Universiteit Hasselt
Faculteit Toegepaste Economische Wetenschappen

Calibrating Unsupervised Machine Learning Algorithms for the Prediction of Activity-Travel Patterns

Proefschrift voorgelegd tot het behalen van de graad van
Doctor in de Toegepaste Economische Wetenschappen
aan de Universiteit Hasselt te verdedigen door

Davy JANSSENS

Promotor:
Prof. dr. G. Wets
Copromotor:
Prof. dr. K. Vanhoof

ACKNOWLEDGEMENTS

During the course of my research, I have had the pleasure of collaborating and meeting with many interesting and fine people whose contributions and support directly or indirectly improved the quality of this work. Therefore, I would like to take this opportunity to express my gratitude and acknowledgements.

Prof. dr. Geert Wets and Prof. dr. Koen Vanhoof, respectively my promoter and co-promoter, are the very first persons whom I truly want to thank. They made this Ph.D. research possible and largely improved and elaborated the content of the dissertation through their scientific guidance and invaluable support. I am particularly grateful because they always helped me to develop interesting ideas, get into contact with other researchers in the field and encouraged me to publish the results of my research in international journals. I wish to acknowledge dr. Tom Brijs, for his reviews and comments on my research results and final papers, from which the dissertation benefited.

I am also grateful that I had the opportunity to work together with the Urban Planning Group in the Netherlands, especially with Prof. dr. Harry Timmermans and Prof. dr. Theo Arentze, who allowed me to use their data and carry out experiments on the original Albatross model. Their experience and previous work guided me into the field of activity-based modelling. Furthermore, I am grateful to Prof. dr. Philippe Toint, Prof. dr. Konstadinos Goulias and Prof. dr. Harry Timmermans, my external jury members, for providing me with detailed remarks and critiques that improved the final shape of the dissertation.

I am indebted to Yu Lan from Tsinghua University. Although we were geographically separated, I am very grateful that I had the opportunity to conduct some joint research work with him and that I was able to get to know such a truly gentle person.

I also wish to acknowledge all my friends and (very large number of) colleagues at our (ever growing) research group and at the university with whom I have had the pleasure of working together and having fun over the years. I would like to thank all of them for contributing to such a nice (working) environment and for the many joyful moments we were able to experience in sometimes highly adorable places.

Finally, I would like to express a special word of gratitude to my parents for their everlasting support. I am also indebted to Kristien for her support and encouragement -especially during the most difficult moments when everything seemed to go wrong-, for her endless patience, continuous love and care.

CONTENTS

Chapter 1
Introduction and Research Motivation

1.1 INTRODUCTION

1.1.1 BACKGROUND

Obviously, the main objective for scientific and academic researchers all over the world is to promote and achieve progress and innovation through research. The mainspring for this common worldwide goal is to improve the standard of living and/or to increase the general intellectual property of society as a whole. While some may argue that the added value of one research domain is more limited in terms of added economic value than the other, the contribution of transportation research towards the society as a whole is significant.

In a research report by the United Nations (United Nations Economic and Social Council, 2001), it has been postulated that the transport sector accounts for about 25 per cent of the total commercial energy consumed worldwide and that it consumes approximately one half of the total oil produced. The International Energy Agency (IEA) predicts that the transport sector will overtake industry as the largest energy user by 2020 (SUT Partnership, 2002).

Unfortunately, the sector has major negative economic, social and environmental side effects. At the environmental level, transport has proven to be the source of nitrogen oxides, sulfur oxides and other volatile organic compounds, all which have negative environmental and health implications. Pollution, environmental degradation, space consumption and green house gases are receiving increasing attention as the immediately detectable externalities of transport and land-use development patterns. At the economic level, accidents and congestions, traffic gridlocks, stress from pedestrian and vehicular conflict, inefficient public transport and urban sprawl are all associated with unsustainable transport systems that indirectly represent costs to society. At the social level, recent research reports seem to suggest that in areas where public transport is often second-rate or absent and where the levels of car ownership are significantly lower, a higher degree of risk for social exclusion is perceived (Transport and Social Exclusion Workshop, 1998). Whereas a good transport system widens the opportunities to satisfy interaction needs, a poorly connected transport system limits economic and social development (Ortúzar and Willumsen, 2002). The transport system thus allows individuals to trade time for space when moving to (activity) locations (Miller, 2003; Rietveld, 1994).

It is clear that all these externalities adversely affect sustainable development. The concept of sustainable development usually refers to the interplay between human society and the environment, with the specific goal to meet the needs of the present without compromising the ability of future generations to meet their needs. While this might be an all-encompassing term, the concept in itself has been widely accepted as a recommendable, if not obligatory planning tool by national governments and international bodies. Rising concerns over these increasingly intolerable externalities have generated particular interest in how transport planning policies might at least moderate the pressures in growth in personal mobility and support the principles of sustainable development (Barret, 1996; Salomon *et al.*, 1993; European Commission, 2001).

1.1.2 TRANSPORT POLICY AND MODELLING

Originally, transport planning policies focused on mastering the massive growth in car mobility. These policies were adopted in an immediate response to the *predicted* growth in (car) mobility. The estimation and forecasting of travel demand and behaviour were handled by a standard methodological approach, commonly referred to as the four-step modelling approach (Ruiter and Ben-Akiva, 1978). This approach was mostly chosen for its convenient mathematical calculus and for its ability to support the policies of infrastructure expansion (Wilson, 1967, Ortúzar and Willumsen, 2002). However, the public scrutiny of prediction, the substantial errors in model forecasts, and the shift in emphasis from long-term investment-based strategies to shorter-term demand-driven solutions, have contributed to the increasing criticism from which these methodologies for predicting travel demand have suffered during the 1980s (Jones *et al.*, 1983). Despite this, improved four-step models still remain frequently used by practitioners to current date, due to their simplicity and ease of understanding.

However, increased concerns about relatively recent phenomena such as congestion, emission and changing land-use patterns, have motivated governments to consider policies aimed at reducing and controlling them (Dijst, 1997). These policies are commonly referred to as travel demand management

(TDM) measures, which objective is to (i) alter travel behaviour without necessarily embarking on large-scale infrastructure expansion projects, (ii) encourage better use of available transport resources and (iii) avoid the negative consequences of continued unrestrained growth in private mobility (Krygsman, 2004). Examples of such measures are the spreading of peak-period traveling through relaxing working, school and shopping hours, congestion charging, and the like. In order to effectively implement and analyze these policy objectives, an increasing amount of awareness emerged with respect to the need for improved understanding of travel behaviour. Obviously, the four-step methodologies that were adopted at that point in time and that were mainly focused on policies of infrastructure expansion, were insufficiently able to achieve this. This resulted in a need for travel demand models that embody a realistic representation and understanding of the decision-making process of individuals and that are responsive to a wider range of transport policy measures.

As some initial modelling efforts slowly started emerging in this field, the use of transportation models to back up transportation policies even became required by law (especially in the United States). This legislation includes the Clear Air Act Amendments (CAAA), the Intermodal Surface Transportation Efficiency Act (ISTEA), and ISTEA's successor, the Transportation Equity Act for the Twenty First Century: TEA-21. However, other initiatives, such as the Travel Model Improvement Program (TMIP), which was established by the Federal Highway Administration; the Federal Transit Administration; the Office of the Secretary, U.S. Department of Transportation; and the U.S. Environmental Protection Agency, have been introduced specifically to encourage improvements in land-use and transportation modelling. European policies are lagging a bit behind when compared to the United States, but the topic slowly starts appearing on multiple European research agendas. For instance in the Scandinavian countries, there is TLEnet, the Nordic Research Network on Modelling Transport, Land-Use and the Environment. A good example in Europe is also the ministry of Transportation, Public Works and Water Management in the Netherlands that actively supports improvements in land-use and transportation modelling, by means of advanced modelling techniques such as the activity-based approach that has been propagated in this dissertation. Following these trends, similar (activity-based modelling) initiatives have recently started in Switzerland (Raney *et al.*, 2003b), Sweden (Algers *et al.*, 2005) and Belgium (Janssens and Wets, 2005; Janssens *et* *al.*, 2005g). Given the significant consequences that transportation policies may have, transportation models are thus often used, with or without a legal imposition.

1.1.3 THE TRAVEL DEMAND NATURE

The major insight that enabled researchers to gain a better understanding of the individual decision-making process is the idea that travel demand is derived from the activities that individuals and households need or wish to perform. Travel is merely seen as a means to pursue goals in life but not as a goal in itself. Therefore, modelling efforts should merely concentrate on modelling activities or on a collection of activities that form an entire agenda which triggers travel participation.

Travel should therefore be modelled within the context of the entire agenda, or in other words, as a component of an activity scheduling decision. The concept of activity scheduling is an important one. A simple example will clarify this (see also Figure 1.1).

Assume a female person spends time at home in the morning, where she has breakfast. Hereafter, this person travels to work by car where she arrives at 9 AM and where she works throughout the day till 17.15 PM. Then this person heads for home, arrives there at 17.30 PM and spends the whole evening at home. It is illustrated in this simple example that this person has a certain schedule of activities and that travel is merely a component of the activity scheduling

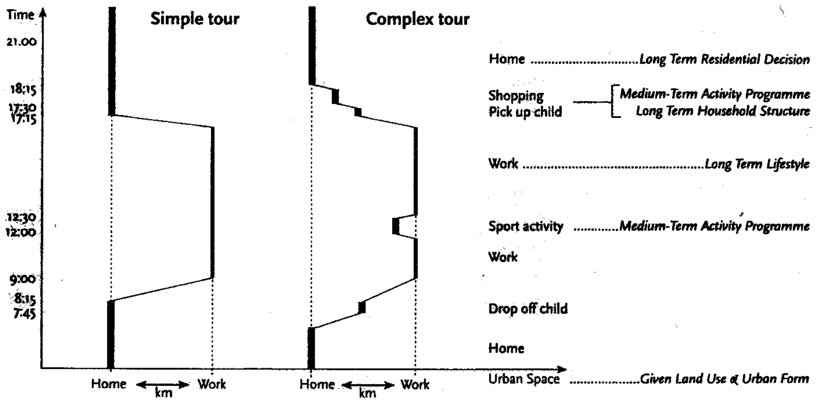

Time
21.00
18:15
17:30
17:15
12:30
12:00
9:00
8:15
7:45
Simple tour
Complex tour
Home ....Long Term Residential Decision
Shopping
Pick up child
Medium-Term Activity Programme
Long Term Household Structure
Work ....Long Term Lifestyle
Sport activity ....Medium-Term Activity Programme
Work
Drop off child
Home
Urban Space ....Given Land Use & Urban Form
Home
km
Work
Home
km
Work

Figure 1.1: An example of an activity scheduling decision

decision. Alternatively, activity scheduling can also be considerably more complex, containing several subtours within the daily schedule. An example is shown in the right part of Figure 1.1. Complexity further increases by several socio-demographic or personal events that may trigger this change in behaviour.

In short, traffic patterns are the manifestation of the implementation of activity programs over time and space. In turn, activity patterns emerge as the interplay between the institutional context, the urban/physical environment, the transportation system and individuals' and households' needs to realize particular goals in life and to pursue activities (Ben-Akiva and Bowman, 1998). This understanding has lead to the formulation of the activity-analysis framework. The fundamental contributions of Hägerstrand (1970), Chapin (1974) and Fried *et al.* (1977) are the undisputed intellectual roots of activity analysis. Hägerstrand has put forward the time-geographic approach that characterizes a list of constraints on activity participation. He made a distinction between "capability constraints" (e.g. a need for sleeping and eating), "coupling constraints" (e.g. dinner with the family assumes that all members of the household are present at the same place and time) and "authority constraints" (e.g. opening hours of shops). This theory postulates that individuals live in a space-time prism in which they can only function by being in different locations at different points in time and by experiencing the time and cost of travel as well as the above listed constraints. Chapin has identified patterns of behaviour across time and space and is more concerned with opportunities and choices instead of constraints. The theory postulates that activity demand is motivated by basic human desires such as ego gratification and social encounters. This theory has later been modified by Fried, Havens and Thall (Fried *et al.*, 1977) who have dealt with some more factors including commitments, capabilities and health. These contributions came together in a study of Jones *et al.* (1983), where activities and travel behaviour were integrated. This was the first initial attempt to model complex travel behaviour.

In order to summarize the above, the work of McNally (2000) can be cited, in which he has listed 5 themes that characterize the activity-based modelling framework:

(i) Travel is derived from the demand for activity participation;

(ii) Sequences or patterns of behaviour, and not individual trips are the relevant unit of analysis;

(iii)Household and other social structures influence travel and activity behaviour;

(iv)Spatial, temporal, transportation and interpersonal interdependencies constrain activity/travel behaviour;

(v) Activity-based approaches reflect the scheduling of activities in time and space.

Activity-based approaches to transportation forecasting therefore aim at predicting which *activities* are conducted *where*, *when*, for *how long*, *with whom*, and the *transport mode* involved. Taking all this into consideration, it is argued that the activity-based framework to travel demand modelling is one of the most detailed frameworks in which travel can be analysed as a daily pattern of behaviour, related to and derived from differences in lifestyles and the activity participation among individuals.

1.1.4 TRAVEL DEMAND MODELS

While a multitude of modelling approaches have emerged over the years, travel demand models can be classified in a number of different ways. In this section, a distinction is made between activity scheduling and simulation models (Timmermans, 2001). Activity scheduling models involve the application of well-developed theoretical constructs to empirical data, with the aim of generating a predictive model that can be used for generalization purposes and for the evaluation of TDM. Activity scheduling models can be subdivided into simultaneous, computational process and constraints-based models (see also Timmermans *et al.*, 2002). Unlike activity scheduling models, simulation models are more "data-driven", by relying upon marginal and conditional probability distributions that are defined for the various choice facets of an activity pattern. For this reason, simulation models are not primarily developed with the intention to explicitly capture the process by which people schedule or execute activities (Timmermans, 2001).

However, one could argue that this difference is one of degree rather than principle, because the activity scheduling process can be indirectly derived from the prediction of each of the individual components of the activity pattern. Therefore, the process of building, testing and applying a particular model is fairly similar in both simulation and activity scheduling models. That is, both approaches finally aim to predict all the typical facets of an activity pattern.

Activity Scheduling Models

The distinction between constraints-based, simultaneous and computational process models is mainly a distinction in terms of the methodology that is used for capturing decision and scheduling behaviour. The theoretical constructs respectively originate from geography, micro-economic theory and psychological decision process theories.

Constraints-Based Models

Constraints-based models typically examine whether particular activity patterns can be realized within a specified time-space environment (Timmermans *et al.*, 2002). These models require as input activity programmes, which describe a set of activities of a certain duration that can be performed at certain times. The space-time environment is defined in terms of locations, their attributes, available transport modes and travel times between locations per transport mode. One of the attributes of interest is the opening hours of the facilities at that location. To examine the feasibility of a certain activity programme, a combinatorial algorithm is typically used to generate all possible activity sequences. The feasibility of each sequence is then tested by checking whether: (a) the interval between the end time of the previous activity and the start time of the next activity is sufficient to perform the activity plus the associated travel time; (b) the activity can start after the earliest possible start time and be finished before the latest possible end time; (c) conditions about the sequencing of activities are not violated. The number of feasible activity schedules is often used as a measure of the flexibility that the time-space environment offers. Geographers have played a dominant role in developing such models. One of the first models in this tradition is Lenntorp’s (1976) PESASP model. A similar model is CARLA, which basically is a combinatorial algorithm for generating feasible activity patterns (Jones *et al.*, 1983). Huigen (1986) proposed another combinatorial algorithm, BSP. This programme is similar to CARLA in that it evaluates the options to maintain the current activity pattern in a changed spatial-temporal setting. However, like PESASP, it does so by exhaustively evaluating all possible sequences of activity/destination combinations. Furthermore, there are minor differences with respect to how constraints are incorporated. It allows that different trips in a chain are made by different modes. Another difference is that it defines available time windows specifically for destinations and not for activities. Another similar model is MASTIC (Dijst, 1995; see also Dijst and Vidakovic, 1997). Its goal is to identify the action space of individuals, using the notion of a space-time prism. A potential action space is defined as the area containing all activity locations that are reachable, subject to a set of temporal and spatial constraints, including type and location of activity bases, available time interval, travel speed and the travel time ratio. Kwan's (1997) GISICAS can be classified as a constraints-based model as well, although it also makes references to computational process models. Given an activity agenda, this GIS-based system begins scheduling by fitting the activities on the agenda into the free time a person has, and orders them into a sequence. Activities with higher priority are ordered first, and the time constraints for performing certain activities are also taken into account. Various search heuristics can be specified to identify the locations where the activities can be carried out. The system then reports a preliminary schedule and also lists the activities that cannot be scheduled. The spatial search is based on a dynamic identification of feasible locations. Compared to other models, constraints-based models lack the necessary mechanisms to predict adjustment behaviour of individuals. When faced with a changed time-space environment, individuals are likely to adjust/reschedule their activity programmes. Consequently, policies may often have less dramatic social impacts as these models suggest. This is especially true in urban contexts where often many potential activity patterns can still be conducted, even after the number of choice alternatives has been reduced. In addition, these models do not provide any information about people's preferences for particular patterns (Timmermans *et al.*, 2002).

Simultaneous Models

Simultaneous models are often based on the assumption of utility-maximising behaviour. Individuals are assumed to schedule their activities such that their utility is maximized. The theory is based on the assumption that choice alternatives can be represented as bundles of attribute levels, for which a particular utility can be derived. Constraints are usually not included in much detail. Activity scheduling behaviour is not addressed specifically in these models but follows automatically from the prediction of the full activity-travel patterns (Timmermans, 2001).

The nested logit formulation became the most frequently applied technique in simultaneous activity-based models of transport demand. However, before this was established, the seminal work by Adler and Ben-Akiva (1979) and by Recker *et al.* (1986a, 1986b) (STARCHILD), rely upon the multinomial logit model. In STARCHILD, an individual's schedule includes activity purpose, duration and location. Constraints on tour-sequences such as timing, location and coupling of activities are incorporated in an external activity program. In this activity schedule model, scheduling is viewed as the choice between types of activity patterns. To this end, activity scheduling behaviour is not addressed separately but follows from the predicted activity-travel patterns. It is assumed that the utility of a specific activity pattern consists of the utilities of its time-component parts. The multinomial logit model was used to predict the choice between these alternative activity patterns. Kitamura *et al.* (2000) presented a sequential, simulation approach to the generation of daily activity-travel patterns. The model, which is also referred to as the Synthetic Travel Pattern Generator (STPG), can equally be classified as a simulation model because it relies on a series of conditional probabilities, each representing the dependency of the attributes of an activity on the past history of activity engagement and travel. However, the STPG system comprises a number of model components for computing probabilities and thereby mainly relies upon multinomial logit models.

Despite its considerably more complex nature, the multinomial logit model was rapidly replaced by the nested logit model, in which the different facets of activity-travel patterns are treated as nests. Kawakami and Isobe (1982, 1988, 1989) were among the first researchers who introduced this model into the field of transportation, by conceptualizing the generation of activity patterns as hierarchical choice processes. However, probably the most popular model in this field is the daily activity schedule program (Ben-Akiva *et al.*, 1996; Bowman, 1995; Ben-Akiva and Bowman, 1995; Bowman *et al.*, 1998; Bowman and Ben-Akiva, 1999). In this model, the starting point is a daily activity schedule, which represents the individual's demand for activity and travel as a multidimensional choice encompassing all the possible combinations of activity and travel. The choice of a daily activity pattern also determines the number of secondary tours. The choices of secondary tour time, destination and mode are assumed to be conditional upon the choice of a daily activity pattern. This implies thus that the utility of a particular alternative in a higher nest is influenced by the utility of the lower level alternatives comprising it; which is a property that can be generalized to all nested logit models. Other nested logit models include the work by Wen and Koppelman (1999), and PETRA (Fosgerau, 1998) which make limiting assumptions, thereby reducing the complexity of the model.

In addition to these nested logit models, there are other attempts for capturing decision and scheduling behaviour by means of utility-maximizing theories. A first example is the PCATS model by Kitamura and Fujii (1998). Unlike the previous models, PCATS assumes a sequential scheduling process in which individuals maximize the utility associated with open time periods, subject to three types of constraints: prism constraints, availability of travel modes and recognition of potential activity locations. In the work by Recker (1995), the household activity pattern is formulated in terms of variants of the pickup and delivery problem with time windows. It is presented in a mathematical programming form. Finally, the work by Bhat and Singh (2000; Bhat, 1999) also uses principles of utility-maximizing behaviour by the presentation of a comprehensive framework for activity-travel generation for workers. In Bhat and Misra (2001, 2002), a similar approach for non-workers has been suggested. In addition to this, Bhat also suggested a series of models to predict more separated components of the activity scheduling decision (Bhat, 1996, Bhat and Singh, 1997). These different models came together in the CEMDAP model (Bhat *et al.*, 2004), which is the only operational model in this category.

Computational Process Models

The third mainstream of activity scheduling models are computational process models. Computational process models have received increased attention over the years because it was claimed by some scholars that utility-maximizing models do not always reflect the true behavioural mechanisms underlying travel decisions. The argument is that people may reason more in terms of context-dependent IF-THEN-ELSE structures when faced with different constraints and circumstances than in terms of truly maximizing utility-based behaviour. For this reason, several studies have shown an increasing interest in computational process models in order to model activity-diary data. The IF-THEN decision rules are also called production systems that specify which decision will be made as a function of a set of explanatory variables (conditions). In its most basic form, a production system is a set of IF-THEN rules which can be represented in the following form: IF (condition=*X*) then (action=*Y*).

The activity scheduling decision or other transport-related decisions are represented as the outcome of such a production system, which consists of several of these decision rules.

The SCHEDULER computational process model, developed by Gärling *et al.* (1989), is the first conceptual framework for understanding the process by which people organize their activities. It was later implemented by Kwan (1997). However, both models are not yet fully operational models.

The AMOS model (Pendyala *et al.*, 1995;1998) is a third example of a typical computational process model. AMOS is an activity-based model that not only tries to simulate the scheduling but also does the adaptation of schedules based on a learning process in which people gain knowledge about the new travel environment. It was one of the first models that considered the option of rescheduling (see also Joh, 2004), at least in a conceptual form, but the model has not been fully operationally implemented to current date. AMOS was applied in papers by Kitamura *et al.* (1995) and by Pendyala *et al.* (1997).

Another model that bears some resemblance with AMOS is the SMASH model (Ettema *et al.*, 1994;2000). SMASH fully concentrates on the process of activity scheduling by adding, deleting or substituting an activity in the schedule or by stopping the scheduling process.

The first fully operational computational process model is the Albatross ("A Learning-Based Transportation Oriented Simulation System") model developed by Arentze and Timmermans (2000). Albatross can be considered as a rule-based system that predicts activity patterns. The proposed scheduling process model in Albatross intends to simulate how individuals frame choices and arrange them into a sequence when they schedule their activities. The scheduling is assumed to be performed in a priority based manner, where the schedule position and timing attributes of higher-priority activities tend to be scheduled first and, if there is space left in the schedule, lower-priority activities are considered next. Albatross considers a pre-defined sequence of choice facets based on an assumed priority ranking of activities by type and an assumed priority ranking of activity attributes. A more thorough discussion of the Albatross model is provided in Chapter 2.

Recently, another model has been made operational for the State of Florida under the name FAMOS: Florida's Activity Mobility Simulator (Pendyala, 2004). FAMOS is a comprehensive multi-modal activity-based micro-simulation system that simulates activity and travel patterns at the level of the individual traveller.

SIMULATION MODELS

The subtle difference between activity scheduling on the one hand and simulation models on the other hand, has lead to some discomfort in the use of terminology when describing a particular model in the research literature. It occurs frequently that activity scheduling models use the term "simulation", while simulation models sometimes pretend to "schedule" activities in time and space. In this section and throughout this dissertation, a number of differences between both approaches have been advanced. The description of the differences will help to understand the fundamental reasons for undertaking the research that has been carried out throughout this dissertation (see also section 1.2).

The first difference was already mentioned previously, i.e. unlike activity scheduling models, simulation modems are not primarily developed with the intention to explicitly capture the process by which people schedule or execute activities. In other words, this means that the subsequence by which people arrive at their scheduling decision (e.g. first making a transport mode decision, than an activity decision, than a timing decision, etc.) is not explicitly modeled. This argument however, is not distinctive since it should be recalled from the previous section that there are also several simultaneous activity scheduling models which also did not explicitly model scheduling processes. Indeed, the activity scheduling behaviour naturally followed from the prediction of the full activity-travel patterns for some activity scheduling models, and this is equally the case for simulation models.

A second important difference is related with the above-mentioned. Given the fact that simulation models are less concerned with activity scheduling principles, they rely more on probability distributions than on well-developed theoretical constructs. This does not imply that the techniques which are used are inferior, but often, simulation models rely more upon the data for deriving decision-theories and to a lesser extent on other a-priori made assumptions or constraints.

This property brings about a third difference, which is that simulation models focus on the evaluation of other TDM than activity scheduling models. This might look odd at first glance since both models predict the same facets (which, where, when, with whom, with which transport mode). Probably, this can be best understood by means of an example. Consider a scenario implementation where a change of the start time of the work episode is proposed for 65% of the population. A feasible way to have such a measure evaluated by an activity scheduling model, is the implementation of a shift in the fixed schedule for work activities after and before a particular point in time (change of start time) for that particular subset of the population. This is more difficult to model by means of a simulation model because of the lack of an explicit schedule of (fixed) activities. Generally spoken, a simulation model will have more difficulties with all kinds of schedule-specific scenarios (another example might be a shift in working hours during a workweek). However, there are other TDM than can be equally well analysed by means of simulation models. An example is the evaluation of population scenarios, such as for instance a change in the composition of single-adult households, household income, car possession, etc. on activity-travel patterns.

While this seems to be an important deficiency, the existence of simulation models is certainly warranted. At a minimum, simulation approaches can be reduced to activity pattern generation models, which can replace conventional trip generation models by converting the assigned patterns to trips. More likely, simulation approaches could replace both the trip generation and distribution models by producing either static (by aggregating time slices) or dynamic (minute-by-minute) origin-destination trip-tables through the simulation of fully specified activity-travel patterns with all activity-scheduling attributes, including locations that correspond to actual geographical locations (Kulkarni and McNally, 2001). Static trip tables can then be input into the mode choice and route choice stages of conventional models, while the dynamic trip tables can serve as input to dynamic traffic assignment or traffic simulation models with the aim of replacing the conventional forecasting processes which are used in these traffic or conventional models. The most important reason for this integration is that the simulation model representation is mostly fairly easy (relying on probability distributions), and is often easier when compared to activity scheduling models. In this respect, simulation models can be some sort of catalyst in making the transition of conventional four-step methodologies towards activity-based models which practitioners often perceive as considerably more complex. This is an important argument in favour of simulation models, certainly in an era where an increased concern seems to exist to move the activity-based framework to practice.

Related with this, is the more recent idea of an integrated multi-agent (traffic-) simulation system (Raney *et al.*, 2003a). The idea here is that a systematic inclusion of transportation network impedance will contribute to better and more robust models. Thus, if one can generate detailed travel plans for each individual, these simulations can execute these plans, while recording for example where conflicts in the form of congestion, delay the plans (Esser and Nagel, 2001). Several groups are developing simulations which can microscopically simulate whole metropolitan areas (e.g. DYNAMIT, 2000; Mahmassani *et al.*, 1995 (DYNASMART); Rickert, 1998 (PAMINA); Gawron, 1998 (LEGO); Rakha and Van Aerde, 1996 (INTEGRATION); Esser, 1998 (OLSIM)). However, the most ambitious and best-known project in this field of research (inclusion of traffic) is TRANSIMS (TRANSIMS, 2003, Smith *et al.*, 1995). The goal of the Transportation Analysis and Simulation System (TRANSIMS) project is to develop a system that combines the functionalities of activity-based travel demand generation, modal choice and route assignment and microsimulation, using advanced methodologies. TRANSIMS predicts trips for individual households, residents, freight loads and vehicles rather than for zonal aggregations of households. Progress has been reported along three lines: creating synthetic populations, simulation of traffic, and generating activity-based transportation demand. The model has caught a lot of attention for its traffic micro-simulation. This module mimics the movement and interactions of travellers throughout a metropolitan region's transportation system, through the execution of their trip plans using a cellular automata model.

But also apart from this detailed inclusion of traffic-simulation, there is a fairly large group of scholars who strongly encourage the use of micro-simulation techniques as such. For instance, Vovsha *et al.* (2002) claim that micro-simulation models allow for much greater realism in the classification of travel demand, when compared to stratified models. Results of validation studies by

Kitamura *et al.* (1997, 2000) (STPG-model; see also under Activity scheduling models) also show that individuals' daily travel patterns can be practically synthesized by micro-simulation. Another micro-simulation application is the RAMBLAS model (Veldhuisen *et al.*, 2000a, 2000b). RAMBLAS aims to assess the intended and unintended consequences of planning decisions related to land use, building programmes and road construction for households and firms. Given the forecasted spatial distribution of dwellings, the distribution of households over dwellings, and the transport network, activity patterns of individuals and households and related traffic flows across the day on the regional road network are predicted. Another application is that by McNally (1995, 1999) and by Kulkarni and Mcnally (2001), in which the use of representative activity patterns (RAP) has been proposed to simulate activity facets such as purpose and duration by drawing from the distributions that are associated with the target pattern. In the identification of RAP's, segmentation (clustering) approaches can be adopted to derive more homogeneous activity patterns, based on variables that are assumed to influence the activity-travel pattern (for instance socio-demographic variables). The identified RAP's are then used for simulating and predicting new activity patterns. Other applications are the MIDAS model (Kitamura and Goulias, 1991; Goulias and Kitamura, 1992; 1996) and the work by Pribyl and Goulias (2004) where individual's daily activity-travel patterns are simulated, incorporating the interactions among members of the household. Cluster analysis is used to classify activity patterns. Decision trees, particularly the CHAID algorithm, are used to take into account the personal and household characteristics. Early micro-simulation work include Brail (1969), Hemmens (1970), Sparmann (1980), Swiderski (1982), Stopher *et al.* (1996).

In addition to the above-mentioned, there are some authors who have adopted simulation techniques as a way of developing synthetic datasets, which can be used in other application areas (Greaves and Stopher, 2000; Stopher *et al.*, 2003) in an effort to reduce the costs of data-collection, as far as permitted by the temporal and spatial resolution of the data.

1.2 Research Motivation

The previous section provided the context for the research motivation and for the methodological contributions that have been advanced in this dissertation.

1.2.1 Activity-Scheduling Versus Simulation Modelling

While the activity-based research area made a significant progress during the last decade, empirical comparisons and benchmarking studies between different types of models are extremely rare. One of the exceptions, comparing the utility-based framework with computational process models, is the work by Arentze *et al.* (2000). In addition to the absolute performance indicators of a particular model, these relative performance indicators are extremely important because they shed light on the more profound and detailed predictive performance of a system. More specifically, it is the only way to get some idea about whether the lack-of-fit of the model is the result of the remaining noise in the data or whether it is due to the model specification as such. Comparisons based on empirical data between the more theory-driven activity scheduling models and the data-driven (micro-)simulation models are, to the best of our knowledge, non-existing. This was one of the main incentives for undertaking the research that has been described in this dissertation.

1.2.2 Evaluating Descriptive Machine Learning

However, benchmarking studies were not only conducted between simulation and activity-scheduling, also *within* each area, alternative techniques and algorithms have been evaluated. In order to achieve this, the Machine Learning domain was taken as a starting point.

Machine learning is a multidisciplinary research field providing a multitude of induction algorithms which aim at acquiring knowledge by learning patterns from data. The domain has assisted researchers all over the world in a large number of application areas in the development of answers towards highly complex problems. The reason for relying on this knowledge domain was that the prediction of travel behaviour in fact implies both the scientific understanding of the mechanisms underlying thought and intelligent behaviour and the possibility to embody this in machines. Machine Learning is particularly well suited for this.

While Machine Learning was relatively unexplored until a couple of years ago in transportation, the area of computational process modelling has introduced several techniques and algorithms that originate from this field. Recently, the area is gaining increased popularity in transportation. It can also be used for solving single facet decision problems, such as the transport mode decision problem.

However, the techniques which are propagated in these researches are often quite straightforward predictive learning algorithms. Machine learning can be

Table 1.1: Overview of Machine Learning tasks and techniques

Machine Learning	**Task**	**Important Characteristic**	**Technique**	**Example**
Supervised (predictive) machine learning	Regression	Predicting a continuous variable	Linear regression, regression trees, neural networks, support vector machines	Predicting sales amount
	Classification	Predicting a categoric dependent variable	Decision trees (CHAID, C4.5), Neural networks, Rule induction (One R), support vector machines	Predicting bankruptcy, transport mode choice, ...
Unsupervised (descriptive) machine learning	Clustering	Identifying homogeneous subpopulations	k-means clustering, latent class clustering, Kohonen neural networks	Market segmentation
	Association analysis	Identifying relationships between items/ variables	Association rules, (Classification based on Association rules (CBA))	Identifying frequently bought products
	Sequence analysis	Identifying relationships between items over time	Sequential association rules, Markov Chains	Identifying time sequence of purchase
	Dependence analysis	Identifying dependencies between items/ variables	Bayesian networks, graphical methods	Identifying dependencies between demographic parameters
(Reinforcement Learning)	Multiple tasks	Learning through experience/ interaction	Q-learning, Temporal differences method	Grid World, Elevator dispatching, Network routing

subdivided into supervised (predictive), unsupervised (descriptive) and reinforcement learning. Some traditional characteristics and examples have been shown in Table 1.1 for each of these machine learning tasks. Predictive machine learning techniques predict the future value of a dependent variable based on patterns learnt from past data. Regression and classification are among the most popular predictive machine learning techniques. In supervised learning, given a set of cases with class labels, the aim of the supervised learning algorithm is to build a model (called classifier) to predict future data objects for which the class label is unknown. Supervised machine learning is perhaps the most frequently adopted type of learning, and has been used in a plethora of application domains such as for the prediction of bankruptcy, credit scoring, sales, etc.

In this dissertation, when considering improvements to the area of simulation or activity scheduling modelling, it was examined whether advanced and more *descriptive* oriented learning algorithms can offer a contribution in these domains. Descriptive machine learning tries to identify patterns or relationships present in the data without presuming a specific dependent variable. Descriptive machine learning are also described as unsupervised learning mechanisms. Unsupervised learning can be defined as the search for a useful structure without labeled classes, optimization criterion, or any other information beyond the raw data. Unsupervised learning can help researchers to discover the whole set of probabilistic relationships existing within the data (*association discovery*) instead of only developing a learning function for one specific dependent variable (supervised learning). It can be seen from Table 1.1 that unsupervised machine learning consists of several tasks such as clustering, association-, sequence- and dependence analysis. Reinforcement learning can in fact also be seen as a form of unsupervised learning, in the way that the only supervisory signal is the reward that is received when it achieves a goal. Having outlined these general ideas and machine learning concepts behind the dissertation, the methodological contributions and choices that were made for both activity scheduling and simulation models can be defined as follows.

1.3 CONTRIBUTIONS OF THE DISSERTATION

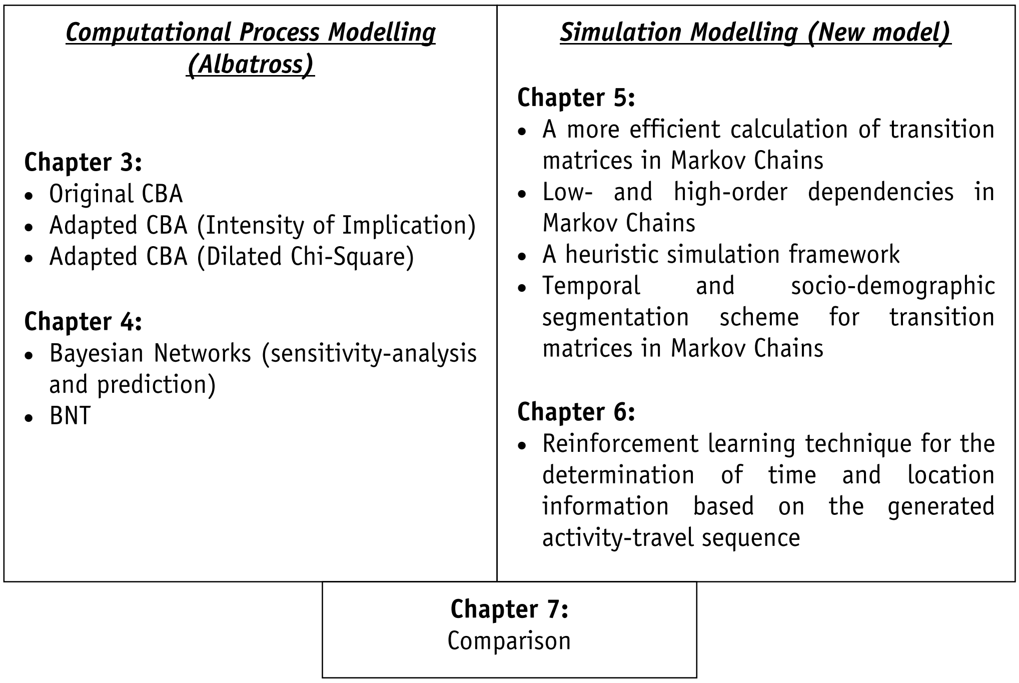
1.3.1 ACTIVITY-SCHEDULING MODELS

As mentioned before, within the area of activity-scheduling models, computational process models are the most susceptible to contributions that originate from the field of Machine Learning. Although it is hard to find a detailed comparison in literature about the functionalities of the different activity scheduling models, some comprehensive reviews are given by Timmermans (2001) and by Guo and Bhat (2001). Based on these reviews, it becomes clear that Albatross is one of the most complete activity-based computational process models to date. Albatross was also the first fully operational activity-based system. The property that Albatross has been developed for the ministry of Transportation, Public Works and Water Management in the Netherlands, probably contributed to this practical orientation. Especially the next version of the system (see Arentze and Timmermans, 2002; Arentze *et al.*, 2003), which currently has passed some final test procedures, has the possibility to evaluate a huge number of policy decisions and TDM. It is obvious that Albatross has received significant appreciation and attention in the scientific literature (Axhausen, 2000, Moons, 2005). Taking these arguments into consideration, it is fair to say that the Albatross model is a justified starting point for this dissertation.

In the original Albatross model, a standard supervised machine learning approach has been adopted, based on a decision tree classification method (CHAID) that has been originally proposed by Kass (1980). Other simple supervised methodologies that have been tested within this framework include C4.5, One R, Zero R and Naïve Bayes in a study by Moons (2005). The latter study also examined the effect of feature selection, a well-known principle in Machine Learning, in Albatross. The Albatross model, the data and a short description of the supervised models that already have been tested within the context of Albatross are introduced in **Chapter 2**.

In order to contribute to the current state-of-the-art, the idea was examined to evaluate the effect of using unsupervised (descriptive) learning systems as the basis for coming to supervised (predictive) learning for the different facets (see Chapter 2) of this transportation model. While this idea of integrating both approaches is still seriously lacking in transportation, it is becoming a more active research topic that is gaining increasing popularity, with some promising predictive results in the field of machine learning (see also Chapter 3). One possible explanation for this increased popularity is that these algorithms are searching globally and are not looking for one specific target attribute. They will therefore contain the *full set* of plausible rules, in which more information can be incorporated. However, this should be interpreted with caution, because the comprehensiveness and the complexity of dealing with the often large number of rules have lead to difficulties and (accuracy versus generality) trade-off questions that are part of a lot of research which is currently going on. In order to deal with this problem, we have also advanced the methodological state-of-the-art by proposing adaptations to these original learning systems. A schematic overview of these contributions is illustrated in the left part of Figure 1.2.

In **Chapter 3**, a classification based on associations (CBA) algorithm is introduced. CBA is one of the best known examples about how descriptive and predictive learning systems can be integrated. The technique focuses on a limited subset of association rules, i.e. those rules where the consequent of the rule is



Computational Process Modelling (Albatross)

Chapter 3:
- Original CBA
- Adapted CBA (Intensity of Implication)
- Adapted CBA (Dilated Chi-Square)

Chapter 4:
- Bayesian Networks (sensitivity-analysis and prediction)
- BNT

Simulation Modelling (New model)

Chapter 5:
- A more efficient calculation of transition matrices in Markov Chains
- Low- and high-order dependencies in Markov Chains
- A heuristic simulation framework
- Temporal and socio-demographic segmentation scheme for transition matrices in Markov Chains

Chapter 6:
- Reinforcement learning technique for the determination of time and location information based on the generated activity-travel sequence

Chapter 7:
Comparison

Figure 1.2: A schematic overview of the contributions of the dissertation

restricted to the classification class attribute. Next, the prototype algorithm has been adapted by coupling it with two other measurements of the quality of association rules: i.e. intensity of implication and an own-developed measure “dilated chi-square”. As mentioned before, the aim of these adaptations is to generate a more accurate and compact decision list, because this is one of the important limitations of integrated approaches as these to current date. These novel contributions have first been tested on several machine learning datasets (see also Janssens *et al.*, 2003a; 2004a; 2005b; Lan *et al.*, 2004; 2006) and later also within the context of the Albatross model (see Janssens *et al.*, 2005e). The second part of Chapter 3 elaborates on these contributions and results.

In the first part of **Chapter 4**, the use of Bayesian networks (BN) has been evaluated. The descriptive nature of Bayesian networks is an important characteristic of this technique (see also Janssens *et al.*, 2003b), which makes that BN are more powerful than CBA because they enable us to conduct detailed sensitivity analyses. This will be illustrated by means of an empirical application. Subsequently, Bayesian networks will be tuned for classification purposes and they will also be tested within Albatross (see also Janssens *et al.*, 2004e; 2004f). In the second part of Chapter 4, the technique is integrated with a decision tree structure (referred to as BNT), in which Bayesian networks are used as the information source for deriving a decision tree. Again (and similar to adapted CBA), the aim of this novel contribution (see also Janssens *et al.*, 2004b; 2005f) is to generate more accurate and compact decision lists than from the straightforward integration of supervised and unsupervised learning.

1.3.2 Simulation Models

As opposed to activity scheduling models, we did not rely upon an existing model within the area of simulation models. The reason is the lack of fully operational and freely available micro-simulation models and the finding that –apart from work by Kitamura *et al.* (2000)– there is no simulation model to current date that explicitly incorporates sequential information in the generation of activity-travel patterns. We found that this may be a deficiency, especially because some “skeleton” activity structure (involving sequence information) is often assumed (and is thus not based on information that is incorporated in the data) and imposed on most simulation models for generating the additional facets of the model. Vaughn *et al.* (1997) also emphasized the importance of an appropriate skeletal structure which imposes constraints and simplifies the simulation of the remaining facets (timing, location, etc.) of the activity-travel pattern. A schematic overview of the contributions for simulation models can be found in the right part of Figure 1.2.

As a first contribution, we have developed and evaluated the implementation of an adapted Markov Chain modelling heuristic and simulation framework in **Chapter 5**. A Markov Chain is a technique for sequential pattern recognition, which is a common task in unsupervised learning. The presented approach is innovative (see also Janssens *et al.*, 2004c; 2004d; 2005c; 2005d) in storing the sequential information in ‘activity bundles’, a term which is introduced to reflect that the information which is kept here represents low- and/or high-order combinations of activities that typically sequentially occur in one particular activity pattern. By doing this, transition probabilities can be calculated in a modified and more efficient way than by means of traditional Markov Chains and the computation of high-order dependencies remains computationally feasible. In addition to this, a novel segmentation procedure has been proposed that is able to cluster sequential activity-travel combinations in terms of socio-demographic or other explanatory information. The segmentation scheme that has been developed is a modified version of a decision tree approach, in the sense that sequential probability information can be used during induction and in the leaves (terminal nodes) of the tree as apposed to the traditional way of only using one single classification attribute (represented by one dependent variable). This type of clustering/segmentation is novel and it may generate promising future applications, that can go beyond the application area of transportation.

While Chapter 5 provided us with a sequence of activities and travel modes, that are derived from sequential dependencies that are present in data, the aim of **Chapter 6** is to allocate time and location information to this framework in order to end up with a more consistent and complete activity pattern. To this end, a reinforcement learning technique has been advanced which makes it possible to allocate location and time information through “trial and error” within a particular space-time prism. Reinforcement learning goes back to the very first stages of artificial intelligence and machine learning and has only been rarely applied within the research area. Consistent with previous research contributions of the dissertation, the reinforcement learning approach is unsupervised in nature. The first contribution of the chapter is the adaptation, evaluation and elaboration of a recently proposed approach (Charypar and Nagel, 2005) towards real empirical data, including a more complex order of activity-travel combinations, the non-restriction to a fixed number of limited activities and the incorporation of real-world and non-fixed travel times. The most important contribution however, is the incorporation of location information in a reinforcement learning environment. As a third contribution, time and location allocation were integrated, which means that the reinforcement *simultaneously* solves the optimal location and time allocation decision.

The performance of the individual components of the simulation model was evaluated separately in Chapters 5 and 6. While these analyses give us a good idea about the predictive performance of the individual facets of a model, a more thorough validation is required in an integrated model. This validation has been provided in **Chapter 7**. In this final chapter, results have been presented that compare the predicted activity patterns of the simulation model with the predicted activity patterns of the Albatross model. The learning algorithms that have been used within Albatross and that were chosen for comparison were selected from the research that has been conducted in Chapters 3 and 4. As a result of this, the contributions of the dissertation were united and a competitive environment was created for both models.

Chapter 2
Data and Architecture of the Albatross Model

2.1 INTRODUCTION

In this chapter, a description of the architecture of the Albatross model is given in order to improve the general understanding and provide the necessary background of the system. Albatross consists of several components that perform specialized functions in the scheduling and the schedule execution process. In our overview, we will focus on the core components of the system, i.e. the Scheduling Engine and the Decision Unit. The latter will be subject to research contributions of Chapters 3 and 4 that were already introduced previously.

Along with the description of the Scheduling Engine and the Decision Unit, an overview of the Albatross-data is given. These data will be used throughout the entire dissertation (also in Chapters 5 and 6), in order to make a feasible comparison. We are grateful to the Urban Planning Group at the TU Eindhoven for providing these data. The chapter concludes with a validation of the predictive capabilities of the model.

2.2 THE ALBATROSS ARCHITECTURE

In order to give a general overview of Albatross, this section discusses the main features of the system. A more detailed description of the Scheduling Engine and the Decision Unit will be provided in sections 2.2.2 and 2.2.3. The overview that is described in these sections mainly relies upon the work by Arentze and Timmermans (2000).

2.2.1 MAIN FEATURES

The Scheduling Engine is one of the main components of the Albatross system. At various moments in the scheduling process, decisions and information about options and conditions for decisions are required. To this end, the Scheduling Engine identifies which condition information is relevant for the Decision Unit, activates the appropriate analytical and rule-based models in the Inference Engine (see infra) to obtain the information and translates the decisions that are returned by the Decision Unit into appropriate operations on the evolving schedule.

The Decision Unit incorporates for each step in the scheduling process a set of decision rules that represent conditional preferences of individuals with constraints regarding the decision options. Only the relevant condition variables (see sections 2.3.1-2.3.6) and decision options are defined in the program code of the system. All decision rules together form a rule-base, which is in fact external to the system and which can be loaded from data files. As mentioned before, in the present system, rules are derived from the data based on principles of supervised learning.

The third main part of the system is an Inference Engine, which consists of a collection of logic-based rules, representing basic knowledge about scheduling constraints. Unlike the decision rules, the Inference Engine is a fixed part of the system, reflecting the assumption that the knowledge it conveys is basic and does not vary across individuals or environments. The incorporated logic-based rules implement dynamic constraints to determine the availability of decision options in each stage of the process, such as for example whether or not an activity fits in a given time slot, possible locations for the activity, etc.

The final two components of the system are a reporter and a scenario agent. Amongst others, the reporter agent provides information to the end user with respect to frequency and contingency analyses and the goodness-of-fit between the observed and the predicted schedules. The scenario agent enables users to define multiple scenarios, e.g. these that are needed for TDM. There are other components which are not mentioned here, i.e. the simulator agent, and the database layers. For a more detailed discussion see Arentze and Timmermans (2000).

2.2.2 THE SCHEDULING ENGINE

The model assumes a pre-defined order of decisions, derived from an assumed priority ranking of choice facets of activities and a priority ranking of activities by type. Decisions are made from high to low priority for each choice facet and within each facet from high to low priority activity. The incorporated model is written in pseudo-code in Figure 2.1 and illustrated by means of a graphical representation in Figure 2.2 (Arentze and Timmermans, 2000).

```
Step 0: For each individual, initialise the current schedule with the given set of fixed activities. Fixed activities are those activities for which the adoption in the schedule, location start time and end time are taken as given (e.g. a work activity)

Step 1: For each individual and for each primary work activity, choose transport mode.

Step 2: For each individual and flexible activity:
  1. Initialise an episode of activity type $a$ by setting the earliest and latest possible start time, the earliest and latest possible end time and minimum and maximum duration to given values
  2. choose whether an activity episode $\tau$ is to be added to the current schedule
  3. if $\tau$is to be added, then
    3.1. set activity type $a(\tau)=a$
    3.2. choose with whom the activity is to be conducted
    3.3. choose the duration of the activity
    3.4. re-define the minimum and maximum duration
    3.5. add it to the current sequence in an arbitrary, preliminary position
  4. if an episode of $a$ has been added, then repeat from 1 (to decide whether a next episode of the activity is to be added)

Step 3: for each individual
  1. define the initial sequence
  2. for each activity and episode
    2.1. choose the time-of-day for the start time of the episode
    2.2. re-define start and end times, given the time-of-day choice
    2.3. add to the schedule in an appropriate position and determine an initial value for the earlier possible start time of the episode

Step 4: For each individual, determine the organisation of trips into tours, as follows:
  1. define the initial sequence
  2. for each activity and episode
    2.1. choose whether the activity is conducted directly after, directly before, in-between specific activities in the sequence, or as a separate trip.
    2.2. add the activity to the sequence in the position that is consistent with the choice of trip type
    2.3. if needed, initialise and add an in-home activity before or after the scheduled activity.
  for each consecutive pair of fixed activities, add an in-home activity in-between the two activities, only if the time gap after subtracting estimated travel time is larger than a pre-defined maximum waiting time.

Step 5: For each individual and each activity episode, determine the transport mode, as follows:
  1. Identify trip-chains as any subsequence of activities beginning and ending at home, and including at least one out-of-home activity
  2. For each trip chain:
    2.1 If the trip chain includes a primary work-activity, then set the transport mode equal to the transport mode of the primary-work activity, else choose the transport mode

Step 6:  For each individual, activity episode:
  1. Choose the location
  2. Determine the travel time
```

Figure 2.1: A pseudo-code representation of the Scheduling engine in Albatross

The keyword *choose* indicates the positions where decision rules deliver inputs. They are indicated by means of ovals in Figure 2.2. As said before, each oval in the figure (decision point) will indicate a set of relevant condition variables for that particular choice facet (see sections 2.3.1-2.3.6). The model first decides on the transport mode for the work activity. Mode choice for work is considered the highest-level decision because this decision determines which person can use the car for a substantial part of the day in cases where there is only one car and more than one drivers' license available in the household.

The second step determines which activities and the number of episodes per activity are added to the skeleton. Time constraints for each candidate activity episode are initialised based on given static household and institutional constraints, if any. When added, the with-whom and duration dimensions are first determined before considering adding a following activity (episode). This reflects the assumption that these two dimensions further define the nature of the activity. For example, individuals may consider a leisure activity of long duration together with others as qualitatively distinct from a leisure activity of short

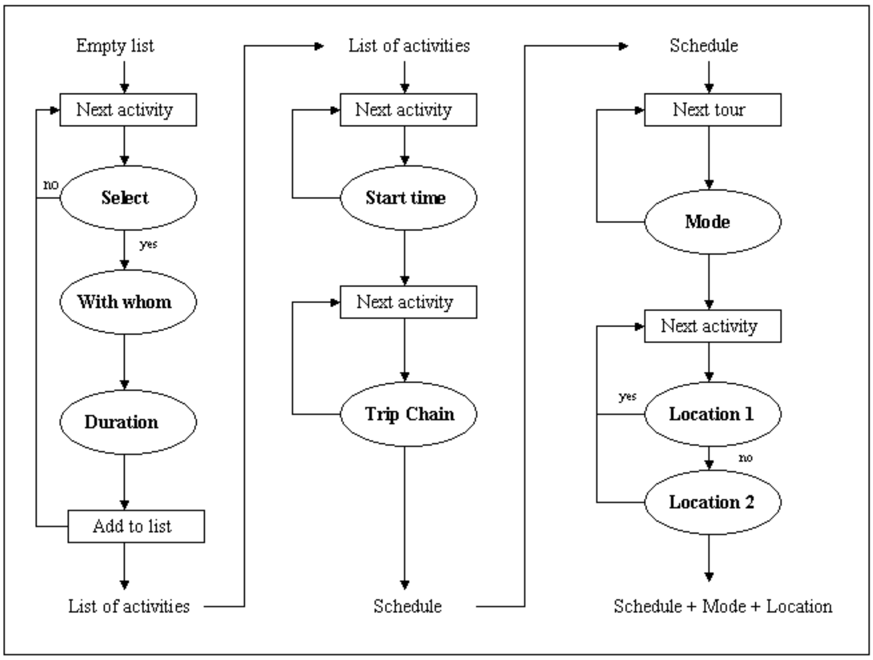


Figure 2.2: A graphical representation of the Scheduling engine in Albatross

duration performed alone. Duration is not exactly specified. Rather the system chooses between typical duration classes for the activity. A duration class defines a normal duration and a range of possible durations within that class.

The third step determines the time of day for each flexible activity in the current schedule. This is modelled as a choice between different time periods assuming a pre-defined subdivision of the day (e.g., early morning, late morning, around noon, etc.). Next, the chosen time of day defines a time interval in which the start time of the activity should fall. The system reconsiders the current position of the activity. In some cases, the choice of time-of-day uniquely determines a position. In other cases, there remains a choice between several feasible positions. Then, the system decides arbitrarily.

The fourth step determines the organisation of trips into tours by choosing for each flexible activity whether it is conducted on a Before stop (directly before another out-of-home activity in the schedule) an After stop (directly after another out-of-home activity), an In-between stop or on a single stop trip. The choices made in this stage are materialised in the schedule in two ways. First, the activity is repositioned if needed to realise the trip type. After this step, the position of the activity is considered definite. A flexible activity being assigned a definite position is considered a candidate for establishing a trip link as well as a fixed activity. Therefore, the procedure also allows the choice of establishing a trip link between flexible activities. Second, in-home activities are inserted where needed to make the origin or destination locations of activities consistent with the chosen trip types. The final two steps are concerned with the allocation (choice) of transport modes and location.

When the decision engine has made a decision for each choice dimension and when the different choice dimensions are coupled together by means of the inference engine, a predicted activity pattern emerges for every person-day.

The present system is still open for improving the handling of shared activities, i.e. activities that are performed jointly by different household members. In this version, there are no mechanisms incorporated to ensure that coupling constraints in the form of start and end times, as well as car availability between activities in different schedules are met (Arentze and Timmermans, 2000).

2.2.3 The Decision Unit

Introduction

For each choice situation in the scheduling procedure of Figure 2.1, the decision-rule base includes a decision table (DT). Each DT consists of a list of condition variables, a list of action variables and a structure that interconnects these variables. The action variable represents the available choice alternatives for that particular choice facet in the model. The condition variables, possible condition states per variable and the action variables are predefined in a DT. Given this fixed structure, every alternative learning mechanism that is evaluated within this context, needs to be converted to the decision table formalism. A brief explanation is given about this formalism, because of the relevance of these conversions (from a set of decision rules to a DT formalism) in this dissertation. In the second subsection, a couple of alternative methods were given that have been tested in Albatross for inducing decision rules from empirical data.

Decision Table Formalism

The DT has been introduced in the late fifties as a tool to structure complex decisions in manufacturing. DT were frequently used to verify the exhaustiveness, the exclusiveness and the consistency of a set of decision rules. Because of this property, for every possible case within the domain a determined response is returned. This behaviour is not guaranteed by means of traditional production systems and it represents a clear advantage of DTs for any modelling purpose. An example of a DT, taken from Arentze and Timmermans (2000), is given in Table 2.1.

The upper part of the table lists the condition subjects C_i (C_1=Distance; C_2=Parking Facilities) for i=1,...,c. The symbol "-" in a condition subject represents the entire domain of the concerned condition, implying indifference for the state of

Table 2.1: Example of a simple decision table

C1	Distance	D<500	500≤D<1000		D≥1000	
C2	Parking Facilities	-	Bad	Good	Bad	Good
A1	Bike	X	X	-	-	-
A2	Car	-	-	X	-	X
A3	Public Transport	-	-	-	X	-

the condition. The lower part of the table contains the action subject A_k (A_1=bike, A_2=car, A_3=PT) for k=1,..., a. The symbols "X" and "-" respectively indicate that action A_k is or is not to be executed for that particular combination of conditions. The second advantage of representing a rule set in the form of a DT, is that the latter provides a suitable formalism for representing various types of interactions between variables, such as conditional relevance and conceptual interaction. To illustrate this, consider the first column in Table 2.1. This is an example of conditional relevance, because the quality of parking facilities is relevant for the choice of transport mode, only if travel distance is equal or larger than 500 metres. Conceptual interaction is present in the different ways of classifying a distance. If parking facilities are good, the critical travel distance equals 500 meter (below this level bike is chosen and otherwise car). If parking facilities are bad, the critical travel distance equals 1000 meter (below this level bike is chosen and PT). As a result, different attribute profiles of locations can lead to the same mode choice. Besides the traditional focus on knowledge validation and verification stages, applications in knowledge acquisition have been reported in previous work. See Lucardie (1994) and Wets (1998) for an extensive overview.

