galus, M., 2009. Plug-in Hybrid Electric Vehicles and Smart Grid: Investigations Based on a Micro-Simulation.

Weis, C., Dobler, C., Axhausen, K., 2010. Stated adaptation survey of household activity scheduling, in: WCTR, Lisbon, Portugal.

Weisstein, W., E., 1999. Normal Distribution.

White-House-NSTC, 2011. A policy framework for the 21st century grid : Enabling our secure energy future (www.whitehouse.gov/sites/default/files/microsites/ostp/nstc-smart-grid-june2011. pdf).

William D'haeseleer, Klees, P., Albrecht, J., De Ruyck, J., Tonon, P., Streydio, J.M., Belmans, R., Dufresne, L., Leduc, B., Proost, S., van Ypersele, J.P., Chevalier, J.M., Eichhammer, W., Terzian, P., 2007. Commission ENERGY 2030 — FINAL REPORT : Belgium's Energy Challenges Towards 2030.

Wood, S., 2012. Generalized Linear Models.

Wu, D., Aliprantis, D., Gkritza, K., 2010. Electric Energy and Power Consumption by Light-Duty Plug-In Electric Vehicles.

Xiao, X., Zheng, Y., Luo, Q., Xie, X., 2012. Inferring social ties between users with human location history. Journal of Ambient Intelligence and Humanized Computing , 1–17.

Zhao, Y., Sadek, A., 2014. Large-Scale Agent-Based Models for Transportation Network Management under Unplanned Events, in: Data Science and Simulation in Transportation Research. IGI Global, Hasselt. Advances in Data Mining and Database Management, pp. 206–231.

Zheng, V.W., Zheng, Y., Yang, Q., 2009. Joint Learning User's Activities and Profiles from GPS Data, in: Proceedings of the 2009 International Workshop on Location Based Social Networks, ACM, Seattle WA.

Zhou, J., Golledge, R., 2006. A Three-step General Map Matching Method in the GIS Environment: Travel/Transportation Study Perspective. International Journal of Geographical Information System 8, 243–260.

Zijpp, N.J.v.d., Catalano, S.F., 2005. Path enumeration by finding the constrained K-shortest paths. Transportation Research Part B: Methodological 39, 545 – 563.

Samenvatting

Onderzoek in mobiliteit bestudeert onder meer het verplaatsingsgedrag van groepen van personen. Daartoe worden verschillende soorten modellen gebruikt.

Macroscopische modellen beschrijven het gedrag van de groep door middel van een stel vergelijkingen waarin de variabelen overeenkomen met kenmerken van de groep als geheel. Micro-modellen beschrijven het gedrag van elk individu en leiden het gedrag van de groep af door aggregatie van de beslissingen die door de individuen werden genomen. Ze laten toe om het gedrag van een entiteit te bepalen door het modelleren van de samenstellende onderdelen. Dat is enerzijds handig omdat het toelaat effecten van diversiteit te evalueren maar anderzijds zijn de resultaten moeilijk te valideren omdat van elkaar verschillende modellen op elkaar gelijkende resultaten kunnen leveren die overeenstemmen met de werkelijkheid. Micro-modellen in mobiliteit zijn over het algemeen stochastisch. Omwille van de complexiteit worden de resultaten van micro-modellen meestal bepaald door micro-simulatie en niet d.m.v. analytische methodes.

In mobiliteitsonderzoek wordt de techniek van *activiteits-gebaseerde modellen* (ActBM) gebruikt; die voorspellen de dagindeling van individuen. Zodra de locatie en periode van alle activiteiten voor de complete populatie gekend zijn, kan de vraag naar verplaatsingen bepaald worden door aggregatie. De meeste huidige ActBMs beschouwen individuen als onderling onafhankelijk.

Agent-gebaseerde modellen (AgnBM) zijn software-mathematische technieken om (optimisatie-)problemen op te lossen d.m.v. autonome entiteiten van allerlei aard, meestal door samenwerking. Deze techniek is op verscheidene manieren bruikbaar in ActBM.