INDUCING DECISION RULES BY MEANS OF SUPERVISED LEARNING IN ALBATROSS

As mentioned before, in the present system, rules are derived from the data based on principles of supervised learning. The original Albatross system uses a standard supervised machine learning approach, based on a tree classification method (CHAID) that has been originally proposed by Kass (1980). In CHAID, the data are successively bisected using a predictor, preserving the ordered nature of the categories where appropriate. CHAID operates on a nominal scaled dependent variable and maximizes the significance of a chi-squared statistic at each partition. CHAID proceeds in steps: it first detects the best partition for each predictor. Then the predictors are compared and the best one is chosen. The data are subdivided according to this chosen predictor. Each of these subgroups are then re-analysed independently, to produce further subdivisions for analysis. Other simple supervised methodologies that have been tested within this framework include C4.5 (Quinlan, 1993), One R (Holte, 1993), Naïve Bayes (Good, 1965) and feature selection and bagging and boosting variants of each of them in a study by Moons (2005). The C4.5 algorithm is more commonly used than CHAID in the Machine Learning literature. C4.5 recursively splits the sample space into increasingly homogeneous partitions in terms of the response variable, until the leaf nodes contain only cases from a single response class. Increase in homogeneity is achieved by a candidate split that is measured in terms of an information gain ratio. We will rely on this principle for developing other alternative methods within the course of this dissertation. The One R algorithm is an extremely simple classifier, containing one single rule, based on the value of a single attribute. The Naïve Bayes classifier uses the naïve assumption of independence to build a conditional independence model of each attribute given the class. The results of this study are summarized in section 2.4. They can be used for benchmarking with our own results in the first part of the dissertation (Chapters 3 and 4). The research presented in these chapters and the work by Moons (2005) can thus be considered as the conclusion of a joint research effort to explore alternative methodologies for developing rule-based models of travel demand, using the activity diary data underlying Albatross.

2.3 THE ALBATROSS DATA

The activity diary data used in this dissertation were collected in 1997 in the municipalities of Hendrik-Ido-Ambacht and Zwijndrecht in the Netherlands (South Rotterdam region) to develop the Albatross model system. The data involve a full activity diary, implying that both in-home and out-of-home activities were reported. The sample covered all seven days of the week, but individual respondents were requested to complete the diaries for two designated consecutive days. Respondents were asked, for each successive activity, to provide information about the nature of the activity, the day, start and end time, the location where the activity took place, the transport mode, the travel time, accompanying individuals and whether the activity was planned or not. Open time intervals were used to report the start and end times of activities. A pre-coded scheme was used for activity reporting. Different administration modes were used. The response rates varied by mode of administration, and typically ranged between 64 and 82 percent. There were substantial problems with incomplete and inconsistent diaries. To diagnose and correct these problems of data quality, an intelligent computer program, called Sylvia (Arentze *et al.*, 1999), was developed and used to improve the quality of the diary data. The corrected diaries were used for the analyses described in this dissertation. After cleaning, a data set containing a random sample of 1649 respondents was formed. The description of these data is similar to previous data descriptions by Arentze and Timmermans (2000) and Moons (2005). The data did not only include full diary information; there are also additional data files such as household attribute data, facilities data, travel time and distance data that provide useful information for analysis. We will rely upon most of these data in the remainder of the dissertation. The data are presented in more detail in Appendix A.

In order to use this information within the context of the Albatross model, separate datasets need to be formed. The data can be roughly separated as a description of general variables that are used for each choice facet of the Albatross model and a description of more specific variables per choice facet. The general variables include household and person characteristics that might be relevant for the segmentation of the sample, including socio-economic variables such as household type, age group, child index and socio-economic class. The general variables also include information about the activity program at a weekly basis with regard to time engaged in work at the household or person level. Finally, the availability of the car at the household level is incorporated in the general variables. Table 2.2 summarizes these general characteristics.

The specific variables that differ per choice facet, have been summarized in the remainder of the section and are described in detail in Appendix B.

Table 2.2: General variables used in the different choice facets of Albatross

Name	Description	Categories
Day	Day of the week	1:Monday...7: Sunday
Csec	Socio-economic class of the household	1: low...4: high
Cage	Age of the oldest person in the household	1: < 25; 2: 25-44; 3: 45-64;4:> 64
Ccomp	Household type	1: single, no work; 2: single, work 3: double, one work; 4: double, two work; 5: double, no work
Cchild	Presence of children in the household	1: none; 2: younger than 6 3: 6-12; 4: older than 12
Gend	Gender of the person	1: male; 2: female
Ncar	Ratio between number of cars and number of adults	1: less than one; 2: one or more
Hwork1	Hours official work of the person per week	0: 0; 1: 1-24; 2: 25-32; 3: 33-38; 4: > 38
Hwork	Hours official work of the household per week	0: 0; 1: 1-32; 2: 33-38; 3: 39-60; 4: > 60

2.3.1 Mode For Work

As mentioned before, the choice of transport mode for primary out-of-home work activities is considered to be the highest-level decision because this decision determines often which person can use the car for a significant part of the day. Especially in households where there is only one car and more than one driving license, this decision is likely to affect subsequent decisions substantially during the rest of the day. The transport-mode alternatives which are considered in Albatross are: slow transport mode (walk, bike); car driver, public transport (bus, train, taxi, etc.) and car passenger. As trips may involve more combinations of several transport modes, the system defines mode choice as choice of the *main* mode. Since we are only looking at the first decision in the schedule; public transport, slow transport and car passenger are always available. The availability of car driver is dependent on the presence of a car in the household and whether or not the person has a driving license. Characteristics of the spouse and schedule-skeleton decisions are included as well as conditions for the decision. To this end, a first group of variables describe the activity program at the level of the person's schedule skeleton and that of the partner, while a second series of variables determine the work-chain for which the choice of transport mode needs to be made. These latter series include work and travel time information. Both series of variables are further detailed in section B.1 of Appendix B. This dataset is referred to as “Mode for work ” in the remainder of the dissertation.

2.3.2 Activity Selection, Travel Party and Duration

As mentioned in Figure 2.1, the second step in the system is concerned with the decision about the activity selection, the travel party and the duration dimensions.

Obviously, the activity selection includes only two choice outcomes: an activity is added or not added to the current schedule. For flexible activities, this judgement is based upon the opening hours of facilities, if applicable and on a minimum duration that is assumed per activity type.

In case of a positive selection decision, the travel party and the duration dimensions are specified before considering the selection of next activities. Choice alternatives for the travel party dimension are: alone, with others exclusively within the own household and with others including persons outside the own household. The “alone” and “others outside the household” options are considered available in every case, while the availability of “others inside the household” is conditional upon the household composition. Only in multiple-person households, this option is considered available.

For the duration facet, Albatross uses a classification of activities in short, average and long duration activities. These activities are defined relative to activity type so that, for example, a long-duration daily shopping activity may still be shorter than a short-duration social activity. The short-duration is always available, while the feasibility of the long and average classes depend upon temporal constraints. The independent variables for all these facets can be divided into program-, and schedule-level variables and other specific variables for each choice facet that determine some constraints (see section B.2 of appendix B). The data discussed in this section are respectively referred to as “Selection”, “With Whom” and “Duration” in the remainder of the dissertation.

2.3.3 ACTIVITY START TIME

For the specification of start time (see Figure 2.1, step 3), the system distinguishes six episodes of the day: before 10 AM, between 10 and 12 AM, between 12 and 2 PM, between 2 and 4 PM, between 4 and 6 PM and after 6 PM. Temporal constraints are the only binding constraints on time-of day choice. For instance, for shopping, service and leisure activities, the opening hours of the facilities further restrict possible start times. For social activities, there are no timing constraints. There are also no logical constraints imposed on possible sequences of activity types. At this stage, the scheduling process is complete in terms of the selection of activities that need to be done that day. At the schedule and activity level, a significant number of independent variables that have been used in the previous steps, re-occur at this stage. These variables are not only added to cover the extra information given by the previous travel party and duration decisions, but they also describe specific conditions for start time decisions (see section B.3 of appendix B). The dataset is referred to as “Start time” in the remainder of the dissertation.

2.3.4 TRIP CHAINING

The fourth step in the system is concerned with the decision about whether or not to include trip-chaining in the current schedule. It is assumed that a trip link exists between two consecutive out-of-home activities, unless they are separated by an in-home activity. Therefore, to organize trips into trip-chains of one or more trips, this step may lead to the repositioning of existing, flexible activities and inserting new in-home activities. Assuming that *A* denotes the concerned activity, *O* an existing out-of-home activity and *H* a home-based activity, the trip chaining choice alternatives for each feasible out-of-home activity are as follows: after stop (*O-A-H*), before stop (*H-A-O*), in-between stop (*O-A-O*) and single stop (*H-A-H*). The single stop option is considered feasible in every case. The feasibility of inserting a after stop, before stop or in-between stop depends on the possibility to bridge the end time and start time of consecutive activities by travelling time. The set of variables that were used to describe the cases at the program-level, schedule-level and activity-level are summarised in section B.4 of appendix B. This dataset is referred to as “Trip Chain” in the remainder of the dissertation.

2.3.5 TRIP CHAIN TRANSPORT MODE

The fifth step assumes that transport mode decisions are made at trip-chain rather than trip level. A distinction is made between trip-chains including a primary work activity and other trip-chains. The same choice alternatives as defined in section 2.3.1 are considered for this choice. The availability of the car-driver option is evaluated based on driving license, the number of cars in the household and use of car by the spouse. A detailed list of independent variables necessary to determine the transport mode, describe the cases at household/individual, activity-program and tour level. The activity-program and tour-level variables are summarised in section B.5 of appendix B. This dataset is referred to as “Mode other” in the remainder of the dissertation.

2.3.6 LOCATIONS

The final step in Figure 2.1 is concerned with the location choice for each flexible out-of-home activity. For each location choice, the system determines a dynamic location choice-set, dependent on the available time-window for the activity, available locations, travel times and flexible activity duration. There are 7 different categories which can be distinguished in the choice facet, i.e. (i) the nearest location from home, (ii) the nearest location in the context of the tour, (iii) the highest-order location within 5 minutes, (iv) the highest-order location within 10 minutes, (v) the highest-order location within 20 minutes, (vi) the highest-order location and (vii) none of the foregoing. Based on the category that is chosen, a specific location is then chosen. As individuals may choose "other" locations than those defined in the choice set (choice 7), the system considers the selection of a travel-time band as a subsequent and additional choice. If the location choice-set includes more than one location in that band, a location is selected randomly. Just as in previous dimensions, each case is described at different levels including the household/individual, activity program/schedule, tour and activity level. The independent variables used for this facet can be found in section B.6 of Appendix B. The data discussed in this section are respectively referred to as "Location 1" and "Location 2" in the remainder of the dissertation.

2.3.7 FULL ACTIVITY DIARIES

As explained at the beginning of section 2.3, activity diary data are used in the original Albatross system. The separate datasets discussed in sections 2.3.1 – 2.3.6 are derived from these full activity diary data. To this end, and to make the separate datasets suitable for the prediction of individual facets of the Albatross model, some variables changed their nature (continuous versus categorical), others were added and some were left out. The 32 (41-9) explanatory variables of the original diary data, containing 2974 person-day diaries (or 2198 household-day diaries), including a total of 4810 tours, have been described in Table 2.3. The other 9 socio-demographic explanatory variables (or a total of 41 explanatory variables) were already introduced in Table 2.2. The dependent variable that is used in this dataset is transport mode choice. The choice alternatives are slow mode (i.e. walk and bike), car driver and car passenger or public transport (bus, train, taxi, etc.). There are other full diary datasets available, containing only work tours, only non-work tours or alternatives with respect to the transport mode decision (only 2 choice alternatives instead of 3), but these are not used within the context of this dissertation.

Table 2.3: Activity pattern and tour characteristics
(AP = activity pattern of the concerned person on the concerned day in which the concerned tour is embedded; C = the concerned tour)

Label	Definition	Categories
Avcar	Car is available in terms of availability driving license and car in household	0: no, 1: yes
Nsec	Number of non-work outhome activities in AP	0: 0, 1: 1, 2: 2, 3: 3-4, 4: > 4
Two	Total time of work in AP (in minutes)	0: 0, 1: 1-90, 2: 91-392, 3: 393-507, 4: 508-545, 6: > 545
Ttot	Total time of primary and secondary work out-of-home in AP (minutes)	1: ≤ 60, 2: 61-150, 3: 151-248, 4: 249-375, 5:376-525, 6: 526-609, 7: > 609
Yserv	There is at least one shopping or service activity in AP	0: no, 1: yes
ySoLei	same for social/leisure outhome activity	0: no, 1: yes
YBget	same for a bring/get person or goods activity	0: no, 1: yes
CBT	Earliest possible begin time of C (in 24 hour notation)	1: < 815, 2: 816-1045, 3: 1046-1335, 4: 1336-1710, 5: > 1710
CET	Latest possible end time of C (in 24 hour notation)	1: < 1230, 2: 1231-1540, 3: 1541-1730, 4: 1731-1950, 5: > 1950
Cdur	Difference between CET and CBT (in minutes)	1: ≤ 30, 2: 31-45, 3: 46-75, 4: 76-105, 5:106-138, 6: 139-190, 7: 191-265, 8: 266-442, 9: > 442
CNout	Number of out-home activities in C	1-4
Ctwo	Total time of work in C (in minutes)	0: 0, 1: 1-70, 2: 71-250, 3: 251-460, 4: 461-520, 5: 521-554: 6: > 554
CTtot	Total time of primary and secondary work out-of-home in C (minutes)	0: 0, 1: 1-135, 2: 136-265, 3: 266-465, 4: 466-520, 5: 521-555, 6: > 555
CyServ	There is at least one shopping or service activity in C	0: no, 1: yes
CySoLei	same for social/leisure outhome activity	0: no, 1: yes
CyBget	same for a bring/get person or goods activity	0: no, 1: yes
Aty1	Type of the first activity in C	1: work, 2: bget, 3: grocery, 4: service, 5: non-groc, 6: leisure, 7: social, 8: other
Aty2	Type of the second activity in C	0: home, 1: work, 2: bget, 3: grocery, service or non-groc, 4: leisure or social, 5: other
Awith	Persons with whom first activity in C is conducted	0: none, 1: only others inside household, 2: others outside household involved

TTbike	Shortest travel time by bike of tour C (in minutes)	0: 0, 1: 1-10 , 2: 11-28, 3: 29-36, 4: 37-56, 5: 57-72, 6: 73-94, 7: 95-132, 8: > 132
Rcabi	Ratio car and bike travel time (in %, shortest tour times)	1: ≤ 21, 2: 22-30, 3: 31-50, 4: 51-100, 5: > 100
Rpubi	Ratio public transport and bike travel time (in %, shortest tour times)	1: ≤ 100, 2: 101-139, 3: 140-200, 4: 201-250, 5: 251-282; 6: > 282
Rpuca	Ratio public transport and car travel time (in %, shortest tour times)	1: ≤ 100, 2: 101-523, 3: 524-646, 4: 647-875; 5: > 875
Textra2	Extra travel time to reach location of order 2 (min. obj. bike time)	0: 0 or not available, 1: ≤ 10, 2: 11-20, 3: 21-24, 4: 25-30, 5: > 30
Textra3	same for order 3	0: 0 or not available, 1: ≤ 12, 2: 13-20, 3: 21-24, 4: 25-34, 5: 35-38, 6: > 38
Textra4	same for order 4	0: 0 or not available, 1: ≤ 24, 2: 25-38, 3: 39-108, 4: 109-124, 5: 125-132, 6: > 132
Pbrget	Partner has a bring or get activity during tour C	0: no, 1: yes
Pserv	Partner has a grocery, service or shopping activity during tour C	0: no, 1: yes
PTmax	Partner maximum objective bike travel time across activities during tour C	0: 0, 1: 1-18, 2: 19-36, 3: 37-59, 4: > 59
yAvSlo	Minimum sum of duration of activities in C plus minimum bike travel time ≤ maximum duration of C (= CDu)	0: no, 1: yes
yAvPu	same for public transport travel time	0: no, 1: yes
Aprim	Primary activity of the tour	1: work, 2: service, 3: bget, 4: social, 5: leisure, 6: other

2.4 VALIDATION OF THE MODEL

It became clear from the previous sections that the Albatross model contains nine different facets, that is to say: nine different decisions that are returned by the Decision Unit. For every dimension, a separate model needs to be built by relying on the set of independent variables. Independent from the model that is used for these predictions, a methodology should be developed to test the validity of the model.

Assessing the predictive performance is by no means a trivial exercise. One could use the same data for both training and estimating the accuracy of the developed model (resubstitution estimate). However, this estimate is often biased towards the training data because the model has been built on the training data as well. Another popular method for performance assessment is the *k*-fold cross-validation method where the data set is typically split into *k* mutually exclusive folds of nearly equal size. The developed model is then trained *k* times, each time using *k*-1 folds for training and the remaining fold for evaluation. The cross-validation performance estimate is then obtained by averaging the *k* validation fold estimates found during the *k* runs of the cross-validation procedure. The variance of this estimate can also be easily computed. The most common value for *k* is 10. The advantage of using cross-validation is that the variance of the resulting estimate is reduced as *k* is increased. The disadvantage of the method is that the training algorithm has to be rerun from scratch *k* times, which means it takes *k* times as much computation to make an evaluation. Within a data mining/machine learning literature, cross-validation is often used for assessing the performance of predictive classification techniques on small datasets. For large datasets, one may also use cross-validation but a single training/test set split up is also quite commonly adopted. In this latter approach, only a subset of the cases is used to build the models (i.e. “training set”), while the other part of the cases is presented as “unseen” data to the models (“validation or test set”). The decline in goodness-of-fit between this “training” set and the validation set is taken as an indicator of the degree of overfitting.

A random sample of 75% of the cases formed the training set; the remaining 25% of the cases were used as a test set in this dissertation. Clearly, the larger the size of the training set, the greater the power of the algorithm to find relationships in the data. Given the size of the total sample, the 25% subset was judged to be sufficient to allow a sufficiently reliable validation test (Arentze and Timmermans, 2000, p. 252). The same splitting criterion has also been used in previous Albatross-related research efforts and thus allows for a fair and equal comparison of results. The splitting criterion has been applied throughout the whole dissertation, except for some of the experiments in Chapter 4, where the 75-25 split will be enhanced by additional cross-validation experiments in order to support the argument of stability that will be introduced in Chapter 4.

Also consistent with previous studies, the predictive performance has been evaluated at three different levels. The first level is the choice facet level. This level simply measures the accuracy of the learning algorithm before full activity-travel patterns are predicted by the Albatross model. The second level of evaluation was carried out at activity pattern level. To this end, Sequence Alignment Measures (SAM) were used to measure the degree of similarity between the observed and the fully predicted activity sequences by Albatross. Finally, an additional validation measure was used, the trip matrix level, to compare the correlation coefficients between the observed and predicted origin-destination matrices. The remainder of this section discusses each of these validation measures in more detail and gives an overview of the empirical results that have been achieved by previous empirical studies.

2.4.1 CHOICE FACET LEVEL

The presentation of the choice facet level does not need a lot of additional explanation. The idea is simple: a learning algorithm acquires knowledge through the observation of the training sample. Next, the knowledge in the model is adopted for the prediction of the nine dependent variables, i.e. for each of the nine choice facets in the Albatross model. Finally, the prediction of the model is evaluated by comparison on a –for the model unseen– test set.

The partial results of the study by Moons (2005) have been summarized in Table 2.4. As mentioned before, a random sample of 75% of the cases were used as a training set; the remaining 25% were used as test set (see supra). For this reason, no variances (standard errors) of accuracies were reported in the study but the size of the test set has been considered as sufficiently large, such that variations were included in the test data distribution and as a result reliable accuracy estimates were reported.

The study examined –among others– the result of extremely simple classifiers in Albatross. The algorithms of these simple classifiers that are shown in Table 2.4 are zero R (containing no rules, simply the majority class is used), one R (containing one single rule) and Naïve Bayes (assuming no dependent relationship between the explanatory variables). These simple classifiers have been compared with the standard CHAID-algorithm that is used in Albatross. Obviously, the higher the accuracy estimate, the better the prediction capability of the learning algorithm. To better understand the meaning and impact of these accuracy percentages, a so-called null-model can be used for evaluation. A null-model is a model that does no partitioning of the condition space and in fact classifies every case in the dataset according to the majority class of the data. The difference between the accuracy of the null-model and the accuracy of the learning algorithm can be seen as a measure of improvement that can be fully attributed through the use of the learning algorithm. In the comparison by Moons (2005), the zero-R algorithm is equivalent to the null-model, and can therefore be used as a measure for comparison.

It was already briefly mentioned in Table 1.1 that regression methods could also have been used as alternative techniques for classification and supervised learning. However, their application is less suited within the Albatross model, given the categorical nature of the dependent variables for each of the nine facets of the model. While this drawback may be solved, it should be pointed out that a regression technique that can be tuned for classification on the one hand and the standard classification methods that are shown in Table 2.4 on the other

Table 2.4: Benchmarking results at choice facet level (accuracy percentages)

Choice Facet	Zero R	One R	Naïve Bayes	CHAID
Selection	66.9	67.7	67.4	72.4
With Whom	35.5	40.8	45.8	50.9
Duration	33.4	34.8	37.0	41.3
Start time	17.2	22.7	31.8	39.8
Trip Chain	53.3	69.9	76.5	83.3
Mode for work	52.5	59.5	64.1	64.8
Mode other	38.8	41.3	45.0	52.8
Location 1	37.5	43.5	47.5	57.5
Location 2	20.0	23.4	28.1	35.4

hand, differ conceptually. First of all, standard regression techniques have a strong linearity assumption which is often an unplausible assumption in reality. Even when the non-linearity restriction can be relaxed through the use of a more advanced regression specification (see Hastie *et al.*, 2001 for a comprehensive overview), it is assumed that the nonlinear specification is known in advance through domain knowledge and/or exists in the data. In order to relax both assumptions (linearity and existence of domain knowledge), a nonparametric approach is recommended. Nonparametric methods are defined as not relying on the estimation of parameters that describe the distribution of the variable of interest in the population. Decision trees or other general rule based methods (see Chapter 3 and 4) are particularly well suited for this purpose. Both techniques are also favoured in terms of interpretation, especially because the relationships between the independent variables of a regression model are somewhat more problematic and less intuitive. Apart from these differences, previous research is mainly inconclusive in terms of predictive comparison and depends upon the model specification of the regression technique, the (in)availability of domain knowledge and application field (e.g. size of the dataset) (Perlich *et al.*, 2004, Stärk and Pfeiffer, 1999).

2.4.2 ACTIVITY PATTERN LEVEL

The assessment of the goodness-of-fit of the Albatross model at pattern level requires the choice of an appropriate measure. The problem at hand, predicting activity patterns, implies that the goodness-of-fit measure needs to be flexible in that it allows the inclusion of sequential and categorical information. Most of the facets of activity patterns, such as mode choice, activity type, etc. are categorical in nature, but the facet of activity scheduling implies sequential information. Most similarity measures developed in transportation science, however, do not properly capture sequential differences among activity patterns. The Sequence Alignment Method (SAM), originally introduced in time use research by Wilson (1998) and later adapted by Joh *et al.* (2001b) towards an activity-based modelling framework is one of the exceptions that is able to capture both sequential and categorical differences among activity patterns (Arentze and Timmermans, 2000). The fundamental features of the Sequence Alignment Method can be summarized as follows.

Let two sequences to be compared, *s* and *g*, and let they have *m*+1 and *n*+1 elements. *S* and *g* can be respectively shown as $s=s[s_0 \ldots s_m]$ and $g=g[g_0 \ldots g_n]$ with $m \geq 0$ and $n \geq 0$. Both sequences are respectively called the source and the target sequence. Similarity is then defined as the total *amount of effort* which is required to equalize sequence $s=s[s_0 \ldots s_m]$, with sequence $g=g[g_0 \ldots g_n]$. To calculate the total amount of effort, SAM uses insertion, deletion and substitution operations. Each operation involves a certain amount of effort. In our comparisons, we assumed that insertion and deletion operations incur the same cost of one unit, while substitution of an element requires twice that cost. Obviously, the lower amount of operations that are needed to equalize two sequences, the more similar the sequences are. In the ideal case, observed sequences are completely similar to the sequences that are predicted by the Albatross model, and SAM will result in a total distance of 0. There is no such upper-bound for the distance measure since this is determined by the length of the sequences under comparison.

The first set of four measures that will be provided in Table 2.5 as goodness-of-fit measures, indicate the uni-dimensional SAM for the different activity pattern attributes separately (activity type, with whom, location, transport mode). The UDSAM indicates a weighted sum of uni-dimensional SAM costs across the three dimensions, whereby activity type was given a weight of two units and the other attributes a weight of one unit. Unfortunately, the conventional SAM can only handle uni-dimensional strings. The uni-dimensional SAM can capture the intra-sequential relationships between elements of an attribute but not the inter-relationships between elements of different attributes. Therefore, a multidimensional extension has been developed by Joh *et al.* (2001a) and is also used as such in the Albatross model. The interpretation of MDSAM is similar as its unidimensional counterpart: the lower the MDSAM measure, the higher the degree of similarity between observed and predicted sequences. SAM and MDSAM are preferably used as relative measures which means that they are particularly suited for comparing the performance of multiple learning algorithms.

Some results of the comparison study by Moons (2005) have been summarized at pattern level in Table 2.5. The values that have been reported are the result of the full activity-travel patterns that are predicted by the Albatross model (see Figure 2.2) by means of the respective predictive learning algorithms that are shown in Table 2.5.

It can be seen from this table that Zero R and One R perform surprisingly well here; the results even seem to suggest that the use of a learning algorithm has no or very little effect on the performance at activity pattern level.

Table 2.5: Benchmarking results at activity pattern level

SAM distance measure	Zero R	One R	Naïve Bayes	CHAID
SAM activity-type	3.130	3.027	3.022	2.777
SAM with whom	3.464	3.312	3.225	3.168
SAM location	3.251	3.184	3.107	3.127
SAM mode	5.018	4.592	4.781	4.626
UDSAM	17.993	17.142	17.156	16.475
MDSAM	8.951	8.474	8.671	8.333

2.4.3 *Trip Matrix Level*

The last measure to evaluate the predictive performance is carried out at trip level (see Table 2.6) in the study by (Moons, 2005). The origins and destinations of each trip, derived from the activity patterns, are used to build OD-matrices. The origin locations are represented in the rows of the matrix and the destination locations in the columns. The number of trips that is undertaken from a certain origin to a certain destination is used as a matrix entry. A third dimension was added to the matrix on which the interactions were broken down. The third dimensions considered are day of the week, transport mode and primary activity. The bi-dimensional case (no third dimension) was considered as well. The measure that will be used for determining the degree of correspondence between the observed and predicted matrices is defined as the correlation coefficient. The correlation coefficient is calculated between observed and predicted matrix entries in general and for three trip matrices that are disaggregated, based on some selected trip facets. As mentioned before, the facets considered include transport mode, day of the week and activity. Also in this case, the reported

Table 2.6: Benchmarking results at trip matrix level

Dimension	Zero R	One R	Naïve Bayes	CHAID
None	0.925	0.928	0.917	0.937
Mode	0.787	0.862	0.842	0.836
Day	0.925	0.937	0.919	0.944
Primary Activity	0.766	0.801	0.800	0.830

correlation coefficients are calculated after application of the full Albatross model (see Figure 2.2) and by means of the respective predictive learning algorithms. In order to calculate the correlation coefficient, cells of the matrix are rearranged into one array and the calculation of the correlation is based on comparing the corresponding elements of the predicted and the observed array. Thus, for the OD-matrices that are disaggregated on the day of the week, the cells of the matrices on weekday, Saturday and Sunday are rearranged into three separate vectors, and these three vectors are then combined into one single vector. Obviously, the correlation coefficient is a relative measure that is preferably used in a comparison with other existing learning algorithms. The higher the correlation coefficient, the better the learning algorithm is capable of predicting activity-travel patterns. An ideal prediction would thus obtain a correlation of 100% but as can be seen from Table 2.6, a fairly high degree of correspondence could already be achieved.

Chapter 3
Classification based on Associations

3.1 INTRODUCTION

As explained in Chapter 1, given a set of cases with class labels as a training set, the aim of classification is to build a model (called classifier) to predict future data objects for which the class label is unknown. A classifier is thus required to learn (i.e. to approximate the behaviour of) a function which maps a vector of independent variables $\left[X_1, X_2, \cdots, X_N\right]$ into one of several classes $[Y_1, Y_2, \cdots, Y_N]$ by looking at several input-output examples of the function. In the case of Albatross, this implied that for every single facet of the model, a classifier needed to be established to use in the Decision Engine of the system. As mentioned before, this type of learning is also commonly referred to as supervised or predictive learning (see Chapter 2 for some results). However, the effect of incorporating unsupervised learning mechanisms remains mostly unexplored in the field of transportation modelling.

One of the best known unsupervised learning mechanisms are association rules. Association rules in fact measure co-occurrence in data. This means they measure, but do not explain, interdependency effects between variables. In other words, association rules are able to quantify the amount of co-occurrence between data but not the reason for its existence (Brijs, 2002). Other descriptive learning mechanisms (than association rules) are probably better suited for this type of reasoning. A good example of such an application is introduced in Chapter 4.

Association rules have received significant attention for extracting knowledge from (large) databases. Their study is focused on using exhaustive search to find all rules in data that satisfy the user-specified minimum support and minimum confidence criteria. For this reason, the extraction of association rules seems at first glance less suited for classification tasks such as those needed within the context of the Albatross model.

However, in recent years, extensive research has been carried out to integrate both approaches. By focusing on a limited subset of association rules, i.e. those rules where the consequent of the rule is restricted to the classification class attribute, it seemed possible to build fairly good classifiers. Associative classification has been first proposed in a classification based on associations algorithm (CBA) (Liu *et al.*, 1998), in which the popular Apriori algorithm (see infra) has been proposed to extract a limited number of association rules with their consequents limited to class labels. These rules are then sorted by descending confidence and are pruned in order to get a minimal number of rules that are necessary to cover training data and achieve satisfying accuracy. CBA is the best known associative classifier. Another associative classifier, ADT (Wang and Zhou, 2000) organizes the rule sets in the tree structure according to its defined relations. The decision tree pruning technique is then applied to remove rules which are too specific. CPAR (Yin and Han, 2003), CMAR (Liu *et al.*, 2001b) and CAEP (Dong *et al.*, 1999) are other examples of associative classification algorithms. They respectively propose expected accuracy, weighted chi-square and growth rate as rule interestingness measures, and all perform classification based on multiple rules that the new sample fires.

This chapter is devoted to the improvement of the CBA algorithm in order to generate a more accurate and compact decision list, which is convenient for decision makers to understand and adopt. The CBA algorithm has been chosen based on the good accuracy results (Liu *et al.*, 2001a, 2001b) that could be obtained on the UCI Machine Learning repository (Blake and Merz, 1998). This repository is frequently used within the field of machine learning, especially when the performance of a new classification system needs to be evaluated and benchmarked. The remainder of this chapter is organised as follows. In section 2, we will give a short introduction into the basic concepts and definitions of association rules discovery. An efficient algorithm to discover all association rules will be provided. In section 3, we will explain how association rules can be tuned for classification purposes. In section 4, the original CBA-system (referred to as “Original CBA”) is applied within the context of the Albatross model in order to examine whether the good results on UCI data can also be maintained for transportation modelling. In section 5, we will propose improvements to the original CBA-system (referred to as “Adapted CBA”). Section 6 quantitatively validates the improvements within the Albatross model. In Section 7, a more explanatory and descriptive analysis of the quantitative results that were obtained in section 6 is provided. The chapter ends with a summary of the most important insights and findings that have been obtained. The final sections also reported some additional topics which are still open for future research.

3.2 ASSOCIATION RULES: DEFINITIONS AND ALGORITHMS

3.2.1 PREFACE

In 1996, it was claimed by (Fayyad *et al.*, 1996) that "our capabilities of collecting and storing data of all kinds have far outpaced our abilities to analyse, summarize, and extract knowledge from data". This tendency is not only perceivable within a business context, but also in more applied areas such as transportation. Trends as the increased use of GPS and PDA devices for collecting location information for instance, will largely contribute to this. In 1993, Agrawal *et al.* (1993) recognized a lack of functionality in database systems for users to take advantage of the huge amounts of retail transactional data. Therefore, they were the first to introduce the technique of association rules to mine a large collection of transactions for hidden patterns of consumer purchase behaviour. Their work was quickly absorbed by other researchers in the machine learning/data mining field who understood the applicability and importance of the technique. Examples of other application domains are cross-selling (Anand *et al.*, 1997), finding co-occuring medical tests from a health information system (Viveros *et al.*, 1996), reducing fall-out in telecommunications systems (Ali *et al.*, 1997) and many others. Within the transportation research community in general, the use of association rules is limited (Keuleers *et al.*, 2001, 2002) and within activity-based scheduling models it has not yet been tested before.

Next, a number of definitions will be introduced that formally describe the technique of association rules. The techniques (and definitions) originate from a business context, but the examples below the definitions give an interpretation in the field of transportation.

3.2.2 DEFINITIONS

Definition 3.1: An item

Let i_k be an item. An item is defined as a combination of an attribute and a value. ■

Example: 1 Child (may be represented as Child=1)

Definition 3.2: A set of items (I)
Let $I = \{i_1, i_2, \ldots, i_k\}$ be a set of all the items that occur in the data. ■

Example: Consider all possible values that variables in the data can take: 1 child, 2 children, 3 children, >3 children, 1 car, 2 cars, >2 cars, etc.

Definition 3.3: A transaction (T)
A transaction T is a subset of items such that $T \subseteq I$. In our application, a household represents a transaction. ■

Example: A particular household (transaction) may be represented by 3 children, 1 car, a low socio-economic class, etc.

Definition 3.4: A transaction database (D)
Let D be a group of transactions, that form a database. ■

Example: Consider 1000 households in the transaction database (D), each household defined by a transaction (T).

Definition 3.5: An itemset (X)
We say that a transaction T contains X, a selection of items in I, if $X \subseteq T$. An itemset that contains k items is a k-itemset. ■

Example: An itemset is a set of items such as {children=4,socio-economic class=low}, which is a subset of T because the household (T) can also be represented by the number of cars. This example represents a 2-itemset.

Definition 3.6: An association rule
An association rule is an implication of the form X => Y, where $X \subset I$, $Y \subset I$ and $X \cap Y = \emptyset$. The implication "=>", means that both itemsets X and Y frequently occur together. The rule does not identify any causal relationships but measures co-occurrences in the data. ■

Example: An association rule expresses which items frequently occur together in the data. For instance, the hypothetical rule Children=4 => Socio-economic class=low indicates that people who have four children, also belong to a low socio-economic class. Note that in this case all items of variables can occur in the right-hand side of the rule, which means that the right-hand side is not restricted to class (dependent) variables in traditional association rules.

Definition 3.7: Confidence of an association rule
The rule *X* => *Y* holds in the transaction set *D* with confidence *c* if *c*% of transactions in *D* that contain *X* also contain *Y*. ■

Example: Assuming that the confidence of an association rule Car=1 => Driving license=Yes is 95%, means that given that people have one car, they also have a driving license in 95% of the cases.

Definition 3.8: Support of an association rule
The rule *X* => *Y* has support *s* in the transaction set *D* if *s*% of transactions in *D* contain $X \cup Y$. ■

Example: The support of an association rule gives an idea about the importance of the rule. Suppose that the support of the rule Car=1 => Driving license=Yes is 2%, this means that both items (Car=1;Driving license=Yes) occur together in 2% of all the cases which are in the dataset.

Given a set of transactions *D*, the problem of mining association rules is to generate all association rules that have support and confidence greater than a user-specified minimum support (minsup) and minimum confidence (minconf) (Agrawal *et al.*, 1993).
As illustrated above, "transactions" can be interpreted as any set of variables that frequently co-occur together. The next section describes the computational details of the algorithms that are used to discover such associations.

3.2.3 ALGORITHMS

Several algorithms have been proposed in the literature to discover association rules (Agrawal *et al.*, 1993, Agrawal and Srikant, 1994, Brin *et al.*, 1997, Mannila *et al.*, 1994). Almost every algorithm has the following general two-phase methodology in common.
The first phase involves looking for so-called *frequent itemsets*, i.e. itemsets for which the support in the dataset equals or exceeds the minimum support threshold that is set by the user (see under section definitions). This is computationally the most complex phase due to the number of possible combinations of items that need to be tested for its support value.

As soon as all frequent items are known, the discovery of all association rules becomes straightforward. That is, if both ABCD and AB are frequent itemsets, then it can be calculated whether the rule AB => CD holds with sufficient confidence by computing the ratio *s*(ABCD)/*s*(AB), in which s(ABCD) is the number of cases that contain ABCD and s(AB) is the number of cases that contain AB. If the computed confidence of the rule equals or exceeds the *minconf* threshold set by the user, then it is a valid rule.

However, testing all possible combinations for their support involves the calculation of 2^n-1 frequencies. To illustrate this, consider the following small example. Suppose there are four items in the data $\{i_1,i_2,i_3,i_4\}$ being respectively equal to {gender=male, car=1, children=2, transport mode=car}. Then, finding all frequent itemsets involves checking the combinations listed in Table 3.1.

Table 3.1: Itemsets that need to be checked for support

1-itemsets	2-itemsets	3-itemsets	4-itemsets
$\{i_1\}$ $\{i_2\}$ $\{i_3\}$ $\{i_4\}$	$\{i_1,i_2\}$ $\{i_1,i_3\}$ $\{i_1,i_4\}$ $\{i_2,i_3\}$ $\{i_2,i_4\}$ $\{i_3,i_4\}$	$\{i_1,i_2,i_3\}$ $\{i_2,i_3,i_4\}$ $\{i_1,i_3,i_4\}$ $\{i_2,i_3,i_4\}$	$\{i_1,i_2,i_3,i_4\}$

It can be seen from this small example that this approach becomes already infeasible for a relatively small number of items. For this reason, the Apriori algorithm, which is based on the downward closure principle which states that "all subsets of a frequent itemset must also be frequent", has been developed by Agrawal and Srikant (1994). This principle simplifies the search for frequent itemsets because for some itemsets, it can be determined in advance that they can never be frequent and by consequence, their support does not have to be checked against the data. For the sake, of clarity, the Apriori algorithm has been portrayed in Figure 3.1.

The first pass of the algorithm simply counts item occurrence to determine the frequent 1-itemsets, i.e. itemsets containing just one item. A subsequent pass, say pass *k*, consists of two phases. First, the frequent itemsets L_{k-1} found in the

```
$L_1$ := {frequent 1-itemsets};
$k$ := 2;  // represents the pass number
while ( $L_{k-1} \neq \emptyset$ ) do begin
    $C_k$ := New candidates of size $k$ generated from $L_{k-1}$;
    for all transactions $T \in \mathcal{D}$ do begin
        Increment the count of all candidates in $C_k$ that are contained in $T$.
    end
    $L_k$ := All candidates in $C_k$ with minimum support.
    $k$ := $k$ + 1;
end
Answer := $\cup_k L_k$ ;
```

Figure 3.1: The Apriori algorithm

(k-1)th pass are used to generate the candidate itemsets C_k. To generate these candidate itemsets, the Apriori candidate generation function is adopted, that consists of two steps: a join and a prune step.

The function is probably best understood by means of an example. Suppose there are five frequent three-itemsets (L_3): {{1 2 3}, {1 2 4}, {1 3 4}, {1 3 5}, {2 3 4}}. The union of the first two itemsets results in {1 2 3 4} which is defined as a candidate four-itemset (C_4) because its other three-item subsets {1 2 3} and {1 2 4} have greater than minimum support (join step). If the three-itemsets are sorted into ascending order, as they are in this example, then we only need to consider pairs whose first two elements are the same, because otherwise the resulting itemset would contain more than four items. Apart from {1 2 3} and {1 2 4}; it is also possible to join {1 3 4} and {1 3 5} and thus result in {1 3 4 5}, which is also a potential candidate four-itemset. However, in the prune step, all itemsets $c \in C_k$ are deleted for which some (k-1)-subset of c is not in L_{k-1}. Therefore, the itemset {1 3 4 5} will be deleted because the itemset {1 4 5} is not in L_3. Consequently, C_4 will only contain the candidate itemset {1 2 3 4}.

Next, the database D is scanned and the support of candidates in C_k is verified against the data as can be seen from Figure 3.1. These two operations (candidate generation and support counting) continue until, according to the downward closure principle, no candidate itemsets can be generated anymore. The outcome of the algorithm is guaranteed to include all frequent itemsets (Agrawal and Srikant, 1994).

3.3 Using Association Rules for Classification

3.3.1 Preface

As already mentioned in the introduction, in the case of the Albatross system, a classifier needs to be established for every facet of the model. To make association rules suitable for the classification task, a classification based on associations (CBA) algorithm (Liu *et al.*, 1998) is presented in this section. The CBA method focuses on a special subset of association rules, i.e. those rules with a consequent limited to class label values (values of dependent variables) only; so-called class association rules (CARs). Thus, only rules of the form A=>c_i where c_i is a possible class, need to be generated. Therefore, the Apriori algorithm portrayed above was slightly modified to build these CARs. In addition to this, another modification was needed because datasets that have been used for classification may contain continuous attributes as well. Mining association rules with continuous attributes has been a major research issue in the past (Srikant and Agrawal, 1996a; Yoda *et al.*, 1997; Wang *et al.*, 1998). The adaptation presented in this section to overcome this problem involves the discretization of continuous attributes based on the classification predetermined class target. There are many good discretization algorithms which can be used for this purpose (Fayyad and Irani, 1993; Dougherty *et al.*, 1995; Janssens *et al.*, 2005a).

3.3.2 Definitions

Definition 3.9: A class
A class can be defined as any dependent variable or output variable for which a prediction will be made or which is considered relevant to be examined. In theory, any item (see definition 3.1) can serve as a class attribute (dependent on the purpose of the research). ■

Example: The number of cars, transport mode, start time, travel party, location, etc. are typical examples of class attributes.

Definition 3.10: Class Association Rule
A class association rule is an implication of the form X => c, where $X \subseteq I$, $c \in C$ and C is a set of class labels. ■

Example: A class association rule expresses which items and class labels frequently occur together in the data. For instance, the rule Car=1 => Driving license=Yes indicates that people who have one car, also have a driving license, given that driving license is a class attribute in the data. Indeed, unlike association rules only class attributes occur in the right-hand side of the rule.

Definition 3.11: Ruleitem
A ruleitem is an expression of the form <*condset*, *c*> where *condset* is a set of items, $c \in C$ is a class label. ■

Example: <{(A, 1), (B, 1)}, (class, 1)>, where A and B are attributes that represent condset.

Definition 3.12: Support of condition set of a ruleitem
The support count of the *condset* (called *condsupCount*) is the number of cases in the data *D* that contain the *condset* of the ruleitem. ■

Example: Suppose that consupCount of the rule A=1^B=1 =>Class=1 is 3, this means that the conditions of the rule (i.e. A=1, B=1), occur together 3 times in the data.

Definition 3.13: Support of ruleitems
The support count of the *ruleitem* (called *rulesupCount*) is the number of cases in *D* that contain the *condset* and are labeled with class *y*. ■

Example: Suppose that the rulesupcount of the rule A=1^B=1 =>Class=1 is 3, this means that the conditions (i.e. A=1, B=1) and the consequent of the rule (i.e. Class=1) all occur together 3 times in the data. This is equal to the support of the CAR.

Definition 3.14: Confidence of ruleitems
The confidence of a ruleitem can be calculated as (rulesupCount / condsupCount) *100% ■

Example: If the support count of the *condset* {(A, 1), (B, 1)} is 3, the support count of the *ruleitem* is 2, then the confidence of the *ruleitem* is 66.7%.

Definition 3.15: Frequent ruleitems
Ruleitems that satisfy the minimum support are called frequent ruleitems, others are called infrequent ruleitems.

For all the ruleitems that have the same condset, the ruleitem with the highest confidence is chosen as the possible rule (PR) representing this set of ruleitems. If there are more than one ruleitem with the same highest confidence, one ruleitem is selected randomly. ■

Example: Suppose, we have two ruleitems that have the same condset:
1. <{(A, 1), (B, 1)}, (class: 1)>.
2. <{(A, 1), (B, 1)}, (class: 2)>.
Assume the support count of the condset is 3. The support count of the first ruleitem is 2, and the second ruleitem is 1. Then, the confidence of ruleitem 1 is 66.7%, while the confidence of ruleitem 2 is 33.3% With these two ruleitems, we only produce one PR (assume $|D|$ = 10): (A, 1), (B, 1) → (class, 1) [support= 20%, confidence= 66.7%].

3.3.3 ALGORITHMS

In this section, the original CBA algorithm (Liu *et al.*, 1998) is introduced. It consists of two parts, a rule generator (called CBA-RG), and a classifier builder (called CBA-CB).

CBA-RG

The CBA-RG algorithm generates all the frequent ruleitems by making multiple passes over the data. In the first pass, it counts the support of each individual ruleitem and determines whether it is frequent. In each subsequent pass, it starts with the seed set of ruleitems found to be frequent in the previous pass. It uses this seed set to generate new possibly frequent ruleitems, called candidate ruleitems. The actual supports for these candidate ruleitems are calculated during the pass over the data. At the end of the pass, it determines which of the candidate ruleitems are actually frequent. From this set of frequent ruleitems, it produces the rules (CARs). Let k-ruleitem denote a ruleitem whose condset has k items. Let F_k denote the set of frequent k-ruleitems. Each element of this set is of the following form: <(*condset*, *condsupCount*), (*c*, *rulesupCount*)>.
Let C_k be the set of candidate k-ruleitems. The CBA-RG algorithm is shown in Figure 3.2.
Line 1-3 represents the first pass of the algorithm. It counts the item and class occurrences to determine the frequent 1-ruleitems (line 1). A frequent ruleitem

```
1  $F_1$ = {large 1-ruleitems};
2  $CAR_1$ = genRules($F_1$);
3  $prCAR_1$ = pruneRules($CAR_1$);
4  for ($k$ = 2; $F_{k-1} \neq \emptyset$ ; $k$++) do
5    $C_k$ = candidateGen($F_{k-1}$);
6    for each data case $d \in D$ do
7      $C_d$ = ruleSubset($C_k$, $d$);
8      for each candidate $c \in C_d$ do
9          $c$.condsupCount++;
10         if $d$.class = $c$.class then $c$.rulesupCount++
11     end
12   end
13   $F_k$ = {$c \in C_k$ | $c$.rulesupCount $\geq$ $minsup$};
14   $CAR_k$ = genRules($F_k$);
15   $prCAR_k$ = pruneRules($CAR_k$);
16 end
17 $CARs = \cup_k CAR_k$;
18 $prCARs = \cup_k prCAR_k$;
```

Figure 3.2: The CBA-RG algorithm (Liu *et al.*, 1998)

has been defined in definition 3.15. From this set of 1-ruleitems, a set of CARs (called CAR_1) is generated by the function genRules (line 2). The function genRules checks whether there are ruleitems that have the same condset. If this is the case, the procedure that has been explained in definition 3.15 and in the example following that definition, was applied. CAR_1 is subject to a pruning operation (line 3) (which can be optional). The function pruneRules uses the pessimistic error rate based pruning method in C4.5 (Quinlan, 1993). It prunes a rule as follows: If rule *r*'s pessimistic error rate is higher than the pessimistic error rate of rule r^- (obtained by deleting one condition from the conditions of *r*), then rule *r* is pruned. For a more detailed discussion of the calculation of the pessimistic error rate, we refer to Quinlan (1993). For each subsequent pass (line 4), say pass *k*, the algorithm performs 4 major operations. First, the frequent ruleitems F_{k-1} found in the (*k*-1)-th pass are used to generate the candidate ruleitems C_k using the candidateGen function (line 5). This function is completely analogous as the Apriori candidate generation function that has been explained in section 3.2.3 (see also Figure 3.1). Hereafter, the algorithm scans the database (line 6-7) and updates various support counts of the candidates in C_k (line 8-10). The algorithm uses the ruleSubset function, which basically takes a set of candidate ruleitems C_k and a data case *d* to find all the ruleitems in C_k

whose condsets are supported by *d*. This and the operations in line 8-10 are also similar to those in algorithm Apriori. The difference is that we need to increment the support counts of the condset and the ruleitem separately, whereas in Apriori only one count is updated. This allows us to compute the confidence of the ruleitem. The updating of both support counts is also useful in rule pruning. After the new frequent ruleitems have been identified to form F_k (line 13), the algorithm then produces the rules CAR_k using the genRules function (line 14). Finally, rule pruning is performed (line 15) on these rules. The final set of class association rules is in *CARs* (line 17). The remaining rules after pruning are in *prCARs* (line 18).

CBA-CB

This section presents the CBA-CB algorithm for building a classifier using CARs (or prCARs) identified by CBA-RG. The original algorithm which is used in CBA is shown in Figure 3.3.

The algorithm will first rank all the CARs. As we will show in the remainder of the chapter, this rank will be subject to the modifications which were implemented. The original ranking is as follows: given two rules r_i and r_j, $r_i > r_j$ (or r_i is said having higher rank than r_j), if (1) conf (r_i) > conf (r_j); or (2) conf (r_i) = conf (r_j),

```
R=sort (R);
for each rule $r \in R$ in sequence  do
    temp = ø;
    for each case $d \in D$  do
        if d satisfies the conditions of r then store d.id in temp and mark
        r if it correctly classifies d;
    end
    if r is marked then
        insert r at the end of C;
        delete all the cases with the ids in temp from D;
        selecting a default class for the current C;
        compute the total number of errors of C;
    end
end

Find the first rule p in C with the lowest total number of resubstitution
errors and drop all the rules after p in C;

Add the default class associated with p to end of C and return C (our
classifier)
```

Figure 3.3: Building a classifier in CBA

but sup (r_i) > sup (r_j); or (3) conf (r_i) = conf (r_j) and sup (r_i) = sup (r_j), but r_i is generated before r_j.

If at least one case among all the cases covered by the rule is classified correctly by the rule, the rule is inserted into the classifier by following this sorted descending sequence order. All the cases the rule covers (i.e. they satisfy the conditions of *r*) are removed from the database. A chronological procedure is followed for this removal, which means that cases of the first classification rule are removed before the cases of the second classification rule. This deletion is repeated for every rule that is inserted into the classifier, each time on a reduced dataset, and therefore the process is largely determined by the sorting criteria (based on confidence) that have been explained above. This procedure is also known as database coverage pruning (Liu *et al.*, 2001b). The rule insertion stops when either all of the rules are used or no cases are left in the database. The majority class among all cases left in the database is selected as the default class. The default class is used in case when there are no covering rules. Then, the algorithm computes the total number of errors, which is the sum of the number of errors that have been made by the selected rules in the current classifier and the number of errors made by the default class in the training data.

After this process, the first rule which has the smallest number of errors on the training set is identified as the cut-off rule. All the rules after this rule are not included in the final classifier since they will only produce more errors (Liu *et al.*, 1998) (see also Figure 3.4 and Figure 3.10). CBA has generated better results than C4.5 in a comparative study by Liu et *al.* (1998), when the algorithm was tested on the UCI Machine Learning repository data. In the next section, we will evaluate whether these results can also be maintained for transportation modelling, when evaluated within the context of the Albatross system.

3.4 Original CBA: Results

3.4.1 Setting Thresholds

Before the CBA model can be tested within Albatross, minimum support and minimum confidence thresholds need to be set. Both parameters were kept low (minimum support was set at 1%, minimum confidence at 10%), with the aim of not excluding interesting rules in advance. However, since the original CBA algorithm sorts the CARs based on confidence, the minimum confidence threshold is unlikely to have an impact on the final accuracy results since all the rules with a high confidence parameter will first be added to the classifier. The rules with lower confidence values are probably situated after the cutpoint *p* and are likely to be discarded. This finding was confirmed in the empirical results for all the facets of the Albatross model (see infra). Equally, the maximum number of conditions that can appear in any CAR needs to be set in advance. This number was restricted to 6 in order to ensure the comprehensibility of every single rule.

3.4.2 CHOICE FACET LEVEL

In order to evaluate the performance of CBA, the data was divided into a training and a test set, as explained before in Chapter 2, section 2.4. In Table 3.2, the number of association rules, class association rules and the number of rules in the final classifier has been depicted for every decision that is taken in the Albatross model. As it can be seen from this table, the low minimum confidence and minimum support thresholds, obviously have an immediate impact on the large number of association rules that were generated by the original Apriori algorithm (first column of Table 3.2). We are able to significantly reduce this number, by only focusing on the CARs, i.e. those rules where the right-hand side of the rule is restricted to a class attribute (second column of Table 3.2). Furthermore, the number of rules in the final classifier, i.e. those rules that are situated before the cutpoint *p*¸ is also significantly lower than the number of CARs (third column of Table 3.2).

This rule selection process is better illustrated in Figure 3.4, where the number of correctly predicted cases is shown for every single rule that is added to the final

Table 3.2: Evolution of the number of AR, CARS and rules in the final classifier

Dataset	Number of association rules	Number of CARS	Number of rules in the final classifier
Duration	55522	16379	147
Location 1	54539	9659	234
Location 2	53808	8372	136
Mode for work	57708	6364	172
Mode other	50899	13636	245
Selection	53735	5721	594
Start time	56551	20786	120
Trip Chain	63567	10488	65
With Whom	55508	18921	222

classifier for the "Trip chain" dataset (by example). The vertical line shows the cutpoint *p* (determined on the training data), the rules before this point are included in the final classifier; the others are discarded. Indeed, the slope after the cutpoint decreases in Figure 3.4 because the total classifier simply classifies a higher number of cases incorrectly than those that could be classified correctly, after a particular point in the classifier (*p*).

The accuracy percentages that indicate the predictive performance on the training and test sets within Albatross are presented in Table 3.3. Results in this table are compared with the original CHAID algorithm that is used in Albatross.

When the average results are compared, it is clear that the CBA classifier outperforms CHAID. Also with respect to the algorithms that were introduced in Chapter 2, CBA achieves better results. Only with regard to the "start time" dataset, CBA performs somewhat worse than CHAID. Despite the good results, it can also be seen that the number of rules is higher, although the comparison between number of rules and number of leaves is not completely perfect for comparison. It was already mentioned in the introduction that the higher number of rules is one of the main limitations of these integrated approaches. Section 3.5 further elaborates on this topic. The predictive findings confirm previous results (Liu *et al.*, 2001a, 2001b) that could be obtained on the UCI machine learning repository and it supports our claim that using unsupervised techniques for classification purposes, holds out considerable promise.

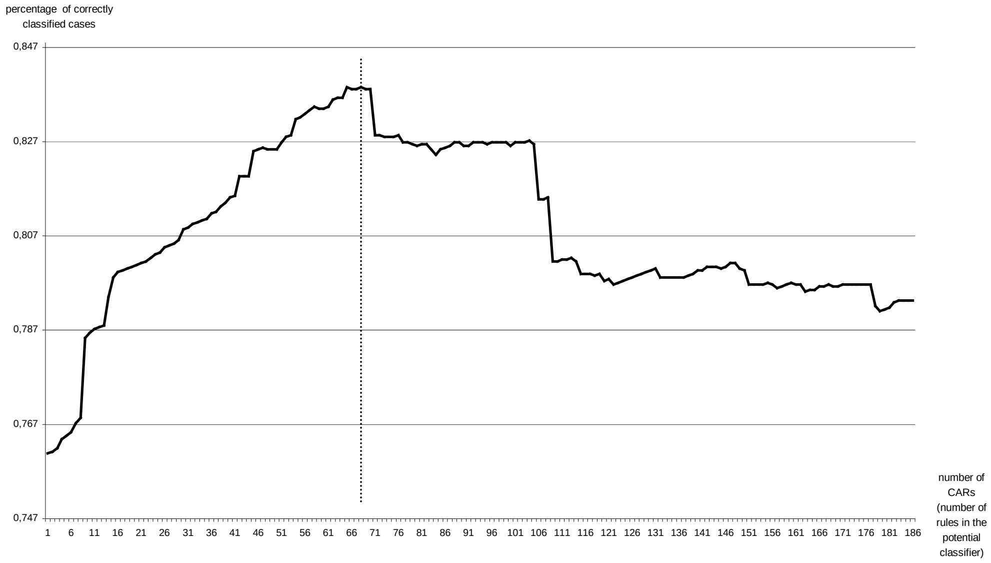


Figure 3.4: Evolution of the number of correctly classified cases for every rule that is added to the final classifier (Trip chain dataset) in original CBA

Table 3.3: Benchmarking results at choice facet level

Dataset	CBA			CHAID		
	Train (%)	Test (%)	Number of rules	Train (%)	Test (%)	Number of leaves
Duration	44.7	39.2	147	41.3	38.8	61
Location1	66.3	62.7	234	57.5	58.9	62
Location2	52.6	41.1	136	35.4	32.6	34
Mode for work	83.5	73.7	172	64.8	66.7	23
Mode other	66.5	60.9	245	52.8	49.5	65
Selection	79.6	78.7	594	72.4	71.6	106
Start time	34.5	33.7	120	39.8	35.4	86
Trip chain	83.9	80.4	65	83.3	80.9	30
With whom	61.1	56.2	222	50.9	48.4	57
Average Accuracy	**63.6**	**58.5**	/	**55.4**	**53.6**	/
Av. Number of rules	/	/	**215**	/	/	**58**

3.4.3 *Activity Pattern Level*

The SAM measure introduced in section 2.4.2 was used to compare the degree of similarity between the predicted and the observed activity patterns. The SAM distance measures, indicating the predictive performance at activity pattern level on the training and on the test set, are presented in Table 3.4. The results show that the performance of the original CBA algorithm is better on the training set than CHAID. However, while results at the test set are still somewhat better than CHAID, a higher degree of overfitting occurred. An analysis of the number of

Table 3.4: Benchmarking results at activity pattern level

SAM distance measure	CBA		CHAID	
	Train	Test	Train	Test
SAM activity-type	1.610	2.712	2.861	2.801
SAM with whom	1.971	3.114	3.225	3.210
SAM location	1.321	3.035	3.181	3.148
SAM mode	2.019	4.414	4.599	4.587
UDSAM	12.871	16.318	16.725	16.629
MDSAM	5.108	8.298	8.457	8.427

rules versus number of leaves (see Table 3.3) further confirmed this finding. With respect to CHAID, no overfitting could be determined at pattern level.

3.4.4 TRIP MATRIX LEVEL

The last measure to evaluate the predictive performance, is carried out at trip level. To this end, the origins and destinations of each trip, derived from the activity patterns, are used to build OD-matrices. In order to determine the degree of correspondence between predicted and observed OD-matrices, correlation coefficients were calculated (see also section 2.4.3). The results at activity pattern level were confirmed by the results at trip matrix level, that is, while the results at the test set are somewhat better for CBA than for CHAID, the degree of overfitting is higher as well (for CBA).

Table 3.5: Benchmarking results at trip matrix level

Dimension	CBA		CHAID	
	Train	Test	Train	Test
None	0.959	0.940	0.954	0.939
Mode	0.911	0.849	0.877	0.846
Day	0.975	0.948	0.960	0.948
Primary Activity	0.902	0.838	0.890	0.832

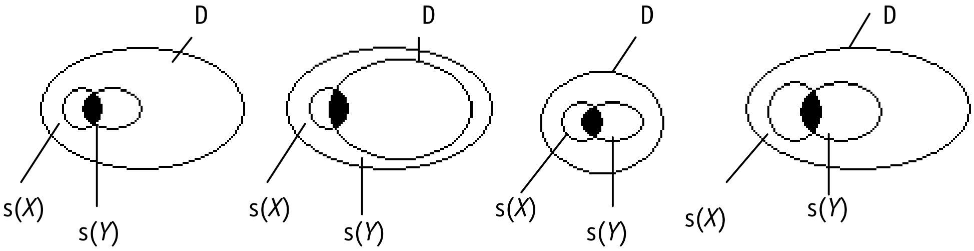
3.4.5 DISCUSSION

The previous sections have illustrated and quantitatively evaluated the use of CBA within the Albatross model. A more qualitative analysis, including a discussion of the variables which appear most frequently in the decision rules, will be provided in section 3.7. The CBA technique uses an adopted version of the Apriori algorithm, which is popular in association rule discovery to generate CARs. The confidence measure is used to determine the sorting of CARs and thus indirectly determines the cutpoint *p* on the training dataset, which forms the final classifier. The empirical results show that CBA was able to achieve better predictive performance at choice facet, activity pattern and trip matrix level when compared to CHAID. The improvement in predictive performance was most dominant at choice facet level. While also at choice facet level, a higher degree of overfitting occurred for CBA, the technique suffered the most from overfitting at activity pattern and at trip matrix level. Obviously, the higher number of rules, which is one of the well-known limitations of integrated approaches as these, may be responsible for this. While it can be seen from Table 3.3 that the amount of the rules is still fairly acceptable in this case, the idea was conceived in the remainder of this chapter to examine whether a more optimal cutpoint could be determined, that simultaneously leads to a reduction in the number of rules and in the degree of overfitting.

3.5 ADAPTING ORIGINAL CBA

3.5.1 PROBLEM STATEMENT

A profound examination of the algorithm presented in Figure 3.3 has revealed that the way by which rules are sorted, can be a potential weakness of the original CBA(-CB) algorithm. Since rules are inserted in the classifier following the sorted confidence order, this will determine to a large extent the composition and the size of the final classifier. Confidence is a good measure for the quality of (class) association rules but it also suffers from certain weaknesses. The aim of this section is to elaborate on these weaknesses and to propose different alternatives. Parts of the subsequent sections are based upon work reported in Janssens *et al.* (2003a; 2004a; 2005b) and in Lan *et al.* (2004; 2006). It can be argued that confidence is not a good parameter to discover the most valuable or interesting rules. An important weakness is that the conditional probability of a rule X => Y is invariable when the size of s(Y) or D varies. The subset of the cases which are covered by the consequent of the rule is given by s(Y), while D is the total number of observations in the dataset. The confidence property is also insensitive to cardinal dilatation (i.e. the size of the subsets increases in the same proportion). Figure 3.5 (taken from Suzuki and Kodratoff, 1998) graphically displays the problem. It can be seen from the figure that the three cases (b, c and d) and the reference case (a) in the figure have the same confidence (indicated by the intersections between the ovals). Let A=s(X), B=s(Y), n=|D|, n_a=|A|, n_b=|B|, and n_{ab}=|AB|. The confidence of rule $X \Rightarrow Y$ is calculated as n_{ab}/n_a. Keeping the numerator and denominator fixed, the confidence is stable when the size of s(Y) or D changes. This is graphically illustrated in Figure 3.5 (b) and (c), where the size of the right inner circle becomes larger in (b) to indicate an increase in *s(Y)*, and where the size of the outer circle decreases in (c) to illustrate a change in the size of D, while the intersection remains the same in both cases when compared to the reference case (Figure 3.5 (a)). Nevertheless,



(a) Reference (b) s(*Y*) increases (c) *D* decreases (d) cardinal dilatation

Figure 3.5: Three cases with constant conditional probability

the rule *X* =>*Y* is intuitively more likely to happen when the size of s(*Y*) increases or when the size of *D* decreases. It is shown that it is not surprising that, when s(*Y*) is close to the size of *D* (this is both the case in Figure 3.5 (b) and (c)), the observations which are covered by the antecedent *X* of the rule, are also included in s(*Y*). Furthermore, the implication will be more meaningful when the size of all the sets grows in the same proportion when compared to the reference case (Figure 3.5 (a)), as it is shown in Figure 3.5(d). For this reason, two novel interestingness measures, i.e. intensity of implication and an own developed dilated chi-square heuristic, were tested to adjust the ranking mechanism in CBA algorithm. The next section elaborates on this.

3.5.2 INTENSITY OF IMPLICATION

Intensity of implication, introduced by Gras and Lahrer (1993) and later adopted and improved by Guillaume *et al.* (1998) and by Suzuki and Kodratoff (1998), measures the distance to random choices of small, even non statistically significant, subsets. In other words, it measures the probability of having as many counter-examples as a randomly-generated rule. The smaller the value of this probability, the higher the value of intensity of implication, and the better the rule.

Consider a database *D*, where |*D*| is the total number of observations in the database, and an association rule $X \Rightarrow Y$. Now, let *U* and *V* be two sets of examples, randomly selected from *D* and being mined with the restriction that both have the same cardinality as *X* and *Y*, i.e., |U| = n_a and |V| = n_b.

Let $N_{u\bar{v}} = |U \cap \bar{V}|$ (with $\bar{V}$ the complement of *V* in *D*), represent the expected number of random negative examples (counter-examples or false positives) under

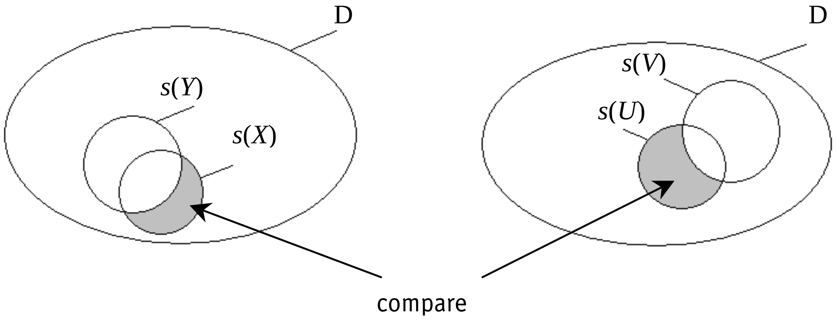


Figure 3.6: Intensity of implication

the assumption that U and V are independent and $n_{a\bar{b}}$ the number of negative samples observed on the rule. Now, if $n_{a\bar{b}}$ is unusually small compared with $N_{u\bar{v}}$ (see Figure 3.6), the one we would expect at random, then we say that the rule *X*⇒*Y* has a strong statistical implication. In other words, the intensity of implication for a rule *X*⇒*Y* is stronger, if the quantity Pr[$N_{u\bar{v}} \leq n_{a\bar{b}}$] is smaller. Intensity of implication is then defined as 1-Pr[$N_{u\bar{v}} \leq n_{a\bar{b}}$]. The random variable $N_{u\bar{v}}$ follows the hypergeometric law, which means Pr[$N_{u\bar{v}} = k$] = Pr[of |*U*| examples selected at random, exactly *k* are not in *V*]. Let $n_u = |U|$, $n_v = |V|$, $n_{\bar{v}} = |\overline{V}|$ and $n=|D|$. It equals

$$\frac{C_{n_{\bar{v}}}^{k} \times C_{n_v}^{n_u-k}}{C_n^{n_u}}$$

Taking into account that $n_u = n_a$, $n_v = n_b$ and n=|*D*|, the intensity of implication can be written as:

$$I = 1 - \sum_{k=\max(0,n_a-n_b)}^{n_{a\bar{b}}} \frac{C_{n_{\bar{b}}}^{k} \times C_{n_b}^{n_a-k}}{C_n^{n_a}}$$

This formula for intensity of implication is suitable as long as the number of samples in the database, i.e. |D|, is reasonably small. Otherwise, the combination numbers in the above formula explode very quickly.

Therefore, Suzuki and Kodratoff (1998) developed an approximation of this formula for big datasets. They argue that if $n_{a\bar{b}}$ is small, an approximation can be applied that uses the well-known Poisson formula (see for instance also (Chen, 1975)). In that case, the above formula for intensity of implication reduces to a much simpler version that is easier to compute:

$$I = 1 - \sum_{k=\max(0,n_a-n_b)}^{n_{a\bar{b}}} \frac{C_{n_{\bar{b}}}^{k} \times C_{n_b}^{n_a-k}}{C_n^{n_a}}$$

$$I \approx 1 - \sum_{k=0}^{n_{a\bar{b}}} \frac{\lambda^k}{k!} e^{-\lambda}$$

With

$$\lambda = \frac{n_a \times (n - n_b)}{n}$$

Keeping the confidence of rule *X*⇒*Y* constant, the intensity of implication varies with the size of s(*Y*), with the size of *D*, and by dilation of *n* when n_a/n, n_b/n and n_{ab}/n stay constant, as Figure 3.7 shows.

Because confidence and support are standard measures for determining the quality of association rules, it would be nice if those could be incorporated in the Poisson approximation formula that was proposed by the Suzuki and Kodratoff. This procedure is quite straightforward and is explained below.

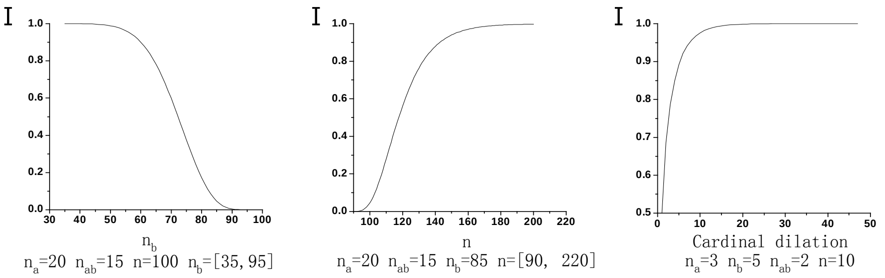


Figure 3.7: Sensitivity analysis of intensity of implication

- Rewriting $n_{a\bar{b}}$ gives

$$
\begin{aligned}
n_{a\bar{b}} &= n_a - n_{ab} \\
&= n_{ab} \times \left(\frac{n_a}{n_{ab}} - 1 \right) \\
&= \frac{n_{ab}}{|D|} \times |D| \times \left(\frac{1}{\frac{n_{ab}}{n_a}} - 1 \right) \\
&= \text{support} \times |\text{cases}| \times \left(\frac{1}{\text{confidence}} - 1 \right)
\end{aligned}
$$

- Rewriting λ gives

$$
\begin{aligned}
\lambda &= \frac{n_a \times (|D| - n_b)}{|D|} \\
&= \frac{\frac{n_{ab}}{|D|}}{\frac{n_{ab}}{n_a}} \times (|D| - n_b) \\
&= \frac{\text{support}}{\text{confidence}} \times (|\text{cases}| \text{-abssupcons})
\end{aligned}
$$

with abssupcons the absolute support count of the consequent of the rule

- Substituting both derivations in the former formula by Suzuki and Kodratoff gives:

$$
I = 1 - \sum_{k=0}^{\text{support} \times |\text{cases}| \times \left(\frac{1}{\text{confidence}} - 1 \right)} \frac{\left(\left(\frac{\text{support}}{\text{confidence}} \right) \times (\text{cases-abssupcons}) \right)^k}{\text{k!}} \times e^{-\left(\left(\frac{\text{support}}{\text{confidence}} \right) \times (|\text{cases}| \text{-abssupcons}) \right)}
$$

By means of this latter formula, we are now ready to adapt the CBA algorithm. This is done by using intensity of implication as the primary criteria when doing the sorting in the first rule of Figure 3.3. Rule r_i has a higher rank than rule r_j if it has a larger value of intensity of implication. When two rules have the same values of intensity of implication, they are ranked according to the confidence sorting mechanism of the original CBA, explained in section 3.3.3. Guillaume *et al.* (1998) claim that the relevance of the discovered association rules can be significantly improved by using intensity of implication. The empirical results of this adaptation are shown in section 3.6. Before doing so, we first briefly elaborate on another own-developed measure for determining the interestingness of a decision rule.

3.5.3 DILATED CHI-SQUARE

Another measure for ranking class association rules that was tested within this dissertation is the dilated chi-square measure. The measure was developed, based on the traditional chi-square test statistic (χ^2), which is a widely used method for testing independence and/or correlation between two variables. Let f_0 be an observed frequency of all the values for a variable, and f be an expected frequency of these values. The χ^2 value can then be defined as

$$\chi^2 = \sum \frac{(f_0 - f)^2}{f}$$

to test the significance of the deviation from the expected values. The statistic can also be applied within the context of classification rules. For each rule $X \Rightarrow Y$ and the training dataset D, a 2*2 contingency table can be derived (Table 3.6):

Table 3.6: A 2*2 contingency table for rule $X \Rightarrow Y$ and dataset D

	Satisfies *Y*	Does not satisfy *Y*	Row Total
Satisfies *X*	m_{11}	m_{12}	Support count of *X*
Does not Satisfy *X*	m_{21}	m_{22}	|*D*|-Support count of *X*
Column Total:	Support count of *Y*	|*D*|-Support count of *Y*	|*D*|

The χ^2 value for rule $X \Rightarrow Y$ can be calculated as

$$\chi^2 = \frac{(m_{11}m_{22} - m_{12}m_{21})^2 |D|}{(m_{11}+m_{12})(m_{21}+m_{22})(m_{11}+m_{21})(m_{12}+m_{22})}$$

However, simply using the traditional χ^2 value will only be favorable in the situation where the distribution of the row total is close to that of the column total distribution (see infra). A dilated chi-square measure was therefore proposed to conquer this shortcoming. In order to do so, the definitions of local maximum χ^2 and global maximum χ^2 were introduced, along with their related properties.

Definition 3.16: Local maximum χ^2
Given a dataset *D*, the local maximum χ^2, denoted as $\mathrm{lmax}\left(\chi^2\right)$, is the maximum χ^2 value for a fixed support count of *X*. ■

Property:

$$\mathrm{lmax}\left(\chi^2\right) = \frac{\left(n_1 n_2\right)^2 |D|}{\left(m_{11}+m_{12}\right)\left(m_{21}+m_{22}\right)\left(m_{11}+m_{21}\right)\left(m_{12}+m_{22}\right)}$$

where

$$\begin{aligned} n_1 &= \min\left(\min\left(m_{11}+m_{12}, m_{21}+m_{22}\right), \min\left(m_{11}+m_{21}, m_{12}+m_{22}\right)\right) \\ n_2 &= \min\left(\max\left(m_{11}+m_{12}, m_{21}+m_{22}\right), \max\left(m_{11}+m_{21}, m_{12}+m_{22}\right)\right) \end{aligned}$$

That is, the local max χ^2 value is arrived at the largest deviation from the expected frequency when the support count of X is given. The property was not theoretically proven but has been heuristically derived and tested on several examples. ■

Definition 3.17: Global maximum χ^2
Given a dataset *D*, the global maximum χ^2, denoted as $\mathrm{gmax}\left(\chi^2\right)$, is the maximum χ^2 value for any possible support count of X. ■

Property:
$\mathrm{gmax}\left(\chi^2\right) = |D|$ ■

Proof: If we for instance suppose that $m_{11}+m_{21} \geq m_{12}+m_{22}$ and $m_{11}+m_{12} \geq m_{21}+m_{22}$, and taking the property above into account:

$$\begin{aligned} n_1^2 &= \left(\min\left(m_{21}+m_{22}, m_{12}+m_{22}\right)\right)^2 \leq \left(m_{21}+m_{22}\right)\left(m_{12}+m_{22}\right) \\ n_2^2 &= \left(\min\left(m_{11}+m_{12}, m_{11}+m_{21}\right)\right)^2 \leq \left(m_{11}+m_{12}\right)\left(m_{11}+m_{21}\right) \end{aligned}$$

Therefore

$$\mathrm{lmax}\left(\chi^2\right) \leq |D| = g\mathrm{max}\left(\chi^2\right)$$

The equation is arrived when $m_{21}+m_{22} = m_{12}+m_{22}$ and $m_{11}+m_{12} = m_{11}+m_{21}$, i.e. the distribution of row total equals that of column total. ■

In order to better illustrate the above properties and definitions, the reader may consider the following hypothetical example.

Example: Suppose three rules have been generated by the CAR algorithm.
r_1: education = university $\Rightarrow$ Transport mode=bike (support count of rule = 32, confidence = 60%), and
r_2: driving license = yes $\Rightarrow$ Transport mode=car (support count of rule = 199, confidence = 99.5%), and
r_3: number of children >4$\Rightarrow$ Transport mode=bike (support count of rule = 2, confidence = 100%).

The contingency tables for these three rules are as follows:

R_1	Car	Bike	Total
Education= university	438	32	470
Education≠ university	12	18	30
Total	450	50	500

2*2 contingency table for rule r_1

R_2	Car	Bike	Total
Driving license=yes	199	1	200
Driving license=no	251	49	300
Total	450	50	500

2*2 contingency table for rule r_2

R_3	Car	Bike	Total
Child≤4	450	48	498
Child > 4	0	2	2
Total	450	50	500

2*2 contingency table for rule r_3

Figure 3.8 Contingency tables for 3 hypothetical rules

The χ^2 values of the three rules are respectively 88.7, 33.4 and 18.1, and the local maximum χ^2 values 287.2, 83.3 and 18.1. It is evident from this example that the χ^2 values are favorable to the situation where the distribution of row total is close to that of column total. For somebody having university education and a driving license, the transport mode will be bike according to r_1, if the choice of the rules is based on simple χ^2 values. However, r$_2$ is intuitively much better than r$_1$, as also indicated by much higher support and confidence values. In addition to this, although the support of r$_3$ is extremely low, r$_3$ has a 100% confidence. The interestingness of r$_3$ may thus have been somewhat underestimated by its current χ^2 value.

Since the χ^2 value seems to have a bias towards total row and column distributions, the measure needed to be adjusted to a more uniform situation. The novel interestingness measure was called dilated χ^2 value, denoted as $dia(\chi^2)$. More specifically, we *heuristically* dilated the χ^2 value according to the relationship between the local and global maximum χ^2 values. The dilation procedure is nonlinear and empirically achieved excellent results on several datasets, as will be demonstrated in next section:

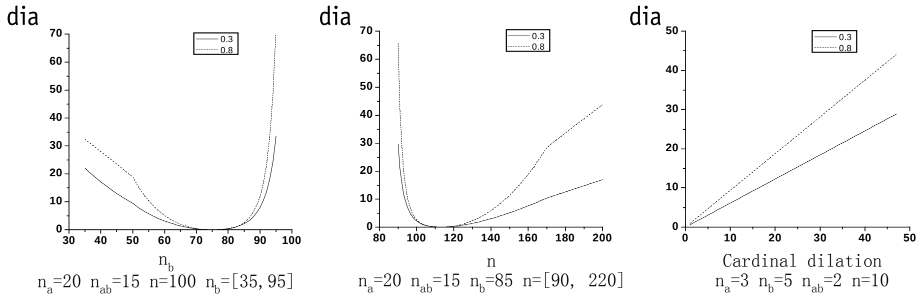
$$\frac{dia(\chi^2)}{\chi^2} = \left(\frac{\text{gmax}(\chi^2)}{\text{lmax}(\chi^2)}\right)^{\alpha} = \left(\frac{|D|}{\text{lmax}(\chi^2)}\right)^{\alpha} \text{,} where \quad 0 \le \alpha \le 1$$

Therefore

$$dia(\chi^2) = \left(\frac{|D|}{\text{lmax}(\chi^2)}\right)^{\alpha} \chi^2$$

The parameter α is used to control the impact of global and local maximum χ^2 values. It determines the degree that chi-square is dilated. The dilated χ^2 values for the three rules are respectively 117.0, 136.1 and 95.1 if $\alpha = 0.5$. These dilated χ^2 values are somewhat more reasonable to our intuition, because in the case that somebody has a university education and a driving license, the transport mode will be car in our example, according to r_2 (which has the largest dilated chi-square value). Also, the interestingness of r_3 seems more reasonable now, when compared to the original chi-squared value, given its 100% confidence.

It can be seen from Figure 3.9 that the dilated χ^2 value is sensitive when the size of *s(Y)* or *D* varies. In this figure, the sensitivity analysis that was reported previously for intensity of implication was also established for the dilated chi-square measure, with α arbitrarily set at 0.3 and 0.8, for the sake of clarity. In the first case, i.e. n_b increases while n_a, n_{ab} and n remain stable, the dilated χ^2 first gradually declines to zero; this is at the point when n$_b$ equals 75. This is the situation when the confidence of the rule is equal to the proportion of class *Y* in the whole dataset *D*, i.e. 0.75. Dilated χ^2 then climbs up sharply if n_b continues to increase, which indicates the negative relationship between *X* and *Y*. Therefore, rules whose confidences are less than their corresponding class proportions are not expected to exist in the final classifier. The similar mechanism occurs in the second case, where dilated χ^2 becomes close to zero when the size of *D* equals 113. The third case shows that dilated χ^2 increases linearly if all subsets are cardinally dilated. In addition to these properties, dilate χ^2 values can estimate interestingness in a more careful manner, such as for instance rules with high confidence and very low support (see example before). Changing the α-parameter does not significantly change the predictive performance, as it will be shown in Appendix E.



n_a=20 n_{ab}=15 n=100 n_b=[35, 95] n_a=20 n_{ab}=15 n_b=85 n=[90, 220] n_a=3 n_b=5 n_{ab}=2 n=10

Figure 3.9. Sensitivity of dilated chi-square

Similarly to intensity of implication, we can now equally adapt the CBA algorithm by means of the developed dilated chi-square formula. This is done by using dilated chi-square as the primary criteria when doing the sorting in the first rule of Figure 3.3. Rule r_i has a higher rank than rule r_j if it has a larger dilated chi-square value. When two rules have the same dilated chi-square values, they are ranked according to the sorting mechanism of the original CBA, explained in section 3.3.3.

3.6 ADAPTED CBA: RESULTS

Before moving on to the results of adapted CBA within the Albatross model, the reader may first briefly consider the results of the adapted CBA algorithms on the UCI Machine Learning repository in Appendix C. It can be seen in Appendix C that adapted CBA-1 and CBA-2, which correspond to the new algorithms that incorporate intensity of implication and dilated χ^2 respectively, perform better than any of the other classifiers in terms of average accuracy. Perhaps even more important, is the number of rules that were generated. The average size of the ruleset was almost one third of the rules that have been generated by original CBA. CBA-2 seemed to be even more compact than CBA-1. The reader may recall from section 3.4 that one of the limitations of integrated (supervised and unsupervised) approaches is the number of rules that is generated. Therefore, our final aim for carrying out these adaptations -to reduce the size of the decision choice sets and the amount of overfitting (see section 3.4.5)- has been achieved; at least for these UCI data. The next section examines whether these good results can also be maintained on the 9 choice facets of the Albatross model.

3.6.1 Choice Facet Level

The accuracy percentages that indicate the predictive performance of Adapted CBA-1,2 within Albatross, are presented in Table 3.7. The results of the original CBA algorithm (see Table 3.3) were also added to the table, for the sake of clarity. With respect to Adapted CBA-1, only the results of the Poisson approximation formula were shown in Table 3.7. As mentioned before, the computational effort was much larger for the hypergeometrical law formula. The detailed results of the hypergeometrical law formula are shown in Appendix D. With respect to Adapted CBA-2, only the best parameter selection (α) has been reported in Table 3.7. The sensitivity analysis for using a different value for α has been shown in Appendix E. It can be seen from Appendix E that the improvement which can be achieved through a proper selection of α is only minor. While there are some small variations, these findings seem to support the stability of the proposed heuristic.

It can be seen from Table 3.7 that Adapted CBA performed slightly worse than original CBA. However, the degree of overfitting and the size of the decision sets are significantly lower. Also, the fact that model performance is not significantly worse for all datasets, has lead us to believe that the rules which are selected in the adapted CBA are more interesting (i.e. contribute more to the classifier) than the rules which are present in the original CBA. In order to examine this in detail, the added value of every rule in the classifier was examined.

Table 3.7: Benchmarking results at choice facet level

Dataset	Adapted CBA-1			Adapted CBA-2			Original CBA		
	Train (%)	Test (%)	Num. of rules	Train (%)	Test (%)	Num. of rules	Train (%)	Test (%)	Num. of rules
Duration	40.7	40.9	17	43.0	41.2	8	44.7	39.2	147
Location1	64.5	68.1	25	59.6	60.8	3	66.3	62.7	234
Location2	26.8	26.3	1	48.3	38.0	55	52.6	41.1	136
Mode for work	74.7	76.8	38	75.6	74.4	12	83.5	73.7	172
Mode other	54.9	54.8	5	68.0	60.5	259	66.5	60.9	245
Selection	79.1	79.2	1	57.5	57.4	1	79.6	78.7	594
Start time	33.3	33.0	69	37.6	36.2	102	34.5	33.7	120
Trip chain	82.7	82.0	21	84.2	83.4	3	83.9	80.4	65
With whom	54.7	48.1	24	55.9	51.1	51	61.1	56.2	222
Average Accuracy	**56.8**	**56.6**	/	**58.9**	**55.9**	/	**63.6**	**58.5**	/
Av. number of rules	/	/	**22.3**	/	/	**54.9**	/	/	**215**

POST-ANALYSIS

As mentioned before, by depicting the number of correctly predicted cases versus the rules that are added to the final classifier, it is possible to get quite a good idea about the quality of every rule that is added to the classifier. The slope of the graph then becomes a quantitative measure for the added value. Figure 3.10 portrays this evolution for all nine facets of the Albatross model. It is shown in the figure that the slope of the graph stops some time after the cutpoint *p*, which means that the rule insertion is stopped because either all of the rules are used or no cases are left in the database. It can be seen that for every decision facet, Adapted CBA-1,2 was able to *faster* obtain a higher number of correctly predicted cases, indicated by the steeper slope. In some cases, the increase is quite spectacular, e.g. for the datasets “Duration”, “Location1”, “Mode for Work” and “Trip Chain”. We therefore conclude that the quality of the decision rules which are added to the classifier was higher for these datasets. The fact that the cutpoint could be achieved earlier is a decent property of Adapted CBA-1,2.

By consequence, the increase in predictive performance is significant for these 4 datasets and the degree of overfitting is lower (see Table 3.7).

However, there are also a couple of datasets ("Mode Other", "Start Time" and "With whom") for which the slope coincides better with original CBA, and by consequence, it is flatter. Despite this, the difference with original CBA still remains that Adapted CBA achieved its cutpoint faster. However, given the lesser quality of the rules that were added, the predictive performance suffers from this (see Table 3.7). An exception to this rule is the performance of the Adapted CBA-2 algorithm on the Start Time dataset.

Finally, there are also a couple of datasets for which Adapted CBA only selected one single rule. This occurred at the Selection dataset (for Adapted CBA-1,2) and at the Location2 data (for Adapted CBA-2). Apart from Adapted CBA-1 at the selection dataset, this was a bad decision of the algorithm, which negatively affected the predictive performance (see Table 3.7).

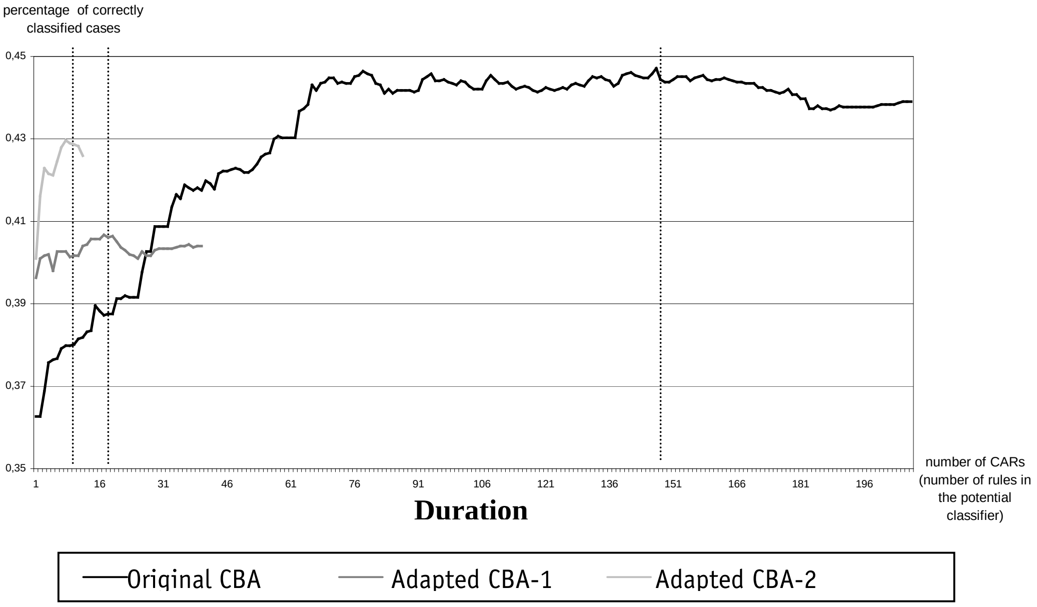


Figure 3.10 (first part): Evolution of the number of correctly classified cases for every rule that is added to the final classifier for CBA, Adapted CBA-1,2

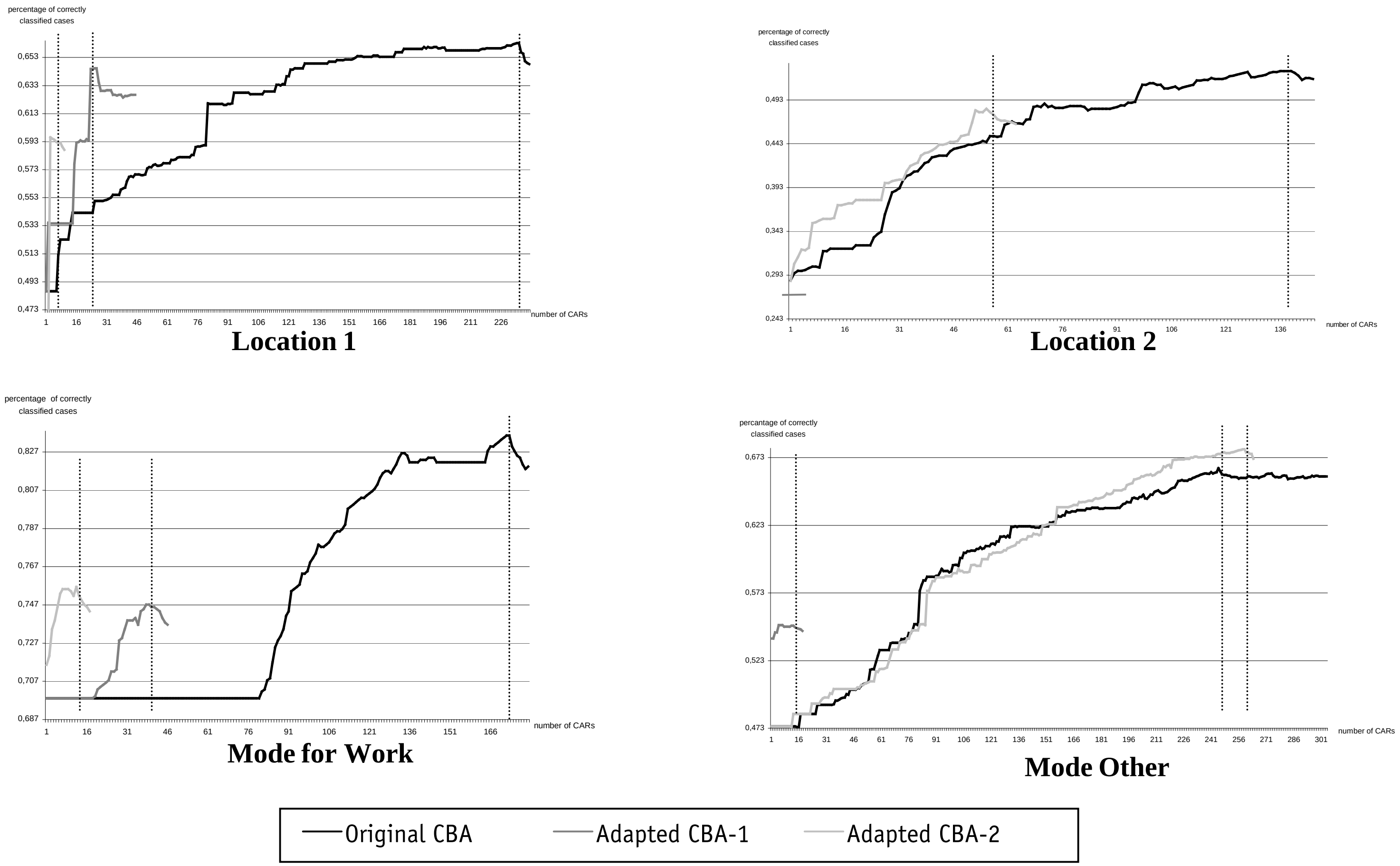


Figure 3.10 (second part): Evolution of the number of correctly classified cases for every rule that is added to the final classifier for CBA, Adapted CBA-1,2

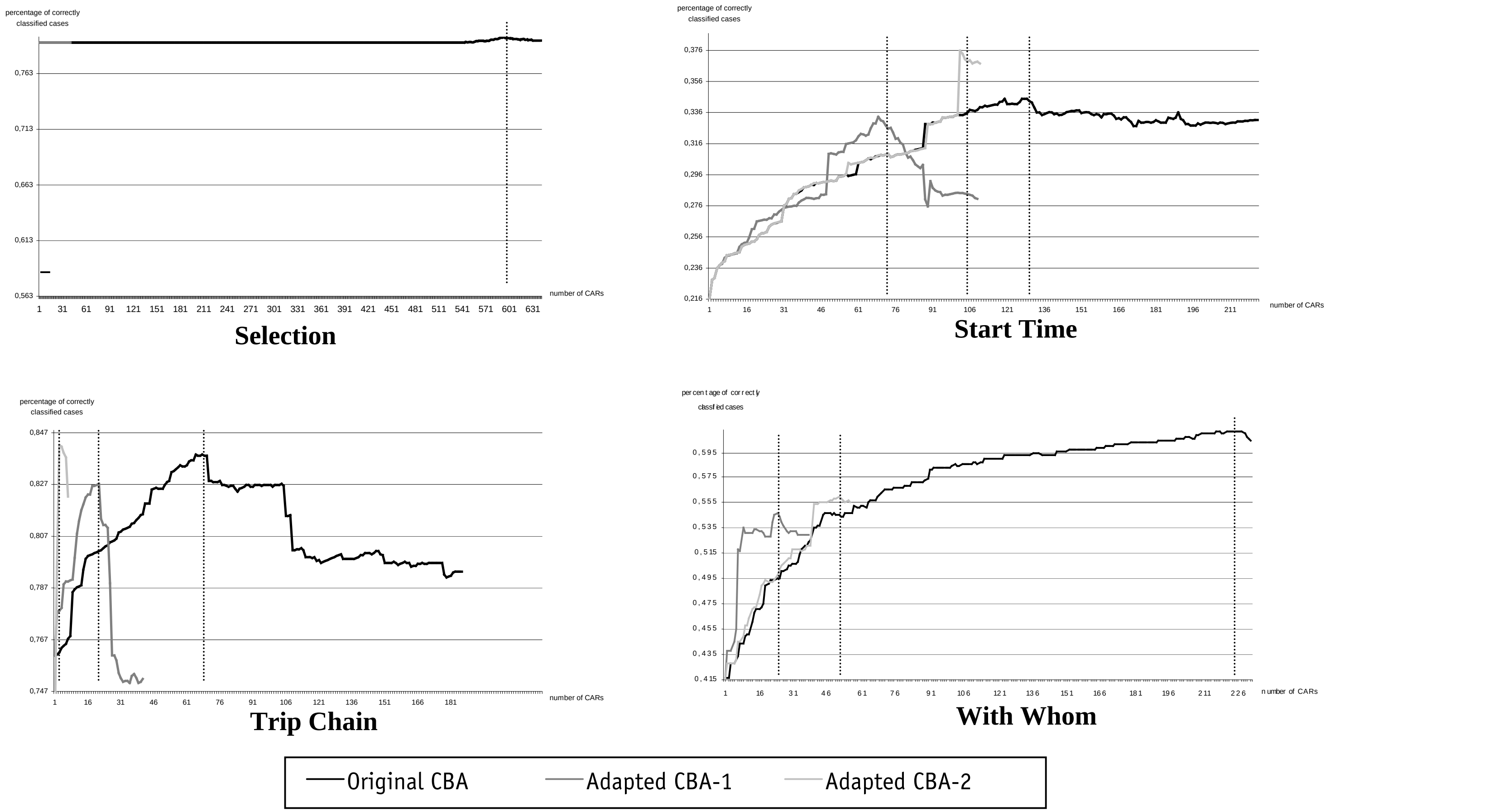


Figure 3.10 (third part): Evolution of the number of correctly classified cases for every rule that is added to the final classifier for CBA, Adapted CBA-1,2

However, not only the predictive performance, but also the computation time of CBA; and especially that of CBA-1,2; is affected by the (number of) rules that are included in the classifier. This can be clearly seen in Figure 3.11, where the computation of the three algorithms has been shown for the different datasets. The three numbers in parentheses that have been listed on the X-axis under the name of the dataset, are equal to the number of rules that are respectively incorporated in the CBA, CBA-1 and CBA-2 classifier. These numbers were already mentioned in Table 3.7 but were repeated in the figure for the sake of clarity. It can be seen that there is clearly a relationship between the size of the classifier and the number of rules. To give one example with respect to the original CBA algorithm: the duration dataset, which has a computation time of 7.09 seconds, contained 147 rules; while the selection dataset has a computation time of 8.350 seconds and contained 594 rules. Similar patterns can be seen for other algorithms (and datasets). It should also be noted that CBA-2 is less efficient when compared to CBA, which is for instance very clear for the "Mode other" dataset, which significantly slowed down the CBA-2 learning algorithm (21.210 seconds), but has only a minor effect on the original CBA algorithm (7.760 seconds), even while nearly the same amount of rules were included in both

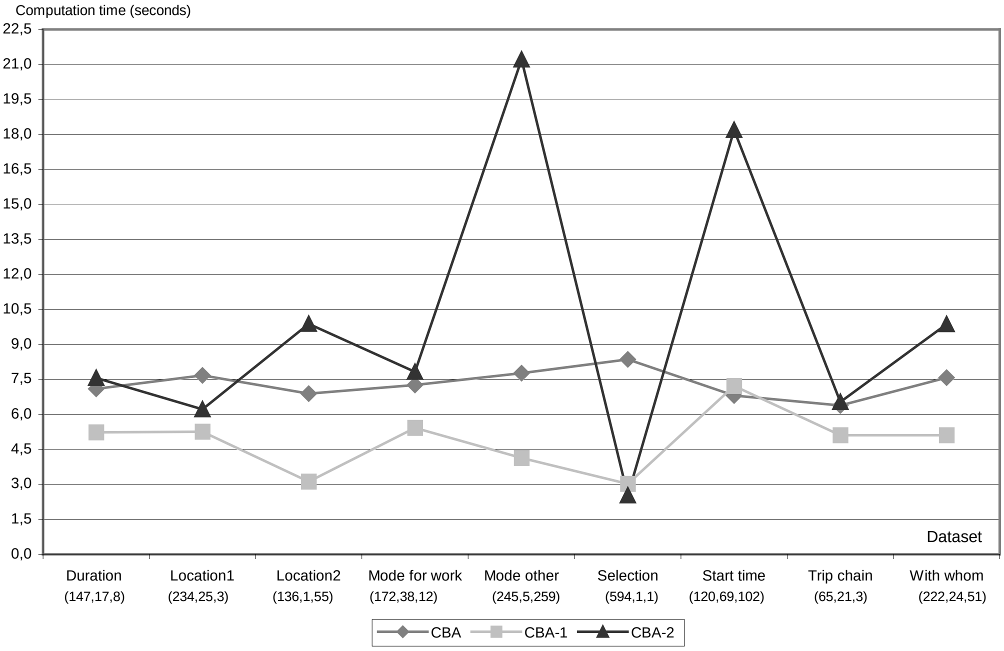


Figure 3.11: Computation time (in seconds) of CBA, CBA-1, and CBA-2

algorithms (259 versus 245 rules). Obviously, this negative side effect is probably the result of the more complex sorting algorithms that have been used in CBA-1,2 and therefore also have required more computation time. However, overall CBA-1 is most efficient for most datasets, which is probably due to the lower number of rules that are incorporated in CBA-1. The algorithms were executed on a Pentium 1.6 Ghz computer with 512 RAM.

3.6.2 ACTIVITY PATTERN LEVEL

The SAM distance measures, comparing the predictive performance at activity pattern level on the training and on the test set for Adapted CBA, are presented in Table 3.8. The results of the original CBA algorithm (see Table 3.4) were again added, for the sake of clarity. The results are in the same line as the results at choice facet level, i.e. original CBA performs slightly better. However, for specific SAM distance measures such as SAM mode, Adapted CBA-1,2 may be preferred.

This result is correlated with the good predictive performance for the mode choice dataset. Another important finding is that the degree of overfitting of Adapted CBA was significantly lower when compared to original CBA. Also this finding is in line with previous results at choice facet level.

Table 3.8: Benchmarking results at activity pattern level

SAM distance measure	Adapted CBA-1		Adapted CBA-2		Original CBA	
	Train	Test	Train	Test	Train	Test
SAM activity-type	2.710	2.712	2.879	2.918	1.610	2.712
SAM with whom	3.210	3.215	3.109	3.210	1.971	3.114
SAM location	3.138	3.142	3.106	3.214	1.321	3.035
SAM mode	4.208	4.218	4.185	4.215	2.019	4.414
UDSAM	16.385	16.412	16.217	16.419	12.871	16.318
MDSAM	8.310	8.356	8.101	8.361	5.108	8.298

3.6.3 TRIP MATRIX LEVEL

Once more, the last measure to evaluate the predictive performance, is carried out at trip matrix level (see Table 3.9). Also in this case, original CBA was added for the sake of clarity. While the results that could be achieved are in favour of Original CBA, the differences are minor. Also at this level, the differences between training and test set are somewhat lower. All results are consistent with previous results at the other levels.

Table 3.9: Benchmarking results at trip matrix level

Dimension	Adapted CBA-1		Adapted CBA-2		Original CBA	
	Train	Test	Train	Test	Train	Test
None	0.956	0.939	0.955	0.940	0.959	0.940
Mode	0.892	0.851	0.887	0.848	0.911	0.849
Day	0.967	0.946	0.964	0.945	0.975	0.948
Primary Activity	0.892	0.836	0.890	0.837	0.902	0.838

3.7 CHAID, CBA, CBA-1 AND CBA-2: QUALITATIVE ANALYSIS

The previous sections have described detailed quantitative analyses about the performances of CBA, adapted CBA-1 and the adapted CBA-2 algorithm. Comparisons were made with the CHAID decision tree algorithm that is used in Albatross. The aim of this section is to proceed with this comparison at choice facet level but now on a more explanatory and descriptive manner. To this end, the most important variables that have been used in the above-mentioned techniques, are discussed. A description of the variables is often mentioned along with the variable, but is provided as well in more detailed lists in Appendix B. The chronology of this section is similar as described in the previous quantitative analyses. A number of rules that were ranked highest by respectively CBA, CBA-1 and CBA-2 were added (for illustrative purpose) per decision facet.

Before doing so, it has to be noted that the identification of most important variables is somewhat different in CBA than in a traditional decision tree approach. With respect to CBA, a weighted variable frequency count was calculated in order to assess the importance of the variables per decision rule. It was mentioned before that the sequence order of the CARs that are entered into the final classifier is important. Therefore, variables in the first rule of the final classifier are considered to be more important than variables in the last rule. A simple weighting factor has been used to assess this difference. Variables with highest weighted frequency counts were thus selected as most important variables. In CHAID, the variables on which the algorithm makes its first splits are considered to be more relevant than variables which occur further down the partitioning process.

3.7.1 DURATION

With respect to the duration facet, the total time of Work1 including travel (Twincl), the total time available for social activities (Tsoc), and the travel party (Awith) are identified as important variables in the original CBA algorithm. Equally important are the maximum available time (Tmax_t), and the availability of the 'long' duration class (Yavail3).

CBA-1 also selected Tmax_t, and Awith but added the Number of instances of the current activity type in the schedule (Iact) and the availability of a grocery activity (Ygroc) to its list of most important variables. In the CBA-2 algorithm, Tmax_t, and Awith were also selected, but Day and Two are important as well in this case.

When compared to the CHAID based approach, there is only one variable which is similar with the variables that have been found important by CBA and CBA-1. This is the Travel party (Awith) variable. Since Awith was also found important by Moons (2005) for *all* the algorithms under consideration in the study, the variable can be regarded as being very reliable and robust for this decision agent. When compared with CBA-2, CHAID has three equally most important variables, i.e. day, Awith and Two. Despite this, the difference in most important selected variables between CBA and CHAID is quite high. Despite the fact that there is a small improvement in accuracy on the test set for CBA (+0.4%), CBA-1 (+2.1%) and CBA-2 (+2.4%) when compared to CHAID (see Tables 3.3 and 3.7), it is probably not really safe to proclaim that CBA-1 and CBA-2 selected better variables and combined them more efficiently than CHAID did.

In Table 3.10, the four most important decision rules have been shown for the algorithms under evaluation in this chapter. All rules are mentioned along with their corresponding support and confidence values. It is logical that the CBA algorithm contains only rules with a very high confidence value because of its sorting criterion. CBA-1,2 may or may not contain rules with high confidence and support values, because of the other criterion of interestingness that has been used. In most cases, the confidence value will be somewhat lower and the support higher in CBA-1,2, but this is not always the case.

Table 3.10: Some important rules in CBA, CBA-1 and CBA-2 (according to sorting criteria) for the duration choice facet

Rule	CBA	CBA-1	CBA-2
1	Twincl=3 ^ Tsoc=0 ^ Awith=0 → Short (Conf.: 0.83; Supp.: 0.015)	Ysoc=0 ^ Awith=0 → Short (Conf.: 0.45; Supp.: 0.17)	Awith=0 → Short (Conf.: 0.45; Supp.: 0.19)
2	Twincl=3 ^ yServ=0 ^ Awith=0 → Short (Conf.: 0.79; Supp.: 0.014)	Awith=0 → Short (Conf.: 0.45; Supp:0.19)	Day=6 ^ Tmax6=3 ^ Awith=1 → Long (Conf.: 0.50, Supp.: 0.05)
3	Tmax3=0 ^ yCar6=1 ^ yAvail3=0 → Short (Conf.: 0.78; Supp.: 0.010)	Tmax2=3 ^ Iact=0 ^ yGroc=0 →Average (Conf.: 0.40;Supp: 0.16)	Day=6 ^ Two=0 ^ Awith=1 → Long (Conf.: 0.48, Supp.: 0.05)
4	Tmax3=0 ^ yCar3=1 ^ yAvail3=0 → Short (Conf.: 0.77; Supp.: 0.011)	Tmax2=3 ^ Tmax4=3 ^ yGroc=0→Average (Conf.: 0.39;Supp: 0.17)	Two=3→Short (Conf.: 0.58, Supp.: 0.03)

3.7.2 LOCATION1

Almost all attributes that the CBA algorithm has found important for the prediction of the location1 decision facet are descriptive variables. The most important variables are day of the week (day), Household type (Ccomp), Number of mandatory out-of-home activities other than work in the schedule (Nsec), and transport mode (Mode). Other variables that are important are the maximum available time in the schedule position of the activity (Tmax) and a variable which indicates whether a trip ends at home (toH) or not.

The CBA-1 algorithm has different variables in its most important rules. On the one hand, the location facet is determined by the previous facets of the Albatross model, which is the activity type (Atype), the transport mode (Mode), and the travel party (Awith). Furthermore, the availability of choice locations given the schedule (yAvail), the number of out-of-home activities (Nout) and the maximum available time in the schedule position of the activity (Tmax) are important variables for CBA-1. CBA-2 did not select any new variables in its most important rules. In this case, only the yAvail and the Mode variables were selected.

When compared to the CHAID based approach, there are only two equal variables. Obviously, the most important variable is the Mode choice variable, also because it appeared in *all* the algorithms in the study by Moons (2005). Also in this case, the variable can therefore be regarded as being reliable and robust for this decision agent.

Having said this and given the fact that the predictive quantitative results of CBA-1 outperformed CHAID by more than 9% (see Table 3.7), it is safe to assume that the algorithm not only selected the best variables but also combined them more efficiently. The better predictive quantitative results are probably the result of the sorting mechanism (interestingness selection criterion) and the fact that the rules stem from a higher "richness" of the rules (i.e. derived from association rules).

The rules that are first added to the classifiers CBA, CBA-1 and CBA-2 are shown in Table 3.11. The reader should especially notice that CBA-2 only contained three rules and achieved an excellent performance of 60.8% on the test set.

Table 3.11: Some important rules in CBA, CBA-1 and CBA-2 (according to sorting criteria) for the location1 choice facet

Rule	CBA	CBA-1	CBA-2
1	Day=7 ^ Ccomp=4 ^ Nsec=0 ^ toH=1 → highest-order loc. (Conf.: 0.94; Supp.: 0.023)	Atype=4 ^ Mode=1 ^ Nout=1 → highest-order loc.(Conf.: 0.92; Supp.: 0.083)	yAvail1=1 ^ yAvail2=0 →nearest loc. from home in the context of the tour (Conf.: 0.30; Supp.: 0.040)
2	Day=7 ^ Ccomp=4 ^ Mode=1 ^ toH=1 → highest-order loc. (Conf.: 0.94; Supp.: 0.023)	Mode=2 ^ Tmax=1 ^ yavail1=0^yAvail5=0→ nearest loc. from home (Conf.: 0.90; Supp: 0.05)	yAvail1=1 → nearest loc. from home in context of tour (Conf.: 0.27; Supp: 0.04)
3	Day=7 ^ Ccomp=4 ^ Tmax=3 ^ Nsec=0 → highest-order loc. (Conf.: 0.93; Supp.: 0.027)	Mode=2 ^ Tmax=1 ^ yavail4=0 ^yAvail5=0→ nearest loc. from home (Conf.:0.89; Supp.: 0.05)	Mode=2 ^ yAvail4=0 ^yAvail5=0 → nearest loc. from home (Conf.:0.75; Supp.: 0.19)
4	Twork=4 ^ Mode=1 ^ Nsec=0 ^ yAvail2=0 → highest-order loc. (Conf.:0.92; Supp: 0.023)	Atype=4 ^ Mode=1 ^ Awith=2 → highest-order loc.(Conf.: 0.87; Supp.: 0.082)	/

3.7.3 Location2

For the location2 facet, there are some variables that were already found important by CBA at the location1 dataset (Mode, Ccomp, tOH, yAvail, and Nout). Also the age of the oldest person in the household (Cage), the gender (Gend), the total time of Work1 (Twoincl) and the type of the previous activity (Aprev) belong to the list of most leading attributes for CBA. This list of variables is highly similar when compared to CBA-2. On the contrary, CBA-1 selected only one rule and one attribute (Avail1). This decision is clearly a very bad choice, given the poor predictive performance of 26.3% that it achieved on the test set.

When compared with CHAID, there are only two equally most important variables (Mode, Nout). Also in this case, the Mode choice variable was found highly significant and was supported by *all* the algorithms under evaluation for other types of classifiers (Moons, 2005) at the location2 facet.

Table 3.12: Some important rules in CBA, CBA-1 and CBA-2 (according to sorting criteria) for the location2 choice facet

Rule	CBA	CBA-1	CBA-2
1	Mode=1 ^ Nout=2 ^ yAvail3=0 → nearest loc. from home (Conf.: 1; Supp.: 0.019)	YAvail1=1→ nearest loc. from home (tour context) (Conf.: 0.27; Supp.: 0.27)	YAvail3=0 ^ YAvail4=0 ^ Yavail5=1→highest-order location within 20 minutes (Conf.: 0.90, Supp.: 0.019)
2	Ccomp=4 ^ Nout=2 ^ Aprev=1 ^ yAvail5=0 → nearest loc. from home (Conf.: 1; Supp.: 0.013)	/	Mode=1 ^ Nout=2 ^ yAvail3=0 → nearest loc. from home (Conf.: 1, Supp.: 0.019)
3	Cage=2 ^ Mode=1 ^ Nout=2 ^ Aprev=1 → nearest loc. from home (Conf.: 0.93; Supp.: 0.014)	/	Ccomp=4 ^ Nout=2 ^ Aprev=1 ^ yAvail5=0 → nearest loc. from home (Conf.: 1, Supp.: 0.013)
4	Cage=2^Gend=2^TOH=1^Twoincl=1 YAvail2=1→ nearest loc. from home (tour context) (Conf.: 0.93; Supp.: 0.013)	/	Gend=2^Tnltot=0^yAvail3=0^yAvail5=1→ highest-order location within 20 minutes (Conf.: 0.78, Supp.: 0.20)

As illustrated before and despite the bad performance of CBA-1, both CBA and CBA-2 achieved better predictive performances than CHAID (respectively +8.5% and +5.4%). Given the small overlap in variables between CHAID and CBA, CBA-2, also in this case it can be concluded that variables were better selected and were combined more efficiently per decision rule. It can be seen from Table 3.12 that there is some overlap in the most important rules for CBA and CBA-2. An example of this finding are the two rules where a confidence-value of 100% was achieved.

3.7.4 Mode for Work

In case of the "Mode for work" choice facet, the ratios between the car travel time and bike travel time (Rcabi) and between public transport and bike travel time (Rpubi) appear in all CBA algorithms. In the general dataset, the average ratio between car and bike is equal to 3.04, while the average ratio between public transport and bike is equal to 2.38. Both indicators (but particularly Rcabi) seem to be more inclined towards the use of bike as transport mode for work, which is probably also supported by the good biking facilities which are available in the Netherlands. In addition to this, the ratio between public transport travel time and car travel time (Rpuca) is also identified by CBA as an important variable and has an average of 2.37. Also important for CBA are general descriptive variables (such as the socio-economic class of the household (Csec), the household type (Ccomp) and the ratio between the number of cars and the number of adults (Ncar)) and transport related variables such as whether the end time of the work episode is situated in the evening peak (Peakn) and the objective travel time to the work location by bike (Tbike). This latter variable has an average value of 27.6 minutes in the dataset.

CBA-1 also contained some partner related variables such as the presence of a bring/get activity (PyBget) and the number of fixed out-of-home activities other than work in the schedule of the partner (Pnsec). In addition to the above-mentioned variables; some other distinctive partner variables were found important in CBA-2 such as the partner status (group) and the total time of work in the schedule of the partner (Pttot). Also important are the total time of work in the own schedule (Two and Ttot).

Unlike in previous choice facets, it cannot be said that there is a low level of correspondence between CBA's and CHAID's most important variables. In CHAID, Rpuca, Ccomp, Ncar and Tbike are important; i.e. all variables which also appear in CBA algorithms. However, when considering Table 3.3 and 3.7, the predictive performances are again significantly in favour of CBA and CBA-1,2 (respectively +7%, +10.1% and +7,7%). For instance, the highest difference (CBA-1) can be better understood when considering Table 3.13. It can be seen in this table that CBA-1 rules have very high confidence values, while its support is also very high when compared to other rules. Therefore, in this case, the significance in predictive performance has to be attributed to a better combination of attributes per decision rule. Also the number of attributes per decision rule was four at maximum, which reduces the chance of overfitting and consequently improves accuracy (on the test set).