Dit doctoraat onderzoekt enkele concrete problemen in de context van micro-modellering. Het bestaat uit twee delen. In het eerste deel worden resultaten van het FEATHERS ActBM gebruikt om concrete in projecten gestelde vragen te beantwoorden. Het tweede deel levert een methode om het route-keuze model te verfijnen.

Samenvatting Deel 1

Elektrische Voertuigen

Productie van elektrische energie evolueert van een systeem met weinig grote centrale eenheden met een groot, vrij stabiel en beperkt regelbaar vermogen naar veel verspreide producenten met kleiner, variabel en moeilijk voorspelbaar vermogen (conversie van zonne-energie en wind-energie). Omdat elektrische energie zeer moeilijk kan worden opgeslaan, moeten productie en consumptie steeds in evenwicht zijn op het distributienet. Dat schept problemen voor het beheer. De beschikbaarheid van de grote gezamenlijke opslagcapaciteit van batterijen in elektrische voertuigen (EV) kan bijdragen tot een oplossing. Hier spelen modellen voor mobiliteit een rol: ze leveren de gegevens om voor verschillende locaties en periodes te bepalen wat de minimale en maximale vraag naar elektrische energie is vanwege EV (m.a.w. waar en wanneer opslagcapaciteit beschikbaar is). Uit het gevoerde onderzoek blijkt dat voor bijna 80% van de Vlaamse populatie de dagelijks afgelegde afstand overbrugd kan worden d.m.v. een uitsluitend elektrisch aangedreven voertuig. Op jaarbasis verbruikt een EV ongeveer evenveel als een gemiddeld gezin voor huishoudelijk gebruik. De productie van elektrische energie is niet problematisch, wel de verdeling van laadbeurten in tijd en ruimte. Verschillende hypothesen voor laadgedrag (overdag, 's nachts, op het werk, etc) zijn doorgerekend en de vraag naar elektrisch vermogen in tijd en ruimte werden berekend.

Coordinatie in Modellen voor Verplaatsingsgedrag

Carpooling voor woon-werk-verkeer is een type-voorbeeld van coördinatie tussen actoren waarbij de eigenschappen op niveau van de gemeenschap afgeleid kunnen worden door het modelleren van gecoördineerd gedrag van individuen. In dit doctoraatsonderzoek is een ontwerp gemaakt voor een Agent-Based Model (AgnBM) voor het simuleren van carpooling voor woon-werk-verplaatsingen in een populatie.

Ook is onderzoek verricht naar de structuur van een web-service die aanbiedingen voor woon-werk-trips accepteert en verzamelt in een netwerkstructuur. Voor elk paar aanbiedingen wordt de kans op succesvolle afloop van de onderhandeling voor carpooling bepaald. Twee aanbiedingen worden in het netwerk met elkaar verbonden als de succeskans voldoende groot is. De kans wordt dan bij de verbinding als een gewichtsfactor geregistreerd. Elke aanbieder van trips vermeldt ook de capaciteit van de beschikbare wagen. De software voor adviesverlening moet in het opgebouwde netwerk een stel verbindingen kiezen die aangeven welke trips samengevoegd moeten

worden. Daarbij mag voor geen enkel voertuig de capaciteit worden overschreden en moet de som van de gewichten van de gekozen verbindingen maximaal zijn. Dit probleem is NP-hard en voor de oplossing is een heuristische methode vereist. Om de keuze van dergelijke methode te ondersteunen zijn kenmerken van het netwerk bepaald op basis van agenda's voorspeld door FEATHERS voor de Vlaamse populatie. De ordegrootte voor het aantal knopen in het netwerk voor Vlaanderen is honderdduizend. Elke knoop heeft meerdere tientallen verbindingen met andere knopen. Het netwerk bestaat uit één zeer grote component (*giant component*) en meerdere veel kleinere.