Table 3.13: Some important rules in CBA, CBA-1 and CBA-2 (according to sorting criteria) for the mode for work choice facet

Rule	CBA	CBA-1	CBA-2
1	Csec=4 ^ Peakn=0 ^ Rpubi=2→car (Conf.: 1; Supp.: 0.06)	Pnsec=0 ^ Rpubi=2 →car(Conf.: 0.93, Supp.: 0.28)	Ncar=1 ^ Two=4 ^ Rcabi=4→slow mode (Conf.: 0.89, Supp.:0.02)
2	Ccomp=2^Tbike=4^Rcabi=1→car (Conf.: 1; Supp.: 0.05)	PyBget=0 ^ Rcabi=1 ^ Rpubi=2→car (Conf.: 0.93, Supp.: 0.29)	Ncar=1 ^ Rcabi=4 ^ Rpubi=1 →slow mode (Conf.: 0.80, Supp.:0.03)
3	Ncar=2^Peakn=1^Tbike=4→car (Conf.: 1; Supp.: 0.05)	Rcabi=1 ^ Rpubi=2 ^ Pybget=0→car (Conf.: 0.92, Supp.:0.30)	Group=3 ^ Ncar=1 ^ Tbike=1→slow mode (Conf.: 0.78, Supp: 0.03)
4	Ccomp=2^ Rcabi=1^ Rpuca=3→car (Conf.: 1; Supp.: 0.047)	Ncar=2 ^ PyBget=0 ^ Rcabi=1→car (Conf.: 0.92, Supp.:0.26)	Ncar=1 ^ Ttot=4 ^ Tbike=1→slow mode (Conf.: 0.76, Supp: 0.04)

3.7.5 MODE OTHER

The "mode other" decision facet is the first decision facet where the most important variables for CBA and CBA-2 are similar. It is therefore not surprising that also the predictive accuracy of CBA and CBA-2 is quite equal. The most important variables can be summarized as general descriptive variables such as Cage, Cchild, Gend and Ncar on the one hand and travel time related variables with respect to bike (Rcabi, Textra, Ttbike) on the other hand.

CBA-1 selected quite some different attributes as its most important variables. The question whether a Grocery (Cgroc) or a social activity (Csoc) is part of the tour, the Partner's maximum bike travel time across activities(Ptmax) or the travel time ratio between public transport and bike (Rpubi), are important variables which are identified by CBA-1 but not by CBA and CBA-2. Also when compared to CHAID, the variables are quite different.

Having said this and given the fact that the predictive quantitative results of CBA and CBA-2 outperformed CHAID by respectively +11.4% and +11% on the test set, it is safe to assume that both algorithms selected the better variables for this decision facet. The fact that CBA-1 achieved a better performance of + 5.3% when compared to CHAID further confirmed this finding.

Table 3.14: Some important rules in CBA, CBA-1 and CBA-2 (according to sorting criteria) for the mode other choice facet

Rule	CBA	CBA-1	CBA-2
1	Ttbike=2 ^ Rcabi=3 ^ Textra3=0 → slow mode (Conf.: 1; Supp.: 0.01)	Twork1=1 ^ Awith1=0 ^ Ttbike=0 → slow mode (Conf.: 0.69; Supp.: 0.12)	Ttbike=2 ^ Rcabi=3 ^ Textra3=0 → slow mode (Conf.: 1; Supp.: 0.01)
2	Cage=3 ^ Aty2=0 ^ Ttbike=6 → car (Conf.: 0.97; Supp.: 0.01)	Twork1=1 ^ Awith1=0 ^ Rpubi=1 → slow mode (Conf.: 0.68; Supp.: 0.13)	Cage=3 ^ Aty2=0 ^ Ttbike=6 → car (Conf.: 0.97; Supp.: 0.01)
3	Ncar=2 ^ Aty2=0 ^ Ttbike=6 → car (Conf.: 0.97; Supp.: 0.01)	Twork1=1 ^ Awith1=0 ^ Cnlout=0 → slow mode (Conf.: 0.63; Supp.: 0.16)	Ncar=2 ^ Aty2=0 ^ Ttbike=6 → car (Conf.: 0.97; Supp.: 0.01)
4	Cchild=1 ^ Awith1=0 ^ Ttbike=6→car(Conf.: 0.97; Supp.: 0.01)	Cgroc=0 ^ Ptmax=0 → car (Conf.: 0.50; Supp.: 0.33)	Cchild=1 ^ Awith1=0 ^ Ttbike=6 →car (Conf.: 0.97; Supp.: 0.01)

3.7.6 SELECTION

It is difficult to determine learning patterns for the selection choice facet because of the skewness of the dataset. 79% of the cases can be explained by simply using a default class.

That is also the reason why Adapted CBA-1 and Adapted CBA-2 only selected one single rule in addition to its default class. CBA-1,2 were thus able to determine that adding other potential rules would further compromise the result on the test data. While the decision of stopping after the first rule was probably the right one, the results of CBA-2 are very unfortunate because of the rather low confidence value of that first rule (see Table 3.15). Obviously, CBA-1 did not suffer from this disadvantage because its first rule achieved a confidence of 100%. CBA and especially CHAID fell into the trap of selecting more rules and achieved worse results than simply using the default class.

Table 3.15: Some important rules in CBA, CBA-1 and CBA-2 (according to sorting criteria) for the selection choice facet

Rule	CBA	CBA-1	CBA-2
1	YAvail=0 → no (Conf.: 1; Supp.: 0.1)	YAvail=0→no (Conf.: 1; Supp.: 0.1)	Tmax2=3 ^ yAvail=1→yes (Conf:0.28; Supp:0.14)
2	Atype=3 ^ Two=4 →no (Conf.: 0.96; Supp.: 0.04)	/	/
3	Atype=2 ^ Tmax5=0 →no (Conf.: 0.96; Supp.: 0.02)	/	/
4	Tmax2=0 ^ A1dur=3 →no (Conf.: 0.95; Supp.: 0.02)	/	/

3.7.7 Start Time

For the start time decision facet, a quite surprising finding is that all of the most important variables for CBA, CBA-1 and CBA-2 are equal. This is also reflected in Table 3.16, where the four most important rules are completely identical for every CBA algorithm. Variables which CBA and CBA-1,2 determined to be important for decision making at the start time level are variables related to the amount of saved bike travel if an activity is linked with an out-of-home activity (DBT, DET), the end time of an out-of-home activity (Etx(*t*)), the maximum available time in *t*-th time interval (Tmax(*t*)), travel party (with), the total time of work (Two) and

Table 3.16: Some important rules in CBA, CBA-1 and CBA-2 (according to sorting criteria) for the start time choice facet

Rule	CBA	CBA-1	CBA-2
1	Tmax5=0 ^ Nsec=0 ^ DBT1=3→ After 6 P.M. (Conf.: 1; Supp.: 0.03)	Tmax5=0 ^ Nsec=0 ^ DBT1=3→ After 6 P.M. (Conf.: 1; Supp.: 0.03)	Tmax5=0 ^ Nsec=0 ^ DBT1=3 → After 6 P.M.(Conf.: 1; Supp.: 0.03)
2	Tmax5=0 ^ With=2 ^ Two=4→ After 6 P.M. (Conf.: 1; Supp.: 0.02)	Tmax5=0 ^ With=2 ^ Two=4→ After 6 P.M. (Conf.: 1; Supp.: 0.02)	Tmax5=0 ^ With=2 ^ Two=4 → After 6 P.M. (Conf.: 1; Supp.: 0.02)
3	Tmax5=0 ^ Nsec=0 ^ DET5=3→ After 6 P.M. (Conf.: 1; Supp.: 0.02)	Tmax5=0 ^ Nsec=0 ^ DET5=3→ After 6 P.M. (Conf.: 1; Supp.: 0.02)	Tmax5=0 ^ Nsec=0 ^ DET5=3 → After 6 P.M. (Conf.: 1; Supp.: 0.02)
4	Tmax5=0 ^ Two=4 ^ ETx6=0→ After 6 P.M. (Conf:0.97; Supp: 0.03)	Tmax5=0 ^ Two=4 ^ ETx6=0→ After 6 P.M. (Conf:0.97; Supp: 0.03)	Tmax5=0 ^ Two=4 ^ ETx6=0 → After 6 P.M. (Conf.: 0.97; Supp.: 0.03)

the number of mandatory out-of-home activities other than work (Nsec). The CHAID based approach used other important variables such as Iact and Atype. CHAID performed slightly better than CBA (+1.7%) and CBA-1 (+0.7%), while CBA-2 achieved slightly better results than CHAID (+0.8%). However, this level of improvement is insufficient to proclaim that one algorithm consistently selected better variables than the other algorithms under consideration.

3.7.8 Trip Chaining

For the trip chaining decision facet, a dummy variable which specifies whether there is an activity with an end time within a 1-hour interval before the first work episode (yCanvo) or after the last work episode (yCantu) is found important in all CBA algorithms (see Table 3.17). Equally important in CBA and CBA-1 is the household type (Ccomp). Also important for CBA is the duration (Xndu) and type of the activity (Xntype), and the time needed to perform a shopping (Tshop) or leisure activity (Tleiso). Important for CBA-1 is the shortest bike travel time available across possible locations (Ad1), the start time of the activity (Tiday), and the presence of a shop activity (yAshop). None of these variables were equally found as most important variables in CHAID.

Also in this case, given the rather small improvement in accuracy for CBA-1 (+1.1%) and for CBA-2 (+2.5%) when compared to CHAID, it is not safe to proclaim that CBA-1 and CBA-2 selected better variables and combined them more efficiently than CHAID did.

Table 3.17: Some important rules in CBA, CBA-1 and CBA-2 (according to sorting criteria) for the trip chaining choice facet

Rule	CBA	CBA-1	CBA-2
1	Ycanvo=0 ^ Ycanna=0→ Single stop (Conf.: 1; Supp.: 0.57)	Ycanvo=0 ^ Ycanna=0→ single stop (Conf.: 1; Supp.: 0.57)	Ycantu=1→ In-Between stop (Conf.: 0.92; Supp.: 0.036)
2	Ycanna=0 ^ Xndu=4 ^ Ccomp=1→ Single stop (Conf.: 1; Supp.: 0.04)	Ycantu=1 ^ Ad1=0 ^ Ccomp=4→In-Between stop (Conf:1; Supp: 0.017)	Ycanvo=0 ^ Ycanna=0→Single stop (Conf.: 1; Supp.: 0.57)
3	Ycanna=0 ^ Xndu=4 ^ Tshop=2→ Single stop (Conf.: 1; Supp.: 0.04)	Ycantu=1 ^ Ccomp=4 ^ YAshop=0→ In-Between stop (Conf:1; Supp: 0.015)	Ycanna=1 ^ Ycantu=0→After stop (Conf.: 0.60; Supp.: 0.19)
4	Ycanna=0 ^ Xntype=3 ^ Tleiso=1→ Single stop (Conf.: 1; Supp.: 0.03)	Ycanna=0 ^ Tiday=6→Single stop (Conf.: 0.98; Supp.: 0.15)	/

3.7.9 WITH WHOM

The last decision facet which is discussed in this section is the with whom facet. The activity type (Atype),the presence of children in the household (Cchild), and Ncar appear in all CBA algorithms (see Table 3.18). In addition to this, the age of the oldest person in the household (Cage), the household type (ccomp), the availability of out-of-home leisure activities and the obviously very logical dummy variable which indicates the presence of others in the household (yAvail), are both present in CBA and CBA-2. A variable which is only selected by CBA-1 is a variable which denotes the availability of car in the *t*-th time interval (yCar(*t*)). When compared with the CHAID based approach, there are only two equal variables (Cchild and Atype).

Despite the fairly equal performance of CBA-1 (-0.3%), original CBA performed significantly better than CHAID (+7.8%). Considering the small number of correspondence of important variables in CHAID and CBA, it is safe to assume that the algorithm selected the best variables and combined them most efficiently.

Table 3.18: Some important rules in CBA, CBA-1 and CBA-2 (according to sorting criteria) for the with whom choice facet

Rule	CBA	CBA-1	CBA-2
1	Atype=1 ^ Ccomp=2→Alone (Conf.:0.92; Supp:0.02)	Atype=1 ^ Cchild=1 ^ Ncar=2→Alone (Conf.:0.88;Supp.:0.04)	Cage=2 ^ Cchild=1 ^ YLeiso=1→Others out HH (Conf.:0.88; Supp.:0.013)
2	Atype=1 ^ YGroc=0 ^ YAvail=0→Alone (Conf:0.91; Supp.:0.03)	Atype=1 ^ yCar2=1 ^ yCar4=1→Alone (Conf.:0.68;Supp.:0.11)	Atype=1 ^ Ccomp=2→Alone (Conf.:0.92; Supp.:0.02)
3	Atype=1 ^ YAvail=0 → Alone (Conf.:0.90; Supp.:0.03)	Atype=1 ^ yCar2=1 ^ yCar5=1→Alone (Conf.: 0.68; Supp.:0.11)	Atype=1 ^ YAvail=0→Alone (Conf.:0.90; Supp.:0.03)
4	Cage=2 ^ Cchild=1 ^ YLeiso=1→Others out HH (Conf.:0.88; Supp.:0.01)	Atype=1 ^ yCar4=1→Alone (Conf.:0.68; Supp:0.11)	Atype=1 ^ Cchild=1 ^ Ncar=2→Alone (Conf.:0.88; Supp.:0.04)

3.8 CONCLUSION

The idea for undertaking the research effort that has been described in this chapter originated from the fact that machine learning techniques which are used in most computational process models are often quite standard supervised classification systems. To this end, it was examined in this chapter whether an unsupervised/descriptive learning technique (association rules) can be used as the basis for coming to efficient supervised learning for the different facets of the Albatross model.

A good example about how unsupervised and supervised learning is the CBA algorithm. CBA focuses on a limited subset of association rules. The algorithm has already been successfully applied and tested within the field of Machine Learning. It was found that the original CBA algorithm generated better predictive accuracy results at choice facet level than the CHAID decision tree approach for most of the datasets under evaluation. This finding proved that the idea of using unsupervised machine learning techniques for supervised learning tasks holds out considerable promise. Unfortunately, the same good results of CBA could not be achieved at pattern and trip matrix level. At both levels, a higher amount of overfitting occurred. To this end, the idea was conceived to examine whether two novel contributions to the original CBA algorithm, referred to as CBA-1 and CBA-2 could reduce the size of the decision rule set and as a result lead to a better predictive performance.

In adapted CBA-1 and CBA-2, the intensity of implication and an own-developed heuristic –dilated chi-square- were used as sorting criteria. The fact that both contributions significantly changed the behaviour of the original CBA algorithm was found at one hand on a quantitative level, where CBA-1 and CBA-2 respectively lead to a better predictive performance for 5 out of 9, and 4 out of 9 datasets when compared to CBA. More importantly, the aim for undertaking the adaptations -which was a reduction of the size of the decision rule set- was achieved for all datasets, both for CBA-1 and CBA-2. At an average scale, the predictive performance was somewhat worse for both CBA-1 and CBA-2 when compared to original CBA, but this is quite normal given the highly significant size reduction in the number of rules. CBA-1 achieved somewhat better average results than CBA-2, both in terms of predictive performance and in terms of number of rules. CBA-1 also requires no parameter selection, which is obviously favoured. The good results of CBA-1 and CBA-2 also paid off at the pattern and choice facet level where a lower degree of overfitting occurred when compared to original CBA. The fact that both contributions changed the composition of the original CBA rule set was also confirmed at a more qualitative and descriptive level by a discussion of the four (most important) rules that were first added to every classification system. This analysis showed that there were not only less rules but also that the rules that have been used per classification system, were different in most cases.

Initial results which are currently being tested on multiclass UCI machine learning data, seem to indicate that our proposed adaptations seems to perform better on binary class than on multiclass datasets. An evaluation of the algorithms on binary class UCI data was already incorporated in Appendix C. If these initial findings are confirmed on more data, an important topic for future research could be to propose another sorting measure which takes specific care for multiclass datasets. It would also be particularly interesting to evaluate the effect of such a measure within the context of transportation, since most of our datasets indeed are multi-class data. Taking our experiments into account, the real challenge for such a new measure should be to simultaneously reduce the size of the decision rule set and improve the original CBA algorithm, and this at an average level.

Chapter 4
Classification based on Bayesian networks

4.1 INTRODUCTION

It was already briefly mentioned in the previous chapter that the major difference between association rules and Bayesian networks is their difference in respectively measuring and modelling co-occurrence in data. Bayesian networks are often used for querying and for making advanced what-if-analyses, which makes that they are probably also better suited for reasoning and explanatory purposes. It is assumed in this chapter that Bayesian networks are well suited for identifying and capturing complex relationships between a set of factors that cause a particular transport behaviour. In addition to the analysing capabilities, Bayesian networks can also be used for classification and prediction. However, the technique is also unsupervised and descriptive in nature and can thus contribute to a more comprehensive overview about how supervised and unsupervised learning can be integrated, in addition to the CBA technique that was described in Chapter 3.

The remainder of this chapter is mainly divided into four major parts. The first part gives an introduction into the basic concepts, definitions and algorithms for Bayesian network discovery and analysis. More specifically, we will detail on the structural learning algorithm that has been used in this dissertation and on parameter learning calculus.

The second part demonstrates that Bayesian networks are potentially very powerful descriptive representation and reasoning tools under conditions of uncertainty. It will be illustrated that they are particularly valuable to capture and visualize the multidimensional nature of complex decisions. It is also shown that they enable one to take into account the many (inter)dependencies that typically exist in complex decision-making processes. Furthermore, the technique is not restricted to the identification of the significant variables but it also enables one to quantitatively evaluate the strengths of the relationships and to reason about and predict choice probabilities. By means of an empirical application, it will be shown how the technique can be used to evaluate and reason about the choice processes that form transport mode decisions.

In a third part, it is examined how Bayesian networks can be used for classification. In particular, we will elaborate on how classification rules can be extracted from a Bayesian network, and how these rules can be used within the context of Albatross.

In a fourth part, the methodological state-of-the-art is advanced by integrating Bayesian networks with decision trees. The idea here is to use Bayesian networks as the information source for deriving a complete decision tree, instead of relying on the original data for doing this. Similar to the advancements that were proposed with respect to CBA, the aim of this contribution is to generate more accurate and compact decision lists/trees.

4.2 BAYESIAN NETWORKS: DEFINITIONS AND ALGORITHMS

4.2.1 PREFACE

The origins of Bayesian networks have to be situated long before the 1980s. Initially, they were only applied on a small scale in mathematics and statistics.

The work of Pearl (1988) is widely accepted as the time by which Bayesian networks were introduced to the artificial intelligence/machine learning community. The first real-world applications of Bayesian networks were MUNIN (Andreassen *et al.*, 1989) and Pathfinder (Heckerman *et al.*, 1992). However, the bloomy days of Bayesian networks are situated in the 1990s thanks to the development of effective algorithms for probabilistic inference and learning from data. Indeed, Bayesian networks were originally only intended to be constructed from domain knowledge, while advancements with respect to learning from data probably denoted the full breakthrough of the technique.

It was already mentioned in Chapter 3 that measures such as confidence can be used to say something about the level of truth, or the certainty of a particular rule. However, especially in tasks of combination and chaining, major problems may be involved with respect to the calculus of those certainty measures (Jensen, 2001). Suppose that we have two rules "if *A* then *B* with certainty *X*" and if "*B* then *C* with certainty *Y*". If we know both *A* and *B*, it is unclear what the certainty of the fact *C* should be. After all, the answer requires a function for combining certainties coming from those two rules. Another problem is chaining. Considering again the above rules, and suppose we only know *A*, then it is also unclear what the certainty of C is. Heckerman (1986) showed that any function for combination and chaining would lead to wrong conclusions.

In the search for mathematically and theoretically sound foundations for doing inference, the Bayes' theorem became one of the most important cornerstones, because it enables combining new data with historical knowledge. Because of this property, Bayesian networks can be considered as being probabilistic expert systems, that can be used for reasoning under uncertainty.

The use of Bayesian networks in transportation has been advocated before (Plach, 1997), but only recently it is gaining increased popularity (Torres and Huber, 2003, Davis and Pei, 2004, Ozbay and Noyan, 2005, Verhoeven *et al.*, 2005). However, within the field of activity-based modelling of transportation demand, its application is still very limited, and to the best of our knowledge, it has never been used before in the context of a fully operational activity scheduling model.

4.2.2 DEFINITIONS

GENERAL CONCEPTS

A Bayesian network consists of two components (Pearl, 1988): first, a directed acyclic graph (DAG) in which nodes represent stochastic domain variables and directed arcs (links, edges) represent conditional dependencies between the variables (see definitions 4.10-4.11) and second, a probability distribution for each node as represented by conditional dependencies captured with the directed acyclic graph (see definitions 4.1-4.8). To formalize, the following conceptualization and definitions are relevant:

Suppose we have a set of possibly related objects $X=\{X_1, X_2, \ldots, X_n\}$. The set can be pictorially represented by a set of nodes, or vertices, each for one element in X. The nodes can be connected by lines, ars or arrows, which are referred to as links or edges. If there is an edge between two nodes X_i and X_j, we use L_{ij} to denote such a link. We will denote L as the set of all links.

Definition 4.1: A graph

A graph $G=(X, L)$ is defined by two sets X and L where X is a finite set of nodes $X=\{X_1, X_2,\ldots, X_n\}$ and L is a set of links (edges). ■

Definition 4.2: Directed link

Let G=(X, L) be a graph. When $L_{ij} \in L$ and $L_{ji} \notin L$, the link L_{ij} is called directed. ■

Definition 4.3: Directed graph

A graph in which all the links are directed, is called a directed graph. ■

Definition 4.4: An adjacency set

Given a graph G=(X, L) and a node X_1, the adjacency set of X_1 is the set of nodes directly attainable from X_1, that is, Adj(X_1)={$X_j \in X|L_{ij} \in L$}. ■

Definition 4.5: A path

A path from node X_i to node X_j is an ordered set of nodes (X_{i1}, ..., X_{ir}), starting in X_{i1}=X_i and ending in X_{ir}=X_j, such that there is a link from X_{ik} to X_{ik+1}, k=1,..., r-1, that is, $X_{ik+1} \in$ Adj(X_{ik}), k =1,...,r-1. ■

Definition 4.6: A closed path

A path (X_{i1}, ..., X_{ir}) is said to be closed if it has the same starting and ending nodes, that is, if X_{i1}=X_{ir}. ■

Definition 4.7: A cycle

A cycle is a closed directed path in a directed graph. ■

Definition 4.8: Directed (a)cyclic graph.

A directed graph is said to be cyclic if it contains at least one cycle. Otherwise, it is called a directed acyclic graph (DAG). ■

Definition 4.9: Parent, child

Given a directed graph G=(X, L) and nodes X_i and X_j in X, X_i is called a parent of X_j, and X_j is called a child of X_i, if there is a directed link from X_i to X_j. ■

Definition 4.10: Edges, dependencies, independencies

Edges in a Bayesian network represent direct conditional dependencies between the variables. The absence of edges between variables denotes statements of independence. We say that variables Y and Z are independent given a set of variables X if P($z|x,y$)=P($z|x$) for all values x, y and z of variables X, Y and Z. Variables Y and Z are also said to be independent conditional on X. ■

Definition 4.11: CPT

A Bayesian network also represents distributions, in addition to representing statements of independence. A distribution is represented by a set of conditional probability tables (CPT). Each node *X* has an associated CPT that describes the conditional distribution of *X* given different assignments of values for its parents. ■

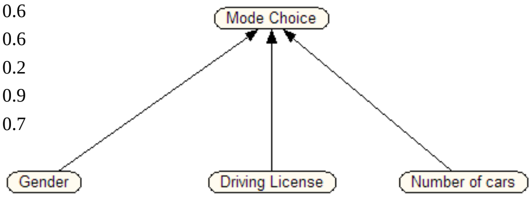
The definitions discussed above were illustrated in Figure 4.1 by means of a very simple hypothetical example. First, the network introduced here clearly is acyclic and directed. Second, the variables "gender", "driving license" and "number of cars" are parents of the "mode choice" variable. Finally, dependent and independent relationships, as well as examples of CPTs are shown in this figure. In the upper CPT for instance, the probability for mode choice being equal to bike, is 0.2, given that gender=male, driving license=yes and number of cars=1.

However, for complex problems, a large Bayesian network will be required, often resulting in a tangle of nodes, which at first glance might look confusing (see also infra, Figure 4.5). Still, it is exactly this property that makes the technique a very powerful representation and visualization tool that enables the user to conceptualize the association between variables. Furthermore, much of the apparent disorder that might exist in a Bayesian network can be reduced by pruning the network. This means that the network can be reduced in size without much loss of relevant information (see infra).

Learning Bayesian networks has traditionally been divided into two categories (Cheng *et al.*, 2002): i.e. structural and parameter learning. Parameter learning determines the conditional probability relationship at each node of the network,

Gender	Driving License	Number of cars	*P* (mode choice = bike)	*P* (mode hoice = car)
Male	Yes	1	0.2	0.8
Male	Yes	>1	0.6	0.4
Male	No	1	0.7	0.3
Male	No	>1	0.4	0.6
Female	Yes	1	0.4	0.6
Female	Yes	>1	0.8	0.2
Female	No	1	0.1	0.9
Female	No	>1	0.3	0.7

CPT Mode Choice



P(Gender = male)	*P* (Gender = female)
0.75	0.25

CPT Gender

P (Driving License=Yes)	*P* (Driving License=No)
0.6	0.4

CPT Driving License

P (Number of cars=1)	*P* (Number of cars>1)
0.2	0.8

CPT Number of Cars

Figure 4.1: A simple Bayesian network with its CPT

given the link structures and the data. It can therefore be used to quantitatively examine the strength of the identified effect. Structural learning determines the dependence and independence of variables and suggests a direction of causation (or association), in other words, the existence or non-existence of the links in the network. As mentioned before, experts can provide the structure of the network using domain knowledge. However, the estimated structure can also be extracted from empirical data. Especially the last option offers important and interesting opportunities for transportation travel demand modelling because it enables one to visually identify which variable or combination of variables influences the target variable of interest. The next section elaborates on both types of learning.

4.2.3 ALGORITHMS

STRUCTURAL LEARNING

The major advantage of learning the structure of the network from the data, compared to building the network using prior domain knowledge, is that this enables the extraction of unknown, useful and understandable knowledge from data. This property is useful both when one wishes to test an assumed structure against empirical data or when one wishes to explore the dependencies in the data. It has to be noted, however, that algorithms that learn the structure of the network, can sometimes have difficulties in capturing the correct (causal) relationships. Causality is extremely difficult to be modeled and captured efficiently by a machine learning algorithm, because it often also involves human reasoning. Therefore, the intuitive interpretation of some directions of arrows in a Bayesian network may look strange. Therefore, it is better to consider the directed arc as an association rather than as a causality relationship per se.

Structural learning can be divided into two categories: search & scoring methods and dependency analysis methods. Algorithms, belonging to the first category interpret the learning problem as a search for the structure that best fits the data. Different scoring criteria have been suggested to evaluate the structure of the network, such as the Bayesian scoring method (Cooper and Herskovits, 1992; Heckerman *et al.*, 1995) and minimum description length (Lam and Bacchus, 1994).

Algorithms, belonging to the second category, view the learning problem differently. Indeed, since a structure encodes many dependencies of the underlying model, the algorithms belonging to the latter category try to discover the dependencies from the data and then use these dependencies to infer the structure. The dependency relationships are measured according to statistical tests, such as Entropy (Herskovits, 1991), Chi-square and the mutual information measure. In order to conduct our experiments, the algorithm developed by Cheng *et al.* (2002) (belonging to the category of dependency analyses methods) has been used. Before detailing this algorithm, two other general definitions have to be introduced.

Definition 4.12: Node ordering
Node ordering specifies a causal or temporal order of the variables of the domain, so that any node cannot be a cause or happen earlier than the nodes appearing earlier in the order. The ordering of the nodes is often specified through domain knowledge information. ▪

Definition 4.13: d-separation
For a DAG G=(V,E); two sets of nodes A, $B \in V$ and $A \neq B$, are **d-separated** by node set $C \subset V \setminus \{A,B\}$, if $\forall$ paths between a node in A and a node in B, $\exists V$ in the path such that either:

1) The connection is either 1) serial at V (i.e. either $\rightarrow V \rightarrow$ or $\leftarrow V \leftarrow$) or 2) diverging at V (i.e. $\leftarrow V \rightarrow$, non-collider), and along with one of those two conditions, it is such that $V \in C$.
2) The connection is converging (i.e. $\rightarrow V \leftarrow$, collider) and neither V nor any of V's descendants are in C.

It is proven in Geiger and Pearl (1988) that the concept of d-separation can reveal all the conditional independence relationships that are encoded in a Bayesian network. In other words, no other criterion can do better. ▪

Since the concept of d-separation is often used to infer structures for Bayesian networks and because the definition above is quite complex, a more intuitive explanation might improve the understanding. Consider a Bayesian network as a tangle of roads, where each node is an intersection with a traffic light. Cars can pass the road when the traffic light allows this. In this case, the road is active. The flow of cars can pass an active road but not an inactive one. When all the traffic lights on one adjacency path between two roads are active, the path is open. If any traffic light in the path is inactive, we say that the path is closed. Since there are two kinds of nodes in a Bayesian network (converging or *colliders* and diverging or *non-colliders*, see definition 4.13), the traffic lights accordingly have two different initial states, inactive and active. Initially, any traffic light that represents a collider-road is inactive for that road; any traffic light that represents a non-collider-road is active for that road. Since a node can be a collider of some paths and non-collider of some other paths, a traffic light can also be active for some roads and inactive for some other roads. Putting a node in the condition-set can be viewed as altering the status of the corresponding traffic light and possibly the statuses of other traffic lights. When all the paths between two nodes are closed by altering the statuses of some traffic lights, we say that the nodes are d-separated by the condition-set corresponding to those traffic lights whose statuses were altered.

Having explained the two concepts above, we are now ready to explain the details of the algorithm that is used in this chapter. The main advantage of the algorithm developed by Cheng *et al.* (2002) is that node ordering is not required and that the algorithm has proven to be efficient (exponential complexity on computational independence tests can be avoided). The algorithm is an extension of the Chow-Liu tree construction algorithm (Chow and Liu, 1968) to a three-phase (drafting, thickening and thinning) Bayesian learning algorithm. An example will illustrate these phases. Suppose we have a dataset which has an underlying Bayesian network structure as depicted in Figure 4.2 (a). The task of

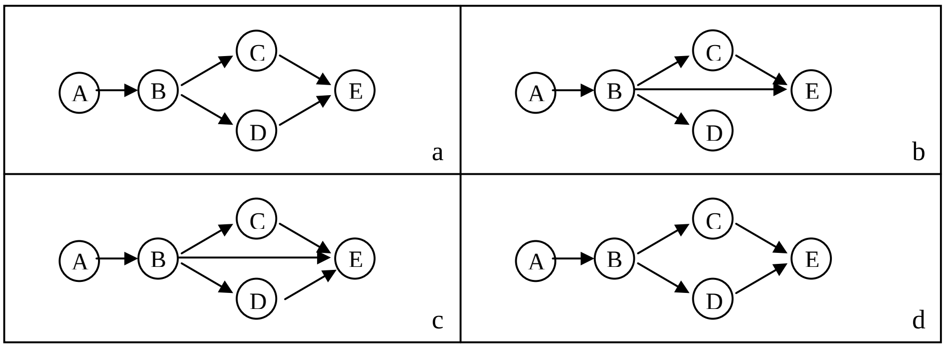


Figure 4.2: An example showing the three phases of the structural learning algorithm (Cheng *et al.*, 2002)

the algorithm is now to discover this underlying network structure from the data. The algorithm will first calculate the mutual information of each pair of nodes as a measure of closeness. The mutual information of two nodes X_i, X_j is defined as:

Definition 4.14: Mutual information between two nodes

$I\ (X_i,X_j)= \sum_{x_i,x_j} P(x_i,x_j)\log\frac{P(x_i,x_j)}{P(x_i)P(x_j)}$,where $P(x_i,x_j)$ is the posterior probability that a particular state of X_i (i.e. x_i) and a particular state of X_j (i.e. x_j) occur together; $P(x_i)$ is the prior probability that a state x_i of X_i will occur and $P(x_j)$ is the prior probability that a state x_j of X_j will occur. The probabilities are summed across all states of X_i and across all states of X_j. When $I\ (X_i, X_j)$ is smaller than a certain threshold ξ, we say that X_i and X_j are marginally independent. ■

Since we have 5 nodes, the mutual information of all 10 pair of nodes is calculated. Suppose that $I(B,D) \geq I(C,E) \geq I(B,E) \geq I(A,B) \geq I(B,C) \geq I(C,D) \geq I(D,E) \geq I(A,D) \geq I(A,E) \geq I(A,C)$ and that all the mutual information is greater than ξ. The drafting phase of the algorithm uses this information to come up with a first “draft”. Every time there is an open path between the two nodes, the algorithm will connect both nodes by an arrow, following the sorted order shown above. The completion of the drafting phase is shown in Figure 4.2(b). Note that arrow (*B*,*E*) is wrongly added and that arrow (*D*,*E*) is missing. In the next phase (thickening), conditional independence tests and d-separation analysis is used to see if we should connect those pairs of nodes. In our example, Arc (*D*,*E*) was therefore added because *D* and *E* are not independent (conditional on *B*). The same type of reasoning explains why arc (*A*,*C*) is not added. After this phase, the graph looks like Figure 4.2(c). It is not sure whether a connection is really necessary, but we can be sure that no real arcs are missing. Since both phase I and phase II can add arcs wrongly, the task of the thinning phase is to identify those wrongly added arcs and remove them. Again, a conditional independence test is used to make a decision, but this time we can be sure that the decision is correct. As a result of this, edge (*B*,*E*) is removed because *B* and *E* appeared to be independent given (*C*,*D*) (see Figure 4.2(d)). A proof of the presented algorithm and a further discussion of an alternative algorithm which does not take node ordering into account (i.e. the more general case of the algorithm described here) can be found in Cheng *et al.* (2002).

Parameter learning

Parameter learning determines the prior CPT of each node of the network, given the link structures and the data. It can therefore be used to examine quantitatively the strength of the identified effect. As mentioned above, a conditional probability table P $(A|B_1 \ldots B_n)$ has to be attached to each variable A with parents B_1, ..., B_n. Note that if A has no parents, the table reduces to unconditional probabilities P(A). According to this logic, for the example Bayesian network depicted in Figure 4.1, the prior unconditional and conditional probabilities to specify are: P(Driving License); P(Gender); P(Number of cars); P(Mode choice|Driving License, Gender, Number of cars). Since the variables “Number of cars”, “Gender” and “Driving license” are not conditionally dependent on other variables, calculating their prior frequency distribution is straightforward. Calculating the initial probabilities for the “Mode choice” variable is computationally more demanding.

In order to calculate the prior probabilities for the “Mode choice” variable, the conditional probability table for P(Mode Choice| Driving License, Gender, Number of cars) was set up in the first part of Table 4.1. Again, this is straightforward mathematical calculus. In order to get the prior probabilities for the Mode Choice variable, we now first have to calculate the joint probability P(Choice, Gender, Number of cars, Driving License) and then marginalize “Number of cars”, “Driving License” and “Gender” out. This can be done by applying Bayes’ rule, which states that:

P(Choice,Gender, Number of cars, Driving License) = P(Choice|Gender, Number of cars, Driving License)*P(Gender, Number of cars, Driving License).

Since “Gender”, “Number of cars” and “Driving License” are independent, the equation can be simplified for this example as:

P(Choice, Gender, Number of cars, Driving License)= P(Choice|Gender, Number of cars, Driving License)*P(Gender)*P(Number of cars)*P(Driving License).

Note that P(Gender=male; Gender=female)=(0.75; 0.25),
P(Driving License=yes; Driving license=no) = (0.6; 0.4),
and P(Number of cars=1; Number of cars>1)=(0.2; 0.8), which are the prior frequency distributions for those 3 variables. By using this information, the joint probabilities were calculated in the second part of Table 4.1. Marginalizing

"Gender", "Number of cars" and "Driving License" out of P(Choice, Gender, Number of cars, Driving License) yields P(Mode Choice=bike; Mode Choice=car) = (0.506; 0.494).

Table 4.1: Conditional and Joint Prior Probability Tables
for the Transport Mode Choice Variable

Conditional Prior Probability Table specifying P(Choice|Gender, Driving License, Ncar)

Gender	Male				Female			
Driving License	Yes		No		Yes		No	
Number of cars	1	>1	1	>1	1	>1	1	>1
Mode Choice bike	0.2	0.6	0.7	0.4	0.4	0.8	0.1	0.3
Mode Choice car	0.8	0.4	0.3	0.6	0.6	0.2	0.9	0.7

Joint Prior Probability Table for P(Choice,Gender,Ncar, Driving License)

Gender	Male				Female			
Driving License	Yes		No		Yes		No	
Number of cars	1	>1	1	>1	1	>1	1	>1
Mode Choice bike	0.018	0.216	0.042	0.096	0.012	0.096	0.002	0.024
Mode Choice car	0.072	0.144	0.018	0.144	0.018	0.024	0.018	0.056

These are the joint prior probabilities for the "Mode choice" variable. Alternatively, probabilities can be calculated automatically by means of probabilistic inference algorithms that are implemented in Bayesian network enabled software. A screenshot of a Bayesian network in the Netica software is given in Figure 4.3 for our example. The different variables in the network are

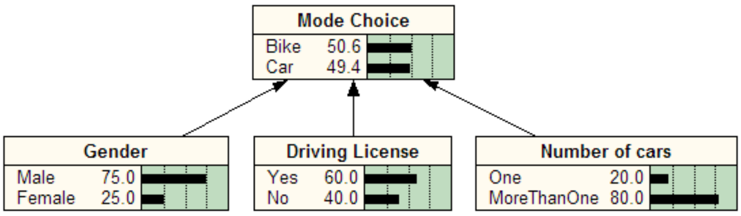


Figure 4.3: Representation of a simple Bayesian network in Netica

represented as boxes and each state in the network is shown with its belief level (probability) expressed as a percentage and as a bar chart. It can be seen from this figure that the joint probability distribution in this figure corresponds to the values that were calculated earlier. The use of these software packages becomes particularly useful in the case when for instance “Gender”, “Number of cars” and “Driving License” are dependent.

Entering evidences

In fact, Figure 4.3 only depicts the prior distributions for each variable. This is useful but not very innovative information. An important strength of Bayesian networks, however, is to compute posterior probability distributions of the variable under consideration, given the fact that values of some other variables are known. In this case, the known states of variables can be entered as evidence in the network. When evidence is entered, this is likely to change the states of other variables as well, since they are conditionally dependent. This is demonstrated by entering the evidence in the network that the “Mode choice” variable is equal to “car”. In this case, evidence on “Mode choice” now arrives in the form of P^*(Mode Choice=bike; Mode choice=car)=(0; 1), where P^* indicates that we are calculating posterior probabilities (i.e. after entering evidences).
Then,

$$P^*(\text{ Choice, Gender, Number of cars, Driving License})=$$

$$P(\text{Number of cars, Gender, Driving License} \mid \text{Mode choice}) * P^*(\text{Mode Choice})=$$
$$(P(\text{Choice, Gender, Number of cars, Driving License})*P^*(\text{Mode Choice}))/$$
$$P(\text{Mode Choice}).$$

This means that the joint probability table for “Choice”, “Number of cars”, “Driving License” and “Gender” is updated by multiplying by the new distributions and dividing by the old ones. The multiplication consists of omitting all entries with “Choice”=“bike”. The division by P(Mode Choice) only has an effect on entries with Mode Choice=“car”, so therefore the division is by P(Mode Choice=“car”). For this simple example, the calculations can be found in Table 4.2. The distributions P^*(Number of cars), P^*(Gender) and P^*(Driving License) are calculated through marginalization of P^*(Choice, Gender, Number of cars, Driving License).

Table 4.2: Posterior Probability Table for the Transport Mode Choice Variable

The Calculation of P*(Choice, Gender, Ncar,Driving License) = P(Choice, Gender,Ncar, Driving Licence|Mode Choice=car)

Gender	Male				Female			
Driving License	Yes		No		Yes		No	
Number of cars	1	>1	1	>1	1	>1	1	>1
Mode Choice bike	0	0	0	0	0	0	0	0
Mode Choice car	0.146	0.291	0.036	0.291	0.036	0.049	0.036	0.113

This means that

P*(Gender=male; Gender=female) = (0.765;0.235);
P*(Number of cars=1; Number of cars>1) = (0.255;0.745);
and P*(Driving License=yes; Driving License=no)= (0.522; 0.478), when evidence was entered that the "Mode choice" variable equals car.

Obviously, the calculation of this example is simple. However, in real-life situations it is likely that conditionally dependent relationships between the "choice" variable and other variables exist as well, and as a result the evidence will propagate through the whole network. In this case, the effect of entering evidences is preferably examined by means of Bayesian network enabled software. An illustration for our example is given in Figure 4.4. When an evidence is entered in the network, this is shown in the figure as a shaded box and as a 100% belief. More information about efficient algorithms for propagation of evidence in Bayesian networks can be found in Pearl (1988) and in Jensen *et al.* (1990).

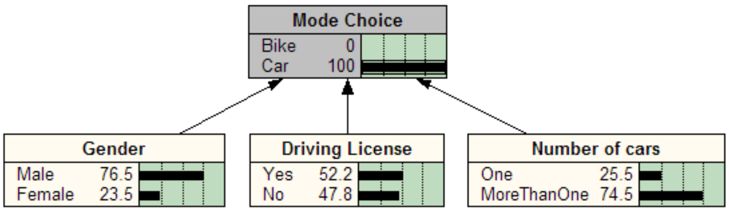


Figure 4.4: Entering evidences in a simple Bayesian network

4.3 Bayesian Networks for Descriptive Learning: Illustration

The aim of this section is to explore the potential value of Bayesian networks to identify and explain complex relationships between variables. By example, we focus in this section on the identification and interpretation of a set of interrelated factors which can have an influence on transport mode choice. The dataset that is used for this empirical study has been described in section 2.3.7. For this descriptive purpose, no distinction has been made between a training and a test set. The logic as the one used in section 4.2 (structural and parameter learning, entering evidences) is maintained here. Parts of this section are based upon work reported in Janssens et *al.* (2003b).

4.3.1 Structural Learning

Given the large number of variables in the dataset (41 independent) and assuming that node ordering is not known, building a network (by means of the algorithm described in section 4.2.3) is not an easy task at all. The final result is depicted in Figure 4.5. It is clear that drawing conclusions from this network is almost infeasible given its complexity. Therefore, a way should be found to reduce the size of the network to make it more comprehensible. This can be accomplished by means of a pruning strategy. As mentioned before, pruning aims to reduce the size of the network without loosing significant information with respect to the variable of interest, in this case “mode choice” (indicated by the variable *choice)*. To this end, the mutual information between pairs of nodes, initially used to build the network structure (see definition 4.14) is needed. However, since our primary interest is to identify which variables influence the transport mode choice, only the mutual information between this main variable and the others is needed. The next section elaborates on this.

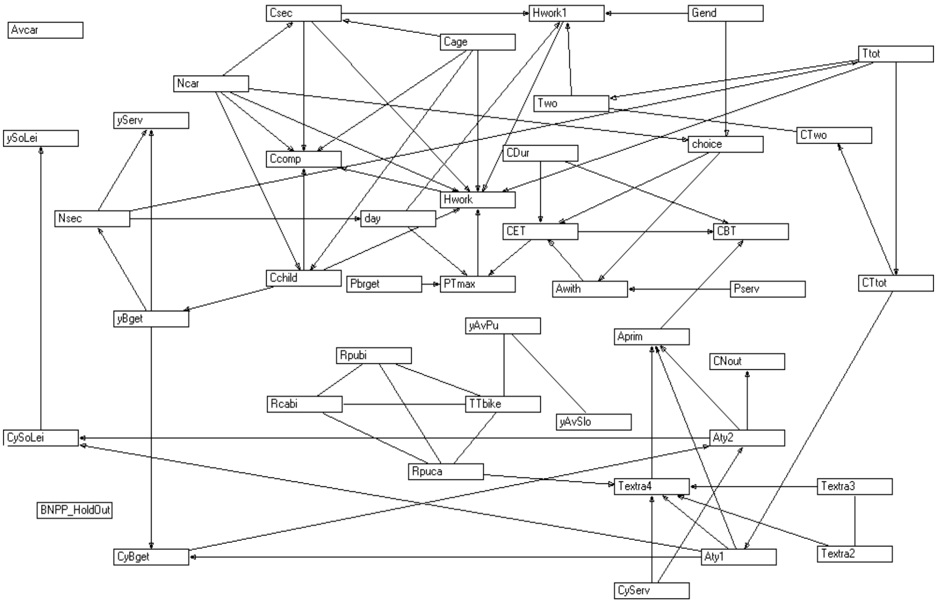


Figure 4.5: An unpruned Bayesian network

4.3.2 PRUNING THE NETWORK STRUCTURE

The mutual information between two nodes is reflected in the expected entropy reduction of one node due to a finding (observation) that is related to the other node. Entropy can be defined as a measure for impurity, disorder and randomness of a particular system. It is for instance often used in the case of decision tree induction, where the aim is to reduce the entropy by recursively splitting the tree. Entropy is measured in bits and is later also used in definition 5.10 in the context of decision tree induction. However, the measure can also be used as an alternative method for pruning the network structure, as it is done in this section. To this end, the dependent variable (transport mode choice) is called the query variable (denoted by the symbol Q), the independent variables are called findings variables (denoted by the symbol F). Therefore, the expected reduction in entropy for network pruning of Q due to a finding related to F can be defined here as being completely analogous to the notation in definition 4.14, where X_i equals Q and X_j equals F.

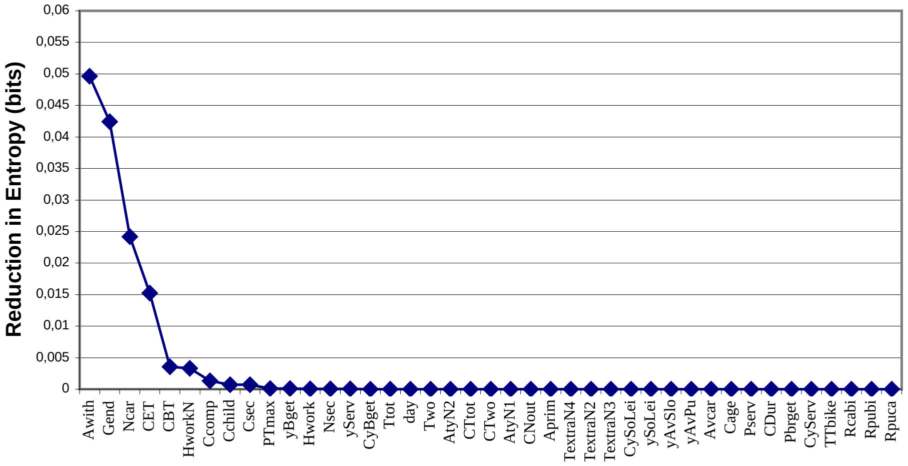


Figure 4.6: The expected reduction in entropy of the transport mode choice variable for the different finding variables.

As depicted in Figure 4.6, by application of definition 4.14, the expected reduction in entropy of the transport mode choice variable can be calculated for the various findings variables. It is shown that there is a huge amount of mutual information between the first five variables and the transport mode choice. Rather soon, however, the reduction in entropy falls to zero, indicating independence between *Q* and *F*. To select the nodes, an entropy reduction of less than 0.001 bits was used as a threshold.

According to this logic, the network can be pruned by discarding those variables that fail to meet this criterion. The pruned network is shown in Figure 4.7. In this figure, joint probability distributions are shown as well. These values are derived by means of the procedure that has been explained in section 4.2.3, subsection Parameter Learning.

4.3.3 Querying The Network: Sensitivity Analysis

The effect that each findings node has on transport mode choice can be measured in a straightforward manner by means of a sensitivity analysis. The sensitivity report is shown in Table 4.3. This table shows the maximum and the minimum posterior probability of the transport mode node due to certain evidences, which are entered in the network. For instance, when for the variable "CET" the value 1 is entered as evidence (which implies that the latest possible end time is before 12:30, see section 2.3.7), the likelihood that somebody uses a

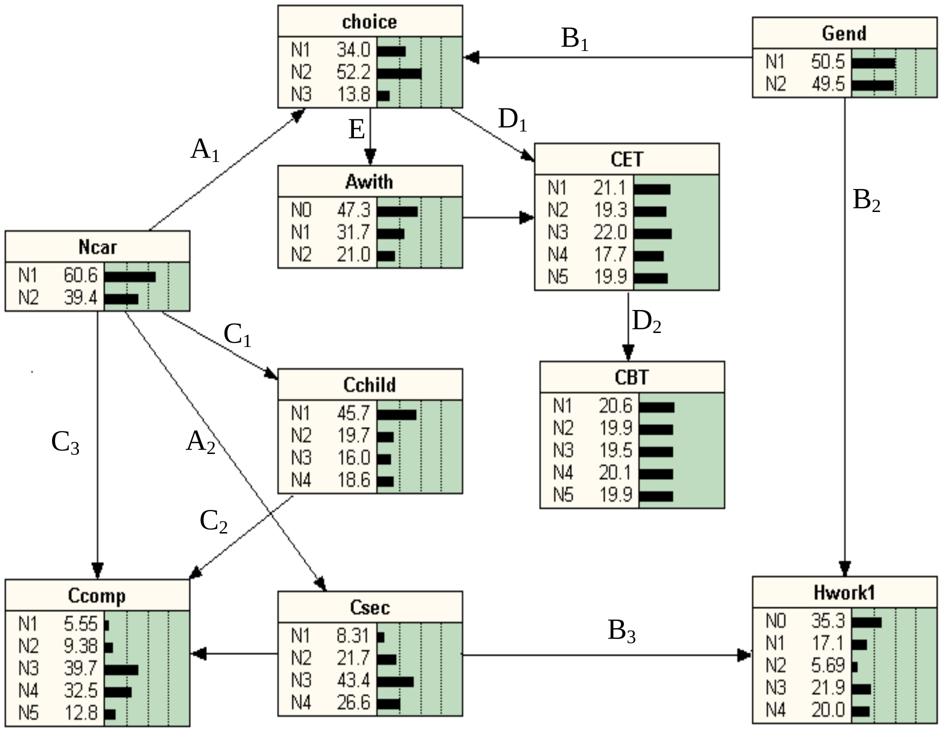


Figure 4.7: The pruned Bayesian network

slow mode of transport (walk or bike) will increase with 18.72% (see Table 4.3), which results in a posterior probability of 52.73% (i.e. from Figure 4.7; 34.01% + 18.72%). Obviously, given the descriptive learning character of the technique, any variable can be chosen for the analysis of sensitivity.

A close inspection of the network, enabled us to come up with five major findings, as shown in Figure 4.7 by the capitals A-E. These findings might help us to get a thorough understanding of the behavioral pattern of individuals with respect to the choice of transport mode. In addition to this, these findings enable one to get quite a good idea about the explanatory and reasoning capabilities of Bayesian networks.

FINDING A

The variable “Ncar” reflects the ratio between the number of cars and the number of driving licenses. A ratio larger than 1 (value 2) means that there is more than one car at someone's disposal. It is quite obvious that those people have a higher probability (10.96%) to use the car as mode of transport. This is shown by Table 4.3 and by arrow A_1 in Figure 4.7.

The network now enables us to characterize this group of people. This can be done by entering an evidence for state 2 of the node "Ncar". As a result, the fourth state of the "Csec" variable (i.e. high socio-economic class of the household) increases by 7.3%, from 26.6% to 33.9%. Arrow A_2 is therefore quite interesting as it depicts that individuals belonging to a high socio-economic household class, are more likely to have more cars than the total number of driving licenses in the household.

Combining the evidences of the fourth state of the "Csec" node together with the second state of the "Ncar" node, further amplifies this finding. Households with a high level of prosperity, possessing more cars than driving licenses have a likelihood of 78% to use the car as the mode of transport: a substantial increase of 25.8%. Several micro-economic (Johansson-Stenman, 2002; De Jong, 1997;

Table 4.3: Sensitivity Analysis with respect to Transport Mode

		CET	CBT	Awith	Gend	NCAR	Hwork1	Ccomp	PTMax	Csec	Cchild
Walk or Bike (34.01%)	Min(%)	-13.13	-12.91	-9.32	-6.70	-9.46	-4.31	-8.76	-0.96	-1.91	-0.82
	value min	4	5	2	1	2	4	2	0	1	4
	Max(%)	+18.72	+11.35	+7.36	+6.83	+6.16	+4.73	+4.45	+3.46	+1.85	+1.21
	value max	1	2	0	2	1	1	5	2	2	3

		CET	Gend	NCAR	Ccomp	CBT	Hwork1	Awith	Csec	Cchild	PTMax
Car (52.53%)	Min(%)	-13.18	-11.91	-7.14	-5.10	-7.85	-8.31	-4.06	-2.06	-2.77	-2.61
	value min	1	2	1	5	2	1	1	2	2	3
	Max(%)	+13.21	+11.66	+10.96	+10.74	+8.04	+7.54	+2.74	+2.29	+1.39	+0.72
	value max	4	1	2	2	5	4	2	1	4	0

		Awith	CET	Cchild	Gend	CBT	Hwork1	NCAR	Ccomp	PTMax	Csec
Public Transport or Car Passenger (13.77%)	Min(%)	-8.87	-5.55	-5.1	-4.97	-3.51	-3.24	-1.50	-2.00	-1.06	-0.39
	value min	0	1	4	1	2	4	2	2	2	1
	Max(%)	+8.88	+6.07	+5.4	+5.07	+4.86	+3.57	+0.98	+0.64	+0.23	+0.19
	value max	1	5	2	2	5	1	1	5	0	2

Pearman and Button, 1976) and macro-economic studies (Johansson and Schipper, 1997) confirm -by means of income elasticity- the general pattern that car use is positively correlated with income. A practical implication could be to let tax policies focus on this very specific niche group.

FINDING B

The second finding states that gender is an important variable in influencing the choice of transport mode. Males are more likely to use the car than females. Females are more likely to use slow modes, to use public transport or travel as a car passenger. Both findings are depicted by arrow B_1 in Figure 4.7 and by Table 4.3.

Another interesting observation arises, due to arrow B_2, when evidence for males is entered. The third and fourth state of the "Hwork1" node reflect that someone works respectively between 38 and 40 hours, and more than 40 hours per week. Both states increase from 21.9% to 34.2% for the third state and from 20.0% to 35.7% for the fourth state due to the evidence entered for males at the "Gender" node. For females, state zero increases from 35.3% to 47.9% and state one from 17.1% to 29.4%. This means that in this dataset, males tend to work longer than females. There also exists a relation between the socio-economic class of the household ("Csec") and the number of working hours ("Hwork1"). This is shown by arrow B_3. From this relation, it can be derived that when someone works many hours, it is more likely that he/she does not belong to a low socio-economic household class (a decrease from 8.31% to 3.96%), rather than being part of a high socio-economic household class (an increase from 26.6% to 27.8%).

By combining the conclusions, i.e. the fourth state of the "Hwork1" variable with the first state of the "Gender" variable and the fourth state of the "Csec" variable, this finding can be amplified. Males who work more than 40 hours per week and who belong to a high socio-economic household class, have a likelihood of 69.2% to use the car as mode of transport, which is a 17% increase. This finding is consistent with earlier work (Grieco and Turner, 1997), which has shown that the lower the income of a household, the more probable it is that women will experience greater transport deprivation as compared to men. Transport deprivation may then take the form of women's use of inferior modes of transport as compared to men.

FINDING C

The presence of children in the household seems another important explanatory factor in the choice of transport mode. Households with very young children -i.e. children younger than 6 years old- are more likely to use public transport than families with older children. This can be explained by the fact that public transport is often provided for free or at least at a discount for this group of children. Entering evidence for the second state of the node Cchild results –due to arrow C_1- in an increased likelihood of 11.3% to have no more cars than driving licenses in the household. By means of arrow A_1 this evidence propagates further through the network and gives a likelihood of 19.2% for public transport, i.e. a 5.4% increase (see also Table 4.3).

Arrow C_2 is responsible for coupling the age of the children in the household with the household type (this should be interpreted in terms of singe and double households and as households where one partner, both or none of the partners work). This seems to be a veritable relationship as it is common knowledge that household composition changes as children grow older. For our dataset, it was found that in the absence of children, the distribution across the different household types is quite uniform, with a small drop for the single and unemployed household type and a peak for the double, both employed household type. The presence of children in the household results in a logical, although quite spectacular, increase in car use (21%) for the double households, where one partner works and for the double households where both partners work. By means of arrow C_3 the evidence propagates through the network.

Again, by combining these conclusions assuming that both partners are employed and have children younger than 6, the likelihood of public transport use increases by 8.2%. This increase is not much higher than the 5.4% increase observed above, but nevertheless, it can be stated that "Ccomp" and "Cchild" together amplify the originally identified relationship, which was based upon "Cchild" only.

This group of people can typically be characterized as young couples who are more willing to use public transport. In order to obtain a higher effect than the 8.2% increase, promotional campaigns could be launched to focus on this specific market segment.

FINDING D

The latest possible end time of a tour (“CET”), has a large impact on all three transport mode choices. It can be seen from Table 4.3 that when the tour is expected to end late, it is more likely that the car or public transport are the preferable modes of transport. This can partially be explained by the peak hours due to the return from work. However, as the activity diary also contains many other activities during this time period (i.e. only 43% work activities), additional explanation is needed. When the anticipated end time of the tour is before noon, a substantial increase (18%) in the use of slow transport modes (walk or bike) is observed. The fact that slow modes are used less if the trip ends late may perhaps be attributed to reasons of unsafety and discomfort associated with these transport modes during the late hours.

It can be seen from Figure 4.7 that the latest possible end time of a tour (“CET”), is correlated with the earliest possible begin time of a tour (“CBT”). However, in this case, analyzing joint effects of both nodes will not influence transport mode choice since nodes are d-separated when an evidence is entered for the node “CET”. In this case, entering evidence for the node “CBT”, will not influence the distribution of the transport mode choice variable (see definition 4.13).

FINDING E

The person with whom the tour is conducted (“Awith”) is especially an important variable for public transport and for slow modes. If the tour is conducted alone, it is more likely (7.36%) that slow modes of transport are chosen. Since one might expect that there is still a significant number of people who do not engage in carpooling, one would expect that a similar conclusion could be drawn for car use. By narrowing down on tours where only a work activity is involved, a slightly different picture arises. Therefore, when a *work* tour is conducted alone, cars along with public transport become the preferred modes of transport. On the other hand, if another person who is no part of the household participates in the tour, the likelihood of being a car passenger increases significantly (8.88%).

In the previous sections, the potential value of Bayesian networks was examined to cope with the complexity of the transport mode choice decision problem for *reasoning and explanatory* purposes. It was found that Bayesian networks are particularly valuable to capture and visualize the multidimensional nature of complex decisions. Especially the property which takes the many (inter)dependencies among the variables into account (that make up the complex decision-making process), makes Bayesian networks potentially valuable for modelling complex decisions. In the next section, it will be investigated to what extent this property can be generalized for *predictive* purposes to the full set of decision agents (where, when, for how long, with whom and which transport mode) that form the complete Albatross system. Parts of the next section are based upon work reported in Janssens *et al.* (2004e; 2004f).

4.4 Bayesian Networks For Classification

4.4.1 Preface

The concepts that were introduced in the previous sections with respect to structural learning, parameter learning and pruning the network, all remain the same when BN are adopted for classification. With respect to the duration facet in Albatross, the three concepts were briefly recapitulated in Figure 4.8. In this dissertation, discretized variables have been used in the Bayesian networks because they are also used and defined as such in the model specification of the Albatross model. However, the technique of Bayesian networks is not restricted towards the use of discretized variables; continuous variables can also be described with parameters of Gaussian or other distributions for continuous random variables.

4.4.2 Choice Facet level

The procedure that is shown in Figure 4.8 is repeated for every dimension of the Albatross model. The final pruned networks for every dimension are shown in Figure 4.9 (a-i).

In order to evaluate the predictive performance on the training and test sets within Albatross, every case was simply presented as a combination of evidences (see section 4.2.3, subsection "Entering Evidences") to the Bayesian network models. Each evidence corresponds thus to a particular value of an attribute per case. The accuracy percentages that indicate the predictive performance on the training and test sets within Albatross are presented in Table 4.4. Results in this

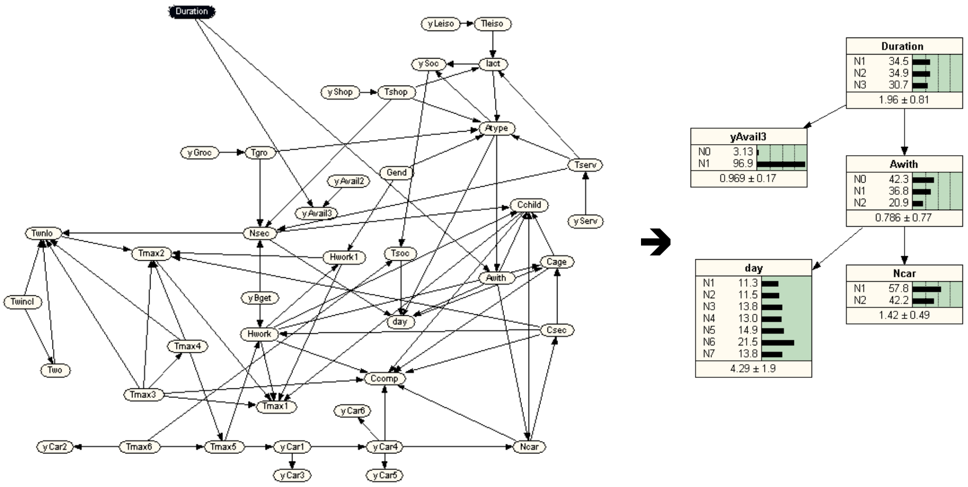


Figure 4.8: An unpruned and a pruned Bayesian network example (duration)

table are compared with the original CHAID algorithm that is used in Albatross. It can be seen that the accuracy percentages of the Bayesian network approach outperform the CHAID decision trees for all nine decision agents of the Albatross model. Especially for the "Mode for work" and for both "Location" decision agents, the increase in predictive performance is significant. In terms of validity, we can conclude that the degree of overfitting is larger for Bayesian networks

Table 4.4: Benchmarking results at choice facet level

Dataset	BN		CHAID	
	Train (%)	Test (%)	Train (%)	Test (%)
Duration	40.9	40.5	41.3	38.8
Location1	69.6	67.9	57.5	58.9
Location2	47.3	42.0	35.4	32.6
Mode for work	76.9	77.9	64.8	66.7
Mode other	58.3	52.1	52.8	49.5
Selection	79.1	79.2	72.4	71.6
Start time	47.7	38.0	39.8	35.4
Trip chain	83.1	82.3	83.3	80.9
With whom	57.7	53.4	50.9	48.4
Average	**62.3**	**59.3**	**55.4**	**53.6**

than for CHAID. However, given the better predictive performance of the Bayesian network models, this test seems to suggest that Bayesian networks are better suited to cope with the complexity of the decision making process for each agent. The next section examines whether this conclusion can also be reached at the level of activity patterns.

4.4.3 ACTIVITY PATTERN LEVEL

As mentioned in section 2.2.3, a learning algorithm needs to be converted to the decision table formalism to evaluate results at activity pattern level, because this facilitates the internal operation within Albatross. For rules this is straightforward, but to derive a ruleset from a Bayesian network model, it requires somewhat more explanation. Figure 4.10 starts from the pruned networks of Figure 4.9 and summarizes the conversion procedure. In the middle part of the figure, evidences are entered for every independent variable, resulting in a probability distribution of the target variable. This process is repeated for every possible combination of states (of independent variables). As already mentioned before, the direction of the arcs is preferably interpreted as an association rather than as a causality relationship. This means that not only child nodes but also parent nodes can influence the probability distribution of the dependent variable. For this reason, evidences need to be entered for every independent variable, regardless of whether these variables are child or parent nodes. By doing this, we get a "full model", that is an enumeration of all possible combinations of states in the Bayesian network model. This means that the number of rules that are derived from the network is fixed and can be determined in advance for a particular network (i.e. per dependent variable). This number is equal to every possible combination of states (values of the condition variables). Therefore, the total number of rules, which has to be derived from the network shown in Figure 4.10 is equal to 5*7*2*4=280, assuming that the with whom attribute is taken as the class attribute. In these cases, the concept of d-separation was ignored in the determination of the rulesets, which means that in the ideal theoretical case, the total number of rules can still be reduced.

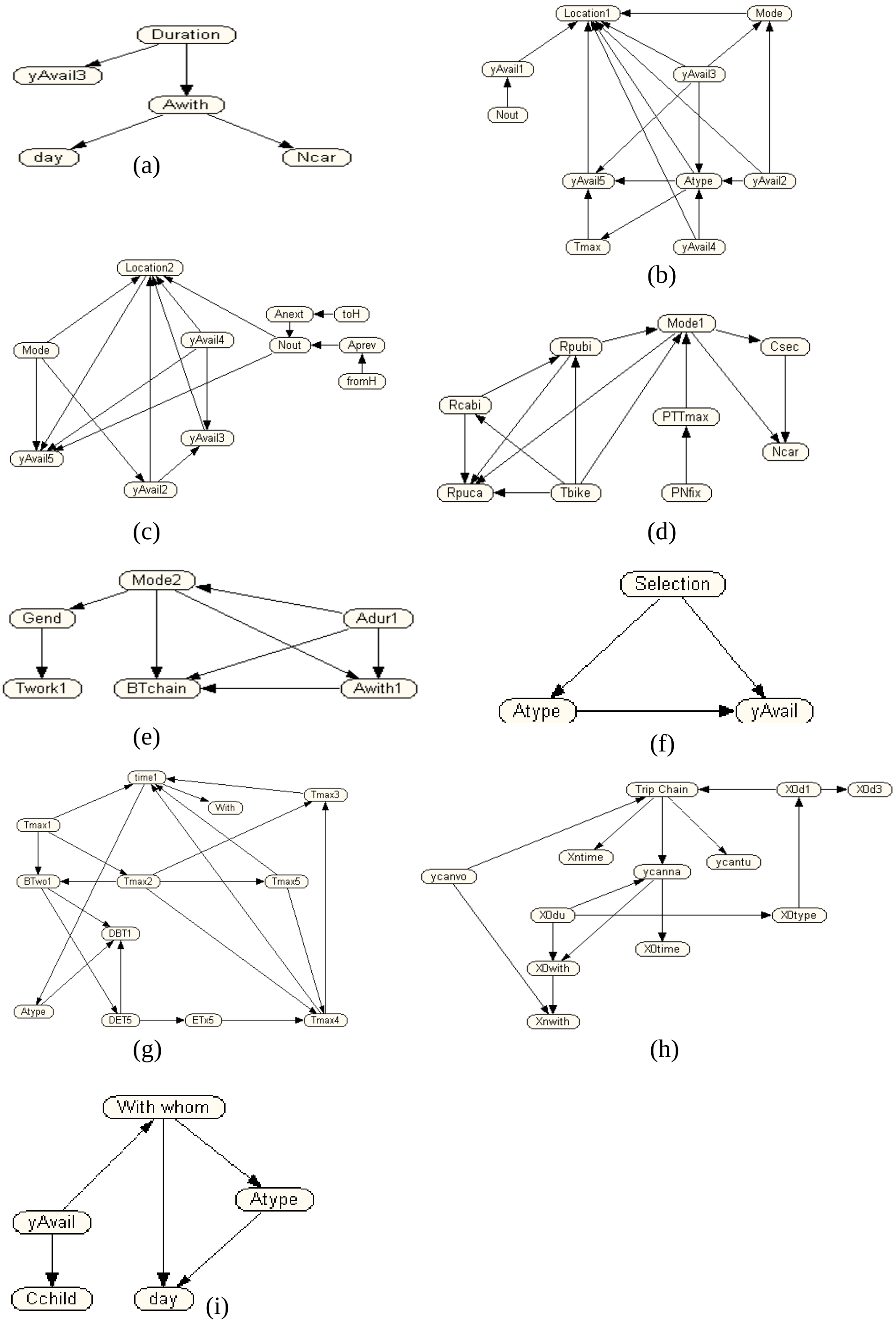
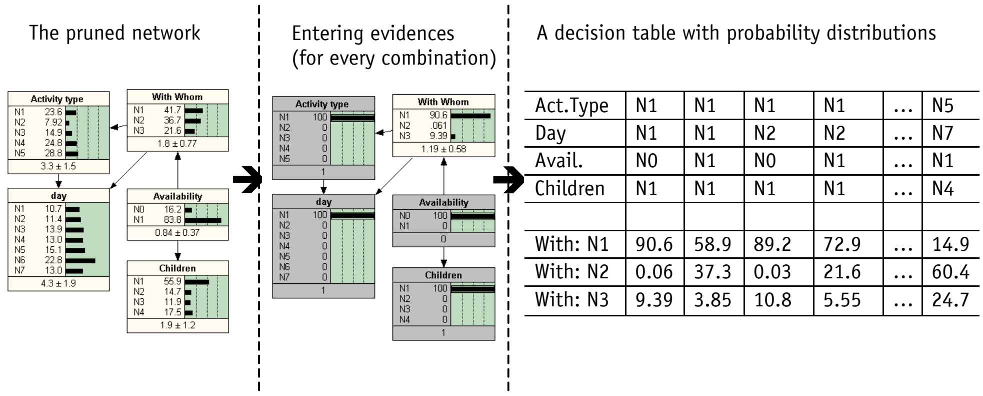


Figure 4.9: Bayesian networks for every decision agent of the Albatross model



Act.Type	N1	N1	N1	N1	…	N5
Day	N1	N1	N2	N2	…	N7
Avail.	N0	N1	N0	N1	…	N1
Children	N1	N1	N1	N1	…	N4
With: N1	90.6	58.9	89.2	72.9	…	14.9
With: N2	0.06	37.3	0.03	21.6	…	60.4
With: N3	9.39	3.85	10.8	5.55	…	24.7

Figure 4.10: Converting a Bayesian network to a decision table format

Converting Bayesian networks to a decision table formalism brings about the undesirable property of combinatorial explosion of the decision rules. Indeed, one may end up with more decision rules than data entries and this an enormous amount of overkill, especially because most decision rules will be redundant and will never be "fired". Every case in the training and test set is thus covered by one decision rule, and the option of appealing to the default class for making a prediction is non-existent. This problem is not present as such at the choice facet level, because data cases are there used as input for "querying" or predicting the dependent variable; while at pattern level, every single possible decision rule is derived from the network. Also, every rule contains the same number of condition variables. For the example shown in Figure 4.10, this number is equal to 4.

The SAM distance measures, indicating the predictive performance at activity pattern level on the training and on the test set, are presented in Table 4.5. It can be seen from this table that while the test set results of BN are still better than the other learning mechanisms that were evaluated before (CBA, Adapted CBA-1,2, CHAID), the discrepancy between the training and test set (overfitting) is large. This finding is especially problematic when the derived networks are used for prediction in another study area. The large number of decision rules in the decision unit is one possible explanation for this. In any case, it is clearly a sub-optimal solution, not only because some of the rules will never be used, but also because the large (fixed) number of conditions does not favour the interpretation and complexity of the model. We will detail on this in section 4.5.3.

Table 4.5: Benchmarking results at Activity pattern level

SAM distance measure	BN		CHAID	
	Train	Test	Train	Test
SAM activity-type	0.061	2.412	2.861	2.801
SAM with whom	0.062	2.814	3.225	3.210
SAM location	2.120	2.735	3.181	3.148
SAM mode	0.063	3.414	4.599	4.587
UDSAM	2.366	14.118	16.725	16.629
MDSAM	1.584	7.298	8.457	8.427

4.4.4 TRIP MATRIX LEVEL

The trip matrix level results (see Table 4.6), comparing BN with the CHAID algorithm, show a similar discrepancy between the training and the test set results than at activity pattern level. It can also be seen from this table that the training set results are better than for CHAID decision trees while the test set results are worse. The amount of overfitting is consistent with what was found at activity pattern level.

Table 4.6: Benchmarking results at Trip matrix level

Dimension	BN		CHAID	
	Train	Test	Train	Test
None	0.964	0.901	0.954	0.939
Mode	0.955	0.821	0.877	0.846
Day	0.971	0.928	0.960	0.948
Primary Activity	0.935	0.818	0.890	0.832

4.4.5 DISCUSSION

The previous sections have quantitatively illustrated the use of Bayesian networks within the Albatross model. Consistent with was done for the CBA algorithm, a more qualitative analysis, including a discussion of the variables which appear most frequently in CHAID and Bayesian networks, will be provided in section 4.6. For an evaluation at choice facet level, every case was presented as a combination of evidences to the empirically derived Bayesian network model. The results at choice facet level show that Bayesian networks generate better predictive performance than CHAID. However, when analyzing the performance of the technique at pattern and trip matrix level, the amount of overfitting was considerably larger for these data than other techniques that were evaluated before. However, the overall finding is the same as in Chapter 3, i.e. using descriptive learning for classification goes fairly well at choice facet level, but the approach looses some of the good performance at pattern and trip matrix level. To this end, and consistent with what was done in Chapter 3, the idea was conceived in the remainder of this chapter to evaluate whether an adaptation to the technique of Bayesian networks can solve these problems.

4.5 TOWARDS A NEW CLASSIFIER: INTEGRATING BAYESIAN NETWORKS AND DECISION TREES

4.5.1 PROBLEM STATEMENT

In the previous sections, it was shown that each decision rule that is used for predicting a particular dependent variable within a network contains the same number of conditions, resulting in potential sub-optimal decision-making. Second, the interpretation of the rules may be an issue. It should be realized that Bayesian networks may link more variables in sometimes complex (see Figure 4.5), direct and indirect ways, making interpretation more problematic.

The idea is proposed in this section to examine the possibility of combining the advantages of decision tree induction in terms of understanding and simplicity with the advantages of Bayesian networks in terms of accuracy. Consequently, a novel classification technique arises that integrates decision trees and Bayesian networks. In the integrated classifier, the idea is proposed to derive a decision tree from a Bayesian network (that is build upon the original data) instead of immediately deriving the tree from the original data. The new heuristic is referred to as a Bayesian Network Augmented Tree (BNT) in the remainder of this section.

In addition to reducing the number of rules and enhancing model comprehensibility, the proposed integration has another advantage. Despite the huge popularity of decision trees, it is a well-known deficiency that the model structure of decision trees can sometimes be instable (see also Breiman, 1996; Bloemer *et al.*, 2003). The reason for this is known as “variable masking”, i.e. if one variable is highly correlated with another, then a small change in the sample data (given several tests) may shift the split in the tree from one variable to another. Even if different decision trees that suffer from variable masking can perfectly arrive at the same final decision, the problem raises questions towards the stability and interpretation of the tree, which can be particularly important for policy makers. To this end, integrating Bayesian networks and decision trees can contribute to a more stable decision tree structure, because the variable correlations have already been taken into account in the Bayesian network, which may reduce the variable masking problem. Parts of this section are based upon work reported in Janssens *et al.* (2005f). To the best of our knowledge, the idea to build decision trees in this way has not been explored before in previous studies.

4.5.2 THE BNT CLASSIFIER

In order to derive a decision tree from a Bayesian network (that is build upon the original data), the information that is contained in the network (that is both the structure of the network and the distributions), is used for building a decision tree. The integration itself is described below.

In order to select a particular decision node in the BNT classifier, the mutual information value that is calculated between two nodes in the Bayesian network is used once more (see definition 4.14). This mutual information value is to some extent equivalent with the entropy measure that C4.5 decision trees use. It was defined as the expected entropy reduction of one node due to a finding (observation) related to the other node. As explained before, the expected reduction in entropy of the dependent variable can be calculated for the various findings variables. Next, the finding variable that obtains the highest reduction in entropy is selected as the root node in the tree. To better illustrate the idea of building a BNT classifier, the reader may consider again the network that was shown in Figure 4.1 by means of example. The dependent variable in this network was “Mode choice” and the different finding variables were “Driving license”, “Gender” and “Number of cars”. In a first step, the expected reduction in entropy between the “Mode choice” and the “Gender” variable can be calculated as follows:

$$I = \mathrm{P}(\mathrm{Mode}_{\mathrm{bike}},\mathrm{Gender}_{\mathrm{male}})*\log\frac{\mathrm{P}(\mathrm{Mode}_{\mathrm{bike}},\mathrm{Gender}_{\mathrm{male}})}{\mathrm{P}(\mathrm{Mode}_{\mathrm{bike}})\mathrm{P}(\mathrm{Gender}_{\mathrm{male}})}+$$

$$P(Mode_{car}, Gender_{male})*\log\frac{P(Mode_{car}, Gender_{male})}{P(Mode_{car})P(Gender_{male})}+$$

$$P(Mode_{bike}, Gender_{female})*\log\frac{P(Mode_{bike}, Gender_{female})}{P(Mode_{bike})P(Gender_{female})}+$$

$$P(Mode_{car}, Gender_{female})*\log\frac{P(Mode_{car}, Gender_{female})}{P(Mode_{car})P(Gender_{female})}$$

The calculation of the joint probabilities P($Mode_i$, $Gender_j$) for *i*={bike, car} and *j*={male,female} is the same as explained in section 4.2.3-subsection parameter learning. The calculation of the individual prior probabilities P($Mode_i$) and P($Gender_j$) is straightforward as well. As a result, the expected result of the formula above is: *I* (Mode choice,Gender) =

$$0.372*\log\frac{0.372}{0.506*0.75}+0.378*\log\frac{0.378}{0.494*0.75}+0.134*\log\frac{0.134}{0.506*0.25}+$$

$$0.116*\log\frac{0.116}{0.494*0.25}=0.00087.$$

In a similar manner, *I* (Mode choice, Driving License) = 0.01781 and *I* (Mode choice, Number of cars)=0.01346 can be calculated.

Since *I* (Mode choice,Driving License) > *I* (Mode choice, Number of cars) > *I* (Mode choice, Gender); the variable Driving License is selected as the root node of the tree (see Figure 4.11). Once the root node has been determined, the tree is split up into different branches according to the different states (values) of the root node. To this end, evidences can be entered for each state of the root node in the Bayesian network and the entropy value can be re-calculated for all other combinations between the findings nodes (except for the root node) and the query node. The node which achieves the highest entropy reduction is taken as the node which is used for splitting up that particular branch of the root node. In our example, the root node "Driving License" has two branches: Driving License=yes and Driving License=no. For the split in the first branch (Driving License=yes), only two variables have to be taken into account: "Number of cars" and "Gender". The way in which the expected reduction in entropy is calculated is the same as shown above, except for the fact that an evidence needs to be entered for the node "Driving License", i.e. P(Driving License=Yes; Driving

License=no)=(1;0) (since we are in the first branch). The procedure for doing this was already repeatedly shown before. Again, I (Mode choice, Gender)=0.02282 and I (Mode choice, Number of cars)=0.07630. Since I(Mode choice, Number of cars)>I((Mode choice, Gender); the variable "Number of cars" is selected as the next split in this first branch. Finally, the whole process then becomes recursive and needs to be repeated for all possible branches in the tree. A computer code has been established to automate the whole process. The final decision tree for this simple Bayesian network is shown in Figure 4.11.

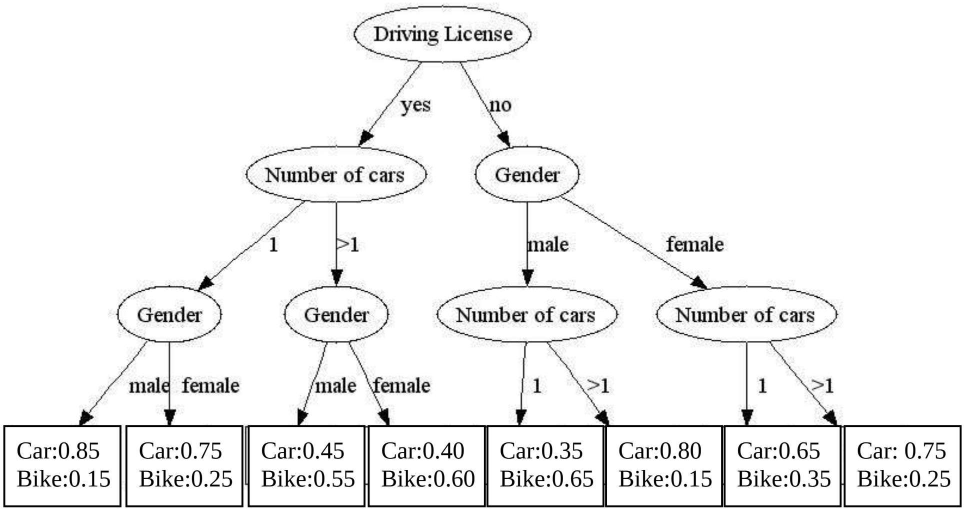


Figure 4.11: The final integrated BNT decision tree classifier (example)

4.5.3 CHOICE FACET LEVEL

The accuracy percentages that indicate the predictive performance for BNT on the training and test sets within Albatross are presented in Table 4.7. It can be seen from this table that the accuracy percentages of BN and BNT are similar.

Obviously, using Bayesian networks as the underlying structure for building the decision trees did not significantly deteriorate the predictive performance. These results illustrate that the idea of integrating Bayesian networks and decision trees holds out considerable promise in terms of predictive accuracy. It was also mentioned in section 4.5.1 that the BNT approach may result in a more stable

Table 4.7: Benchmarking results at choice facet level

Decision Agent	Decision making based on Bayesian networks		Decision making based on integrated BNT classifier	
	Training Set (%)	Validation Set (%)	Training Set (%)	Validation Set (%)
Duration	40.9	40.5	41.0	40.2
Location 1	69.6	67.9	69.4	68.5
Location 2	47.3	42.0	47.3	41.9
Mode for work	76.9	77.9	77.0	78.3
Mode other	58.3	52.1	58.3	52.1
Selection	79.1	79.2	79.1	79.2
Start time	47.7	38.0	42.3	39.3
Trip Chain	83.1	82.3	83.1	82.5
With Whom	57.7	53.4	57.7	53.5
Average	*62.3*	*59.3*	*61.7*	*59.5*

decision tree structure due to the fact that the variable correlations are already taken into account in the Bayesian network. In order to quantitatively assess this effect to some extent, a 10-fold cross-validation method has been used, where the data set is typically split into 10 mutually exclusive folds of nearly equal size. The developed model is then trained 10 times, each time using 9 folds for training and the remaining fold for evaluation. The cross-validation performance estimate is then obtained by averaging the 10 validation fold estimates found during the 10 runs of the cross-validation procedure. As a result of this, multiple training models were built and the variance (standard error) of the average estimate can serve as a quantitative measure for the stability of the BN and BNT models. Average results and standard errors (in parenthesis) on the test set have been reported in Table 4.8. In this table all standard errors are below 0.6 percentage, which is a low deviation. This can for instance be seen when the standard error is deducted from the average accuracy. In this case, the result is still well above the accuracy that has been obtained by CHAID, see Table 4.4. Therefore, it is reasonable to conclude that these results contribute and support the argument of model stability.

Table 4.8: 10-fold crossvalidation results (validation set)

Decision Agent	Decision making based on Bayesian networks (%)	Decision making based on integrated BNT classifier (%)
Duration	40.3 (0.3)	40.6 (0.4)
Location 1	67.5 (0.4)	68.7 (0.5)
Location 2	41.7 (0.3)	41.7 (0.4)
Mode for work	77.6 (0.3)	78.2 (0.3)
Mode other	52.1 (0.2)	52.2 (0.3)
Selection	79.2 (0.3)	79.2 (0.5)
Start time	38.4 (0.5)	39.3 (0.5)
Trip Chain	81.9 (0.6)	82.9 (0.4)
With Whom	53.1 (0.4)	53.4 (0.4)
Average	59.1 (0.367)	59.6 (0.411)

However, our experiments only lead to an unambiguous added value if the interpretation of decision rules derived from BNT is superior to the interpretation of decision rules that are derived from Bayesian networks. On the one hand, model complexity can be approximated by the total number of decision rules that is derived from a model. For Bayesian networks, this total is equal to the product of the number of possible states per variable (see section 4.4.3). For decision trees, the total number of rules equals the total number of leaves in the tree. However, while this may give an idea about the complexity of the full model, it does not give any indication about the ease of interpretation of a single decision rule. This latter form of individual rule complexity can be measured quite easily for Bayesian networks as it can be approximated by the number of independent variables which are present in the network, because every independent variable is used in every decision rule. For decision trees, the complexity of the derived decision rules can be approximated by the "depth" of the decision tree. The depth of a decision tree is equal to the number of levels that occur in a decision tree. Indeed, if the structure of the decision tree is rather flat, the ease of understanding of an individual decision rule is easy, since the number of independent variables that is used for predicting the dependent variable is limited. As a result of this, the understanding of the joined impact of these variables on the dependent variable is facilitated.

In Table 4.9, an indication is given about the model complexity in terms of number of rules and in terms of the complexity of every single rule. It can be seen from this table that the BNT classifier significantly improves the individual rule complexity in terms of the number of independent variables that is used in the decision rules. While it remains difficult to analyse the joined impact of several independent variables, it is still possible for relatively low numbers (let's say at most 5 or 6), and it will become almost totally incomprehensible for higher numbers. A significant achievement is obtained in this respect for the facets "start time", "trip chain", "location2" and "mode for work". The reader should also note that the depth of the decision tree that is shown in the third column of Table 4.9, indicates the maximum number of levels in the tree. This means that for most branches in the tree, this maximum number will not be achieved and less independent variables will be used in the decision rules, making comprehension easier than in Bayesian networks. In addition to this, and as mentioned before, this is not the case for Bayesian networks, since the number of independent variables in every decision rule is constant for every decision agent (see second column in Table 4.9). It would be possible however, to adopt some kind of pruning mechanism on the rules, by which the number of independent variables could be reduced. This was not done for two obvious reasons. First, the Bayesian networks that were used for prediction, as well as the BNT classifier which uses the networks as its underlying structure for calculating the splits in the tree, were already pruned networks (see section 4.4.2). Adding another post-pruning stage to the decision rules that are derived from these networks can potentially result in overpruned results, and it wipes out the original idea for using Bayesian networks, which is to analyse the joined impact of a large number of variables. Second, the rules that were derived from the BNT classifier were not pruned either, and this enables a fair comparison between both algorithms.

Table 4.9: Comparison of model complexity and individual rule complexity with respect to Bayesian networks and integrated BNT classifier

	Individual rule complexity		Model complexity	
Decision Agent	Bayesian networks (number of independent variables in network)	BNT ("depth" of decision tree, in maximum number of levels)	Bayesian networks (total number of rules)	BNT (number of leaves/number of nodes)
Duration	4	2	84	6/4
Location 1	9	7	5760	253/215
Location 2	10	6	9216	131/124
Mode for work	8	5	36864	187/64
Mode other	5	4	432	108/45
Start time	11	5	983040	210/58
Trip Chain	11	6	124416	384/175
With Whom	4	3	280	70/16

The right part of Table 4.9 describes the model complexity in terms of the total number of rules that is used in each decision agent. Based on Table 4.9, we have to conclude that the BNT classifier is a huge improvement in terms of model complexity over the Bayesian network approach and this for all decision agents. However, the opposite is obviously true in terms of computation time, as is shown in Table 4.10. As mentioned before, BNT relies upon BN for building a decision tree. As a result of this, the algorithm first needs to construct the BN in a first stage, and then only in a second stage derive the final BNT. This two-stage process obviously augments the computation time that is needed for BNT. The computation time that is shown in Table 4.10, is mainly determined by the number of variables in the dataset. A large number of variables, increases the likelihood of dependencies between variables and hence requires additional computation time for BN and BNT.

Table 4.10: Computation time (in seconds) of BN (structural learning) and BNT

Dataset	Duration	Location 1	Location 2	Mode for work	Mode other	Start Time	Trip Chain	With Whom
BN (seconds)	67.3	27.1	14.2	6.2	73.3	114.5	54.8	64.7
BNT (seconds)	82.5	72.2	55.4	41.8	92.4	167.3	110.4	76.8

4.5.4 ACTIVITY PATTERN LEVEL

The SAM distance measures, comparing the predictive performance at activity pattern level on the training and on the test set for BNT are presented in Table 4.11. The results of the BN approach were again added for the sake of clarity. While BN still achieved better performance on the test set than BNT, we have to conclude that the amount of overfitting is considerably smaller for BNT. On the other hand, it can be seen that the results of the BNT algorithm on the test set are in the same order as previous results that were obtained by CHAID, CBA and other previously tested classification systems.

Table 4.11: Benchmarking results at activity pattern level

SAM distance measure	BNT		BN	
	Train	Test	Train	Test
SAM activity-type	2.511	2.713	0.061	2.412
SAM with whom	2.915	3.010	0.062	2.814
SAM location	2.983	3.103	2.120	2.735
SAM mode	4.117	4.215	0.063	3.414
UDSAM	15.489	16.305	2.366	14.118
MDSAM	8.107	8.353	1.584	7.298

4.5.5 TRIP MATRIX LEVEL

Finally, the performance of BNT is evaluated at trip matrix level (see Table 4.12). Also, in this case, BN was added for the sake of clarity. The results at trip matrix level are consistent with previous results at other levels, i.e. a lower degree of overfitting, and performances that are in the same order of magnitude when compared to other previously tested classification systems.

Table 4.12: Benchmarking results at trip matrix level

Dimension	BNT		BN	
	Train	Test	Train	Test
None	0.946	0.938	0.964	0.901
Mode	0.877	0.853	0.955	0.821
Day	0.952	0.949	0.971	0.928
Primary Activity	0.865	0.839	0.935	0.818

4.6 CHAID, CBA, BN/BNT: QUALITATIVE ANALYSIS

The previous sections described detailed quantitative analyses about the performances of the BN and BNT classifiers. At choice facet level, comparisons were also made with the CHAID decision tree algorithm that is used in Albatross. Consistent with was previously done for CBA, the aim of this section is also to conduct the same comparison at choice facet level but now in a more explanatory and descriptive manner. We will discuss the most important variables that have been used in BN and compare them with variables that are used in CHAID and CBA. For BN (and BNT), all variables can be found in Figure 4.9 (a-i). Since BNT uses the same variables as BN, a discussion of the variables that have been used in BNT becomes superfluous.

Alternatively, the reduced variable set that was selected by BNT can also be used as input in a standard unpruned decision tree (C.4.5) which is then built upon the reduced dataset (i.e. which only consists of the selected set of variables). To this end, the same variables occur in both trees (when no pruning is assumed) and differences in performance are only the result of the different structure of the tree (i.e. the positioning of the variables in the level of the tree), because the unpruned C4.5 tree was built upon the reduced dataset, while BNT was built upon the BN structure. Some initial experiments were conducted to evaluate such an approach. Based on these first initial results, it seems that a decision tree that is built upon the reduced dataset differs from a decision tree that is built from a Bayesian network. This means that even if the same set of variables is used, most variables were positioned at another level in the tree. It should be noted here that comparisons between the decision tree structures are based on our own observation and impression, no measure has been used here to quantitatively support this impression. Apart from research which has been conducted within the domain of natural language processing trees (where Kernell methods are sometimes used to get a more reliable estimate about the degree of correspondence between two or more trees), little is known about quantitative measures for decision tree structure comparison.

It is clear that the observed difference needs to be ascribed to the underlying network structure (in BNT), because it explicitly incorporates dependencies and relationships in the DAG. BNT explicitly uses this information to build its decision tree (see section 4.5.2), while a normal tree that is built upon the data, does not.

The chronology of the remainder of this section is similar as described in previous quantitative analyses. A description of the variables is often provided with the variable name, but for the more detailed list, we refer again to Appendix B.

4.6.1 DURATION

With respect to the duration facet, Figure 4.9a shows that travel party (Awith), car availability (Ncar) and the day of the week (Day) and the availability of the 'long' duration class (yAvail3) occur in the BN/BNT model. In the CHAID based decision tree, the scheduled work time (Two), the travel party dimension (Awith), day (Day), car availability (Ncar) and the type of activity (Atype) have an important impact on the duration choice. It can therefore be concluded that there is a fairly high degree of correspondence between the CHAID based decision tree and the BN/BNT model. This is also reflected at a more quantitative level, where only a small improvement of +1.4% was achieved in favour of BNT and +1.7% in favour of BN.

4.6.2 LOCATION1

It can be seen in Figure 4.9b that with respect to the location1 decision facet, the BN/BNT model is on the one hand determined by the previous facets of the Albatross model, which is the Activity type (Atype) and the transport mode (Mode). Furthermore, the availability of choice locations given the schedule (yAvail), the number of out-of-home activities (Nout) and the maximum available time in the schedule position of the activity (Tmax) are important variables for the BN/BNT model. In fact this set of variables is completely analogue to the variables that were considered most important for CBA-1 and CBA-2. When compared to the CHAID based approach, there are only two equally most important variables. When the above observations are coupled with the good performances of CBA-1,2, and with the finding that BN and BNT respectively outperformed CHAID by 9% and by 9.6%, it is safe to assume that the CBA-1,2, BN and BNT algorithms selected the best variables for achieving a high degree of accuracy on the test set.

4.6.3 LOCATION2

For the location2 facet (see Figure 4.9c), there are some variables that were already found important by BN and BNT at the location1 dataset (Mode, yAvail$_t$ (t=2,..5) and Nout). Also the variable which indicates whether a trip ends at home or not (toH) and the type of the previous activity (Aprev) were found important by BN and BNT. In fact, all these 8 (out of 10) variables were also considered most important for CBA and CBA-2. The two remaining variables are the type of next activity (Anext) and the toH counterpart, fromH. When compared with CHAID, there are only two equally most important variables (Mode, Nout). Also in this case, when these findings are coupled with the good performances of CBA and CBA-2 at the location2 level, and with the finding that BN and BNT respectively outperformed CHAID by 9.4% and by 9.3%, it can be concluded that CBA, CBA-2, BN and BNT algorithms selected the best variables for achieving a high degree of accuracy on the test set.

4.6.4 MODE FOR WORK

Also for the “mode for work” decision facet, a similar conclusion can be reached as for the location1 and location2 decision facets. It can be seen from Figure 4.9d that the ratios car/bike travel time (Rcabi), the public transport/bike travel time (Rpubi) and the public transport/car travel time ratio (Rpuca), were identified as important variables by the BN/BNT model. Also important are general descriptive variables (Csec and Ncar) and the objective travel time to the work location (Tbike). All 6 (out of 8) variables were also considered most important for CBA.

The two remaining variables are partner related variables such as the Number of out-of-home activities (Pnfix) in the schedule of the partner and the maximum bike travel time across activities in the schedule of the partner (PTTmax). When compared with CHAID, only 3 of these variables (Rpuca, Tbike, Ncar) can also be found at an important level in the tree. Once more, when these findings are coupled with the good performances of CBA, CBA-1,2 at this facet, and with the finding that BN and BNT respectively outperformed CHAID by 11.2% and by 11.6%, it can be concluded that CBA, CBA-1,2, BN and BNT algorithms all selected better variables than CHAID did.

4.6.5 *Mode Other*

With respect to the “mode other” facet, Figure 4.9e shows that the duration (Adur1) and the travel party (Awith1) of the first activity in the tour, the begin time of trip chaining (Btchain), the gender (Gend) and the time for a work1 activity (Twork1) have an important impact on the “mode other” decision facet. When compared with CHAID and CBA there is a very low level of correspondence in the selected variables. For this reason, the good results that we were able to obtain by the CBA algorithm could not be achieved by BN/BNT. The latter two approaches both achieved a fairly small improvement of 2.6%. Therefore, we can conclude that the variables that were selected by CBA probably better represented the decision behaviour for the “mode other” decision facet.

4.6.6 *Start Time*

In Figure 4.9g it is shown that at least 11 variables were found to have a significant impact on the start time decision. The first set of variables ($\text{Tmax}_{t,}$ *t*=1,..,5) represents for each time interval, the available time in the current schedule. Important to notice is also that the travel party decision seems to be important with respect to the start time decision. Furthermore, a variable which indicates whether there is a work activity with start time in time interval 1 of the schedule (“Btwo1”), a variable which indicates whether there is an out-of-home activity with end time in time interval 5 (“ETx5”) and the type of the activity (“Atype”), are important for start time decision making. Finally, factors dealing with saved travel time (“DBT1” and “DET5”) were found to be significant by the BN/BNT.

When compared with CBA/CHAID, the level of correspondence is rather low (4 of the most important variables in BN/BNT also appear in CBA). In addition to this, a high degree of overfitting can be observed for BN (Table 4.4). However, when compared to BNT, the degree of overfitting is considerably lower. Despite the fact that an improvement of 2.6% could be established for BN and 3.9% for BNT, and given the rather small degree of correspondence when compared to CHAID and CBA, the level of improvement is still insufficient to proclaim that BN and BNT consistently selected better variables than CBA and CHAID.

4.6.7 Trip Chaining

Also for the "Trip Chaining" decision facet, a similar conclusion can be reached. It can be seen from Figure 4.9h that also in this case a large number of variables were found significant. However, neither the CBA algorithm, nor BN/BNT significantly outperformed CHAID. BN and BNT achieved an improvement of +1.4% and +1.6%, but this level is too low to proclaim that one algorithm consistently selected better variables than the other algorithms under consideration.

4.6.8 With Whom

The last decision facet is the With Whom facet. Unlike in most previous comparisons, there is quite a large degree of correspondence between BN/BNT and CHAID's most important variables. In CHAID, Day, Cchild, Ycar4 and Atype are important, while three out of four of these variables also appear in the BN model in Figure 4.9i. The fairly good improvement of BN (+5.0%) can still be attributed to the different nature of the technique. However, CHAID and BNT are both decision trees and for a better understanding of the better performance of BNT (+5.1%), a further analysis like the one described at the beginning of section 4.6 is desirable. Similar to what was mentioned before, it could also be seen in this case that even if the same set of variables is used, most variables were positioned at another level in the decision tree. It is clear that this observed difference needs to be ascribed to the underlying network structure (in BNT), because it explicitly incorporates dependencies and relationships in the DAG. BNT explicitly uses this information to build its decision tree (see section 4.5.2), while a normal tree that is built upon the data (such as CHAID), does not.

4.7 Conclusion

It was assumed in this chapter that Bayesian networks are well suited for identifying and capturing complex relationships between a set of factors that cause a particular transport behaviour. For this reason, the idea within this chapter was to examine their performance within activity-based transportation modelling. The technique is also unsupervised and descriptive in nature and can thus contributed to a more comprehensive overview about how supervised and unsupervised learning can be integrated in the context of the Albatross model and in addition to the CBA technique that was described in Chapter 3.

Bayesian networks are often used for querying and for making advanced what-if-analyses, which makes that they are probably also better suited for reasoning and for explanatory purposes than CBA. By means of an empirical application, it has been shown how the technique can be used to evaluate and reason about the choice processes that form transport mode decisions. In addition to the analysing capabilities, Bayesian networks were obviously also used for classification and prediction within the Albatross model. It was found that the BN model generated better predictive accuracy results at choice facet level than the CHAID decision tree approach. In some cases, the improvements were highly significant. However, when looking at the results at pattern and trip matrix level, the overall finding is the same as in Chapter 3: using descriptive learning for classification goes fairly well at choice facet level, but the technique looses its very good performance at pattern and trip matrix level because of high overfitting. To this end, and consistent with what was done in Chapter 3, it was evaluated whether an adaptation to the technique of Bayesian networks could reduce the size of the decision rule set and improve predictive performance accordingly. Bayesian networks were used as the information source for deriving a complete decision tree (BNT), instead of relying on the original data for doing this. At choice facet level, the predictive performance of both BN and BNT were comparable. However, the improvement was especially important with respect to the understanding and interpretation of the individual rule set. BNT was able to significantly improve the individual rule complexity in terms of a reduction of the number of independent variables per rule. The exercise especially paid off at pattern and trip matrix level where a lower degree of overfitting was established when compared to BN. The chapter was concluded with a more qualitative and descriptive analysis by a discussion of the most important variables that appeared in BN/BNT. Variables were also compared with CHAID and with CBA. It was discussed in that section that the level of correspondence between BN/BNT and CBA was rather high, while only for few datasets a high similarity was found between BN/BNT and CHAID. The better performance has lead us to believe that BN/BNT and CBA selected better variables for some of these datasets. However, initial additional analyses also seemed to suggest that even if the same set of variables is used in a traditional decision tree algorithm and in BNT, most variables were positioned at another level of the tree. This confirmed the variable masking problem and the instability of decision tree structures, which were identified as one of the most important reasons to integrate BN and decision trees. In order to evaluate the degree of stability of BN and BNT, 10-fold cross validation experiments have been conducted and promising results of these experiments have been reported. Given the good performance of BNT, an important topic for future research is an evaluation of the technique on more datasets and on different application domains.

Chapter 5
The Identification, Segmentation and Prediction of Sequential Dependencies in Activity-Diary Data

5.1 INTRODUCTION

It was already mentioned in Chapter 1 of the dissertation that the process of building, testing and applying a particular model is fairly similar in both simulation and activity scheduling models. That is, both approaches aim to predict full activity-travel patterns, along with all the typical activity-based facets (when, where, which activity, etc). The most obvious difference is the fact that simulation approaches are mostly driven by the data itself and by the structures and relationships which are incorporated in the data (and often also rely upon probability distributions), while activity scheduling models often make additional assumptions to find the best representation and reproduction of activity-travel patterns. Let us take the Albatross system as an example, where the prediction of every facet of the decision unit, has been steered by several independent variables (see Appendix B). However, it is not sure what the influence of other explanatory variables is on these different decision outcomes. Another example are the Scheduling and the Inference Engine of the system which are not fully data-dependent (Arentze and Timmermans, 2000). The system takes thus domain knowledge or system-defined knowledge into account that goes beyond the pure extraction of knowledge from the data. Other differences and examples of existing micro-simulation models were already introduced in section 1.1.4-subsection simulation models of Chapter 1.

In this second part of this dissertation, the development and the first empirical results of a new data-driven simulation procedure will be reported. The presented approach is almost completely data-driven, and assumes few additional assumptions. The major a-priori made assumption is related to the prediction order of the different decision facets. That is, the algorithm first predicts activity and transport mode dimensions, and these dimensions are only at a later stage complemented by time and location facets. There is no general consistency or agreement within the literature which prediction order needs to be followed, or in other words, which dimension uniquely determines the other dimensions. The main reason for this is that all dimensions are highly interconnected, and the correct prediction order can probably only be approximated by sufficient empirical benchmarking results.

There were several reasons for undertaking the simulation research effort that is described in this second part of the dissertation. First, the idea was to examine whether (data-driven) simulation is capable of adequately representing/ replicating activity travel behaviour. To this end, the outcome of Chapters 5 and 6 of the dissertation are used as input in Chapter 7, where an initial attempt has been made to empirically compare the Albatross activity scheduling model with our developed simulation effort. Both research areas have not yet been empirically examined before. Second, having advanced unsupervised (descriptive) learning in an activity scheduling model in Chapters 3 and 4, the same approach was adopted with respect to the area of (data-driven) simulation in this part of the dissertation. To this end, unsupervised (descriptive) learning algorithms will be used in Chapters 5 and 6. Third, the research problems that needed to be addressed to replicate activity-travel patterns, enabled us to come up with several methodological contributions to the current state-of-the-art.

The most important insight that differentiates our model from other existing simulation (and activity-scheduling) models, is the fact that they do not account for sequential information and sequential dependencies in the identification of representative activity patterns. Apart from a study by Kitamura *et al.* (2000), sequential information has not been taken explicitly into account in simulation modelling. Kitamura (2000) has claimed that there are reasons to believe that behaviour is path dependent on the past behavioural trajectories of the respective individuals. There have been several studies that examine the role of state dependence in activity and travel behaviour from temporal perspectives. Applications include studies by Kitamura and Kermanshah (1983), Goulias and Kitamura (1997) and Kasturirangan *et al.* (2002). These studies confirm that the choice of activity is dependent on the preceding activity engagement. Also in a more intuitive interpretation, there is little doubt that sequential information is omnipresent in activity diary data. To give a simple example: during one particular day, it is highly probable that the combination have breakfast, travel and working occurs frequently together in a diary, since people obviously first have breakfast during the morning and then need some kind of transportation to arrive at their work location. There are many other sequence pair combinations that can be revealed in this way. In this chapter, we will show that activities and travel are sequentially correlated in activity-diary data. The main objective for identifying this sequential information is to come up with a pattern of activity-travel combinations that are used as the basis for simulating other dimensions such as location and time information (see Chapter 6). The reader may notice that this approach differs from the more common “skeleton” approach in activity scheduling models, where a skeleton is defined as the subsequence of fixed activities. In our context, the simulation effort needs to be interpreted as the sequence of activities and transport modes that have been identified as being sequentially correlated during a time period (i.e. a full day), regardless of the character (fixed or flexible) of the activity.

The remainder of this chapter is organized as follows. First, we will briefly elaborate on the research that has been done in the past to identify sequential dependencies in data. In that section, our choice for transition matrices will be motivated, as well as their formalization and use within Markov Chain modelling. Additional statistical significance tests will also be introduced to allow for testing both (i) the preconditions to use a first-order and/or higher-order transition matrices as they are used in Markov Chains and (ii) the stationarity condition. In section 3, we will show in a problem statement that there are three important drawbacks with respect to the use of (higher-order) Markov Chains in our simulation framework. The section also describes the need for developing modified techniques in the identification of more accurate transition probabilities. Section 4 introduces these algorithms. Furthermore, in section 5, a novel segmentation procedure has been proposed that is able to cluster sequential activity-travel combinations in terms of socio-demographic (modified decision-tree approach) and time information (bifurcation points). Sections 4 and 5 finalize the full “knowledge model”. The controlled simulation procedure which is used for generating new activity-travel sequences, based on this “knowledge model”, has been presented in section 6. In section 7, all the above sections have been tested and empirically validated by means of pattern- , trip- and activity-level performance indicators.

This chapter aims to propose advancements to the methodological state-of-the-art on several points (see also Janssens *et al.*, 2004c; 2004d; 2005c). One advancement is the introduction and modification of the computation of low- and higher-order transition probabilities as they can be used in Markov Chains and the development of a simulation framework in the area of transportation.

Also, a segmentation procedure will be introduced that enables one to cluster transition matrices in terms of time information (relaxation of the stationarity condition, see infra) by means of the technique of the identification of bifurcation points. Finally, a similar segmentation scheme has been developed in terms of socio-demographic information. This procedure uses a modified version of a decision tree, in the sense that sequential probability information can be used during induction and in the leaves of the tree as apposed to the traditional way of only using one single classification attribute (represented by one dependent variable).

5.2 SEQUENTIAL PATTERN RECOGNITION

5.2.1 PREFACE

Sequences have been the subject of research in many disciplines, among which archaeology (McBrearty, 1988), biology (e.g., DNA sequence analysis –Raftery and Tavaré, 1994; Lipman and Pearson, 1984), chemistry (Xu and Agrawal, 1996), computer sciences (Sabherwal and Robey, 1995), economics (Hopp, 1987), econometrics (Bollerslev *et al.*, 1992), history (Abbott, 1995), linguistics (Jonz, 1989), meteorology (Raftery, 1985; MacDonald and Zucchini, 1997), psychology (Cohen *et al.*, 1990) and sociology (Abbott and Hrycak, 1990; Katz and Proctor, 1959; Logan, 1981).

A sequence can be defined as a succession of events. An event is a transition from one discrete state to another, situated along a time continuum (Abbott, 1995). In this chapter, events represent activities that occur in a persons' diary. Traveling is considered as an activity as well, while transport mode is added as an additional attribute in this case (see infra).

Similar to the technique of association rules, sequence pattern recognition is mainly descriptive in nature. Most of the basic algorithms for sequential pattern mining are based on the Apriori algorithm that was already proposed in Chapter 3. A series of Apriori-like algorithms have been proposed for sequential association rule mining: AprioriAll, AprioriSome, DynamicSome in (Agrawal and

Srikant, 1995), GSP (Srikant and Agrawal, 1996b), SPADE (Zaki, 2001), FreeSpan (Han *et al.*, 2000) and Pre xSpan (Pei *et al.*, 2001).

Apart from sequential association rules, a lot of work has been done within the area of Markov Chains to represent information about sequential dependencies among events. Markov Chains are probabilistic models which were introduced by Andrej Andreevic Markov at the beginning of the 20th century. Their application domains have been numerous, including geography, biology, meteorology, music, and many others. For comprehensive treatments of Markov Chains and their applications see e.g. (Bharucha-Reid, 1960; Dynkin, 1965; Kemeny and Snell, 1976; Kemeny *et al.*, 1976; and Doob, 1990).

According to Abbott (1995), both sequence analysis types can be classified by the length of the considered succession of events. Methods focusing on low-order (e.g. first-order) combinations only consider one transition at a time. Markov models are the most popular methods for doing this. Gradually, higher-order dependencies can be taken into account. This gave rise to *n*-th order Markov models and also to sequential association rules. Whereas Markov models treat sequences step-by-step (i.e. transitions from one state to another discarding the sequence as a whole), sequential association rules treat them as whole units (Abbott, 1995; Prinzie and Van den Poel, 2005). The central issue is whether there are patterns in the sequences, either over the whole sequences or within parts of them (Srikant and Agrawal, 1996b). Consider for instance the frequent itemset {i_1, i_2, i_3, i_4} where i_1, i_2, i_3 and i_4 are respectively "sleep", "work", "travel by car" and "medical visit". By means of sequential association rules, it is then perfectly possible to have a sequential association rule as follows: IF i_1 THEN i_4; regardless of what happens in between. Accordingly, sequential association rules identify patterns where for instance i_4 temporally comes always after i_1 and thus consider the full sequence and not subsequence information.

In addition to this, another important difference is that items in a frequent itemset cannot re-occur. As a result, it is not possible to identify rules like IF i_1 and i_2 THEN i_1. Therefore, re-occurring items within one sequence such as for instance Sleep-Work-Sleep cannot be identified by means of traditional sequential association rules. Since re-occurring items are highly frequent in activity-travel patterns, sequential association rules, cannot be immediately adopted.

When translated to a transportation framework, two additional reasons in favour of using Markov Chains can be given. Based on the work that has been done by Bhat (1999; Bhat and Singh, 2000), Damm (1980) and Hamed and Mannering (1993), there is evidence that activity-travel behaviour can be subdivided into different time-windows (and thus favour subsequences and not full sequences). In addition to this, higher-order Markov Chains take into account what happened between two temporally separated events, in case of misfit of lower-order Markov Chains (Abbott, 1995). Finally, it has to be noted that, there is also a large family of latent variable models which have not been dealt with in this dissertation, that can be jointly used along with Markov Chains. In latent variable and latent class models, the categorical variables of behaviour are assumed to be an imperfect relection of another set of variables that are unobserved. These unobserved variables are called latent and their categories are called classes. The models can be integrated with Markov Chains and can for instance be used as Mixed Markov Latent Class models (MMLC) (see examples in Langeheine and van de Pol, 1990, Goulias, 1999) that have been used to describe stochastic processes in discrete space and discrete time. Markov Chains are also particularly well suited in the analysis of longitudinal (panel) data. It is claimed by Kitamura (2000) that a more coherent and accurate forecasting is possible through the use of longitudinal data, because they also capture dynamics of travel behaviour (Goodwin *et al.*, 1990), as opposed to the more commonly used cross-sectional data. Likewise, models and techniques that are capable of analyzing and predicting longitudinal data, such as for instance Markov Chains become more favoured.

Taking all these arguments into consideration, Markov Chains are considered to be a justified starting point that can assist in the identification of sequential dependencies for our application area. In the next subsection, the basic principles of Markov Chains will be introduced along with a brief discussion about how the technique can be applied in our simulation framework.

5.2.2 Markov Chains: Definitions and Conceptualisation

A transition matrix reveals information about the underlying structure of the data sequence and is in fact the core knowledge representation of Markov Chains. In this section, we first make a distinction between first- and higher-order transition matrices and then elaborate on their application in the context of our simulation framework.

First-Order Transition Matrices

In Markov Chains, the goal is to simulate (predict) a discrete random variable X_t, taking values in the finite state space {1,...,*m*}, as a function of the values taken by previous observations of this variable. In order to predict the value taken by X_t, the Markov property is used as a necessary condition in Markov Chains.

Definition 5.1: Markov property

The Markov property says that the present time *t*, can be entirely explained by the first lag (*t-1*), so that we can write:

$P(X_t = i_0 \ |X_0 = i_t, \ldots, X_{t-1} = i_1) = P(X_t = i_0 \ |X_{t-1} = i_1) = q_{i_1 i_0}(t)$, where $i_t, \ldots, i_0 \in \{1, \ldots, m\}$. ■

Considering all combinations of i_1 and i_0, we can now construct a transition probability matrix *Q*.

Definition 5.2: Transition probability matrix

A transition probability matrix typically looks like:

$$Q = \begin{array}{c|cccc|c}
 & & & X_t & & \\
X_{t-1} & 1 & \cdots & \cdots & m & \\
1 & q_{11} & \cdots & \cdots & q_{1m} & q_{1tot} \\
\vdots & \vdots & \ddots & & \vdots & \\
\vdots & \vdots & & \ddots & \vdots & \\
m & q_{m1} & \cdots & \cdots & q_{mm} & q_{mtot} \\
 & q_{tot1} & & & q_{totm} & N(= q_{tottot})
\end{array}$$

The matrix of transition probabilities provides a compact and unique description of the behaviour of a Markov Chain. Each element in the matrix represents the probability of the transition from a particular state (represented by the row of the matrix) to the next state (representing the column of the matrix). Assuming a fixed number of possible states and the stationarity condition (see infra), the transition to and from every state can be described by a single matrix. Each of the rows sums to 1. Finding this transition matrix *Q* and the initial distribution P(X_0=i_t) determines the Markov Chain, which can be used to calculate in generality for any time (*t*=1, 2, ..., *T*), the probabilities of being in one of the finite number of states {1,...,*m*}. ■

Definition 5.3: Transition(al) frequency matrix Q_{freq}

A transition frequency matrix (Q_{freq}) is similar to a transition probability matrix (definition 5.2), except for the fact that the entries in the matrix (n_{ij}) represent frequencies instead of probabilities. Any probability matrix can be derived from a transition frequency matrix. Row totals in a transition frequency matrix do no longer sum to 1. ■

Definition 5.4: Stationarity condition

A Markov Chain satisfies the stationarity condition if transition probabilities do not depend on the time *t*. It means that at whatever time point *t* the chain is looked at, transition probabilities are the same. ■

Usually the Chi-square statistic is employed on structural contingency tables, in order to determine if there is a dependence between 2 (or many) categorical variables. However, the statistic can also be applied on transition frequency matrices, which are in fact a special case of contingency tables (Bishop *et al.*, 1977; Gottman and Roy, 1990; Bakeman and Gottman, 1986), as it is done in the following two definitions.