Aanpassing van Agenda's

Activiteits-gebaseerde modellen voorspellen agenda's voor een dag of een week. In realiteit worden agenda's aangepast tijdens de uitvoering. Aanpassingen zijn het gevolg van onverwachte gebeurtenissen of naderhand geplande activiteiten. In dit doctoraat is een software framework ontwikkeld voor het evalueren van gedragsmodellen voor aanpassing van agenda's. Het gaat om een hybride simulator: voor het gedrag van de individuen wordt micro-simulatie gebruikt en de toedeling van het verkeer op het wegennet gebeurt macroscopisch. In een simulatie werd de capaciteit van hoofdwegen in Vlaanderen verlaagd n.a.v. een incident. Voor elk kwartier wordt de toestand herberekend. Het resultaat van de macroscopische toedeling wordt gebruikt om de perceptie van elk individu te bepalen en diens inschatting van de drukte op het wegennet in de komende uren. Met die gegevens past elk individu de eigen agenda aan. Het effect van die aanpassing wordt weer doorgerekend in de verkeerstoedeling voor het volgende kwartier.

Samenvatting Deel 2

Het micro-model voor een individu is samengesteld uit meerdere sub-modellen. Veel daarvan zijn discrete-keuze modellen: die voorspellen de uitkomst van de keuzes die het individu moet maken in verschillende situaties. Voorbeelden van keuze-modellen zijn: kiezen van de *locatie* voor het uitvoeren van een welbepaalde activiteit, kiezen van een volgende *activiteit*, kiezen van een *vervoersmiddel* (fiets, trein, wagen, etc). De keuzeverzameling (de verzameling waaruit men moet kiezen) is meestal niet groot en in sommige gevallen voorafbepaald.

Echter, keuze van de *route* om van een oorsprong naar een bestemming (OB) te gaan leidt tot moeilijk handelbare discrete keuzemodellen (i) omdat de keuzeverza-

meling zó groot is dat het individu die meestal niet geheel kan vatten, (ii) omdat de opties door overlapping van routes niet onderling onafhankelijk zijn en (iii) omwille van het rekentechnisch probleem dat de keuzeverzameling voor elk OB-paar afzonderlijk moet worden bepaald (het courant gebruikte model voor Vlaanderen verdeelt de ruimte in ongeveer 2500 zones wat tot 6.250.000 OB-paren leidt). Om een realistisch model voor route-keuze te creëren, moet een realistische keuzeververzameling worden opgesteld. Enerzijds moet de keuzeset beperkt blijven om praktische redenen en moeten onrealistische opties worden geweerd, anderzijds moet die set voldoende groot zijn om geen realistische opties uit te sluiten. Er bestaan meerdere technieken om dergelijke sets op te stellen op basis van een gegeven netwerk (bijvoorbeeld OpenStreetMap). Kandidaat-routes worden gekozen op basis van meerdere criteria (maximale omweg t.o.v. kortste-afstand pad, maximale extra duur t.o.v. kortste-tijd pad, vermijden van kruisingen waar links afdraaien problematisch is, afkeer of voorkeur voor verkeerslichten, etc).

In dit doctoraat wordt voorgesteld hieraan een criterium toe te voegen dat verband houdt met de *structuur* van de gekozen route in het netwerk. Hiertoe is volgende hypothese geformuleerd en geverifiëerd: *voor trips met een doel kiest men een route die bestaat uit een klein aantal goedkoopste paden in het netwerk.* De qualificatie *goedkoopste* kan betrekking hebben op afstand, tijd of een andere kost die aan het pad wordt toegekend. Het toe te voegen criterium vereist dat de kansverdeling voor het minimum aantal goedkoopste componenten, waaruit de paden in de keuzeverzameling zijn samengesteld, overeenstemt met de realiteit. Het nieuwe criterium wordt voorgesteld omdat er routes bestaan waarvoor de omweg-factor voldoende klein is maar die een onwaarschijnlijke structuur hebben. Die moeten uit de keuzeverzameling worden geweerd.