Definition 5.5: χ^2 for testing stationarity

Stationarity can be tested by means of an "omnibus" method (Gottman and Roy, 1990) that divides a sequence (in this case activity-travel patterns) into *D* time periods, thereby yielding *D* subsequences. Then, transitional frequency matrices are computed for each time period (*d*) and a statistical chi-square test will compare the individual transition frequency matrices with the overall one. The expected transitional frequencies and the χ^2-statistic can be respectively computed as:

$$E_{ij}(d) = n_{itot}(d)\frac{n_{ij}}{n_{itot}} \forall i, j$$

$$\chi^2 = \sum_{d=1}^{D} \sum_{i=1,j=1}^{m} \frac{(n_{ij}(d) - E_{ij}(d))^2}{E_{ij}(d)}$$

The tested hypothesis is that the transitional probabilities are constant across time periods. It is expressed as:

H$_0$: $q_{ij}(d)=q_{ij}$, $\forall$ d=1,2,..., D

H$_1$: $q_{ij}(d)\neq q_{ij}$, $\forall$ d=1,2,..., D ■

Obviously, the most important question in sequence pattern recognition is whether the state of a system at a certain point in time t depends or is completely independent from what happened previously. Also in this case, a simple chi-square statistic has been developed (Bishop *et al.*, 1977, Everitt, 1992). The Chi-square statistic tests whether transitions are a first-order Markov process, that is, if the state of the system at time t depends on the state at time t-1.

Definition 5.6: χ^2 for testing a first-order Markov process

For a sequence of m states, there are mxm=m^2 first-order transition frequency entries, as represented by the general transition frequency matrix (see definition 5.3). These observed transition frequencies are compared with what would be expected if the state of the system at time t was independent from its previous state t-1. The best estimates are given by the maximum likelihood estimates (Bishop *et al.*, 1977; Everitt, 1992):

$$E_{ij} = \frac{n_{itot} \times n_{totj}}{N} \forall i,j \in \{1,...,m\}$$

These expected frequencies are then replaced in the well known χ^2 formula:

$$\chi^2 = \sum_{i,j} \frac{(n_{ij} - E_{ij})^2}{E_{ij}}$$

When using the χ^2 test, the following hypothesis can be made about the transitional frequency matrix:

H$_0$: X_t is independent from X_{t-1}, e.g. $n_{ij} = \frac{n_{itot} n_{totj}}{N}$

H$_1$: X_t is dependent from X_{t-1}, e.g. $n_{ij} \neq \frac{n_{itot} n_{totj}}{N}$

■

Alternatively, the likelihood ratio G² may be used for testing whether first-order sequential dependencies exist in data (Lemay, 1999).

Definition 5.7: G² for testing a first-order Markov process

$$G^2 = 2\sum_{ij} n_{ij} \log \frac{n_{ij}}{E_{ij}}$$

χ^2 and G² should have asymptotically the same behavior and should be close too each other for reliable results. ■

However, both tests should be interpreted with caution. A non-significant statistic may for instance show that there is no dependence between *t-1* and *t* transitions, but it does not rule out the possibility that sequential dependencies are present in the data. For instance, it is possible that the sequential information is of second-order, that is for instance a relationship between *t-2* and *t*.

Instead of testing the whole transition matrix, as it is done in definitions 5.6 and 5.7, it is certainly warranted in some research domains to test the statistical significance of specific transitions in the matrix. Especially, when the transition matrix is used for analysis and explanatory purposes, such significance tests are essential. However, in our application, the final aim is simulation and as a result, is less vulnerable to not significant sequence pair combinations since these may occur in real activity diaries as well. Tests which can be used are the simple test of proportions (Spiegel, 1980), Sackett's or Gottman's Z-score (Bakeman and Gottman, 1986), or Chi-square and G² statistics of individual transitions.

HIGHER-ORDER TRANSITION MATRICES

Definition 5.8: Higher-order transition probabilities

In general, a ℓ-th order Markov Chain can be specified with the transition probabilities: $P(X_t=i_0 \mid X_0=i_t, \ldots, X_{t-1}=i_1) = P(X_t=i_0 \mid X_{t-\ell}=i_\ell, \ldots, X_{t-1}=i_1) = q_{i_\ell \ldots i_0}(t)$

where $i_t, \ldots, i_0 \in \{1, \ldots, m\}$. ■

There are two different methods for calculating higher-order transition probabilities: the simple higher-order transition (frequency and probability) matrix and the *n*-grams method (Scholtes, 1991; Suen, 1979).

Simple higher-order transition matrices are built by considering the number of transitions from state *i* to *j*, *k* states before, without taking into account the states in-between. The method is easy to comprehend and is in fact similar to definition 5.1, with this difference that transitions from *t-k* to *t* are considered for every *k*-order transition matrix. The order of a transition matrix can be tested fairly easy by means of classical statistics. For continuous and binary variables, the (Pearson) autocorrelation coefficient can be used to quantify the degree of linear relationship between state *t-k* and *t* (Kendall and Ord, 1990; Gottman, 1981). For nominal variables, we need to rely on the Chi-square statistic to assess the order of the system (Lemay, 1999). Again, testing significance of the transitions may be performed on transitional frequency matrices at lag *k* using either the χ^2 or the likelihood ratio Chi-square G^2, as described in the previous section. The associated p-value will determine if the transitions depend on the specified lag *k*.

The second approach that deals with higher-order transitions is the *n*-gram approach (Scholtes, 1991; Suen, 1979). *N*-grams are formed by concatenating the last *n* states of the same variable into a single entity. Contrary to the previous method, the *n*-gram method takes into account intermediary states. For instance, when *n*=2, all possible combinations of consecutive states are concatenated into a single symbol; for a dual variable that is for instance 00, 10, 01, 11. When considering higher order *k*, we take into account pairs, integrating intermediary states. For instance, for a third-order gram (3-gram), possible states are 000, 001, 010, 011, 100, 101, 110, 111. A fruitful method for determining the order in this case is by means of information theoretical methods, and more specifically by means of entropy and conditional entropy. The rationale behind this method is that the information (as computed by the entropy) brought by past state symbols should significantly increase if the sequence really depends on past states. Only a few methods exist for analyzing the order of *n*-grams Markov Chain processes. The method that is introduced in this section is one by Attneave (1959), Gottman and Roy (1990). They suggest the following procedure for testing the significance of a higher-order dependence (*n*-gram) of a sequence.

Definition 5.9: Testing a higher-order Markov process (*n*-grams)

1. First, compute the entropy H_i for each *i*-gram.
2. Compute the entropy difference D_i, between the entropy of *i*-gram (H_i) and entropy of (*i-1*)-gram (H_{i-1}); that is $D_i = H_i - H_{i-1}$
3. Compute the difference between entropy differences, i.e.: $T_i = D_i - D_{i+1}$
4. Finally, compute $\chi^2 = 2N\log_e 2T_i$

Computationally, the procedure can be simplified since $T_i = D_i - D_{i+1}$, where $D_i = H_i - H_{i-1}$ and $D_{i+1} = H_{i+1} - H_i$. Therefore $T_i = 2H_i - H_{i-1} - H_{i+1}$

The hypotheses that are tested are stated as following:

$H_0(k)$: the process is a *k*-th order Markov Chain

$H_1(k)$: the process is a (*k+1*)-th order Markov Chain

Hypotheses are tested at the preferred α-level with $\chi^2 = 2N\log_e 2T_i$ (N is the length of the sequence). ■

Conceptualisation

Having explained a set of definitions and statistical tests that can be adopted within the context of Markov Chains, we are now able to detail on how the technique can be used for simulating activity-travel patterns.

To this end, we need to reconsider the discrete random variable X_t, taking values in the finite state space $\{1,\ldots,m\}$. Each value in this set is an activity that occurs in a persons' activity pattern. Travelling is considered as an activity as well, however transport mode is added as an additional attribute in this case (see also Arentze *et al.*, 2001). Alternatively, only activities may be used to identify sequential dependencies in the data. However, the clear disadvantage of such an approach is that the sequential dependencies with respect to the transport mode facet, for example (different transport modes) during different times of day or (different transport modes) after very specific activities, cannot be taken into account. For this reason, the parameter *m* contains the total of non-travel activities and transport modes that occur in an activity pattern.

Our goal is now to simulate (predict) (see section 5.6) the value taken by X_t as a function of the values taken by previous observations of this variable. Obviously, the most important question here is to investigate which number of previous observations can best explain the current observation in the activity pattern. In the limit, the current value taken by X_t can be entirely explained by the previous observation (Activity *t-1*) (i.e. first lag). Analogously, in the limit, it might only be possible to accurately explain the current value of X_t by the last *k-1* observations (Activity *t-1*, Activity *t-2*, ..., Activity *k-1*) (i.e. *k-1*th lag) in which *k* represents the length of the activity pattern. However, when the current value can only be explained by a relative large number of previous observations, it is unlikely that the information that is identified is suitable for generalisation (prediction) purposes. On the other hand, a low number of previous observations might not be sufficient to explain the current observation either.

Higher-order Markov Chains can be better comprehended by means of the following example. Suppose for instance the case where the variable X_t originally takes the values in the state space {1,2,3}, where "1" stands for instance for "Sleeping", "2" stands for "Eating" and "3" stands for "Working". When one wants to take into account that X_t is not only explained by the first lag (*t*-1) but also by a second lag (*t*-2), the state space can be redefined as {(1,1)} (2-gram), indicating that this person was Sleeping at time *t*-2 and at time *t*-1, and with {(1,2), (1,3), (2,1), (2,2), (2,3), (3,1), (3,2), (3,3)} defined similarly.

By means of example, the corresponding transition matrix *Q* for ℓ=2 and *m*=3 is then:

$$Q = \begin{array}{ccc|ccccccccc|}
 & & X_t & 1 & 1 & 1 & 2 & 2 & 2 & 3 & 3 & 3 \\
X_{t-2} & X_{t-1} & X_{t-1} & 1 & 2 & 3 & 1 & 2 & 3 & 1 & 2 & 3 \\
\hline
1 & 1 & & q_{111} & 0 & 0 & q_{112} & 0 & 0 & q_{113} & 0 & 0 \\
2 & 1 & & q_{211} & 0 & 0 & q_{212} & 0 & 0 & q_{213} & 0 & 0 \\
3 & 1 & & q_{311} & 0 & 0 & q_{312} & 0 & 0 & q_{313} & 0 & 0 \\
1 & 2 & & 0 & q_{121} & 0 & 0 & q_{122} & 0 & 0 & q_{123} & 0 \\
2 & 2 & & 0 & q_{221} & 0 & 0 & q_{222} & 0 & 0 & q_{223} & 0 \\
3 & 2 & & 0 & q_{321} & 0 & 0 & q_{322} & 0 & 0 & q_{323} & 0 \\
1 & 3 & & 0 & 0 & q_{131} & 0 & 0 & q_{132} & 0 & 0 & q_{133} \\
2 & 3 & & 0 & 0 & q_{231} & 0 & 0 & q_{232} & 0 & 0 & q_{233} \\
3 & 3 & & 0 & 0 & q_{331} & 0 & 0 & q_{332} & 0 & 0 & q_{333}
\end{array}$$

However, as we can see from this simple example, there are several transitions in Q that can never occur. For instance, it is impossible to go from the row defined by X_{t-2}=1, X_{t-1}=1 to the column defined by X_t=1, X_{t-1}=2, because of the different value taken by X_{t-1}. The entries of these elements are called structural zeros. For any combination of ℓ and m, one can rewrite Q in a more compact form by excluding the number of structural zeros. This collapsed or reduced form of Q for ℓ=2 and m=3 can be denoted by (Pegram, 1980):

$$Q = \begin{array}{cc|ccc|}
 & & & X_t & \\
X_{t-2} & X_{t-1} & 1 & 2 & 3 \\
\hline
1 & 1 & q_{111} & q_{112} & q_{113} \\
2 & 1 & q_{211} & q_{212} & q_{213} \\
3 & 1 & q_{311} & q_{312} & q_{313} \\
1 & 2 & q_{121} & q_{122} & q_{123} \\
2 & 2 & q_{221} & q_{222} & q_{223} \\
3 & 2 & q_{321} & q_{322} & q_{323} \\
1 & 3 & q_{131} & q_{132} & q_{133} \\
2 & 3 & q_{231} & q_{232} & q_{233} \\
3 & 3 & q_{331} & q_{332} & q_{333}
\end{array}$$

The number of different states is equal to m^{ℓ}, and there are (m-1) independent probabilities in each row of the matrix Q, the last one depending on the others since each row is a probability distribution summing to one. The total number of independent parameters to estimate from the data is thus equal to m^{ℓ} (m-1) (for a more elaborated discussion see also Raftery, 1985; Berchtold and Raftery, 2002; Raftery and Tavaré, 1994).

Only after the optimal number of lags has been determined, and after a segmentation procedure has been applied (see section 5.5), sequences can be simulated based on the sequential information which is incorporated in the model. It is assumed that an optimal skeleton (i.e. the activity-travel pattern that has been predicted based on that lag that achieves the best match to the observed patterns) is able to generate better additional facets (such as time and location information) than a suboptimal skeleton. The assumption that was made in this respect is that other factors guide the allocation of time and location information to activities than sequential dependency information (see Chapter 6). Unfortunately, there are at least three important drawbacks with respect to the straightforward application of Markov Chains in our activity-diary simulation framework.

5.3 PROBLEM STATEMENT

First of all, it is obvious that as the order ℓ of the chain and the number of possible values *m* increase, the number of independent parameters increases exponentially and becomes too large to be estimated (Berchtold and Raftery, 2002). It even becomes infeasible to estimate the number of parameters for a relatively small number of possible values and for low-order Markov Chains. The problem was already identified in Raftery (1985) and a mixture transition distribution (MTD) model has been proposed in Berchtold and Raftery (2002) to come up with a solution. Table 5.1 relies upon their problem identification. The table gives the number of independent parameters to be estimated for different combinations of ℓ and *m*. With values of *m* going till 23 and with order ℓ sometimes up to 10 and higher (see infra), the infeasibility of the straightforward use of Markov Chains becomes clear for our application.

Secondly, Markov Chains were originally designed for modelling only *one* particular stochastic process of a discrete random variable X_t. It is still almost always used in this respect. The technique is for instance often applied for modelling *one* DNA string, for weather data during *one* year, etc. This obviously is a problem for our application domain since activity diaries of different respondents typically consist of multiple and (sometimes) independent sequences. We are able to get around this difficulty by carrying out as many runs of the Markov Chain as there are sequences in our data. However, this solution inevitably leads to a third problem, which is related with the estimation of the transition probabilities in Q.

Table 5.1: Maximal Number Of Independent Parameters For Markov Chains

Number of different values (m)	Order (ℓ)	Number of independent parameters to be estimated
3	2	18
5	2	100
	3	500
	4	2500
	5	12500
10	2	900
	3	9000
	4	90000
	5	900000
15	2	3150
	3	47250
	4	708750
	5	10631250

Transition probabilities are estimated through empirical frequencies which are observed in the data. They are often interpreted as a simple case of maximum likelihood estimates (MLE's) and are calculated by $N_{ij}/N_{i.}$ in which N_{ij} stands for the number of transitions from state i to state j in the whole dataset and $N_{i.}$ stands for the number of transitions starting from state i in the dataset. By calculating the transition probabilities for all the sequences at once, the independent character of each sequence in the data is in fact ignored, which may result in estimates which are seriously biased by specific combinations that may appear in one particular sequence. This problem is illustrated in Figure 5.1 by means of random extracts from 6 activity diaries. It can be seen from this figure that first-order transition probabilities in the pair FF are seriously flawed ($N_{FF}/N_{F.}$=19/33=0.58) by the occurrence of the number of family visits in the first sequence. In this case the diary of the first respondent implicitly receives in fact more weight in the calculation of the transition probabilities than diaries of the others respondents, which is not desirable and incorrect. For this reason, we have developed alternative methods for storing the sequential information (sequences of activities) in 'activity bundles', a term which is introduced to reflect that the information which is kept here represents low- and high-order combinations of activities that typically sequentially occur in *one* particular activity sequence (see section 5.4).

Activity Pattern 1: T_cFFFFFFFFFFFFFFFFE
Activity Pattern 2: T_cEEFREREERFT_cFT_cFFT_cFET_cF
Activity Pattern 3: RREFEFEE$T_c$$T_c$R
Activity Pattern 4: EEFFT_cFT_cFRR$T_c$$T_cRT_c$RR
Activity Pattern 5: FFT_cFFRE
Activity Pattern 6: EET_cFRRE
With T_c= **T**ransportation, with **c**ar as transport mode, F=visit **F**amily, E=**E**at, R=**R**ead

	T_c	E	R	F
T_c	0.13	0.07	0.20	0.60
E	0.21	0.36	0.14	0.29
R	0.17	0.42	0.33	0.08
F	0.18	0.12	0.12	**0.58**

Markov Chains (MLE)

	T_c	E	R	F
T_c	0.12	0.03	0.15	0.70
E	0.23	0.40	0.08	0.29
R	0.10	0.53	0.30	0.07
F	0.21	0.20	0.28	**0.31**

Alternative calculus of transition probabilities

Figure 5.1: Comparison of first-order transition probabilities
(Markov Chain's MLE versus our modified calculus)

It can be seen (for illustrative purposes) from Figure 5.1 that –when we compare the sequence pair FF with our developed methodology– our approach potentially does not suffer from this problem (only a weighted value of 0.31) as the independent character of every sequence is maintained (see next section for more details about this).

Having defined the drawbacks with respect to the use of Markov Chains in our simulation framework, we are now ready to develop an alternative technique which is able to approximate transition probabilities for higher-order Markov Chains and which does not suffer from the disadvantages mentioned above. The developed algorithms have the same aim as the previously mentioned MTD model, which is a reduction in the estimation of the number of parameters (i.e. a solution to the first problem statement in this section). Our approach differs from the MTD model in its alternative calculation of transition probabilities (thereby solving the 2nd and 3rd problem statement). However, the MTD model adopts weight parameters, expressing the effect of each lag on the present state *X*, and for this reason, it seems more suitable for analyses and policy evaluation.

Parts of the following sections are based upon research that was initially presented in Janssens *et al.* (2004c; 2004d) and that was later elaborated in Janssens *et al.* (2005c).

5.4 Building Transition Probabilities by means of Activity Bundles

The way by which the transition probabilities are calculated, uniquely determines the quality of the transition probability matrix. In order to avoid that transition probabilities are calculated for all sequences at once, as it is often done in Markov Chains, the idea of calculating probabilities for each respondent by means of activity bundles was developed. Thus, the main advantages of using activity bundles is that (i) transition probabilities are no longer *immediately* estimated by ignoring the independence between sequences, but are first stored in activity bundles per sequence and (ii) they represent a more intelligent framework to calculate the plausible combinations of states and can deal more efficiently with the combinatorial explosion of calculations. Activity bundles have to be seen as an intermediate, but crucial step before building transition probability matrices for the whole sample population. Indeed, since each activity bundle will represent the correct (i.e. not flawed by one particular sequence which receives more weight) low and high-order combinations per respondent, transition probabilities for the sample population can safely be derived by summing up equally occurring activity bundles over different respondents. We will elaborate on two different approaches for storing this sequential information in activity bundles.

5.4.1 Approach 1: Activity Bundles with Most Frequently Occurring Combinations

The first approach aims at selecting the most frequently occurring combination of elements within one particular sequence. The algorithm which is used to construct an ℓ-th-order activity bundle is introduced in Figure 5.2. Hereafter it is illustrated by means of a simple example.

```
Set k:=length of the sequence (diary)
Set $\ell$:=1                                    // $\ell$ is the order of the activity bundle
Do while an $\ell$-th order activity bundle can still be constructed ($\ell$<k)
Begin
If $\ell$=1 then
  begin
    identify all unique elements u, with $u \in \{1,\ldots,m\}$
    for each unique element u do
      begin
        set i:=current unique element
        identify all the elements ij which follow immediately after i, with $j \in \{1,\ldots, m\}$
        for each ij do count the number of times that ij occurs ($n_{ij}$)
        select the maximum value of $n_{ij}$ (break tie if necessary) and store this ij in the
        first-order activity bundle
      end
  end
else
  begin
    read the ($\ell$-1)-th order activity bundle
    for each combination A in the ($\ell$-1)-th order activity bundle do
      begin
        set A:=current combination
        identify all the elements Aj which follow immediately after A, with $j \in \{1,\ldots, m\}$
        for each Aj count the number of times that Aj occurs ($n_{Aj}$)
        select the maximum value of $n_{Aj}$ (break tie if necessary) and store this Aj in the
        $\ell$-th order activity bundle
      end
  end
$\ell$:=$\ell$+1
end
```

Figure 5.2: Algorithm for building activity bundles
with most frequently occurring combinations

It can be seen from this figure that the construction of higher-order activity bundles is based upon the activity-bundle which immediately precedes the current higher-order activity-bundle. By doing this, activity bundles are built in a more efficient manner. The idea originates from the discovery of frequent itemsets in the Apriori algorithm for association rule mining (Agrawal *et al.*, 1993), which also only uses the (ℓ-1)-th order itemsets to generate candidate ℓ-th order itemsets. Therefore, unlike Markov Chains, not all the different combinations (m^{ℓ}) have to be evaluated and computation difficulties will be reduced significantly.

Consider the following example to illustrate this algorithm:
T_c–W–R–T_c–W–TV–T_c–W–TV–T_c–F
where T_c=Transportation, with car as transport mode; W=Work ; R=Read ; TV=Watch Tv and F=Visit Family.
The algorithm starts with first-order activity bundles. The unique elements (*u*) in the diary are T_c, W, R, TV and F. The elements which follow immediately after T_c are W and F, indicated as T_cW and T_cF. It this simple example, it is obvious that n_{TW}=3 and n_{TF}=1. Therefore, the pair T_c-W will be stored as an element of a first-order activity bundle. Doing the same for every unique element *u*, means that also the couples W-TV, R-T_c and TV-T_c have to be added as elements to the first-order activity bundle.
Next, we move on to the second-order activity bundle. There are four different combinations *A* in the first-order activity bundle. In the first combination *A*:=T_c-W, the elements *Aj* which follow immediately after *A* are R and TV, indicated as AR and ATV, with frequencies n_{AR}=1 and n_{ATV} =2. Accordingly, the combination A-TV (A:= T_c-W) will be added to the second-order activity bundle. Adding W-TV-T_c and R-T_c-W is straightforward. However, a tie occurs when A:=TV-T_c. Indeed, in this case n_{AW}= n_{AF} =1. The standard rule to break this tie is that the element which occurs most frequently in the sequence has to be chosen. The idea behind this rule is that this particular activity might be valued higher by this respondent. When there is no element which occurs most frequently in the diary, the combination is chosen at random. In this case, it is obvious to add the combination TV-T_c-W. Building higher-order activity bundles is easy for this simple example. An overview is given in Table 5.2.

Table 5.2: A higher-order example of activity bundles (Approach 1)

Order	Combinations
1	T_c-W; W-TV; R-T_c; TV- T_c
2	T_c-W-TV; W-TV- T_c; R-T_c-W;TV- T_c-W
3	T_c-W-TV- T_c; W-TV- T_c-W; R-T_c-W-TV; TV- T_c-W-TV
4	T_c-W-TV- T_c-W; W-TV- T_c-W-TV; R-T_c-W-TV-T_c; TV- T_c-W-TV- T_c
5	T_c-W-TV- T_c-W-TV; W-TV- T_c-W-TV- T_c; R-T_c-W-TV-T_c-W; TV- T_c-W-TV- T_c-F
6	T_c-W-TV- T_c-W-TV-T_c; W-TV- T_c-W-TV- T_c-F; R-T_c-W-TV-T_c-W-TV
7	T_c-W-TV- T_c-W-TV-T_c-F; R-T_c-W-TV-T_c-W-TV-T_c
8	R-T_c-W-TV-T_c-W-TV-T_c-F

Based on these activity bundles and in correspondence with Markov Chain terminology, transition probability matrices can be constructed. Taking for instance the elements in a second-order activity bundle, the corresponding (trivial) second-order transition probability matrix will look like:

$$Q= \begin{array}{cc|ccccc} & & & & X_t & & \\ X_{t-2} & X_{t-1} & T_c & W & R & TV & F \\ \hline T_c & W & 0 & 0 & 0 & 1 & 0 \\ W & TV & 1 & 0 & 0 & 0 & 0 \\ R & T_c & 0 & 1 & 0 & 0 & 0 \\ TV & T_c & 0 & 1 & 0 & 0 & 0 \end{array}$$

Indeed, since there is only one most frequent combination of elements per activity bundle, the transition probability matrix will only increase by one and for instance not by two for the combination T_c-W-TV, as the data would suggest.

The same procedure will be followed for every sequence. This means that activity bundles have to be built for every activity sequence which is in the sample data and that the transition probability matrix has to be updated for every sequence as well. Note that rows will be added in this process whenever this is necessary (i.e. when the current elements in a particular activity bundle contain different combinations than previous elements) and that the number of columns will remain the same. After the whole procedure is completed, the numbers will be normalized such that each row of the transition probability matrix sums to one. A computer code has been established to automate the full process.

It can be seen from Table 5.2 that the number activity bundles per order still stays very low, even when higher-order bundles are considered. The fifth-order bundle contains only 4 elements for instance. Multiplied by 5 columns means that only 20 parameters need to be estimated. This number is of course extremely low when we compare it with the estimation of the transition probabilities for a fifth order Markov Chain for this small sequence, which would contain 12500 independent parameters to be estimated. Obviously, the major part of these parameters will be zero, since these combinations won't occur in the data.

However, the comparison above is not completely fair since Markov Chains do not only consider the most frequently occurring combination of elements, but they take every possible combination into account by means of a simple case of maximum likelihood estimation procedure. In order to enable us to make a better comparison, a second approach has been developed, which is on the one hand equal to the technique of Markov Chains (it also uses MLE) but which on the other hand still constructs bundles per activity pattern and therefore maintains the independent character of every sequence.

5.4.2 Approach 2: Activity Bundles with Maximum Likelihood Estimates

Unlike only adopting a majority rule in approach 1, allowing minority combinations is also possible if these minority combinations are given a lower score. Maximum likelihood estimates are used to calculate the scores which are given to each combination. As mentioned before, the simple case of MLE's are calculated by dividing N_{ij} by $N_{i.}$ where N_{ij} stands for the number of transitions from state *i* to state *j* in the data and $N_{i.}$ represents the number of transitions starting from state *i* in the pattern. Obviously, this approach will result in more elements per activity bundle since every possible combination, which is in the data, will now be stored. The difference with Markov Chains is that frequencies are estimated and stored per sequence. The algorithm, which is quite similar to the algorithm given in approach 1, is shown in Figure 5.3.

```
Set k:=length of the sequence (diary)
Set $\ell$:=1                              // $\ell$ is the order of the activity bundle
Do while an $\ell$-th activity bundle can still be constructed ($\ell$<k)
Begin
If $\ell$=1 then
  begin
        identify all unique elements u, with $u \in \{1,\dots, m\}$
        for each unique element u do
           begin
              set i:=current unique element
              calculate the number of transitions that start from state i in the activity
              pattern ($n_{i.}$)
              identify all the elements ij which follow immediately after i, with $j \in \{1,\dots, m\}$
              for each ij do count the number of times that ij occurs ($n_{ij}$)
              store each ij and each weight ($n_{ij}/n_{i.}$) in the first-order activity bundle
           end
  end
else
  begin
        read the ($\ell$-1)-th order activity bundle
        for each combination A in the ($\ell$-1)-th order activity bundle do
           begin
             set A:=current combination
             calculate the number of transitions that start from A in the activity pattern ($n_{A.}$)
             identify all the elements Aj which follow immediately after A, with $j \in \{1,\dots, m\}$
             for each Aj count the number of times that Aj occurs ($n_{Aj}$)
             store each Aj and each weight ($n_{A j}/ n_{A.}$) in the $\ell$-th-order activity bundle
           end
  end
$\ell$:=$\ell$+1
end
```

Figure 5.3: Algorithm for building activity bundles with MLE

In order to illustrate this algorithm, reconsider the random extracts from 6 activity diaries, which were shown in the problem statement:

Activity Pattern 1: T_cFFFFFFFFFFFFFFFE
Activity Pattern 2: T_cEEFREREERFT_cFT_cFFT_cFET_cF
Activity Pattern 3: RREFEFEE$T_c$$T_c$R
Activity Pattern 4: EEFFT_cFT_cFRR$T_c$$T_cRT_c$RR
Activity Pattern 5: FFT_cFFRE
Activity Pattern 6: EET_cFRRE

The first-order activity bundles of these activity patterns are shown in Table 5.3. The weights of each combination are shown in brackets.

Table 5.3: A First-order Example of Activity Bundles (Approach 2)

Seq. number	First-order Combinations
Activity pattern 1	T_c-F (1); F-F (0.94); F-E (0.06)
Activity pattern 2	T_c-E (0.2); T_c-F (0.8); E-E (0.33); E-F (0.17); E-R (0.33); E-T_c (0.17); F-R (0.17); F-T_c (0.5); F-F (0.17) ; F-E (0.17) ; R-E (0.67); R-F (0.33)
Activity pattern 3	R-R (0.5); R-E (0.5); E-F (0.5); E-E (0.25);E-T_c (0.25); F-E (1); T_c-T_c (0.5); T_c-R (0.5)
Activity pattern 4	E-E (0.5); E-F (0.5); F-F (0.25); F-T_c (0.5); F-R (0.25); T_c-F (0.4); T_c-T_c (0.2); T_c-R (0.4); R-R (0.5); R-T_c (0.5)
Activity pattern 5	F-F (0.5); F-T_c (0.25); F-R (0.25); T_c-F(1); R-E (1)
Activity pattern 6	E-E (0.5) ; E-T_c (0.5) ; T_c-F (1) ; F-R (1) ; R-R (0.5); R-E (0.5)

Based on these activity bundles, the final first-order transition matrix can be constructed for this example by aggregating the same bundles of activities across the different activity patterns. Unlike in approach 1, matrix entries increase by the *weight* which is computed in each of the activity bundles.

The final first-order transition matrix will look like:

$$R= \begin{array}{c|cccc} & & & X_t & \\ X_{t-1} & T_c & E & R & F \\ \hline T_c & 0.70 & 0.20 & 0.90 & 4.20 \\ E & 0.92 & 1.58 & 0.33 & 1.17 \\ R & 0\ .50 & 2.67 & 1.50 & 0.33 \\ F & 1\ .25 & 1.23 & 1.67 & 1.85 \end{array}$$

Normalizing this table such that each row sums to one, gives exactly the same transition probability matrix as the one shown in the problem statement. It should be clear that this algorithm is comparable with the Markov Chain approach. However, since each element in an activity bundle represents the correct low and high-order combinations per respondent, each respondent (activity pattern) receives the same weight and estimates of transition probabilities promise to be less biased than in the Markov Chain approach. Although this algorithm is clearly computationally more demanding than the first approach, the number of different parameters which has to be estimated, is still significantly lower than the number which needs to be estimated in Markov Chains (see also section 5.7.3).

5.5 The Need for a New Segmentation Approach

5.5.1 Preface

Until now, we assumed that there is only one transition probability matrix (per lag/order) which is both representative for every respondent and for every time frame during the day. However, there is accumulated empirical evidence which suggests that activity-travel patterns are (highly) correlated with the socio-demographic information (Greaves and Stopher, 2000; Veldhuisen *et al.*, 2000a) of the respondent and that different transport behaviour (and thus different activity-travel patterns) exist for different time windows during the day (Bhat and Singh, 2000; Hamed and Mannering, 1993). The reader may recall that the presence of different transition probability matrices for different time windows in the day is in fact a relaxation of the stationarity condition as introduced in definition 5.4. When dealing with time segmentation, only one explanatory variable (i.e. time of day) needs to be taken into account. Therefore, the statistical test of stationarity of a system that has been introduced in definition 5.5, and that mainly examines the change of dynamics before and after a certain moment in time, can be extended by considering different splitting points that lead to a significant statistical difference (see section 5.5.2). Things get more complicated when segmentation is done in terms of socio-demographic information because of different explanatory variables. In this case, a modified version of a decision tree was developed, such that the dependent variable in the tree explicitly takes sequential information per socio-demographic variable into account. Both approaches have been described in the next two sections.

5.5.2 Segmentation by means of Bifurcation Points (Temporal Segmentation)

Definition 5.5 has introduced a statistical test for examining whether there is a change of dynamics before and after a particular cutpoint. For instance, the test can be used to examine whether transition matrices are significantly different in the time periods ranging from 24PM-8AM; from 8AM-16PM and from 16PM-24PM.

(see section 5.7.4 for an empirical validation). Obviously, the test can equally be used for a segmentation in more segments. The choice of these cutpoints can be done arbitrarily, relying for instance on domain knowledge. While this is a good procedure to get an initial idea about whether our transition matrices satisfy the stationarity assumption or not, splitting a sequence at different single points in time is unlikely to result in the most optimal segmentation because it is not at all driven by the information which is incorporated in the data. Therefore, by the iterative application of definition 5.5 (omnibus test), for all possible splitting points of a sequence, we are able to point out moments in time where the system bifurcated into significantly different type of dynamics. Such a procedure is also valid to evaluate whether the identified pivotal moments in the data match with the moments defined by a priori domain knowledge. The points that radically transform the dynamics of a system are called bifurcation points. The methodology is frequently used in complex dynamical system theory (see for instance Haken, 1983; Barton, 1994, Lemay, 1999). The procedure to identify these bifurcation points is straightforward:

1. Determine the level of significance (α).
2. Set a time window by which transition matrices need to be compared. In the limit, this time window can be set equal to 1 minute but this will lead to a computational explosion of the calculations. In the artificial example given above (24PM-8AM; 8AM-16PM and 16PM-24PM), the three time windows are set equal to 8 hours, and the potential bifurcation points are set at 8AM, 16PM and 24PM. Accordingly, every time window defines potential bifurcation points (n_1,n_2,n_3). The first potential bifurcation point is defined as n_{min}, the last as n_{max}.
3. Construct a transition matrix for every time window, i.e. three transition matrices in this example.
4. Calculate the χ^2 value to evaluate whether the dynamics of the system is subject to a segmentation into *d* (i.e. 3 by example) time periods (see definition 5.5) by application of the omnibus test.
5. Store the p-value for this omnibus test.
6. Redefine the time window ranging from n_{min} to n_{max} by adding one time window to n_{min}, thereby setting n_{min}:=n_{min}+1. Recalculate the transition matrices for the new time windows. In our example a new

transition matrix needs to be computed ranging from 24PM-16PM. The transition matrix that was computed before, for 16PM-24PM, remains the same.

7. Re-calculate the omnibus test for $\mathrm{n}_{\mathrm{min}}$ to $\mathrm{n}_{\mathrm{max}}$. Equally, substract one time period from *d*; i.e. *d*:=*d*-1. In our example, it is thus evaluated whether the dynamics of the system is subject to a segmentation in two time periods.
8. Store the p-value for this omnibus test.
9. Repeat steps 6 till 8 until n_{min}:=n_{max} or until *d*:=1.
10. Plot the p-values for every omnibus test

The procedure will be empirically illustrated in section 5.7.5.

5.5.3 SEGMENTATION BY MEANS OF FULL DECISION TREES (SOCIO-DEMOGRAPHIC SEGMENTATION)

Things get more complicated when transition probability matrices need to be segmented in terms of socio-demographic information. Indeed, unlike in the previous case, there is now a combination of different explanatory variables that have a potential influence on the transition probability matrix. To this end, a novel segmentation scheme has been developed that is a modified version of a decision tree approach. Especially CART decision trees were used in a number of previous studies (Greaves and Stopher, 2000; Vaughn *et al.*, 1999) in the context of transportation modelling for segmentation. The best known application of this technique is probably the TRANSIMS project, where the CART algorithm is used in the "Activity Generator Module" to produce an accurate classification of household characteristics based on household travel behaviours.

However, in traditional (classification) decision trees, the dependent variable at the leaf simply contains a finite number of possible values and is often discrete in nature. The novel algorithm that is proposed in this chapter differs in two ways from this common way of thinking. First, the dependent variable can no longer be immediately observed from the data but is the result of a learning methodology (see section 5.4) and second, the dependent variable explicitly takes sequential information into account. As such, transition probability matrices are used as dependent variables in the construction of the trees.

Obviously, the most important decision that needs to be made when developing a decision tree is the splitting criterion. One possible approach would be to compute the difference between two matrices, for instance by the calculation of the sum of squared errors (SSE) between two matrices. However, such an approach would only look at differences at a particular level in the tree, while traditional decision trees take criteria into account that calculate the benefit of moving from one level in the tree down to the next level (for instance by means of gain ratio). In addition to this, it would only be computationally feasible to construct binary decision trees by means of these procedures, which is a major limitation when compared to traditional decision trees.

Fortunately, one of the most widely measures that is adopted in decision trees, i.e. gain ratio, can be applied in a quite straightforward manner in our approach as well. The use of gain ratio as a split criterion favours splits into increasingly homogeneous partitions in terms of the dependent variable (class attribute), because the best split is the one with the most homogeneous daughters. In the limit, leaf nodes (i.e. nodes that have no offspring nodes) will therefore only contain cases from a single response class. Gain ratio is a measure which is derived from information theory. Information theory defines the quantity of information conveyed by a particular message as being inversely proportional to the predictability of that message. When a message is entirely certain (that is, its probability is 1), then the quantity of information conveyed is zero. When a message is nearly improbable (that is, its probability is almost 0), a maximum quantity of information is needed to receive such a message. The degree of uncertainty of a message can be represented by the probability of that message, or in terms of traditional decision trees, by the probability of that class. As mentioned before, when all this can be translated to transition probability matrices, a new decision tree approach will emerge that discerns itself from traditional decision trees in the sense that the dependent variable can no longer be immediately observed from the data but is the result of a learning methodology and second, the dependent variable explicitly takes sequential information into account.

The entries in transition probability matrices are used for achieving this. Adopting gain ratio in the context of transition probability matrices will then result in *homogeneous* transition probability matrices. For a better understanding, the reader may reconsider the following definitions that are used in traditional decision tree induction (Quinlan 1993). Each definition will later be re-introduced such that it becomes feasible in the context of transition probability matrices.

INFORMATION THEORY

Definition 5.10: Entropy of a set *T*

The entropy of particular (sub)set *T* is equal to:

$\text{Info } (T) = -\sum_{i=1}^{k} \left(\frac{freq(C_i, T)}{|T|} \right) \times \log_2 \left(\frac{freq(C_i, T)}{|T|} \right) \text{bits}$, with *T* a set of cases, C_i a class *i*, |*T*| the number of cases in *T, k* the number of classes and freq (C_i,*T*) the number of cases in *T* that belongs to class C_i. ■

Example: Assume that we face a 2 class-problem, 2 cases with class “yes” and 3 cases with class “no” in the leaf node of a decision tree. The entropy of this set of cases *T* is then represented as: Info $(2,3) = -\frac{2}{5}\log_2(\frac{2}{5}) - \frac{3}{5}\log_2(\frac{3}{5}) = 0.971$ bits.

Definition 5.11: Entropy after a (sub)set has been partitioned on a test *X*:

$\text{Info}_X(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \times \text{info}(T_i)$, where $|T_i|$ represents the number of cases that belongs to the partition *i* and |*T*| represents the number of cases in *T*. ■

Example: In fact the above formula simply calculates the average information value, taking into account the number of instances that go down each branch in the tree. In addition to the information value of the first branch (2 cases with class “yes”; 3 with “no”), we assume that there are two additional branches for this particular split (*X*), resulting in 4 cases with class “yes” (zero “no”) in the second branch and 3 cases with class “yes” and 2 with class “no” in the third branch. Thus, the information value of this split consists of three branches, and can be computed as follows:

Info ([2,3],[4,0],[3,2])=(5/14)x0.971+(4/14)x0+(5/14)x0.971=0.693 bits.

Definition 5.12: Gain criterion

The gain criterion measures the information that is gained by partitioning a training set using a particular test *X*. Gain criterion is defined as: Gain (*X*)= info (*T*)-info$_X$(*T*). ■

Example: The information that is gained by partitioning a particular split as defined above is equal to: Gain (*X*)=Info ([9,5])-0.693= 0.247 bits
In C4.5 (and ID3) the test is chosen that maximizes the information gain because one may expect that the remaining subsets in the branches will be the most easy to partition. However, this is by no means certain because we have looked ahead only one level deep in the tree. Despite this, the gain criteria proved to perform well in practice. However, there is another serious deficiency with the gain criterion. When attributes have a large number of possible values, giving rise to a multiway branch with many child nodes, a problem arises with the information gain calculation. The problem can be best understood when an attribute has a different attribute for every instance in the dataset, in the most extreme case (e.g. an ID-code). Suppose for instance that we have 14 ID-codes, where each ID-code contains one case. The computation then becomes Info ([0,1])+ Info ([0,1])+ Info ([1,0])+ ... + Info ([0,1]), which is equal to zero.
Consequently, the information gain of this attribute is equal to the information value at the root of the tree. This value is larger than any other attribute and ID code will inevitably be chosen as the splitting attribute. To compensate for this trend to favour multi-way branching attributes, gain ratio is often used.

Definition 5.13: Gain ratio

Gain ratio (*X*)=$\frac{\text{gain}(X)}{\text{split info (X)}}$, where split info (*X*) indicates the information that is generated by partitioning *T* into *n* subsets. It is calculated by:

$$-\sum_{i=1}^{n}\frac{|T_i|}{|T|}\times\log_2\left(\frac{|T_i|}{|T|}\right)$$ ■

Example: The gain ratio of the ID-attribute that was illustrated above is equal to: 0.940/3.807=0.246, which largely reduces the original gain criterion for this attribute.

Re-Introducing Information Theory

Definition 5.14: Consistent with definition 5.10, the entropy of one row *i* in a frequency matrix Q_{freq} can be defined as:

$$\text{Info } Q_{\text{freq}}(i) = -\sum_{j=1}^{m}\left(\frac{(Q_{\text{freq}}(i,j)}{n_{itot}}\right)\times\log_2\left(\frac{(Q_{\text{freq}}(i,j)}{n_{itot}}\right)\text{bits}, \forall \text{ row } i,$$

with $Q_{\text{freq}}(i)$ the i^{th} row of the frequency matrix Q_{freq}; $Q_{\text{freq}}(i, j)$ the matrix entries defined by the i^{th} row and the j^{th} column, that is the frequency that the element(s) in row *i* is (are) followed by the element in column *j*; n_{itot} the row total in Q_{freq} and *m* the number of columns. ■

Example: Assume that we have the following (first-order) transition frequency matrix:

Q_{freq}=

X_{t-1}		X_t			
	T_c	E	F	R	n_{itot}
T_c	5	18	2	5	30
E	1.7	3.68	4.09	0.53	10
F	6.25	10	8.75	5	30
R	0	1.7	5.8	2.5	10

The entropy of row two is then: $\text{Info}Q_{\text{freq}}(2)$=

$$-\frac{1,7}{10}\log_2(\frac{1,7}{10})-\frac{3,68}{10}\log_2(\frac{3,68}{10})-\frac{4,09}{10}\log_2(\frac{4,09}{10})-\frac{0,53}{10}\log_2(\frac{0,53}{10})=1{,}72 \text{ bits}$$

The entropy of other rows in the matrix can be calculated in a similar way.

Definition 5.15: Entropy of a full transition (probability or frequency) matrix.

The entropy of a full transition matrix is equal to $\text{Info(Q)}=\sum_{i=1}^{m}\frac{n_{itot}}{N}\times\text{Info}(Q_{\text{freq}}(i))$

■

It can be seen from this formula that every row in the transition matrix is weighted in proportion to the number of times a particular sequence starts with the element(s) which is represented in that particular row of the matrix.

Example: The entropy of the full transition probability matrix, shown in the example above is equal to: (30/80)x1,56+(10/80)x1,72+(30/80)x1,95 +(10/80)x1,38 =1,70 bits.

Definition 5.16: Consistent with definition 5.11, the entropy of a full transition probability matrix after a (sub)set has been partitioned on a test *X*, can be calculated as:

$\text{Info}_X(\text{Q})=\sum_{i=1}^{n}\frac{|T_i|}{|T|}\times\text{info}(Q_i)$, where $|T_i|$ represents the number of cases that belongs to the partition i and $|T|$ represents the number of cases in T. ■

Example: Assume that the first branch is specified by the transition probability matrix introduced in the example given in definition 5.14, and that a second branch contains a transition probability matrix that has an entropy equal to 1,50 bits. Both branches respectively represent 4 and 3 cases for this particular split (*X*). The calculation is as follows: (4/7)x1,70+(3/7)x1,50=1,61 bits.

The gain and the gain ratio criterion for transition probability matrices only depend upon a particular split. Accordingly definitions 5.12 and 5.13 can be re-introduced for transition probabilities without modification.

In order to use these modified principles of information theory in a new decision tree segmentation scheme, the following mathematical conceptualisation has been introduced.

CONCEPTUALISATION

S the total sample of activity diaries, consisting of *n* sequences, indexed $i = 1,\ldots,n$

$\mathbf{X_k}$ explanatory socio-demographic attributes, with $k = 1,\ldots,K$.

Y dependent variable, represents the transition probability matrix *Q* (Y=Q); for all sequences ($\forall\ i$)

T Final decision tree based on sequential information, comprised of nodes (*Ns* and *L*) and branches. Leaf nodes are specified by *Q*.

N the current node in *T*, splitting the current subset of S into subsets N_{kt}

$\mathbf{N_{kt}}$ represents the subsets of a split by *N*; splits at a value *t* based on an independent variable X_k, such that X_k=t; $\forall t_k$

$\mathbf{t_k}$ set of possible values of *t* such that there exist observations in *N* having X_k=t; $\forall k$=$1,\ldots,K$; and with N=X_k.

Ns set of active decision nodes in *T* that split *S* into different subsets.

L set of inactive decision nodes that cannot split *S* into additional subsets because n_{min} or G_{min} are not satisfied. In this case they become leaf nodes *L*.

$\mathbf{n_{min}}$ Parameter that determines whether a particular branch in the tree is split into additional nodes or not. Splitting is stopped when the number of individuals that belong to either of the child nodes N_{kt} is less than the number defined by n_{min}

$\mathbf{G(N_{kt})}$ Gain ratio (as defined by definitions 5.12 to 5.16) of a transition matrix that is built on the subset of N_{kt}.

Max $\mathbf{G(N_{kt})}$ represents the global maximum of all the gain ratios per level in the tree, $\forall N_{kt}$. Max $G(N_{kt})$ is used to select the optimal decision node *N*.

$\mathbf{G_{min}}$ Minimum Gain ratio that determines whether a particular branch in the tree is split into additional decision nodes or not. Splitting is stopped when Max $G(N_{kt})$ is smaller than G_{min}.

Having defined this mathematical conceptualisation, a decision tree procedure can now be introduced in Figure 5.4. A computer code has been established to automate the full process. The next paragraph elaborates on an example to illustrate this procedure.

Example:

We can for instance assume that we take the first 3 of the 6 activity-travel patterns that were introduced in section 5.3. Now let us assume that socio-demographic information is also provided with this example. For the sake of simplicity, it is assumed that only 3 socio-demographic attributes are known; i.e. gender; age and education. The specific values for each activity pattern are:

Diary 1: male, older than 45 years and high educated

Diary 2: female, between 18 and 24 and low educated

Diary 3: male, between 25 and 44 and low educated

The initialisation procedure in Figure 5.4 is quite simple for this example. The value n_{min} and G_{min} are respectively set equal to 1 and 0, X_k is defined as gender, age and education, respectively for k=1,…,3; with K=3. The set of active decision nodes *Ns* is also fixed as {gender, age, education}. Note that this set is not always equal to the variables X_k for k=1,…,K, since one might decide not to use certain variables as decision nodes, for example in case of ID-number, which might be perfectly relevant as an attribute but not as a decision node. Finally, the set of leaf nodes is initialised as empty.

Since the set of decision nodes *Ns* is not empty and the set of leaf nodes is empty, the two first checks in Figure 5.4 can be omitted. After this, the procedure will select the most optimal decision node for the root node of the tree.

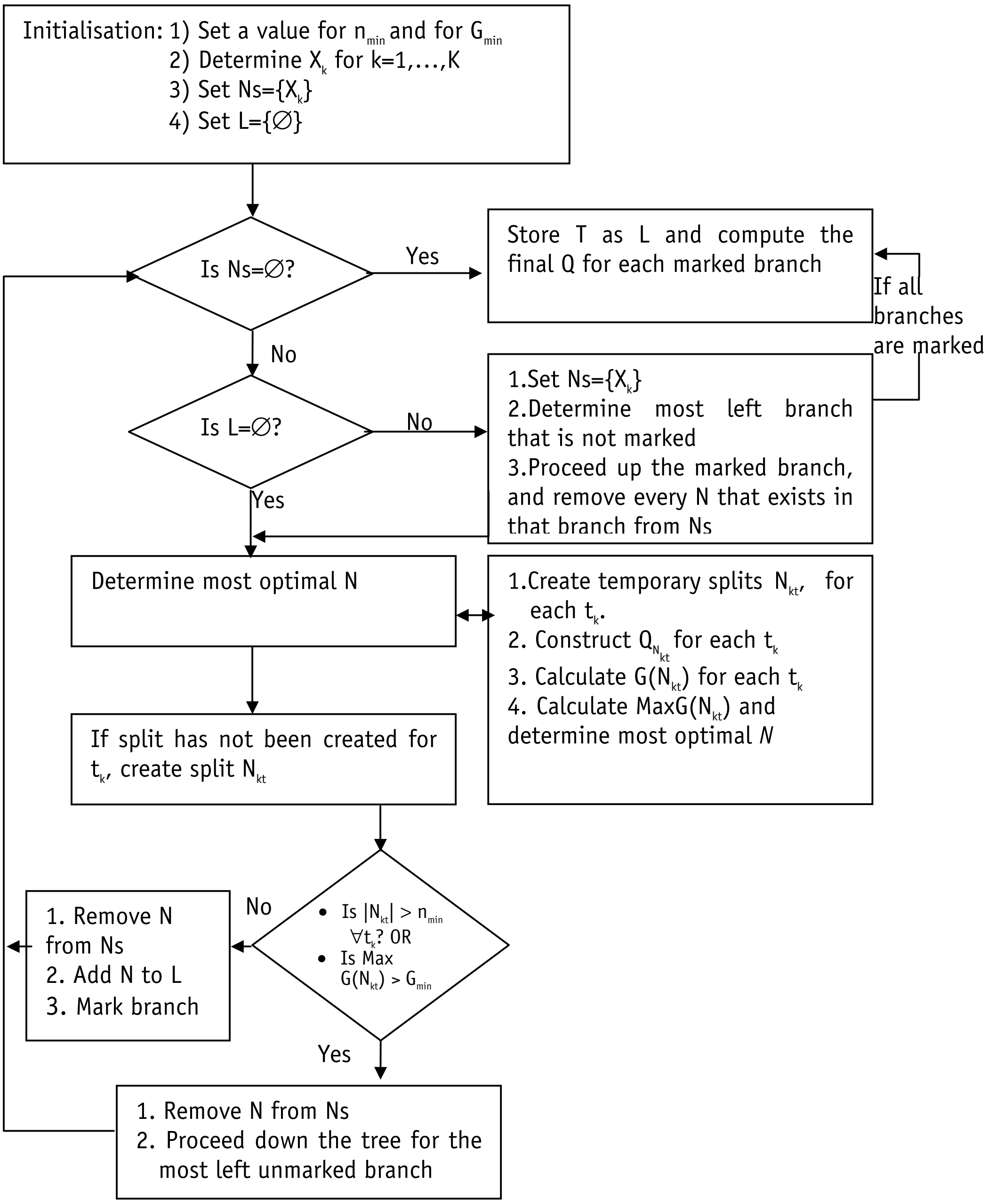
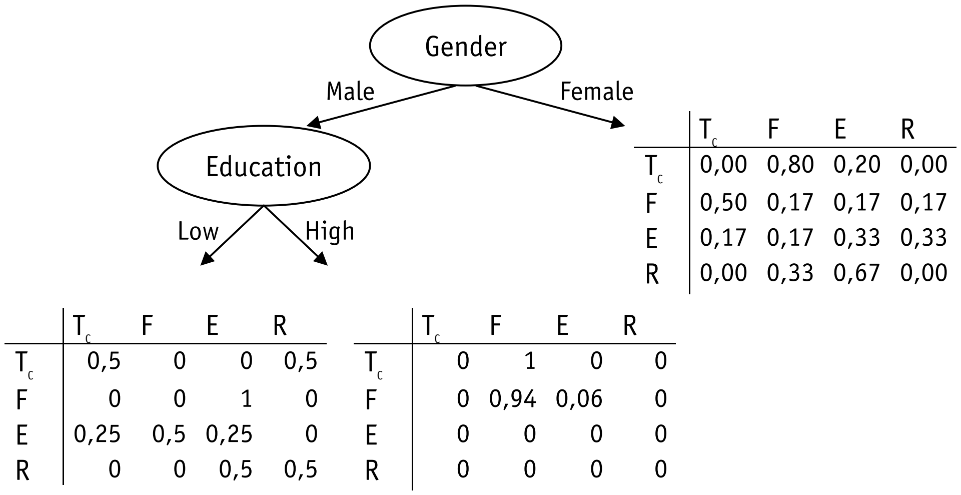

Initialisation: 1) Set a value for n_{min} and for G_{min}
2) Determine X_k for k=1,…,K
3) Set Ns={X_k}
4) Set L={∅}
Is Ns=∅?
Yes
Store T as L and compute the final Q for each marked branch
If all branches are marked
No
Is L=∅?
No
1.Set Ns={X_k}
2.Determine most left branch that is not marked
3.Proceed up the marked branch, and remove every N that exists in that branch from Ns
Yes
Determine most optimal N
1.Create temporary splits N_{kt}, for each t_k.
2. Construct $Q_{N_{kt}}$ for each t_k
3. Calculate G(N_{kt}) for each t_k
4. Calculate MaxG(N_{kt}) and determine most optimal *N*
If split has not been created for t_k, create split N_{kt}
No
• Is $|N_{kt}| > n_{min}$ $\forall t_k$? OR
• Is Max $G(N_{kt}) > G_{min}$
1. Remove N from Ns
2. Add N to L
3. Mark branch
Yes
1. Remove N from Ns
2. Proceed down the tree for the most left unmarked branch

Figure 5.4: Description of the procedure for building a decision tree based on sequential information.

First, each decision node N that belongs to Ns, is divided into temporary splits N_{kt} such that X_k=t $\forall N_{kt}$ for k=1,...,3. Then, $Q_{N_{kt}}$ is constructed for each t_k. Next, the gain ratio is calculated as explained by definitions 5.12-5.16, and the attribute that achieves the highest gain ratio in the tree is selected to carry out the split at the current level (=root level) of the tree. The attribute "Gender", with a gain ratio of 0.469, achieves the highest value (MaxG($\mathrm{N_{kt}}$)) and is thus chosen as the best split for this tree at root level. After actually creating this split, the next step first verifies whether the number of observations in the child nodes, is greater than the minimal value n_{min} and greater than G_{min}. While this is the case for the first branch (gender=male), it is not for the second branch (gender=female). For this reason the decision node "Gender" is removed from the set of active decision nodes Ns and added to the set of leaf nodes. The branch for which the decision node did not satisfy the n_{min}-check (i.e. N_2) is marked to indicate that it has been fully exploited. The procedure now restarts from the beginning. While the first check still is not yet satisfied, the set of leaf nodes is no longer empty. First, the set of active decision nodes in temporarily set equal to the set decision nodes X_k. This is necessary to let a particular variable occur multiple times in different branches in the tree. Second, the most left unmarked branch in this tree is now identified. In this example this unmarked branch is specified by Gender=male. Again, the most optimal N needs to be determined for this branch. This means that temporary splits need to be created for the branch gender=male and the maximum gain ratio need to be computed. In this case, the variable education achieves the largest gain ratio. Now, both remaining branches (i.e. education=low; education=high) do not satisfy the n_{min}-check. Indeed, for the branch gender=male; there is only one case that belongs to education=low and one case that belongs to education=high. This means that both branches need to be marked. While Ns is still not empty, all branches are now marked and the final decision tree along with its final Q are stored and shown in Figure 5.5.



	T_c	F	E	R
T_c	0,00	0,80	0,20	0,00
F	0,50	0,17	0,17	0,17
E	0,17	0,17	0,33	0,33
R	0,00	0,33	0,67	0,00

	T_c	F	E	R
T_c	0,5	0	0	0,5
F	0	0	1	0
E	0,25	0,5	0,25	0
R	0	0	0,5	0,5

	T_c	F	E	R
T_c	0	1	0	0
F	0	0,94	0,06	0
E	0	0	0	0
R	0	0	0	0

Figure 5.5: The final sequential information decision tree (example)

Once the segmentation for all the transition probability matrices has been done, every sequence can use the appropriate transition matrix by proceeding its socio-demographic information down the tree.

5.6 SIMULATING ACTIVITY-TRAVEL PATTERNS

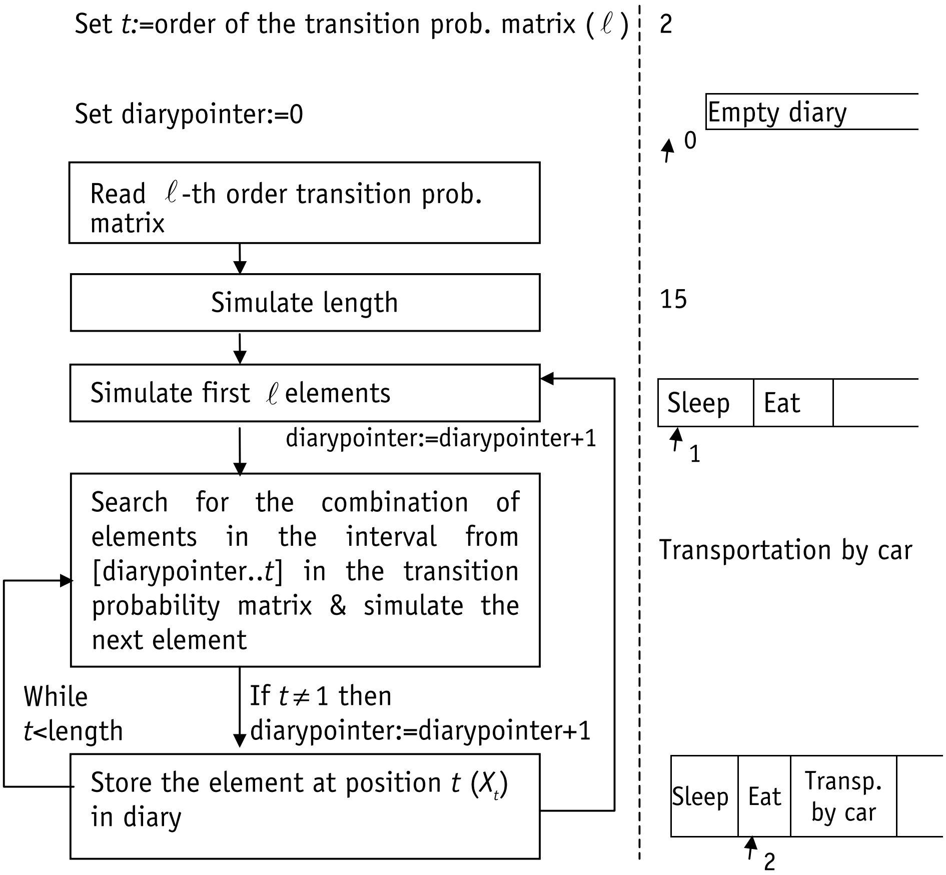
5.6.1 BASIC SIMULATION

The previous sections have defined the "knowledge model" that will be used to steer the basic simulation procedure that is introduced in this section. To summarize the above, the knowledge incorporated in the model is sequential dependency information of low and high-order, differentiated by time of day and by socio-demographic information. While the method indirectly takes time information into account to develop a more accurate skeleton of activities and transport modes, specific time information was not yet allocated to activities. The same can be said with respect to location information. As mentioned before, we have made the assumption that other factors than sequential dependency information guide the allocation of time and location information to activities (see Chapter 6). For this reason, both facets were not yet incorporated in the current simulation procedure.

The aim of the simulation procedure is to predict the value taken by X_t as a function of the values taken by previous observations of this variable (as they are incorporated in the transition probability matrices). Before doing so, socio-demographic information is simulated, based on a simple sampling procedure, where the distribution per socio-demographic variable is used to guide the simulation procedure. Once the socio-demographic information is generated per respondent, the specific socio-demographic information is routed down the segmentation tree and the corresponding transition probability matrix is used in the procedure for simulating the activity patterns, described in Figure 5.6.

The left part of this figure shows the different steps of the procedure; the right part shows the real outcome of these steps by means of an example. The procedure starts by initialising the values of the indexes *t* and "diarypointer". The index *t* is preferably interpreted as the position of the activity in the activity pattern, whereas the "diarypointer" is a kind of technical index that keeps track of the lag, which is used in the simulations. The diarypointer is always initialised at position zero; the index *t* is variable and is set equal to the order of the transition probability matrix. The second order transition probability matrix is taken as an example (see right part). Reading the transition probability matrix is the first logical step. Next, the length of the activity pattern is generated. This implies for this step that a random number is generated, based on a given sample distribution of the length of the activity pattern. The decision was made to incorporate this dimension into the generated activity patterns since some people fill out their diaries carefully (or simply perform more activities), while others are more imprecise. In our example, it is assumed that 15 activities will be generated. In order to take advantage of the segmentation with respect to time information (by means of the bifurcation point procedure, see section 5.5.2), the simulation approach also uses the length of the simulated activity travel pattern to come up with a rough segmentation per pattern. That is, if the bifurcation point procedure determined that 3 time intervals contained statistically significant transition matrices, the generated activity-travel pattern is split up in three equal parts; where the first 5 activities will be generated by means of the first transition matrix, activities 6-10 are generated by means of the second transition matrix and activities 11-15 are generated by means of the third transition matrix. This is an arbitrarily defined procedure, but the alternative - using only one transition matrix- is even worse if the process does not satisfy the stationarity condition (see section 5.7.4 for an empirical illustration). These steps were not shown in Figure 5.6 for the sake of clarity.

Once the length of the (to be predicted) pattern has been simulated, the first ℓ elements of this pattern are generated. The initial sequential probability distributions in the sample data are used for this. This means that the first ℓ elements of the sequence are generated from the prior probability distributions, which are in the data and not from the empirically constructed transition matrices. Assume that a sleep and an eat-activity are the first two simulated elements. The diarypointer can now be augmented from zero to one in order to keep track of the two lags that are used in the example. Note that these and the subsequent steps of the simulation procedure will only occur when the order of the activity bundles (ℓ) does not exceed the simulated length.



Set *t*:=order of the transition prob. matrix (ℓ)
2
Set diarypointer:=0
Empty diary
0
Read ℓ-th order transition prob. matrix
Simulate length
15
Simulate first ℓ elements
Sleep
Eat
1
diarypointer:=diarypointer+1
Search for the combination of elements in the interval from [diarypointer..*t*] in the transition probability matrix & simulate the next element
Transportation by car
While *t*<length
If $t \neq 1$ then diarypointer:=diarypointer+1
Store the element at position *t* (X_t) in diary
Sleep
Eat
Transp. by car
2

Figure 5.6: Description of the simulation procedure

The next step is to search for the combinations of elements in the transition probability matrix in the interval [diarypointer..*t*]. This means for our example that the Sleep-Eat-combination is looked up in the transition matrix and that the distribution which is in this row of the table is used as a constraint for simulating the next activity (also referred to as intra-sequential simulation, see infra). If no combination of elements is found, the procedure stops simulating elements for this particular activity pattern (not shown in Figure 5.6).

After the "diarypointer" and the index *t* are augmented, the chosen element ("transportation by car" in our example) is stored at position *t* (i.e. 3) in the activity pattern. The simulation procedure is repetitive, i.e. when the prediction of the value X_t is based on two lags, then the next value to be predicted becomes X_{t+1}, which is based on X_t (predicted in previous step) and on X_{t-1} (also referred to as inter-sequential simulation, see infra). This repetition continues until the generated activity pattern equals the simulated length of the pattern. This procedure is repeated for every activity pattern in the data set.

5.6.2 CONTROLLED SIMULATION

In the previous section, the term "intra-sequential" simulation has been briefly mentioned. The term means that only the intelligence information that is captured in the order of the transition probability matrix, is taken into account for the simulation of the next elements in the activity-travel pattern. The example that was given previously, where a breakfast and a working activity are often separated by travel, since people often need (and report) transportation before starting the work activity, is intra-sequential when this activity-travel pattern is simulated from a third-order transition probability matrix. In contrast to this, inter-sequential dependencies arise from a new simulation loop as it was described in the basic simulation procedure above. While simulation loops are clearly interconnected (the elements that are simulated by means of loop A serve as input for a new loop B), some logical constraints might be lost in inter-sequential loops. This is a relatively small problem for interconnections in terms of activities because transition matrices are constructed for different time intervals, preserving the simulation of activities that are not randomly distributed over different time intervals (sleep activity for instance).

However, additional problems arise when transport modes are simulated inter-sequentially. Common sense let us believe that transport modes that are used in the beginning of the activity-travel sequence may re-occur in trips that are conducted at a later moment throughout the day and will most often be used for returning home as well. Conceptually, our method is able to deal with this problem by preventing that an odd number of entries of a particular transport mode occur in a sequence. This may indicate that different transport modes are used for the return and the departure of a particular trip. However, there are possible exceptions like driving around by car, (without a stop or activity reported) or moving the car to a different parking space closer to home, where an odd number of transport modes is feasible. In case of an odd number of transport modes, the activity-travel pattern can be re-simulated consecutively, until a particular parameter r_1 is attained that accounts for those exceptions as explained above.

Also, the number of trips/tours that are conducted in one activity-travel pattern may be limited. Since location information is not incorporated at this stage in the simulation framework, a tour refers to any appearance/subsequence of (out-of-home) activity(ies) between two transport modes. Also in this case, if the simulation approach exceeds a particular value *b* (maximum number of tours), activity-travel pattern can be re-simulated consecutively. Again, a parameter r_2 is used to account for exceptions.

5.7 Empirical Results

5.7.1 Data Preparation

Obviously, all the separate data sets (concerning the different decision facets), that were used in the Albatross model, are no longer needed in this simulation framework. However, the full activity diary data and information which is in the Albatross model, is used in these experiments. Obviously, a couple of data transformations –which are mainly of technical nature– need to be carried out, such as for instance the transformation of the data into separate sequences of activity-travel combinations. A computer code has been established to automate this transformation process. Also in this case, our data were separated into a training and test set, consistent with the procedure that has been applied in

Chapters 2 till 4. In the remainder of this chapter, all the definitions and sections that have been explained previously, will now be empirically applied and validated.

5.7.2 First-Order Transition Probabilities

Intuitive interpretation

The collapsed form of the first-order transition probability matrices are 23 by 23 matrices for both developed algorithms (see sections 5.4.1 and 5.4.2), as 23 different activity and travel categories were distinguished in the activity diary. At first glance, the sequential information that both approaches have revealed seems intuitively logical and useful for low-order combinations. For the sake of clarity, we illustrate this by means of the following 5x5 matrix, which is taken from the non-normalized 23x23 matrix that was developed in approach (algorithm) 2 (see Table 5.4).

Table 5.4: A 5x5 first-order transition frequency matrix (empirical data)

				X_t			
	X_{t-1}	Sleep	Eat/drink	Transport (car)	Work (out-of-home)	In-home Leisure	…
	Sleep	3.7	640.7	149.9	0	319.5	…
Q =	Eat/drink	71.29	7.0	228.9	34.2	481.5	…
	Transport(car)	28.1	113.4	0.1	251.2	178.5	…
	Work	1.1	42.3	585.9	5.1	0.01	…
	In-home Leisure	293.9	145.4	118.1	0	551.6	…
	…	…	…	…	…	…	…

It is for instance easy to understand that a 'Sleep'-activity is most often (weighted value of 640.7) followed by an 'Eat/drink'-activity. On the other hand, the 'Sleep'-activity is never immediately followed by a 'Work' (out-of-home)-activity since people first need transportation to arrive there. The numbers on the diagonal represent people who in succession (erroneously or not) report the same activity. The transition matrix that is calculated by means of approach 1, shows similar logical relationships. Obviously, only integer numbers occur in this matrix.

Evidence of First-Order Sequential Dependencies

Obviously, building first-order sequential dependency matrices, is only useful in the case that first-order sequential dependencies are really present in the data. It was explained before that the Chi-square and the G^2 statistic (see definitions 5.6 and 5.7), are able to test whether first-order sequential dependencies are present in the data, that is, if the state of the system at time *t* depends on the state at time *t-1*. Unfortunately however, when empirically testing the 23x23 matrix, some expected values are below 5, thereby violating the basic assumptions for the use of this test. In addition to this, there are quite some cells in the matrix that contain zero's, representing sequence-pair combinations that never occur in the data. As a result of this, a significant Chi-Square or G^2-value is easy to achieve. Both problems can be solved by creating a more compact transition matrix that generalizes entries in the matrix into broader activity categories. To this end, an aggregated 9x9 transition matrix was constructed, containing in-home, bring/get, shopping/service, leisure/social, work and other activities and car, slow modes and public transport as transport modes in the matrix. By consequence, the basic Chi-square assumption was not violated. Also for this matrix, both chi-square and G^2 were highly significant ($p<0.0001$) for all transition matrices (approach 1, approach 2 and MLE), thereby not accepting the H_0-hypothesis. This should not be surprising of course, since it is both clear from common sense but also from the analysis of the partial transition matrix shown in Table 5.4 that the transitions in the cells are significantly different than would be expected if the combinations are randomly sequenced. To this end, it is safe to assume that the state of the system at time *t* depends on the state at time *t-1,* and that first–order sequential dependencies are present in the data.

Approach 2 Versus MLE

As mentioned before, instead of using approach 2 for the computation of transition probabilities, simple MLE can be used. While approach 1 seemed less fair in comparison with MLE, it was claimed in section 5.4 that approach 2 preserved the independent character of every sequence and by consequence, estimates seemed to be less biased by specific combinations that appear in one particular sequence.

In order to evaluate this empirically, transition probability matrices (23x23) were respectively constructed by means of simple MLE estimates and by means of the algorithm that was introduced in Approach 2. Differences between both transition matrices (Approach2-MLE) were then calculated in order to empirically evaluate the magnitude of the problem as introduced in section 5.3. When the difference has a negative sign, this indicates that MLE estimates are biased due to an overestimation of the probability because of the repetitive inclusion of these specific activity combinations in some activity-travel patterns, while they are almost non-existing in other patterns (see example in the problem statement in section 5.3). A positive sign on the other hand is the result of an underestimation of the probability by means of MLE, because specific combinations occur more uniformly over all respondents and thus result in a higher probability per respondent by means of Approach 2. The experiment has been graphically displayed in Figure 5.7. It can be seen from this figure that differences in probability between both approaches range from –10% to +8%, which is quite substantial. The negative signs (indicating an overestimation of the transition probabilities) are most remarkable for sequence pair combinations that start and end with In-home leisure, In-home non-leisure and out-of-home non-leisure activities. Positive signs (indicating an underestimation of transition probabilities) occur more frequently. Probably the most important sequence pair combinations that differ are combinations that start with "car driver", "walk" and "public transport" and that are immediately followed by work (out-of-home). Those sequence pair combinations are respectively underestimated 6.8%, 5.5% and 7.8% by MLE. Given the important character of these sequence pair combinations for trip generation, the more accurate estimation of transition probabilities is an important strength in comparison with the standard probability estimation (MLE) in Markov Chains. Amongst others, underestimates also occur for the combinations Work (in-home) --- In-home Leisure (5,2%); In-home-leisure --- Sleep (5,7%) and In-home-non-leisure---In-home-leisure (6,2%).

Based on these results, the development of Algorithms 1 and 2 has proven to be justified and useful.

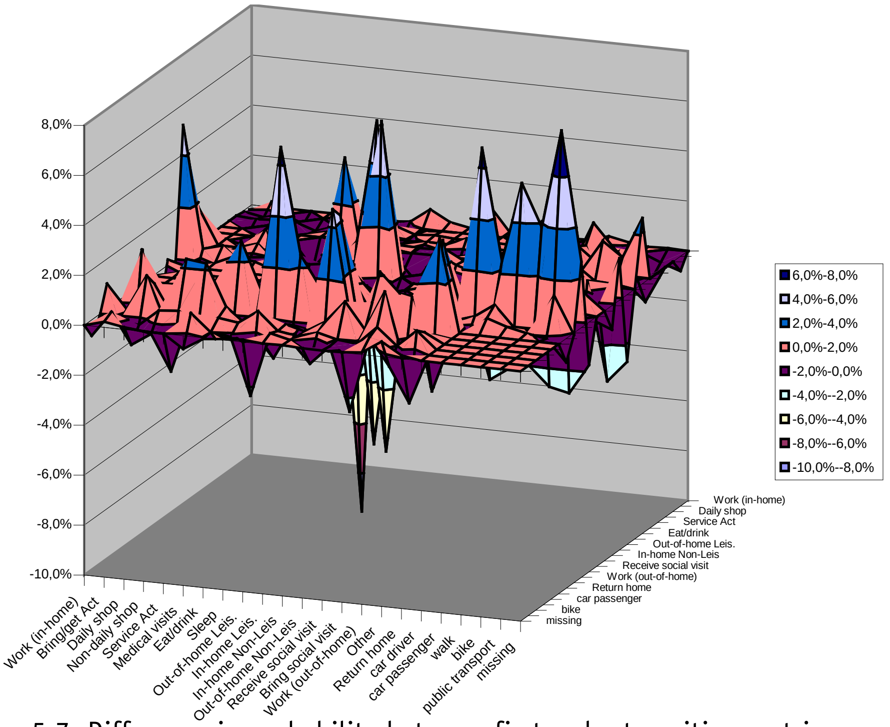


Figure 5.7: Difference in probability between first-order transition matrices, comparing MLE with Approach 2 (Approach 2 - MLE)

5.7.3 HIGHER-ORDER TRANSITION PROBABILITIES

INTUITIVE INTERPRETATION

In this section we consider a second-order transition matrix that has been constructed by means of the *n*-grams method that was introduced in section 5.2.2. Once more, only a 5x5 matrix (from approach 2) is reported for the sake of clarity in Table 5.5.

Second-order transition probability matrices are not different when compared to a first-order matrix, except for the additional dimension which need to be added to these tables. Obviously, also the size of the matrix will become variable. In our first approach (majority rule), the size of Q is 283x23; while in the second approach it is 325x23. It can be seen from Table 5.5 that the same previously given intuitive explanations seem to be valid. Making an intuitive interpretation

Table 5.5: A 5x5 second-order transition frequency matrix (empirical data)

			X_t					
	X_{t-2}	X_{t-1}	Sleep	Eat/ drink	Transp. (car)	Work (out-of- home)	In-home Leisure	…
	Sleep	Eat/drink	45	3	211.5	0.5	214.5	…
	Eat/drink	Transp.(car)	1.5	3	0	236.8	6.1	…
$Q=$	Transp.(car)	Work	0	47.5	539.4	3	0	…
	Work	Transp.(car)	16.3	163.4	0	31.7	202.6	…
	Eat/drink	In-Home Leis.	33.4	37.1	142.9	0	371.4	…
	…	…	…	…	…	…	…	…

for higher-order (>2) combinations is almost infeasible because long combinations of sequence elements are unable to be fully captured by the human brain. In these cases, quantitative evaluation measures seem to be better suited (see infra).

Evidence of Higher-Order Sequential Dependencies

While first-order sequential dependencies have proven to be highly present in activity-travel patterns, it is also useful to examine the presence of higher-order dependencies. As mentioned before, the required statistical tests depend upon the method that is used for calculating higher-order transition probabilities. Two methods were distinguished: the simple higher-order transition (frequency and probability) matrix (ignoring the information in between two states) and the *n*-grams method.

The first method can be tested by means of a simple χ^2-test and by constructing transitional frequency matrices between state *t-k* and state *t*. The χ^2-values for each order were calculated and only the second lag (order) proved to be significant. When a full activity diary is considered, there is no intuitive reason why there needs to be a sequential dependency between an activity which is reported at position *t* in the diary and an activity that is reported at position *t*+5 for instance, given the heterogeneity by which people fill out their diaries. However, when transition matrices are built for specific time intervals, a different pattern emerges. For instance, given the fixed format that is used for every diary (starts and ends at 3 AM), the second, third and fourth lag have proven to be significant during a morning time interval (6AM–9AM). This is also logical because the overall heterogeneity in diaries is reduced by concentrating on one specific time interval. This finding is a first indication that the stationarity condition does not hold for our data (see also section 5.7.4).

The second method that is used for constructing higher-order sequential dependencies, takes in-between information into account between state *t* and state *t-k* (*n*-grams-method). Definition 5.9 has introduced a method which can be used to test higher-order dependencies by means of *n*-grams. Results are shown in Table 5.6 for some higher-order (grouped into broader activity categories) transition probability matrices. It can be seen from this table that the second- till the fifth-order transition matrices are significant, which means that the present state of the system is significantly influenced by the sequential dependencies which are taken into account 2, 3, 4 and 5 lags before. As mentioned before, since this matrix is not used for analyses and explanatory purposes, this test has no further implications for our simulation as such. The test only indicates that higher-order dependencies are present in the data and that a simulation procedure that takes these higher-order transition matrices into account, may result in more accurately simulated activity-travel patterns. However, the ultimate validation is the comparison between the observed and the simulated activity-travel patterns on the test set, even if a non-significant order is used in the transition matrix. Obviously, it is likely that there is a strong correlation between the significance of the matrix and the accuracy of the simulated activity-travel pattern (see infra). Given these findings, the term "higher-order transition probability matrix" is referred to as the matrix that takes sequential information in between two states into account (*n*-grams) in the remainder of this chapter.

Table 5.6: Evidence of higher-order sequential dependencies (*n*-grams method)

Order (*k*)	2	3	4	5	6	7	8	9	10
P	0.001	0.01	0.03	0.04	0.08	0.11	0.15	0.19	0.24

Own Approaches Versus MLE

It was already shown in Figure 5.7 that approach 2 resulted in transition probabilities that range from –10% to +8% compared with MLE in Markov Chains (first-order dependencies). As explained, the under- or overestimation in MLE was the result of treating all different independent activity patterns (of different respondents) as one activity-travel pattern, thereby ignoring the independence over different respondents. This argument still holds for higher-order dependencies. However, in addition to this, another important reason is the computational explosion that was already introduced in the problem statement in section 5.3.

For a better understanding, the reader may consider the size of some final transition probability matrices for approaches 1 and 2 in Table 5.7. Each value in the table has been experimentally derived from our data for these approaches. They represent the total number of rows in the transition frequency matrix. The total number of columns is always a constant. A theoretical comparison is also made with Markov Chains to illustrate the computational efficiency of Approaches 1 and 2. High-order Markov-chain transition matrices were not empirically built because it is simply impossible given the computational explosion of the calculations.

This significant decrease in computation complexity is mainly due to the facts that only the activity bundle which immediately precedes the current higher-order activity bundle is used for the construction and that only combinations which actually occur in the data are considered. In Markov Chains all activity combinations are considered, and even when the major part of these combinations will have a zero probability (because most combinations never occur in the data) extra passes are needed through the data and accordingly extra computer memory is consumed.

Table 5.7: Maximal number of independent parameters For Markov Chains

Order	Approach 1	Approach 2	Markov Chain
1	23	23	23
2	283	325	529
3	1233	1596	12167
4	2451	3761	279841
5	3880	6535	6436343
6	4803	8974	1.48E+08
7	5190	10362	3.4E +09
8	5193	10714	7.83E +10
9	4904	10342	1.8E+12
10	4443	9593	4.14E+13

5.7.4 TESTING FOR STATIONARITY

It was explained in definition 5.5 how it could be examined whether data violated the stationarity condition or not. When no stationarity is found in the data, different transition probability matrices need to be developed for different time periods, thereby empirically determining the different (time) cutpoints (bifurcation points) that result in separate time windows during the day. However, before this can be done, the existence or non-existence of stationarity in the data is examined in this section. This is done by assuming that heterogeneous activity-travel patterns exist for different time periods. Different time periods have been considered in Table 5.8 based on some a-priori made assumptions. Chi-square values are computed by application of definition 5.5. The result of this statistical test is more meaningful if the data is pre-processed first by equalizing the length of the different activity sequences. This takes away the effect of detail by which people fill out activity diaries, and it treats all time periods uniformly. To this end, the training data was preprocessed in equal time periods of 30 minutes. The activity or travel mode that the respondent initially reported was thus assigned to the corresponding time interval. As a result of this, all diaries contain 48 entries over a 24-hour period.

Table 5.8: The result of the stationarity test on a-priori defined cutpoints

Number of time periods	Cutpoints for time periods	p-value
2	24PM-12AM; 12AM-24PM	0.27
3	24PM-8AM; 8AM-16PM;16PM-24PM	0.019
4	24PM-6AM;6AM-12AM;12AM-18PM;18PM-24PM	0.015
6	24PM-4AM;4AM-8AM;8AM-12AM;12AM-16PM;16PM-20PM; 20PM-24PM	0.062

It can be seen from this table that there are only two time periods which are significantly different. For a further explanation, we refer to the following section.

Nevertheless, it should already by pointed out here that these results show that we were unable to conclude that stationarity was present in our data. In other words, based on these initially chosen cutpoints, different transition probability matrices should be developed for these time periods. However, as mentioned before, it is unclear until now whether these cutpoints are optimal bifurcation points, because no data-information as such, is used in the determination of these points. In order to achieve this, we need to iteratively apply the omnibus test for all possible splitting points of a sequence, as described in the procedure in section 5.5.2. The results of this procedure are described in the following section.

5.7.5 Segmenting Transition Probability Matrices

Bifurcation Points (Temporal Segmentation)

The iterative application of the omnibus test is a computationally demanding procedure. In order to reduce the computational burden, we have defined time periods of 60 minutes. Consequently, there are 24 potential bifurcation points in the beginning of the procedure. As explained before in section 5.5.2, time windows will gradually be combined together, ending up with two time windows in the end, and every time defining new potential bifurcation points. The level of significance in our experiments was set at 5%. An evolution of the p-values is shown in Figure 5.8. The first p-value in the figure is the result of a comparison between 24 time windows (i.e. one transition matrix for every hour in the day). While there are obviously large (significant) differences between transition matrices that are built during morning periods (e.g. 3AM-4AM) and noon periods (e.g. 12AM-13PM), the differences are non-significant when the full day is considered. The reason for this is that during the majority of the day (except for some specific time periods), activity-travel combinations are more or less randomly distributed and majority patterns flatten out the significant differences. In other words, the dynamics of the system do not change (alter) significantly every hour. In order to determine a more significant change in dynamics, more aggregated time periods need to be considered. For this reason, it may be somewhat surprising that time periods dividing the diary in for instance 8 time periods (i.e. every three hours) were found not significantly different during one day. Also in this case, while there are significant differences between the time periods 3AM-6AM and 12AM-3PM; the majority of the three-hour during time periods appeared to be non-significant. The significant effect starts to appear from 5 time periods (i.e. every 4 hours and 48 minutes) and ranges till 3 time periods (every 6 hours). As was already shown in Table 5.8, two time periods (lasting 12 hours each) were found not significantly different. One possible explanation is that the (frequently occurring) work-, sleep- and travel-combinations appear fairly equal in both time windows.

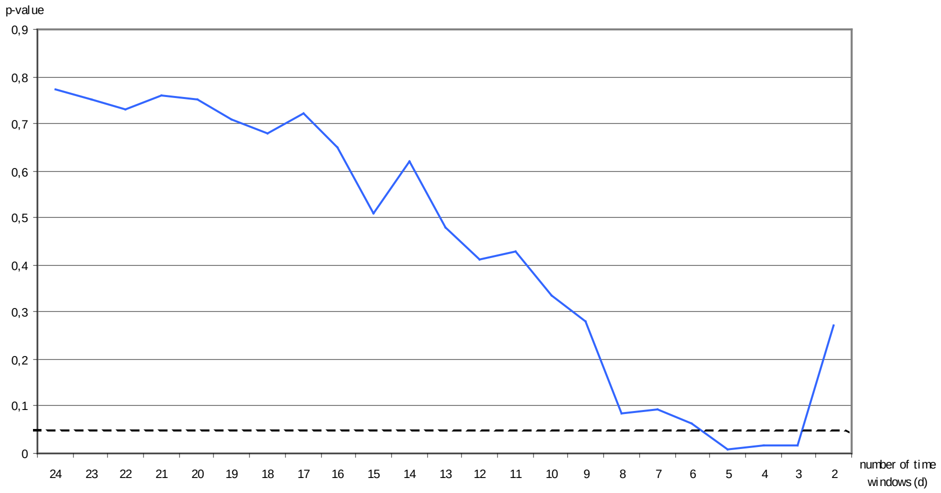


Figure 5.8: Evolution of p-values for the procedure described in section 5.5.2

Obviously, these are important results for the simulation framework that is described in section 5.7.6. Indeed, it has now been experimentally determined that we should rely upon different transition probability matrices when predicting activity-travel combinations for these different time windows during the day. As mentioned before, despite the fact that our simulation method does not yet explicitly simulate time information as such (see Chapter 6), we need to take this information into account in order to simulate the most accurate activity-travel combinations. In our simulation, we will rely upon the first finding that was found significant, i.e. 5 time periods, defined as 3AM – 7:48AM; 7:48AM-12:36PM; 12:36PM-17:24PM; 17:24PM-22:12PM and 22:12PM-3AM.

Full Decision Trees (Socio-demographic Segmentation)

Segmenting transition matrices in terms of socio-demographic variables, assumes the execution of the procedure that was explained in section 5.5.3. The n_{min} parameter and the minimum gain ratio (G_{min}) were respectively arbitrarily set at 75 cases and at 0.05 to prevent overfitting of the tree on the training data. An example decision tree that was built for our data is shown in Figure 5.9. In addition to the structure of the decision tree, every decision node shows the number of cases that go down that branch, the maximum gain ratio and the information value that was achieved. Every leaf node, containing different transition probability matrices was indicated by (L). It can be seen from this tree that the variable "number of cars" ("Ncar") was the most important variable in the tree, followed by Household type ("Hhtype"), gender ("Gender") and socio-economic class ("Sec"). Having applied temporal and socio-demographic segmentation and all the required statistical tests that were introduced before, the full "knowledge model" is finalized and we are now ready to move on to the simulation of new activity-travel patterns.

5.7.6 Simulating Activity-travel Patterns

Preface

The previous sections have described a number of improvements and refinements to the transition probability matrix as it is used in traditional Markov Chain modelling. Improvements were proposed with respect to (i) a more efficient and more accurate calculation of the transition probabilities (approaches 1 and 2); (ii) the use of a different number of lags (lower-order versus higher-order transition matrices) and (iii) the use of a segmentation scheme (temporal and socio-demographic) in the development of these transition matrices.

```
(start)
| NCAR=1  (900, INFO = 2.727906, G = 0.058563)
| | HHTYPE=3  (311, INFO = 2.724312, G = 0.123689)
| | | GENDER=1  (309, INFO = 2.725375, G = 0.057060)
| | | | SEC=3  (134, INFO = 2.625469, G = 0.101251)
| | | | | BIKEAV=1  (124, INFO = 2.604755, G = 0.125868)
| | | | | | CARAV=1  (117, INFO = 2.602145, G = 0.098740)
| | | | | | | CHILDREN=2  (L)
| | | | | | | CHILDREN=1  (L)
| | | | | | | CHILDREN=3  (L)
| | | | | | | CHILDREN=4  (L)
| | | | | | CARAV=unknown  (L)
| | | | | BIKEAV=unknown  (L)
| | | | SEC=1  (L)
| | | | SEC=unknown  (L)
| | | | SEC=2  (96, INFO = 2.684212, G = 0.128843)
| | | | | BIKEAV=1  (83, INFO = 2.650120, G = 0.130249)
| | | | | | CHILDREN=2  (L)
| | | | | | CHILDREN=1  (L)
| | | | | | CHILDREN=3  (L)
| | | | | | CHILDREN=4  (L)
| | | | | BIKEAV=unknown (L)
| | | | SEC=4  (L)
| | | GENDER=2  (L)
| | HHTYPE=1  (L)
| | HHTYPE=4  (239, INFO = 2.697180, G = 0.074627)
| | | GENDER=1  (212, INFO = 2.680736, G = 0.080723)
| | | | AGE=2  (153, INFO = 2.663894, G = 0.089576)
| | | | | SEC=3  (L)
| | | | | SEC=1  (L)
| | | | | SEC=unknown  (L)
| | | | | SEC=2  (L)
| | | | | SEC=4  (L)
| | | | AGE=3  (L)
| | | | AGE=4  (L)
| | | | AGE=1  (L)
| | | GENDER=2  (L)
| | HHTYPE=5  (140, INFO = 2.651551, G = 0.114153)
| | | GENDER=1  (132, INFO = 2.650076, G = 0.095268)
| | | | SEC=3  (L)
| | | | SEC=1  (L)
| | | | SEC=unknown  (L)
| | | | SEC=2  (L)
| | | | SEC=4  (L)
| | | GENDER=2  (L)
| | HHTYPE=2  (160, INFO = 2.560276,   G = 0.139788)
```

```
(continued)
| | | CARAV=1  (154, INFO = 2.550203, G =
              0.105228)
| | | | NBIKES=4  (L)
| | | | NBIKES=1  (76, INFO = 2.505884,
                G = 0.125667)
| | | | | AGE=2  (L)
| | | | | AGE=3  (L)
| | | | | AGE=4  (L)
| | | | | AGE=1  (L)
| | | | NBIKES=unknown  (L)
| | | | NBIKES=2  (L)
| | | | NBIKES=3  (L)
| | | | NBIKES=0  (L)
| | | CARAV=unknown  (L)
| | HHTYPE=unknown  (L)
| NCAR=unknown  (164, INFO = 2.627780,
              G = 0.087061)
| | SEC=3  (L)
| | SEC=1  (L)
| | SEC=unknown  (L)
| | SEC=2  (L)
| | SEC=4  (L)
| NCAR=2  (290, INFO = 2.656737,
        G = 0.062742)
| | GENDER=1  (267, INFO = 2.649541,
            G = 0.061261)
| | | SEC=3  (91, INFO = 2.520677,
            G = 0.109946)
| | | | CHILDREN=2  (L)
| | | | CHILDREN=1  (L)
| | | | CHILDREN=3  (L)
| | | | CHILDREN=4  (L)
| | | SEC=1  (L)
| | | SEC=unknown  (L)
| | | SEC=2  (L)
| | | SEC=4  (131, INFO = 2.639051,
            G = 0.094076)
| | | | CHILDREN=2  (L)
| | | | CHILDREN=1  (L)
| | | | CHILDREN=3  (L)
| | | | CHILDREN=4  (L)
| | GENDER=2  (L)
| NCAR=5  (L)
| NCAR=0  (L)
| NCAR=3  (L)
| NCAR=4  (L) (end)
```

Figure 5.9: A final sequential information decision tree (empirical data)

The aim for doing all this is to end up with the most “optimal” predicted subsequence of activities and transport modes, i.e. that particular predicted activity travel pattern that coincides best with the observed activity travel pattern. It is assumed that a more accurately predicted activity-travel pattern, finally also results in more accurate predictions of the time and location facets of the simulation. The aim of this section is to report upon the contribution for each of these three improvements when this ultimate validation measure (i.e. the predicted activity-travel pattern) is adopted.

Data were divided into a training and a validation set, thereby using the training set for building the model (transition matrices, segmentation tree, etc.), while the unseen test data were used for validation. Activity-travel patterns were simulated both for the training and the test data. The goodness-of-fit for the simulated diaries was measured by comparing the generated activity patterns with the observed patterns in the training and the test dataset. The comparison was measured using the following two indicators:

- Pattern level attributes (number of tours)
- Trip level attributes (trip rates)

In addition to this, the computational complexity will be evaluated in the final paragraph of this section.

More advanced and comprehensive evaluation measures such as SAM (see also previous Chapters 3 and 4) will be re-introduced in Chapter 7, when the prediction of the full activity-travel pattern (including location and time information) has been established.

Pattern Level Attributes: Number Of Tours

Pattern level attributes give an indication about the performance of the simulation framework at the highest level. Although there are other indicators at pattern level (for instance number of activity episodes per activity category), the evaluation was made at this level by comparing the *mean* number of tours in the observed and the generated patterns. A tour is defined as a subsequence of activities that start and end at the same base location. Since location information is not yet incorporated in this simulation framework, a tour refers to any appearance/subsequence of (out-of-home) activity(ies) between two transportation activities.

Without Segmentation

The first table in this section (Table 5.9) compares the mean number of tours between the observed and the predicted patterns for the training and test dataset. The comparison is made by differentiating between the order of the transition matrices which are used (ℓ) to build the transition probability matrices and between both approaches. No segmentation has been used, which means that only one transition matrix per order has been used for the prediction of the training and test data.

Table 5.9: Comparing the observed and predicted mean number of tours by differentiating between the order of the activity bundles (without segmentation)

Training dataset				Test dataset			
Observed Tours (mean)	Order of Q	Predicted Tours (mean) (Approach 1)	Predicted Tours (mean) (Approach 2)	Observed Tours (mean)	Order of Q	Predicted Tours (mean) (Approach 1)	Predicted Tours (mean) (Approach 2)
2.801	ℓ=1	1.223*	1.722*	2.435	ℓ=1	1.321*	1.621*
	ℓ=2	1.874*	1.975*		ℓ=2	1.745*	1.954*
	ℓ=3	1.941*	1.949*		ℓ=3	2.148*	2.128*
	ℓ=4	2.481*	2.563*		ℓ=4	2.312	2.316
	ℓ=5	2.692	2.732		ℓ=5	2.414	2.424
	ℓ=6	2.690	2.779		ℓ=6	2.721*	2.621*
	ℓ=7	2.781	2.821		ℓ=7	2.732*	2.730*
	ℓ=8	2.212*	2.262*		ℓ=8	2.012*	2.003*
	ℓ=9	1.931*	1.951*		ℓ=9	1.521*	1.621*
	ℓ=10	1.201*	1.312*		ℓ=10	0.927*	1.222*

* Statistically significant difference in means (observed vs predicted) at the 95 percent level of confidence

There are a number of conclusions which can be derived from this table. First, the results of the training set, at the left part of the table, give an indication about how well the framework is capable of capturing and simulating the information which is incorporated in the training data. It can be seen that high-order combinations are not well suited to generate reliable patterns of activities. This seems counter-intuitive at first sight. Indeed, one might expect that the more sequential information that is incorporated in the transition probability matrices, the more reliable the simulations tend to be. This turned out to be only true to some extent (only till ℓ=7). Indeed, recall Figure 5.6, where it was explained that one of the first steps is to draw the first ℓ elements from the prior sample distributions. This means for high numbers of ℓ that the simulated activity-travel patterns are much larger than the diaries in the sample data, which damages the accuracy of the results.

It can also be seen that although there is little difference between approach 1 and approach 2, the latter generates slightly better results for low-order combinations. For both approaches, transition matrices with orders ranging from 5 till 7, turned out to generate no significant difference with respect to the training dataset. When we compare these results with the data which is generated for the test set, it appears that 6th and 7th order transition matrices slightly overfit the training data, i.e. the good performance on the training data could not be kept on the unseen test data. In this case, the 4th and 5th order transition matrices seem to generate the best fit. Again, while approach 2 gives a slightly higher accuracy, there are no important significant differences between approaches 1 and 2.

With Segmentation

A completely different picture arises when time and socio-demographic segmentation is taken into account. This means, that transition matrices are made dependent on time windows and on socio-demographic information when simulating new activity-travel patterns. It can be seen from Table 5.10 that low-order combinations are preferred in this case. An explanation of this result may be that, since one is looking at subpatterns of a full activity-travel pattern (for instance dividing the activity travel pattern in five subsequences), higher-order transition matrices further deteriorate results. A similar pattern was already noticed with respect to the simulation results where no segmentation was used. However, a shift could thus be observed from a fifth-order transition matrix as being the optimal lag, towards first and second-order lags.

Table 5.10: Comparing the observed and predicted mean number of tours by differentiating between the order of the activity bundles (with segmentation)

Training dataset				Test dataset			
Observed Tours (mean)	Order of R	Predicted Tours (mean) (Approach 1)	Predicted Tours (mean) (Approach 2)	Observed Tours (mean)	Order of R	Predicted Tours (mean) (Approach 1)	Predicted Tours (mean) (Approach 2)
2.801	ℓ =1	2.695	2.732	2.435	ℓ =1	2.215	2.232
	ℓ =2	2.698	2.737		ℓ =2	2.314	2.319
	ℓ =3	2.501*	2.512*		ℓ =3	2.112*	2.141*
	ℓ =4	1.921*	1.856*		ℓ =4	1.543*	1.510*
	ℓ =5	1.872*	1.821*		ℓ =5	1.541*	1.432*

*Statistically significant difference in means (observed vs predicted) at the 95 percent level of confidence

Trip Level Attributes: Trip Rate

Trip level attributes are lower in hierarchy, which means that not the whole pattern but the individual trip is taken as the relevant unit of analysis in the evaluation. Typically, trips are differentiated here by means of the main purpose for which the trip is undertaken. The mean trip rate is used as an evaluation measure. The mean trip rate is defined as the mean number of trips that a person has done during one particular day.

Without Segmentation

Table 5.11 shows the predicted patterns for the test set. It is obvious that the performance indicator "Trip Rates" compares sequences at a more detailed level (it does not compare large patterns but individual trips). This can also be seen from the results which are shown in Table 5.11. Indeed, low-order transition matrices seem to result in more accurate results than higher-order transition matrices (larger than 5th order transition matrices were not shown for this reason). More specifically, transition matrices ranging from orders 1 till 3 seem to give the best predictions for the first approach. The same could be said for the second approach, but in this case transition matrices of order 3 clearly outperform the others.

The training set results were not shown because the overfitting was rather low, which means that also for the training set, the low-order transition matrices achieved the highest accuracy.

Table 5.11: Comparing observed and predicted mean trip rates by differentiating between the order of Activity Bundles (Test Set) (without segmentation)

Purpose	Observed Trip Rate (mean)	Predicted Trip Rate (mean)									
		Approach 1					Approach 2				
		ℓ =1	ℓ =2	ℓ =3	ℓ =4	ℓ =5	ℓ =1	ℓ =2	ℓ =3	ℓ =4	ℓ =5
Work	0.735	0.749	0.741	0.631*	0.598*	0.551*	0.744	0.742	0.634*	0.603*	0.557*
SL	0.569	0.601	0.592	0.518*	0.493*	0.359*	0.597	0.595	0.496*	0.513*	0.375*
Service	0.496	0.509	0.513	0.434*	0.411*	0.379*	0.502	0.511	0.413*	0.410*	0.373*
B/G	0.274	0.280	0.282	0.315*	0.312*	0.217*	0.284	0.281	0.311*	0.209*	0.210*
Other	0.134	0.141	0.140	0.160*	0.152*	0.101*	0.147	0.144	0.165*	0.153*	0.108*

Work: Work out-of-home activity; SL: Social or Leisure out-of-home activity; Service: Shopping or other service related activity; B/G: Bringing/Getting persons or goods; Other: Other out-of-home activity. *Statistically significant difference in means (observed vs predicted) at the 95 percent level of confidence

With Segmentation

The same conclusion was reached when time and socio-demographic segmentation was taken into account. However, in this case the first- and second-order transition matrices are preferred and this for both approaches (see Table 5.12).

Table 5.12: Comparing the observed and predicted mean trip rates by differentiating between the order of Activity Bundles (Test Set) (with segmentation)

Purpose	Observed Trip Rate (mean)	Predicted Trip Rate (mean)									
		Approach 1					Approach 2				
		ℓ =1	ℓ =2	ℓ =3	ℓ =4	ℓ =5	ℓ =1	ℓ =2	ℓ =3	ℓ =4	ℓ =5
Work	0.735	0.811*	0.742	0.721	0.612*	0.521*	0.823*	0.693	0.739	0.632*	0.614*
SL	0.569	0.641*	0.576	0.559	0.491*	0.359*	0.471*	0.510*	0.560	0.461*	0.421*
Service	0.496	0.571*	0.521	0.511	0.421*	0.368*	0.431*	0.435*	0.488	0.441*	0.401*
B/G	0.274	0.285	0.280	0.311*	0.235*	0.221*	0.236*	0.235*	0.267	0.306*	0.221*
Other	0.134	0.314*	0.121	0.142	0.161*	0.163*	0.153*	0.157*	0.126	0.101*	0.09*

*Statistically significant difference in means (observed vs predicted) at the 95 percent level of confidence

COMPUTATIONAL COMPLEXITY

The computational complexity of the two approaches that have been advanced in this section, is mainly determined by the order of the activity bundles (ℓ), the adopted approach and by the number of states (*m*). The results on the computation time of different orders, can be seen from Table 5.13. Approach 2 is clearly more computationally demanding than Approach 1, which is logical, given the fact that more detailed activity bundles (using maximum likelihood estimates) need to be calculated. In our experiments, the number of states did not vary, so we experienced no additional computational burden from that parameter. It should also be noted that the reported times only involve the "knowledge acquisition" phase of the proposed method, i.e. the extraction of sequential information that is incorporated in the data.

Table 5.13: Computation time of multiple orders (approach 1 versus approach2)

	ℓ =1	ℓ =2	ℓ =3	ℓ =4	ℓ =5
Approach 1 (seconds)	45	272	456	912	2124
Approach 2 (seconds)	73	389	498	1102	2376

5.8 CONCLUSION

In this chapter, a data-driven heuristic simulation procedure of activity and travel information has been proposed. Both dimensions are only at a later stage complemented by means of time and location information. However, it was assumed that a more accurately predicted activity-travel pattern, finally also results in more accurate predictions of the time and location facets of the simulation. The simulation procedure uses knowledge information that is embedded in the form of transition matrices. Each element in a transition matrix represents the probability of the transition from a particular state (represented by the row of the matrix) to the next state (represented by the column of the matrix). The simulation procedure that has been proposed in this section, differs from most other existing activity-based models, because it explicitly accounts for sequential information and sequential dependencies that are present in activity diaries. Tests showed that these dependencies were highly present in our activity-diary data. This is an important finding because the model was almost completely data-driven. As a result of this, its predictive performance and accuracy largely depends upon the information in the data. In order to improve prediction capabilities, the model accounted both for lower and higher-order sequential dependency information that might be present in the data. For reasons of efficiency and accuracy, modified heuristic computation algorithms that use the concept of activity bundles, were developed and tested in the chapter.

In the second part of the chapter, it was shown that the use of only one transition probability matrix which is both representative for every respondent and for every time frame during the day, is insufficient. To this end, a segmentation procedure has been introduced that enables one to cluster transition matrices in terms of time and socio-demographic information. The first segmentation used the technique of the identification of bifurcation points; the latter used a modified version of a decision tree, in the sense that sequential probability information was used during induction and in the leaves of the tree as opposed to the traditional way of only using one single classification attribute.

The empirical results, which compared the predicted mean number of tours and the mean trip rates with the observed values in the data, seem to indicate a difference in the situation without segmentation (where 1 matrix is used) and with segmentation (where multiple matrices were used). In the case of no segmentation, the fourth and fifth order transition matrices seem to generate the best fit. When multiple transition matrices were used per segment, first and second order matrices performed better. The observation was consistent at pattern level (number of tours) and at trip level (trip rate). The fact that sequences per segment are much shorter when compared to full length diaries, may be an intuitive explanation for this finding, but more profound research should substantiate this. Further research should also be conducted in order to get a better idea about the relative performance of the techniques that have been advanced in this chapter against other clustering techniques that can be used to complement Markov Chains such as the MMLC application in Goulias (1999) and the Latent Class clustering application in Kim and Goulias (2004).

Chapter 6
Allocating Time and Location Information through Reinforcement Learning

6.1 INTRODUCTION

It was explained in the previous chapter how a sequence of activities and travel (transport modes) can be generated by means of low- and high-order sequential dependencies that are incorporated into transition probability matrices. However, now that the activity-travel sequence is known, time and location information still need to be allocated to end up with a more complete activity pattern. However, it seems less likely that time and location information can also be efficiently modeled by means of sequential dependency information that is in the data. To this end, it is assumed in this chapter that location and time information are determined by means of *interaction* within a particular space-time prism (and thus not independently). A technique has been postulated where a machine learning algorithm can learn its optimal starting and end times and location information through interaction with this environment. In our application, the environment is first of all pre-determined and bounded by the fixed sequence in which activities are performed, but other important factors are taken into account as well (see infra). The idea that people learn by interacting with the environment is inherent to the nature of learning as such. Whether somebody is learning to drive a car or holding a conversation, one is aware about how the environment responds to his/her actions, and about how he/she can have an influence on that environment through his/her behaviour. A modelling technique which aims to embody and simulate this behaviour in a (machine) learning environment, is called Reinforcement Learning, which is in fact a synonym for *learning by interaction* (Kaelbling *et al.*, 1996).

Reinforcement learning goes back to the very first stages of artificial intelligence and machine learning. As a result of this, the applications of reinforcement learning are situated in the basic roots of artificial intelligence, such as for instance game playing (Littman, 1994; Tesauro, 1992; Tesauro, 1994; Thrun, 1995) and robotics (Schaal and Atkeson, 1994; Mahadevan and Connell, 1992). However, there are also numerous other application domains such as for instance in elevator dispatching (Crites and Barto, 1996), natural systems (behaviour of ants for instance, Barto and Sutton, 1981); production scheduling (Schneider *et al.*, 1998); but also in a transportation-related context such as in intelligent lane selection (Moriarty and Langley, 1998) for achieving a higher traffic throughput. Within an activity-based framework, the reinforcement learning technique has been first applied by Arentze and Timmermans (2003) in the context of learning and adaptation and only very recently by Charypar *et al.* (2004), and by Charypar and Nagel (2005) in a time allocation problem.

During learning, the adaptive system (also called agent) tries some actions (i.e., output values) on its environment, then, it is reinforced by receiving a scalar evaluation (the reward) of its actions. Reinforcement learning tasks are generally treated in discrete time steps. Stated otherwise, at each time step t, the learning system receives some representation of the environment's state X_t, it takes an action a, and one step later it receives a scalar reward r, and finds itself in a new state X_{t+1}. The reinforcement learning algorithms selectively retain the outputs that maximize the received reward over time. The two basic concepts behind reinforcement learning are trial and error search (to improve the agents behaviour in an unknown environment) and delayed reward.

The difficulty in reinforcement learning is that we cannot reward the right action, because we do not know yet what that correct long-term action is, that’s in fact why we are learning. In other words, it is relatively easy to say what the best action is for the next state (because we get feedback on the actions taken), but it is less clear what combination of actions gives us the highest reward for that person during a particular day. Related to this is the trade-off between exploration and exploitation. To obtain a lot of reward, a reinforcement learning agent must prefer actions that it has tried in the past and found to be effective in producing reward. But to discover such actions, it has to try actions that it has not selected before. The agent has to *exploit* what it already knows in order to obtain reward, but it also has to *explore* in order to make better actions selections in the future.

In the remainder of this chapter we will elaborate on one particular algorithm (Q-learning, which is an abbreviation for “Quality-learning”) that has been introduced by Watkins (1989; Watkins and Dayan, 1992) and that can be used for solving the reinforcement learning problem. Q-learning is one of the most applied algorithms in reinforcement learning (Sutton and Barto, 1998). It was mentioned before that Charypar and Nagel (see Charypar *et al.*, 2004; Charypar and Nagel, 2005) used the Q-learning technique for time allocation within the area of activity-based transportation. However, their application was somewhat limited because they focused only on one artificial example (i.e. 1 respondent), and used a fixed sequence of only four activities to solve the allocation problem. Compared to their approach, there are some important contributions that have been added in this chapter. The first contribution is the evaluation towards real empirical data, including a more complex order of activity-travel combinations (see Chapter 5), the non-restriction to 4 activities and the incorporation of real-world and non-fixed travel times. Related with these contributions, is the evaluation on a larger empirical dataset, which means that time allocation is not restricted to only one activity-travel pattern. The most important contribution however, is the allocation of location information in the simulation of activity-travel patterns. Furthermore, the time and location allocation problem were treated and integrated simultaneously, which means that the respondents' reward is not only maximized in terms of minimum travel duration, but also *simultaneously* in terms of optimal time allocation.

With respect to the allocation of start and end times for a given fixed sequence of activities, it may not be sufficient to model the time allocation problem as a pair of the activity type and starting time. After all, being at work at 4 PM but having arrived at 9 AM, is a different state than being at work at 4 PM but having arrived at 1 PM. Therefore, the time that was already spent at that activity (duration) also needs to be taken into account. Obviously, the activity type is equally important because time allocation for shopping is different than time allocation for work activities. Time allocation is thus assumed to depend on the triple: type of the activity, starting time of the activity and the time already spent at that activity. This triple may be augmented in the future with other factors that are believed to have an impact on the time allocation problem. The only requirement for these dimensions is that it should be possible to specify scalar rewards. The scalar rewards *r* (which were already briefly mentioned before) are defined as reward tables, which -for each activity and each arrival time (starting time of activity)-, give the reward for staying one more time slice as a function of the duration of the activity. Rewards are thus given as a "utility per time slice", which corresponds to a marginal utility.

With respect to the allocation of location information, the travel time between two locations (origin and destination locations) is used and is made dependent on the transport mode that has been chosen for travelling from one location to another. Indeed, travel durations between two locations are obviously not equal over different transport modes, so it is warranted to take this dimension into account.

The remainder of this chapter has been organized as follows. In section 2, a short introduction into the basic concepts and definitions of Q-learning is given. In section 3, we will detail by means of artificial examples how Q-learning can be applied to the time and location allocation problem. Section 4 describes the empirical section, where a more detailed explanation is given about how the reward tables were derived from the empirical data, which parameter selection criteria have been used and which validation measures were adopted. The chapter ends with initial empirical conclusions.

6.2 Definitions and Algorithms in Reinforcement Learning

Definition 6.1: States

A reinforcement learning problem consists of discrete states x ($\in X$, a finite set). A particular state is defined by a number of dimensions which are assumed to characterize the current state. The number of states is finite and it is defined in advance (i.e. before learning starts). ■

Example: In a time allocation problem, a current state x can be characterized by the triple: activity, starting time of activity, time already spent at activity (duration). Working, 15 PM, 7 hours already spent at work is thus an example of a state.

Definition 6.2: Actions

At every discrete state x, a set of discrete actions a ($\in A$, a finite set) can be taken. Actions may be probabilistic or discrete. The number of actions is finite and is defined in advance. ■

Example: For a discrete state x, an action may be to stay at the current activity (state x_t) or to move on to the next state (x_{t+1}).

Definition 6.3: Rewards

At each discrete time step, the agent observes state x_t, takes action a_t, observes new state x_{t+1}, and receives an immediate reward *r(x,y)*. *r* is a function of the transition from state *x* to the new state *y*. ■

Example: If somebody is already working for 15 hours and has started working at 6 AM; working for another hour (new state *y*), may result in a low immediate reward. On the contrary, if somebody started working at 6 AM but is only working for 1 hour at the moment; working for another additional hour may result in a high immediate reward.

Definition 6.4: Probabilistic versus deterministic worlds

Transitions may be probabilistic, that is to say, *y* (and *y* being any new state, e.g. x_{t+1}) and *r* are drawn from probability distributions $P_{xa}(y)$ and $P_{xa}(r)$, where $P_{xa}(y)$ is the probability that taking action *a* in state *x* will lead to state *y* and $P_{xa}(r)$ is the probability that taking action *a* in state *x* will generate reward *r*. We have $\sum_y P_{xa}(y) = 1$ and $\sum_r P_{xa}(r) = 1$. The deterministic world is a special case with all transition probabilities equal to 1 or 0. For any pair *(x,a)*, there will be a unique state y_{xa} and a unique reward r_{xa} such that:

$$P_{xa}(y) = \begin{cases} 1 & \text{if } y = y_{xa} \\ 0 & \text{otherwise} \end{cases}$$

$$P_{xa}(r) = \begin{cases} 1 & \text{if } r = r_{xa} \\ 0 & \text{otherwise} \end{cases}$$

■

Example (deterministic world): If somebody is already working for 15 hours, and that person started working at 6 AM (state *x*) and the algorithm decides to stay at the current activity (action=stay), then there is only *one unique* state y_{xa}, which is defined by the triple: working, starting at 6 AM, duration 16 hours (assuming a discrete time window *t* of 1 hour). Equally, in the other case, if the action is to move to another activity (which is defined in advance by the sequence order); there is also only one unique state y_{xa}, which is defined by: sleeping (assuming that sleep is the next activity in the sequence order), starting at 22 PM (15hours + 6 AM), duration 0 hours (assuming a discrete time window *t* of 1 hour and that no travel is needed).

Definition 6.5: Expected Rewards

When we take a particular action *a* in state *x*, the reward that we expect to receive is:
$E(r) = \sum_y r(x,y)P_{xa}(y)$, where $P_{xa}(y)$ is the probability that taking action *a* in state *x* will lead to state *y,* and *r* is a function of the transition *x* to *y,* as mentioned before in definition 6.3.

In some models, rewards are associated with states, rather than with transitions, that is *r* = *r*(*y*). The agent is not just rewarded for arriving at state *y* - it is also rewarded continually for remaining in state *y*. This is just a special case of *r* = *r*(*x*,*y*) with $r(x,y) = r(y)\ \forall x$

and thus:

$$E(r) = \sum_y r(y)P_{xa}(y)$$

Kaelbling (1993) defines a globally consistent world as one in which, for a given *x,a*; E(*r*) is constant. Rewards *r* are bounded by r_{min}, r_{max}, where r_{min}<r_{max} (r_{min}=r_{max} would be a system where the reward was the same no matter what action was taken. The agent would always behave randomly). Hence for a given *x,a*; $r_{\min} \le E(r) \le r_{\max}$. ■

Example (deterministic world): If somebody started working at 6 AM, and is only working for 1 hour at the moment; working for another additional hour, may result in a reward of 0.25. If the action is to stay at the current activity, the next state (started working at 6 AM, working for 2 hours) may result in a reward of 0.35. The expected total reward in the deterministic case is equal to 0.6. This means that all possible actions for all possible states can be evaluated in this manner, by selecting state-action combinations that maximize the total reward.

Definition 6.6: Discounting factor

The agent should not only be interested in immediate rewards, but in the *total* discounted reward. In this measure, rewards received *n* steps into the future may be worth less than rewards received now, for instance by a factor of γ^n where 0≤γ≤1: R=r_t+γr_{t+1}+$\gamma^2 r_{t+2+\ldots}$. The *discounting factor* γ defines how much the expected future rewards, affect decisions now. Genuine immediate reinforcement learning is the special case γ=0, where we only try to maximize immediate reward. Low γ means that the agent should pay little attention to the future. High γ means that potential future rewards have a major influence on decisions now – and that one is willing to trade short-term loss for long-term gain. ■

Example: Working from 6 AM to 16 PM may give somebody a higher total expected reward than working from 10 AM to 20 PM, because in the first case there may be some time left to carry out leisure activities during the evening. Rewards in the future (or for instance during evening) may thus receive a higher reward by means of a high discounting factor.