Het uitgevoerde onderzoek start van een netwerk van wegen en een verzameling *GPS traces*. Die traces bevatten informatie over de routekeuze gemaakt voor duizenden verplaatsingen. Elke GPS registratie bestaat uit een tijdsaanduiding en een stel coördinaten. De toegepaste methode omvat volgende stappen:

1. *trip-detectie*
2. *map-matching*
3. bepaling van het *minimale aantal goedkoopste paden* waarin een gegeven pad in een netwerk kan gesplitst worden. Die goedkoopste deel-paden worden *basiscomponenten* genoemd.
4. opsomming van *alle minimale decomposities* van een gegeven pad in een netwerk.

Trip-detectie en Map-matching

Bij *trip-detectie* wordt een sequentie van geregistreerde GPS punten opgedeeld zodat elke deel-sequentie overeenkomt met juist één trip.

Map-matching verwerkt de GPS registraties per trip. Uit een sequentie van registraties wordt een sequentie van verbindingen in het wegennet afgeleid. Hiertoe werd een nieuwe techniek ontwikkeld die even snel werkt als de bestaande methoden maar die rekening houdt met alle gegevens in de GPS trace bij het bepalen van de vermoedelijke reeks van netwerk verbindingen gebruikt in de trip, Efficiënte gekende methoden verwerpen, om technische redenen, weinig beloftevolle kandidaat-sequenties wanneer nog slechts een gedeelte van de trace is verwerkt, waardoor informatie ongebruikt blijft.

Decompositie van Routes

Zodra de gevolgde paden in het wegennet gekend zijn door map-matching, wordt voor elk pad bepaald wat het *kleinste aantal basiscomponenten* is waaruit het is samengesteld. Denk aan hiërarchische routering waarbij iemand de kortste of snelste weg vanuit het vertrekpunt naar een oprit van een snelweg volgt en vanaf de uitrit opnieuw het kortste of snelste weg naar de bestemming. Dit doctoraat stelt een efficiënt algorithme voor om het minimum aantal basiscomponenten in een pad te bepalen. Hiermee zijn duizenden trips verwerkt en de kansverdeling voor het aantal componenten in minimale decomposities werd bepaald. Die kansverdeling bevestigt de geformuleerde hypothese. Paden met meer dan vijf componenten vertegenwoordigen ongeveer vijf procent van het totaal, zowel in een dataset voor Vlaanderen als in een dataset voor Milaan.

Een pad kan in het algemeen op meer dan één manier opgedeeld worden in een minimum aantal basiscomponenten. Elke decompositie wordt bepaald door *splitspunten*: dit zijn de knopen in het pad die de grens vormen tussen basiscomponenten. Het hoger vermelde algorithme dat het minimum aantal basiscomponenten bepaalt, levert tevens een collectie van elkaar niet overlappende verzamelingen van knopen in het pad. Uit elk van die verzamelingen moet exact één knoop worden gekozen om een minimale decompositie te vinden, maar niet elke combinatie is geldig. Dat is interessant want het betekent dat de datasets informatie bevatten die mogelijk bruikbaar is.

De laatste bijdrage van dit doctoraat is een methode bestaande uit vijf stappen en gebaseerd op grafentheorie, om alle mogelijke minimale decomposities van een pad op te sommen in polynomiale tijd. Mede omdat de verzamelingen waaruit men

splitsknopen moet kiezen niet allemaal even groot zijn, komt niet elke kandidaat-splitsknoop even dikwijls voor in de opsomming van alle minimale splitsingen. Het belang van een knoop wordt bepaald door het aantal decomposities waarin de knoop als splitsknoop voorkomt. Die frequentie van voorkomen is in de praktijk eenvoudig bepaalbaar omdat de opsomming van de minimale splitsingen in polynomiale tijd kan worden uitgevoerd. Dat maakt het mogelijk om grote hoeveelheden routes uit GPS te analyseren.

Dit resultaat kan nieuw onderzoek voeden waarin het belang van elke knoop wordt gecorreleerd met de attributen van de knoop in het netwerk (aanwezigheid van signalisatie etc). Het doel daarvan is om, door uitsluitend gebruik te maken van wiskundige technieken en informatie uit GPS traces, knopen in het netwerk te identificeren die interessant zijn voor het voorspellen van gebruikte routes.