Definition 6.7: Policy and value function

The agent acts according to a policy π which tells it what action to take in each state *x*. A policy that specifies a unique action to be performed a=π (*x*) is called a *deterministic* policy - as opposed to a *stochastic* policy, where an action *a* is chosen from a distribution P_x^π with probability $P_x^\pi(a)$. The task for the agent is to find an optimal policy - one that maximizes the total discounted expected reward.

The total discounted expected reward can be summarized by means of a value function, which is called the *value* of state *x* under policy π, and which is represented by $V^\pi(x)$.

$$V^\pi(x_t)=E(R)=$$

$$E(r_t)+\gamma E(r_{t+1})+\gamma^2 E(r_{t+2})+\ldots$$

$$E(r_t)+\gamma[E(r_{t+1})+\gamma E(r_{t+2})+\gamma^2 E(r_{t+3})+\ldots]$$

$$E(r_t)+\gamma V^\pi(x_{t+1})$$

$$\sum_r rP_{x_t a_t}(r)+\gamma\sum_y V^\pi(y)P_{x_t a_t}(y)$$

For any state *x*, there is a unique value $V^*(x)$ which is the best that an agent can do from state *x*. Optimal policies π^* may be non-unique, but the value V^* is unique. All optimal policies π^* will have: $V^*(x)=V^{\pi^*}(x)$. ■

Definition 6.8: Q-learning

Q-learning is a popular learning algorithm that can be used for solving the reinforcement learning problem. The strategy that the Q-learning agent adopts is to build up *Quality-values* (Q-values) $Q(x,a)$ for each pair (x,a). If the transition probabilities $P_{xa}(y)$ and $P_{xa}(r)$ are explicitly known, Dynamic Programming finds an optimal policy by starting with random $V(x)$ and random $Q(x,a)$ and repeating forever (or at least until the policy is considered good enough):

```
For all $x$
    For all $a$
        $Q(x,a) := \sum_{r} rP_{xa}(r) + \gamma \sum_{y} V(y) P_{xa}(y)$
        $V(x) := \underset{a \in A}{Max}\ Q(x,a)$
```

For the deterministic world, Q-values can thus be defined as:

$$Q(x,a) := r(x,a) + \gamma \sum_{b \in A} \max Q(y,b)$$

This equation is thus equal to the reward given for the action pair (x,a) plus γ times the maximal expected cumulative reward that can be obtained in the resulting action pair (y, b). ■

Definition 6.9: The Q-learning algorithm

Definition 6.8 has described the final solution after learning (steady state). However, this state is initially not known. The actual learning process can be described as follows:

1. Initialize the Q-values.
2. Select a random starting state x which has at least one possible action to select from.
3. Select one of the possible actions. This action will get you to the next state y (or x_{t+1}).
4. Update the Q-value of the state action pair (x, a) according to the update rule below.
5. Let $x = y$ and continue with step 3 if the new state has a least one possible action. If it has none go to step 2.

The update rule is given by
$Q_{t+1}(x,\ a)$=$(1$-$\alpha)Q_t(x,a)$+$\alpha[R(x,a)+\gamma\max\limits_{b} Q_t(y,b)]$, where
$Q_t(x,\ a)$ is the Q-value at the current time-step, $\max\limits_{b} Q_t(y,b)$ is the maximal expected cumulative reward that can be obtained in the resulting action pair (y, b) and $Q_{t+1}(x,\ a)$ is the updated value. α is the learning rate and is a parameter of the algorithm. The learning rate α controls how much weight we give to the reward just experienced, as opposed to the old Q-estimate. One typically starts with α =1; i.e. giving full weight to the new experience. As α decreases, the Q-value is building up an average of all experiences, and the odd new unusual experience won't disturb the established Q-value much. As time goes to infinity, α will go to 0, which would mean no learning at all, with the Q-value fixed. ■

Definition 6.10: Exploration versus exploitation

In each state the agent basically can choose from two kinds of behaviour: either it can explore the state space or it can exploit the information already present in the Q-values. By choosing to exploit, the agent usually gets to states that are close to the best solution so far. By this option, it can refine its knowledge about that solution and collect relatively high rewards. On the other hand, by choosing to explore the agent visits states that are further apart from the currently best solution. By doing so, it is possible that it finds a new, better solution than the one already known.

A parameter –the exploration rate $p_{explore}$– can be used to to set the behavior of the Q-learning algorithm. In every step, with a probability of 1-$p_{explore}$ the agent exploits the information stored in the Q-values, with probability $p_{explore}$ the agent chooses a random action in order to explore the state space. ■

Examples for definitions 6.7-6.10 will be given in the following section.

6.3 ALLOCATING TIME AND LOCATION INFORMATION (EXAMPLE)

In this section, a hypothetical example has been presented to improve the understanding of Q-learning. The behaviour of the Q-learning algorithm is first explained with respect to the time allocation problem (section 6.3.1); location allocation is dealt with in section 6.3.2 and the integration of time and location allocation is treated in section 6.3.3.

6.3.1 TIME ALLOCATION BY MEANS OF Q-LEARNING

For this first application and for the sake of clarity, the presence of travel modes has been ignored in the fixed sequence of activities (see Chapter 5). Transport modes will be re-introduced later in section 6.3.2. There are a number of other simplifying assumptions which are made to better understand the behaviour of the decision agent that can be summarized as follows (all assumptions will be relaxed later on in the chapter in the empirical section):

- Learning rate $\alpha = 1$ (see definition 6.9)
- Discounting factor γ=0.8 (see definition 6.6)
- Exploration maximal, i.e. $P_{explore}$=1 (see definition 6.10)
- Time already spent at activity (duration): Only 3 discrete time slots: 0 hours, 6 hours or 12 hours
- Fixed order of only 4 activities (1 sequence), i.e.: Home – Work – Shop – Leisure
- A state *x* is characterized by the activity, starting time of activity and time already spent at activity (duration) (see definition 6.1)
- For a state *x*, an action may be to Stay ('S') at the current activity (state x_t) or to Move on ('M') to the next state (x_{t+1}) (see definition 6.2)
- No travel time between two activities (ignorance of travel modes)

It has to be emphasized that the parameter setting of the algorithm (discounting factor, learning and exploration rate) do not influence the learning solution as such, although they may have an impact on the time that is needed for the algorithm to converge to an optimal solution (see also Charypar and Nagel, 2005). Obviously, the number of discrete time slots and activities do have an impact on the solution outcome, as it will be shown in the empirical section.

In addition to the assumptions above, reward tables are also artificial and extremely simple in this example, as shown in Table 6.1.

Table 6.1: An example of a simple reward table

	Home			Work			Shopping			Leisure		
Start time/ Duration	0 h	6 h	12 h	0 h	6 h	12 h	0 h	6 h	12 h	0 h	6 h	12 h
0:00 A.M.	0	6	0	0	0	0	0	0	0	0	3	0
6:00 A.M.	0	4	0	0	3	5	0	0	0	0	3	1
12:00 A.M.	0	2	0	0	0	0	0	5	1	0	3	4
6:00 P.M.	0	1	0	0	0	0	0	0	0	0	3	0

It can be seen from Table 6.1, that the reward of working 0 hours is 0 and is independent of the starting-time of the work-activity. Arriving at work at 6 A.M gives somebody a reward of 3 (units) at the moment he/she is working for 6 hours (i.e. from 6 AM-12:00 AM) or a reward of 5 (units) at the moment the person is working for 12 hours (i.e. from 6AM-6 PM). Arriving at work later than 6 AM gives no reward at all. The reward tables for home, shop and leisure are similar.

Let us now reconsider the Q-learning algorithm that has been introduced in definition 6.9. Since $\alpha = 1$ and $\gamma = 0.8$, the update rule for our simple example is equal to $Q_{t+1}(x,\ a)=R(x,a)+0.8\max_b Q_t(y,b)$, where $Q_{t+1}(x, a)$ is the updated Q-value and $Q_t(x, a)$ is the Q-value at the current time-step.

In the first step of definition 6.9 (initialisation), all the q-values of every state-action pair are set equal to zero.

Next, a random starting state *x* will be chosen, which has at least one possible action to select from. In our example, the starting state may be equal to: Work activity, start time 0:00 AM, duration 6 hours.

The third step selects one of the possible actions, which will bring us to the next state *y* (x_{t+1}). Because the exploration probabilitiy was set maximal, i.e. $P_{explore}$=1, the decision agent will always choose a random action in order to explore the state space in an attempt to find a new, better solution than the one already known. On the contrary, when $P_{explore}$=0, the decision agent will choose the action that has the largest Q-value (thus not randomly). When both actions have the same Q-value, the action that has first achieved this Q-value will be chosen.

In our example, the decision has been taken to Move on to the next activity, which is a random choice because $P_{explore}$=1. Since the sequence order is fixed in this example, the next activity (Shopping) is known in advance. Also the full state is known in advance, it is equal to 0:00 AM + 6 hours = 6 AM.

Now, the Q-value for the state Q(x,a)=Q(Work;Start 0:00 AM;Duration = 6 hours, *Move*) is equal to 0. This value is simply equal to the reward which has been defined in Table 6.1. The second part of the update rule ($0.8\max\limits_{b} Q(y,b)$) is only computed if this state has been visited at least one time before. Otherwise, no state can be selected that maximizes the expected cumulative reward. In this example, the state has only been visited once before by the initialisation procedure, which has set all Q-values equal to 0. Table 6.2 shows the states that have been visited by the agent in every loop, while Table 6.3 illustrates the progress of the Q-values for every *state-action* pair during the execution of the algorithm. Q-values in this table are intermediary results and the solution has not yet converged (see infra).

In the final step, the state x will be set equal to the state y (=Shopping, Start 6:00 AM, duration 0 hours). In this artificial example, no travel time has been taken into account. However, it should be noted that in a realistic scenario (where travel time is considered), the start time of state y should be augmented with the travel time which is needed to get from state x to state y (see section 6.3.2). For now, the algorithm continues with loop 2, which starts again at step 3 of the algorithm procedure. The Q-values stay equal to zero until the 5th loop. In this loop, the action is *Stay,* which will bring us to the state of Leisure, 6:00 PM, duration 6 hours, which reward is 3. The state has not yet been visited before, so

Table 6.2: Visited states per loop (Numbers denote the loop number)

	Home			Work			Shopping			Leisure		
Start time/Duration	0 h	6 h	12 h	0 h	6 h	12 h	0 h	6 h	12 h	0 h	6 h	12 h
0:00 A.M.	7	8			1							
6:00 A.M.				9 25	10 26		2	3	4			
12:00 A.M.	13 17			14 18			11 15 19 27			12 16 20 28	21 29	
6:00 P.M.	22 30 34	23	24	31 35			32 (36)			5 33	6	

the total Q-value is also equal to 3. It should be noted that the state-action pair, represented as (Leisure, 6:00 PM, 6 hours; Move) remains intact and is still equal to 0. Another noteworthy moment is from the 22nd till the 24th loop. In the 22nd loop, the Q-value is equal to 1, which is similar to the calculation in the 5th loop. However, when we move on to the 23rd loop, the Q-value becomes equal to −1. Indeed, since we are using the time that somebody has already spent at the current activity as a dimension, the reward that can be derived is equal to −1 since we moved from reward 1 (22nd loop) to reward 0 (23rd loop) (marginal utility function). The 24th loop is the first where an update value has to be calculated. The immediate reward is equal to 0, but the second part of the update rule looks at the *latest* updated Q-value for every action, takes the *largest* Q-value *over all the actions* and multiplies this by the discounting factor. In this case the latest updated Q-value for the action pair (*y*,*b*): (Work, 6 A.M., 0 hours; Stay) is 3 (see loop number 9) and for (Work, 6 AM, 0 hours; Move) it is 0 (initialisation). For this reason, the updated Q-value of the 24th loop is equal to 0 + 0.8*Max[(Work, 6 A.M., 0 hours; Stay); (Work, 6 AM, 0 hours; Move)]= 0+0.8*Max[3;0]=2.4. The computation for the other loops is similar (see Table 6.3).

Once all the Q-values have been computed, a policy (chart) can be constructed according to which the learning agent will behave (see definition 6.7). The final action will be determined when the Q-values that are calculated in Table 6.3 have become stable. For example, the state Shopping, start time 12:00 AM and duration 0 hours will be determined by the latest value that has been achieved at loops 11, 15, 19 and 27, in the assumption that the 4th visit to that state leads to a stable Q-value. The value is 2.4 and the corresponding action is *Move*. As mentioned before, if there are two states with an equal Q-value and a different action, the action that has first achieved the Q-value will be chosen. This is often the case in our artificial example where the Q-value is 0 and where the action is *Move*. In these cases the action *Stay* will be preferred (*Stay* was initialised before *Move*). The full policy chart for this example has been show in Table 6.4.

Table 6.3: Q-values and state-action pairs

Loop	Action	Q-value	Loop	Action	Q-value	Loop	Action	Q-value
1	Move	0	13	Move	0	25	Stay	3+0.8 max(10)=3
2	Stay	0	14	Move	0	26	Move	0
3	Stay	0	15	Move	0	27	Move	0+0.8* Max(3;0)=2.4
4	Move	0	16	Move	0	28	Stay	3
5	Stay	3	17	Move	0	29	Move	0+0.8*Max (1;0)=0.8
6	Move	0	18	Move	0	30	Move	0
7	Stay	6	19	Move	0	31	Move	0
8	Move	0	20	Stay	3	32	Move	0+0.8*Max (3;0)=2.4
9	Stay	3	21	Move	0	33	Move	0+0.8*Max (1;0)=0.8
10	Move	0	22	Stay	1	34	Move	0+0.8*max (0;0)=0
11	Move	0	23	Stay	-1	35	Move	0+0.8*max (0;2.4) =1.92
12	Move	0	24	Move	0+0.8* Max(3;0) =2.4			

It should be noted that this is a suboptimal solution which has not converged. A suboptimal solution to a reinforcement learning problem is a solution that was not completely solved by the agent, i.e. some of the Q values do not correspond to the steady state values. In this case the agent will nevertheless find a cycle, albeit possibly not the optimal (i.e. maximized reward per 24 hours) one. There are two important reasons for not reaching this solution for this example.

The first reason is that the agent needs to visit every possible state-action pair infinitely often (Mitchell, 1997). Stated otherwise, the loops in the experiment

Table 6.4: Policy chart for 35 loops

	Home			Work			Shopping			Leisure		
Start time/ Duration	0 h	6 h	12 h	0 h	6 h	12 h	0 h	6 h	12 h	0 h	6 h	12 h
0:00 A.M.	S	S	M	S	S	M	S	S	M	S	S	M
6:00 A.M.	S	S	M	S	S	M	S	S	M	S	S	M
12:00 A.M.	S	S	M	S	S	M	M	S	M	S	M	M
6:00 P.M.	S	M	M	M	S	M	M	S	M	S	S	M

need to be large enough. Obviously, the 35 loops that were considered in the example (for the sake of clarity) are clearly insufficient. The number of possible state-action pairs determines the size of the learning problem and the number of times that an agent needs to visit a particular state-action pair before an optimal solution can be reached. This research topic is a subdomain within the field of reinforcement learning that goes beyond the scope of this dissertation.

The second reason (closely related with reason 1), is the number of trials that has been conceived. In the development of Policy chart 6.4 for this example, Work activity, start time 0:00 AM, duration 6 hours have been used as the starting state. This obviously represents only one trial. In the empirical section, 10 different trials have been used (where each trial is defined by a different random starting state).

14 Setting the iteration number (number of loops) equal to 100000 or 1000000, and with number of trials set to 10, will lead to an optimal policy chart that is shown in Table 6.5. A computer code has been established to automate this process.

Table 6.5: Policy Chart for iterations going to infinity (Steady-state situation)

	Home			Work			Shopping			Leisure		
Start time/ Duration	0 h	6 h	12 h	0 h	6 h	12 h	0 h	6 h	12 h	0 h	6 h	12 h
0:00 A.M.	S	M	M	M	S	M	M	S	M	M	M	M
6:00 A.M.	S	M	M	S	M	M	S	M	M	M	M	M
12:00 A.M.	M	M	M	M	M	M	S	M	M	S	S	M
6:00 P.M.	M	S	M	M	M	M	M	M	M	S	M	M

Based on this policy chart, start and end times can be allocated to a fixed sequence of activities. The procedure for doing this is quite simple and is illustrated in Table 6.6.

Table 6.6: Determining start and end times based on a Policy chart

	Home			Work			Shopping			Leisure		
Start time/ Duration	0 h	6 h	12 h	0 h	6 h	12 h	0 h	6 h	12 h	0 h	6 h	12 h
0:00 A.M.	S	M	M	M	S	M	M	S	M	M	M	M
6:00 A.M.	S	M	M	S	M	M	S	M	M	M	M	M
12:00 A.M.	M	M	M	M	M	M	S	M	M	S	S	M
6:00 P.M.	M	S	M	M	M	M	M	M	M	S	M	M

The algorithm will first choose a random start state. Let's say Shopping, 0:00 A.M., duration 6 hours is chosen. The corresponding action in the policy chart is *Stay*. As a result, the next state is equal to Shopping, 0:00 A.M., duration 12 hour (action=*Move*). Then, the next states are equal to Leisure, 12:00 A.M., duration 0 hours (action=*Stay*), Leisure, 12:00 A.M, duration 6 hours (action=*Stay*) and Leisure, 12:00 A.M, duration 12 hours (action=*Move*). Home, 0:00 A.M., duration 0 hours (action=*Stay*) and Home 0:00 A.M., duration=6 hours (action=*Move*) are the following states. Next, Work, 6:00 A.M., duration=0 hours (action=*Stay*) and Work, 6:00 A.M., duration=6 hours (action=*Move*) have been reached. Finally; Shopping 12:00 A.M., duration=0 hours (Action=*Stay*), Shopping 12:00 A.M., duration=6 hours (Action=*Move*) and Leisure 6:00 P.M., duration=0 hours (Action=*Stay*); Leisure 6:00 P.M., duration=6 hours (Action=*Move*) have been visited by the agent. Now, the loop has been completed for one day. Obviously, the algorithm continues to move on and checks whether the solution converges. The check is satisfactory and the start and end times for every activity in the fixed sequence order are thus determined as follows:

Home	: 00:00 A.M.-- 6:00 A.M.
Work	: 6:00 A.M. -- 12:00 A.M.
Shopping	: 12:00 A.M. -- 6:00 P.M.
Leisure	: 6:00 P.M. -- 00:00 A.M.

It can be seen from the policy chart that any other start state in the policy chart such as for instance Leisure, 0:00 A.M. duration= 6 hours; Home, 6:00 P.M., duration = 6 hours or Home, 0:00 A.M., duration= 12 hours, etc. will lead to the same solution.

Finally, some additional remarks need to be made with respect to the use of the Q-learning algorithm to solve the time allocation problem. First, cycles can also be multiples of 24 hours. For example, an agent can have one full day where it gets up early and goes to bed late, alternated with a less full day where it gets up later and goes to bed earlier. Second, an interesting side-effect of the structure of Q-learning is that the result of the computation is not only the optimal "cycle" through state space, but also the optimal "paths" if the agent is pushed away from the optimal cycle. For example, if an activity takes considerably longer than expected, the Q values at the arrival state will still point the way to the best continuation of the plan (Charypar *et al.*, 2004). To this end, the technique may be used within the context of within-day rescheduling, which is becoming an active topic of research in activity-based transportation research. However, it has been claimed (Nagel and Marchal, 2003; Charypar *et al.*, 2004) that if the technique is to be used for large scale multi-agent transportation simulations, the all-day activity plan (i.e. sequence order of activities) needs to be pre-planned in advance before the simulation starts in order to deal efficiently with re-scheduling. The procedure that has been described in Chapter 5 may offer a solution to come up with this fixed pre-planned order.

6.3.2 LOCATION ALLOCATION BY MEANS OF Q-LEARNING

Consistent with the time allocation problem, location allocation can also be solved by means of Q-learning. For this purpose, it is assumed that people try to minimize the travel between two locations in order to have more time available to carry out activities and realize goals.

Travel distance may not be an optimal measure for determining the burden of travel because it is plausible in a realistic situation that the distance between location A and location B is shorter than the distance between location A and C, while the travel time may be longer (for instance because of a better road network). Furthermore, it is possible that there is a difference in the transport mode that is used. For instance, there may be an efficient highway between two locations (for car use), but a poor road network for slow modes. For this reason, it is assumed that the travel time between two locations is representative for the burden that arises because of this travel, and that a differentiation needs to be made with respect to the transport mode that has been used. As a result, the presence of transport modes (ignored in the previous section) has to be re-introduced in the activity patterns (sequences) when dealing with the location allocation problem.

Translated to a context of Q-learning, this implies that reward tables also depend upon the travel time between two locations (this is not similar as the travel time between activities, see Charypar and Nagel, 2005) and on the transport modes that have been used to reach these locations. The solution procedure is similar as in the time allocation problem, but is somewhat more complicated.

Again, consider a simple example with the following assumptions to better understand the behaviour of the decision agent (assumptions will be relaxed later on in the empirical section):

- Learning rate $\alpha = 1$ (see definition 6.9)
- Discounting factor $\gamma=0.8$ (see definition 6.6)
- 5 locations (locations A-E in Figure)
- Two travel modes: Car and Walk
- A similar activity sequence as in previous section (fixed order), but travel modes are included in the sequence, e.g.:
  Home – *Car* – Work – *Car* –Shop – *Walk* – Leisure
- A state *x* is characterized by an activity and an origin location.
- For a state *x*, an action is to choose a destination location which is available for the next activity. When the destination location and the travel mode (available because of the fixed sequence order) have been determined, the reward for that particular action can be determined.
- Activities can only be carried out at a limited number of locations.
- Assume that the following activity locations are possible for each activity:

Home	: Location A
Work	: Location B
Shop	: Location C or D
Leisure	: Location C or E

Assume that the 5 locations (A-E) are spatially distributed in areas as graphically illustrated in Figure 6.1. As mentioned before, the spatial distribution can be equally (and perhaps more efficiently) represented by the travel time between two locations, as shown in Table 6.7 by means of example.

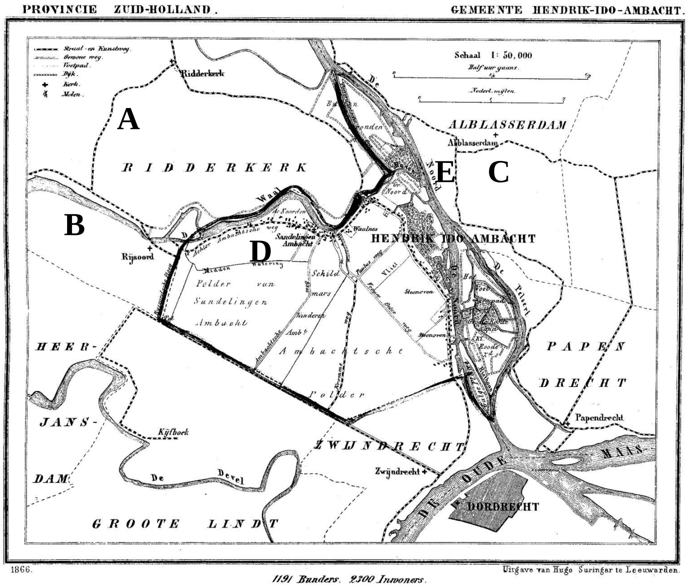


Figure 6.1: Spatial distribution of 5 possible locations in study area (example)

A simple linear transformation (by means of interpolation) can be used to determine the reward/cost (see Table 6.8) based on this travel time table. To this end, the overall maximum travel time per transport mode is taken as a minimum reward value in the interpolation procedure. It is obvious that this transformation can still be further refined, for instance through the use of an alternative transformation function or by the segmentation of the utility perception over a

Table 6.7: Travel times between pair of locations (in minutes)

	Car					Walk				
Origin/ Destination	A	B	C	D	E	A	B	C	D	E
A	0	5.5	15.1	18.2	21.5	0	23.2	65.2	43.2	62.1
B	5.5	0	32.3	3.1	25.1	23.2	0	78.6	33.2	71.6
C	15.1	32.3	0	15.5	12.1	65.2	78.6	0	84.1	6.2
D	18.2	3.1	15.5	0	5.2	43.2	33.2	84.1	0	69.1
E	21.5	25.1	12.1	5.2	0	62.1	71.6	6.2	69.1	0

group of (clustered) individuals with similar utility. Also, the maximum utility (cost) has now been determined arbitrarily per transport mode and can become the subject of additional refinements. More detailed work in this respect, applied to an activity-based rescheduling framework, has been done by Joh *et al*. (2003; 2004).

Since the aim of the decision agent is to minimize travel duration, it may be argued that the reward of travel should be negative (and thus represents a cost for the agent); a decision which becomes even more important in the integrated approach (see section 6.3.3). For this isolated location allocation problem, using positive values would obviously lead to the same results.

Table 6.8: Reward table based on travel time table (interpolation)

	Car					Walk				
Origin/ Destination	A	B	C	D	E	A	B	C	D	E
A	0	-2	-5	-6	-7	0	-5	-12	-8	-11
B	-2	0	-10	-1	-8	-5	0	-14	-6	-13
C	-5	-10	0	-5	-4	-12	-14	0	-15	-2
D	-6	-1	-5	0	-2	-8	-6	-15	0	-13
E	-7	-8	-4	-2	0	-11	-13	-2	-13	0

The solution procedure that is used to arrive at a steady state (and to come up with a stable policy chart) differs somewhat with respect to the time allocation problem, especially because the action space is different. Taking our example into account: Home – *Car* – Work – *Car* –Shop – *Walk* – Leisure and recalling that Home and Work can only be carried out at location A and B; it is obvious that the agent only has to decide about the location of the Shopping and the Leisure activity. For Shopping, the agent can only choose between locations C and D; for Leisure, the choice is between locations C and E. The remainder of this section illustrates the use of definition 6.9 for this problem.

After the initialisation, a random starting state *x* will be chosen, which has at least one possible action (i.e. destination) to select from. It should be recalled that a state is defined by an origin location and an activity. For instance the states (C, Shop) and (D, Shop) have two possible actions (destination locations), i.e. Location C or E. As a result, the Leisure state has no possible actions to select from. Therefore, it can not be chosen initially and will never be visited by the agent. Assume that the agent first visits state (C, Shop). In the third step of the algorithm in definition 6.9, the decision agent will choose a random action in order to explore the state-space in an attempt to find a new, better solution than the one already known ($P_{explore}$=1). There are only two possible actions at the state (C, Shop), i.e. C or E. Let us assume that action (destination) C has been chosen. Since the sequence order is fixed in this example, the next activity is known in advance (Leisure). Now, we can compute the Q-value for the state Q(*x*,*a*)=Q(C, Shop; C) = 0. The second part of the update rule ($0.8 \max_b Q(y,b)$) can be omitted in this case because there is no action at the next state (Leisure) for which a Q-value can be calculated. For this reason, the value is simply equal to the reward table which has been defined in Table 6.8. It should be noted that we need to rely upon the reward table for Walk, because Walk is used to get from Shopping to Leisure. The next destination location to decide about is for the state (B, Work). Assume that action (destination) C has been chosen. When formula $Q_{t+1}(x,\ a)=R(x,a)+0.8\max_b Q_t(y,b)$ is applied, it is clear that the immediate cost (R(*x*,*a*)) is equal to -10 (reward table *Car*). The second part of the update rule looks at the *latest* updated Q-value for every action, takes the *largest* Q-value *over all the actions* and multiplies this by the discounting factor. In this case the latest updated Q-value for the action pair (*y*,*b*), is both zero for Q(C, Shop; C) (because of loop 1) and for Q(C, Shop; E) (because of initialisation). The total reward (cost) in this case is thus equal to -10. The computation for the other loops is similar and has been illustrated in Table 6.10. Table 6.9 shows the states that have been visited per loop.

Table 6.9: Visited states per loop (Numbers denote the loop number)

Origin/activity	Home	Work	Shop	Leisure
A	/	/	/	/
B	/	2,4,6,8,10,12,14...	/	/
C	/	/	1,3,7,11, ...	/
D	/	/	5,9,13,15...	/
E	/	/	/	/

Table 6.10: Q-values and state-action pairs

Loop	Action	Q-value
1	C	Q(C, Shop;C) = 0
2	C	Q(B, Work;C)= -10+0.8*max(Q(Shop,C;C),Q(Shop,C;E)) = -10
3	E	Q(C, Shop;E) = -2
4	D	Q(B, Work;D) = -1 +0.8*max(Q(Shop,D;C),Q(Shop,D;E))= -1
5	C	Q(D, Shop;C) = -15
6	C	Q(B, Work;C) = -10+0.8*max(Q(Shop,C;C),Q(Shop,C;E) =-10
7	E	Q(C, Shop;E) = -2
8	D	Q(B, Work;D) = -1 +0.8*max(Q(Shop,D;C),Q(Shop,D;E))= -1
9	E	Q(D, Shop;E) = -13
10	C	Q(B, Work;C)= -10+0.8*max(Q(Shop,C;C),Q(Shop,C;E)) = -10
11	C	Q(C, Shop;C) = 0
12	D	Q(B, Work;D) = -1 +0.8*max(Q(Shop,D;C),Q(Shop,D;E))= -11.4
13	E	Q(D, Shop;E) = -13
14	D	Q(B, Work;D) = -1 +0.8*max(Q(Shop,D;C),Q(Shop,D;E))= -11.4
15	C	Q(D, Shop;C) = -15
...	...	...

Once all the Q-values have been computed, a policy (chart) can be constructed according to which the learning agent will behave. The final action will be determined when the Q-values that are calculated in Table 6.10 have become stable. It can be seen that Q(B,work; C)=-10 and Q(B,work; D) = -11.4 for these 15 loops. For this reason, the policy will be to choose action C at the state (B, Work). The final policy chart looks thus like Table 6.11. This policy chart appears to be stable when the number of loops approximate to infinity (100000 and 10000000). A computer code has been established to automate this process.

Table 6.11: Policy chart for iterations going to infinity

Origin/activity	Home	Work	Shop	Leisure
A	/	/	/	/
B	/	C	/	/
C	/	/	C	/
D	/	/	E	/
E	/	/	/	/

The optimal location allocation for this sample sequence is thus equal to:

Home – *Car* – Work – *Car* –Shop – *Walk* – Leisure
| | | |
A B C C

Two final remarks need to be made with respect to this optimal solution. In the empirical section (see section 6.4), work and home locations were randomly simulated for every respondent. Work and home locations are thus assumed to remain fixed per respondent during the complete location allocation phase and are not determined by the Q-learning agent. This random simulation has been drawn from the set of home locations that appear in the data. Despite the fact that work and home locations are not determined by the Q-learning agent, they are considered to be critical in the interaction process with the agent because they anchor the set of spatial information that is developped by an individual and condition the search for other locations through segments of space. In fact, this assumption closely resembles the anchorpoint theory as suggested by Golledge (1975, 1978). Other anchorpoints such as shopping locations, or other commonly recognized, known and often-used places in the environment can be incorporated within the same framework. Second, since the location of the Work activity is always B in our example, one might think that the agent would have decided to carry out the Shopping activity at location D, since the cost is only equal to –1, compared to –10 for the other alternative (location C) (see reward Table 6.8). However, as illustrated before, the agent had the intelligence of looking into the future and not being trapped into a local optimum. Therefore, the solution which is proposed in this section with respect to location allocation, has a lot of similarities with the well-known shortest path algorithm (Dijkstra, 1959). The advantages for using Q-learning for this purpose are that time allocation can be easily integrated and that the agent returns the best action at every possible state. The integration procedure is discussed in the section below.

6.3.3 Integrating Time and Location Allocation in Q-Learning

The previous two sections have independently considered time and location allocation. Obviously, dealing with both allocations simultaneously, leads to some important advantages. The first advantage is that the reward is not only maximized in either the time or the location facet, but the *total* reward in a day (i.e. reward that arizes from determining optimal start and end times and the cost that arizes from travelling between locations) will be maximized by means of an integrated approach, which is obviously more realistic. The second advantage is that flexible travel times between two locations can be considered. In the time allocation section (see section 6.3.2), the presence of travel times has been ignored. Obviously, this assumption is superficial and needs to be relaxed. For instance, Charypar *et al. (*2004) deal with the problem by assuming a constant travel time between two different *activities* (they only considered activities and not locations). They also experimented with a variable travel time per activity pair, but they only did this for one particular test case. Therefore, no differentiation has been made over different respondents, thereby implicitly assuming that every activity pair always has a similar travel time, independent from the activity location and from the transport mode that has been used to reach these locations. Based on the location allocation section, it now becomes possible to consider variable travel times between two locations and take this information into account in time allocation. Locations that are far/close to each other will require a longer/smaller travel time and will benefit from a more realistic time allocation. To this end, travel times between two locations are looked up in travel time tables (such as for instance Table 6.7).

States and Actions

In order to make this integration, states and actions have to be redefined when compared to individual time and location allocation.

A ***state*** is determined by an activity *a*, an origin location *l*, a start time *b*, and a duration *d*.

Also in this case, there are two possible ***actions***:

- *Stay*: Keep performing activity *a* at location *l* for another time slot.
- *Move*: Move to location *l′* for a next activity *a′* (by means of a travel mode *tm*). Take the travel time into account between location *l* and location *l′*. If there is no travel mode between activity *a* and activity *a′*, *a′* is performed at the same location as *a*, i.e. $l = l'$. Otherwise $l' \in L(a')$, where $L(a')$ denotes the set of all possible locations for activity *a′*.

Transitions Between States

It logically follows from the previous section that in case transportation is needed to move to another state (thus in case of the action *Move*), the start time of state *y* should be augmented with the travel time that is needed to get from state *x* to state *y*. This becomes possible because the destination activity has been determined at this point and because travel times are known per origin-destination pair. However, due to the use of discrete time intervals (see time allocation section), the start time of state *y*, is set equal to the start time of the interval which is closest to the sum of the start time of state *x* and the travel time that is needed to get from state *x* to state *y*.

Time and Location Allocation in Pseudocode

The integrated time and location allocation program consists of 5 general steps and can be shown as follows in pseudo-code:

```
1. Read sequences
2. Read reward tables for activities (start time and duration) and travel mode
   (between each pair of locations)
3. Assign locations to each activity in the sequence, which the agent can visit.
   One random location will be assigned to in-home and work activities. The other
   out-of-home locations are selected from all the other possible locations that
   appear in the data.
4. For each sequence
Apply definition 6.9
```

5. Output the optimal time and location allocation policies per sequence when Q-values become stable in the 4th step.

6.4 Empirical Section

This section describes the empirical results after application of the integrated time and allocation procedure that has been described above. Before doing so, more detailed explanation is given about the parameter values and reward tables that have been used in the experiments.

6.4.1 Parameter Values

In the previous sections, it was mentioned that there are basically 3 parameters that play an important rule in Q-learning: the discounting factor γ, the learning rate α and the exploration rate $P_{explore}$. Other important factors are the time interval that has been used and the maximum duration that an activity can last.

Discounting Factor

The discounting factor should be close to 1 since we are interested in finding the daily time and location plan that maximizes the *cumulative* reward. For the utility of a plan it does not matter when a certain reward is earned (incrementally increasing the discounting parameter from 0 to almost 1 may give us an idea when the reward is earned) only important is *that* it is earned. Also, the agent should look as much as possible into the future because someone's decision behaviour is not restricted to the next activity/state either. In the example given in definition 6.6, it was mentioned that a high reward may be accorded to leisure activities that are carried out during the evening and this may have an influence on the decision process during the day (for instance one may quit working earlier). However, it can be theoretically proven that setting the parameter equal to 1 can lead to diverging Q-values.

On the other hand, for reasons of efficiency, low discount parameters are best as they reduce interdependency of the Q-values and therefore lead to higher learning speeds. However, low discount parameters inherently prefer short activities. This can result in undesirable results such as for example long activities (e.g. "Work") may be left out completely while short activities (e.g. "Shopping") are repeated over and over again (Charypar and Nagel, 2005).

In our new integrated setting, one should deal carefully with the discounting factor. In particular, the discount per time slot should take the travel time - needed to reach the next state- into account. For example when γ is set equal to 0.99 and the time slot is 15 minutes and if it takes 45 minutes to reach activity *a′* at location *l′*, the value of γ equals $\gamma^{45/15} = \gamma^3 = 0.97$. As a result of this, the discount per time interval *t*, is able to take care of the travel time that was needed. In another example; assume that the action is *Move* between the states (Eat, location 301, 6:00 PM, 30 minutes) and (Sleep, location 301, 6:30 PM, 0 minutes). There is no time delay between these two states and therefore no discount is needed. Indeed, recall from definition 6.6 that the discounting factor only defined how much expected future rewards, affect decisions now. Since we are not talking about future values in this latter case, no discount is needed. By consequence, every time the updating rule has been applied, the discounting factor needs to be recalculated. In our experiments, we determined that a discounting factor of 0.95 looked far enough into the future and was also able to reach a solution that converged.

Learning Rate

The next parameter to decide about is the learning rate α. It has been proven by Watkins and Dayan (1992) that Q-learning only converged to a steady state if some preconditions were met (for more information see Watkins and Dayan (1992)). However, for purely deterministic worlds, as the one discussed in this chapter, α can be set equal to 1. The reason is that since the system is discrete and finite, the trajectory eventually needs to come back to a state where it was before. Once this point has been reached, the system will do exactly the same as in the previous "round" (see previous policy charts). A learning rate of 1 will then lead to the most optimal and fastest learning.

Exploration Rate

Another important parameter is the exploration rate $P_{explore}$. It was mentioned before that by choosing to explore the agent visits states that are further away from the current best solution. By doing so, it is possible that it finds a new, better solution than the one already known before. With probability of $p_{explore}$, the agent will choose a random action in order to explore the state space. The safest option here is to set $P_{explore}$ equal to 1, in order to be sure that the best solution has been found in the end. Obviously, this decision negatively affects the learning speed.

Time Interval

The second but last parameter to decide about is the time interval. It will be shown in the following sections that the time interval is the parameter which has the largest impact on the size of the learning problem (and on the time to reach a solution that converges). Obviously, since we explicitly use travel times in our integrated framework, those travel times determine to a large extent the detail of the time interval that needs to be chosen. Indeed, when the travel time between two locations is for instance 20 minutes, it is too imprecise to set the time interval equal to 1 hour. In this context the reader may recall from section 6.3.3 that the start time of state *y* is set equal to the start time of the interval which is closest to the sum of the start time of state *x* and the travel time that is needed to get from state *x* to state *y*. If the time interval would be set equal to 1 hour and travel time is 20 minutes, we run the risk of dealing with a lot of unspecified time in which no activity can be carried out. The average travel times between all the locations in the study area is equal to 212 minutes for walk as transport mode, 60 minutes for bike, 13 minutes for car and 61 minutes for public transport. Therefore, we determined that a 15 minute interval will enable us to end up with sufficiently detailed time allocation. Setting the time interval below 15 minutes will lead to much slower learning speeds for the algoritm to converge and it will probably not lead to a significant improvement in accuracy either (see also section 6.4.3).

Maximum Duration per Activity

Finally, a decision needs to be made about the maximum duration that an activity can last. For our application, the maximum duration was set equal to 10 hours. When compared to real diary data, this decision corresponded to 99,4% for all the activities that have been reported. Setting the maximum duration equal to

12 hours would further increase this percentage to 99,9% but this would lead to slower learning times while the gain is only minor.

6.4.2 REWARD TABLES

Once the parameters have been set, the most important input for Q-learning are the reward tables that need to be specified correctly. The section below described which reward tables have been used in the experiments.

TIME REWARD TABLES

It was already mentioned before that the size of the time reward table depends to a large extent on the time interval that has been used. In the example reward table in Table 6.1, the size of the table is equal to 48 cells. This is equal to the number of discrete time steps that fits into the maximum duration (columns of the reward table), multiplied by the number of discrete time steps that fits into a 24-hour day-period (rows of the reward table), multiplied by the number of activities (i.e. 3*4*4=48). In our experiments, the size of the reward table is equal to 41*96*17=66912 cells. The main problem is where to get these values from. As one alternative, an elaborated stated preference experiment can be developed that is able to quantitatively assess the reward that people experience per start time and per time unit that was spent per activity. However, even in such an experiment, it seems more desirable to use larger time intervals than a 15-minute interval.

As a second-best alternative, it was examined how frequency information that is available in the activity diaries can be used to come up with these values. While frequency is certainly not a synonym for reward, the idea might work fairly well if we have a look at the purpose of our experiment. In our application, the aim is to come up with a time allocation (per activity), that corresponds best with the information that is present in the data. Obviously, some direct relationship is needed between the unsupervised learning model and the data to achieve this. So, even though people may not like it to get to work at 7 A.M. (and may report a low reward in a realistic situation), the learning model will assign a lot of activities starting at that point in time if this happens frequently often in the data.

However, the frequency information cannot be used entirely without any modification. A simple example can illustrate this. Suppose that somebody has reported to have end sleeping at 3.15 A.M. and that the time that he/she was already sleeping was 15 minutes. The reward table that is defined by a 3.00 A.M. starting time and a 15 minute duration, needs be incremented by 1 unit. However, assume now that a second person reported to have ended sleeping at 4.00 A.M. and that the time that he/she was already sleeping was 60 minutes. Now, in this case, not only the reward table that is defined by a 3.00 A.M. starting time and a 60 minute duration needs to be updated, but the 15-, 30- and 45-minute interval needs to be updated as well. A simple program has been established to automate these kind of conversion procedures.

An example of such a possible reward table has been shown in Figure 6.2 for the work or study (out-of-home) activity. The axes report the start time, duration and frequency of the activity. It can be seen from this figure that most people start to work in the period between 7.30 A.M. and 8.30 A.M. and the number of people

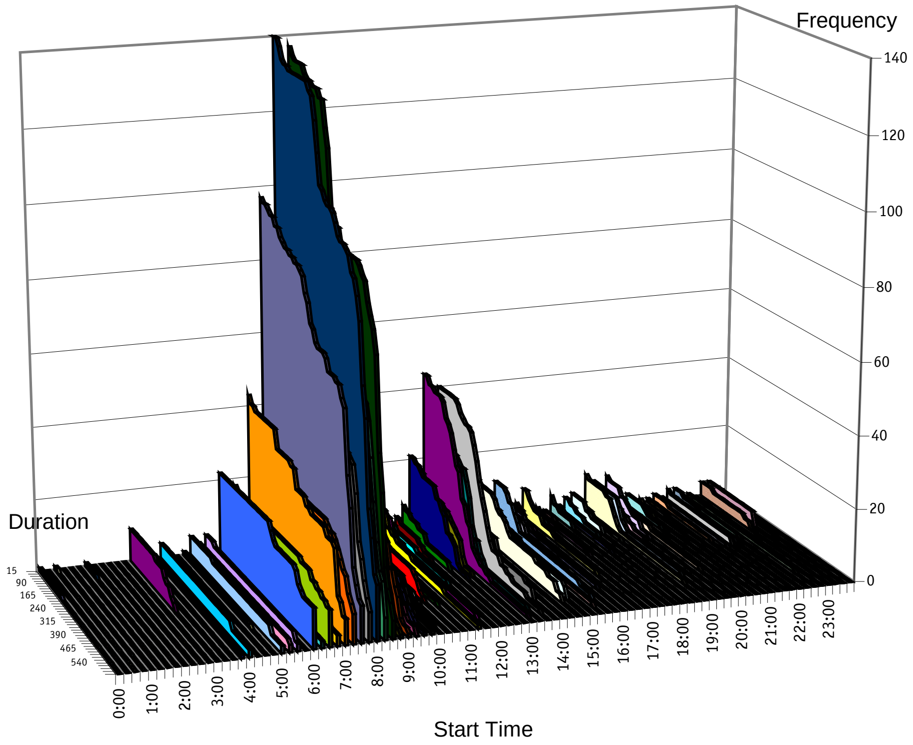


Figure 6.2: Deriving a data-driven time reward table

that will continue to work decreases when duration increases. An important increase was also found between 1.00 P.M. and 1.30 P.M., where the work activity was often resumed (and thus reported) after lunch. Based on the 15 minute time interval that was used in the figure, it was possible to approximate the 3936 cells (=41*96) that are needed for this reward table. As stated before, using less detailed reward tables (which means broader time intervals), is not an option given the average travel time that is needed for the car transport mode between all the locations in the study area (13 minutes). Reward tables have been developped for the following 17 activity categories: work or study in-home, bring or get persons or goods, daily shopping, non-daily shopping, service activities, medical visits, eating or drinking, sleeping, out-of-home leisure, in-home leisure, in-home non-leisure (household tasks), out-of-home non-leisure, receive social visit, bring social visit, work or study out-of-home, return home (e.g. drop bags) and “other” activities.

Location Reward Tables

Once all the time reward tables have been constructed, one should also pay attention about how to build location reward tables. It was mentioned before that location rewards depend on the travel time between two locations and on the transport modes that have been used to reach these locations (see section 6.3.2). Fortunately, travel time information between pairs of locations is available in our data (see Appendix A, Table A.4). All travel times in these tables are measured in minutes travel on the shortest route under free-floating conditions. For our experiments, matrices 1 till 4 in Table A.4 were used for the transport modes: walk, bike, car and public transport. Unfortunately, the simple linear interpolation procedure that was used to convert travel time information to reward tables in section 6.3.2, can no longer be applied in the integrated framework.

The aim of the integrated framework is to find an optimal time allocation (i.e. maximize reward) and in the meantime reduce the burden for travel (i.e. minimize travel duration). This is most conveniently modelled when a (small) negative reward (i.e. cost) is assigned to travel, while rewards for time allocation are positive values. However, the magnitude of the location cost versus time reward relationship is not known (see infra: interaction weight). Furthermore, one should take into account that a travel time of 45 minutes by foot may not have the same cost than the same travel duration by car. In order to solve both problems, a transition function is needed. In the context of our experiments, the following simple conversion function has been evaluated:

$$\text{Location Reward}=-w_i \times \text{travel\_time}^{w_{mode}}$$

In this formula, w_i (interaction weight) does not vary with the transport mode that is used. The parameter is only used to assess the relationship between location rewards (costs) and time rewards. When w_i>1, the cost of travel will be weighted relatively more in the integrated final solution than rewards that would be obtained by means of an optimal time allocation. So, most probably, the agent will try to optimize its travel by all means and will care less about the time that the activity has been carried out. When w_i<1, the agent will pay less attention to the travel cost. The agent will use travel to achieve its most optimal start time in order to get a better reward due to a more optimal time allocation. W_i is the only parameter that regulates the relationship between both reward tables; the time reward table is considered to be fixed (see previous section).

The other parameter w_{mode} needs to be set per transport mode. The parameter is used to evaluate the cost that the user has experienced per transport mode for a given travel time. A simple example of the behaviour of the function has been shown in Figure 6.3, where w_i has been kept constant and w_{mode} varied.

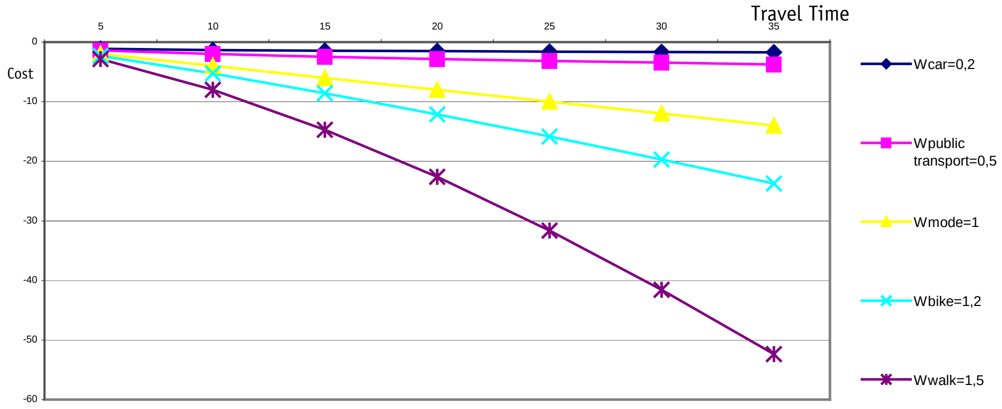


Figure 6.3: Plot of the function that is used to convert travel times to costs (rewards). w_{mode} is varied along with travel time; w_i is kept constant

It can be seen from the figure that for higher travel times, the cost that the user experiences is higher when travel time increases. This property is a precondition of the model. Obviously, the cost depends upon the transport mode that has been used and can be controlled by means of the parameter w_{mode}. In this example, a larger value has been assigned to w_{mode} for slow travel modes, because it was assumed that people are more reluctant towards using slow transport modes if they face a longer travel time. Obviously, these assumptions need to be compared with empirical results. Several empirical scenario's were tested; 8 of those were illustrated in Table 6.12 and empirically evaluated in the following section.

The scenario's can be subdivided into three different subcategories. Scenario A is an outlier. The scenario empirically evaluates whether the intuitive decision to assign larger values to w_{mode} for slow travel modes is correct (see Figure 6.3). To this end, a counter-scenario has been defined which evaluates the behaviour of the algorithm if the agent would have used weights that result in a larger cost for fast modes than for slower modes when travel time increases. Scenario's B till and E are developed to evaluate the effect of the interaction weight only. The mode specific costs were respectively kept constant at 0.2; 0.5; 1.2 and 1.5. On the contrary; scenario's F, G and H were used to vary mode-specific weights while the interaction weight remained constant.

Table 6.12: An overview of some of the parameters that have been varied in experiments

Scenario number	W_{mode}				w_i
	Car	PT	Bike	Walk	
A	1.5	1.2	0.5	0.2	1
B	0.2	0.5	1.2	1.5	0.4
C	0.2	0.5	1.2	1.5	1
D	0.2	0.5	1.2	1.5	2
E	0.2	0.5	1.2	1.5	5
F	0.3	0.6	1.1	1.2	1
G	0.4	0.7	1.4	1.6	1
H	0.6	0.8	1.5	1.7	1

6.4.3 Empirical Results

In order to evaluate time and location allocation, sequences of activities and travel are initially taken from the original data and not from the simulation procedure that was described in Chapter 5. As a result of this, the bias that might arise due to a possible prediction error of activity-travel patterns (Chapter 5), is kept separated from time and location allocation.

Time Allocation

The application of the procedure described in section 6.3.3 and further specified in sections 6.4.1 and 6.4.2, resulted in a time allocation that has been evaluated for the 8 different scenario's (see Table 6.13).

Table 6.13: Frequency distribution results of time allocation for scenario's A-H

	Obs. in original data	Q-learning (Time allocation)							
		A	B	C	D	E	F	G	H
3A.M.-6A.M.	**0.075**	0.094	0.082	0.087	0.085	0.083	0.083	0.082	0.089
6A.M.-9A.M.	**0.117**	0.189	0.101	0.091	0.173	0.202	0.102	0.103	0.097
9A.M.-12A.M.	**0.113**	0.178	0.097	0.093	0.170	0.201	0.099	0.101	0.098
12A.M.-3P.M.	**0.130**	0.113	0.152	0.162	0.112	0.100	0.164	0.154	0.166
3P.M.-6P.M.	**0.148**	0.102	0.121	0.109	0.121	0.092	0.125	0.129	0.121
6P.M.-9P.M.	**0.228**	0.201	0.246	0.251	0.207	0.211	0.233	0.232	0.234
9P.M.-12P.M.	**0.166**	0.110	0.182	0.192	0.121	0.102	0.180	0.181	0.180
12P.M.-3A.M.	**0.023**	0.014	0.019	0.015	0.011	0.009	0.014	0.018	0.015

The second column in this table shows the frequency distribution per time interval of activities and transport modes as represented in the original data. The most important conclusion that can be derived from this table, is the finding that both the interaction weight and the W_{mode} parameter, determine to a large extent the accuracy of the Q-learning solution. It should be emphasized that the correct w_i and W_{mode} parameters are respectively a reflexion of the respondents' attitude (or at least the attitude that was reported by them) towards the location cost versus time reward relationship and towards the use of transport modes when travel time increases.

To better understand this, let us take a look at Scenario B. This scenario proved to have generated reliable results that are close to the frequency distribution that was observed in the original data. The reason for this is the interaction weight of 0.4. It was mentioned before that when w_i<1, the agent will pay less attention to the travel cost and will use travel to achieve its most optimal start time in order to get a better reward. Keeping the w_{mode} parameter the same and further increasing the interaction weight w_i (scenario's C, D and E) leads to less accurate predictive results. Especially in scenario's D and E, the second and third time interval (6 A.M.-9 A.M. and 9 A.M.-12 A.M.) were overestimated. In this case, the cost of travel will be weighted relatively more in the integrated final solution than rewards that would be obtained by means of an optimal time allocation. So, the agent tried to optimize its travel by all means and cared less about time allocation.

Another important conclusion that can be derived from Table 6.13, is the finding that w_i and W_{mode} seem to be interchangeable to some extent. In other words, non-optimal W_{mode} parameters can be compensated by a more optimal choice of the w_i parameter and vice versa. For instance, scenario's F, G and H have used the less optimal interaction weight of 1 (compare with scenario C) but this could be compensated by a modification of the W_{mode} parameters. This is an important finding (see also evaluation at location allocation), especially because the best of these 3 scenario's (G) performed better than the scenario with w_i=0.4 (scenario B). The same trend can also be seen in scenario A, where a non-optimal choice of the w_{mode}-parameters has lead to poor predictive results.

Location Allocation

For the validation of location allocation, origin-destination matrices were derived from the original dataset. The distinct locations in the dataset are defined by means of different zip codes and are labelled as Rotterdam-Noord (1-3); Rotterdam-Zuid (4-6); Hendrik-Ido-Ambacht (7-9); Zwijndrecht (10-15), Paependrecht, Sliedrecht (16-19) and Elsewhere (20), as it was also shown in Table A.5. Subsequently, locations were allocated for each of the 8 scenario's that have been defined before. Correlation coefficients between the original dataset and the allocated locations were shown in Table 6.14. As opposed to time allocation, scenario's D and E produced the best results for location allocation. However, the finding is consistent with our previous conclusion, where it was found that in case w_i >1, the cost of travel was weighted relatively more in the integrated final solution (scenario's D and E) and as a result the agent optimized its locations choice and cared less about time allocation. It can be concluded that the performance of scenario G (which produced the best result for time allocation) is also satisfactory for location allocation. The experiments enabled us to get an idea about the parameter choice for this dataset that resulted in an acceptable trade-off between time and location allocation.

Table 6.14: Correlation coefficients between OD-matrices for scenario's A-H

Scenario	A	B	C	D	E	F	G	H
Correlation coefficient	0.821	0.831	0.871	0.892	0.895	0.864	0.872	0.869

6.5 Conclusion

This chapter further completed the simulation of activity patterns by the development of a methodology that was able to allocate time and location information to sequences that consist of activities and transport modes. It seems less likely that time and location information can be efficiently modeled by means of sequential dependencies. To this end, a technique of reinforcement learning (Q-learning) has been chosen where somebody can learn its optimal starting and end times and location information in interaction with the given fixed sequence of activities and transport modes.

During learning, the Q-learning agent tries some actions (i.e., output values) on its environment. Then, it is reinforced by receiving a scalar evaluation (the reward) of its actions. In a first implementation, we assumed that time allocation is dependent on the type of activity, the starting time of the activity and the time already spent at that activity. Also, the sequence of different activities determined the time allocation. Indeed, two sequences that contain a similar activity which has the same starting time and the same time spent at that activity, do not have to (and often will not) receive the same time allocation for that particular activity, as a result of the different sequence order in which other activities occur in both diaries. Technically, the agent will come up with another optimal path, a different policy chart and as a result also a different time allocation for both sequences. The location allocation problem was initially also solved in the assumption that the allocation is dependent on the travel time between two locations and on the transport mode that has been chosen to reach these locations. Also in this case, it is obvious that the sequence information of activities and transport modes largely determines the allocation.

Then, in our final implementation, the idea to integrate time and location allocation simultaneously, has been conceived. Dealing with both allocations simultaneously, leads to some important advantages. The first advantage is that the reward is not only maximized in either the time or the location facet, but the *total* reward in a day (i.e. the reward that arizes from determining optimal start and end times and the cost that arizes from travelling between locations) will be maximized by means of an integrated approach, which is obviously more realistic. The second major advantage is that flexible travel times between two locations can be incorporated. In our first time allocation implementation, it was impossible to achieve this, due to the lack of location information.

The most important drawback of this integrated implementation, is that the magnitude of the importance between the time and location relationship cannot be immediately observed from the data. To this end, a simple conversion function and different scenario's were evaluated in the empirical section. Further research could for instance use other alternative techniques (for instance stated preference) to better specify and understand this relationship. In our experiments, time allocation was evaluated by comparing the allocated frequency distribution of activities and transport modes with the observed frequency distribution that is present in the original data. Location allocation was evaluated by the calculation of a correlation coefficient between origin-destination information that is available from the original data with origin and destinations that are allocated. The first empirical results seem to indicate that time information is allocated quite well, but correlation coefficients of OD-matrices seem to be somewhat lower than what we are used to in the Albatross model.

Finally, it should be noted that sequences of activities and travel are taken from the original data and not from the simulation procedure that was described in Chapter 5. The main aim for doing this was to result in an optimal parameter choice for our dataset, without having to care about the bias that will arise due to a prediction error of activity-travel patterns. This optimal parameter choice can then be used in the integrated framework that has been presented in the following chapter.

Chapter 7
Final Conclusions and Comparisons

7.1 INTRODUCTION

The performance of the individual components of the simulation model was evaluated separately in Chapters 5 and 6. For example, predicted activity-travel patterns (Chapter 5) were not yet integrated in the validation of time and location allocation in Chapter 6. Furthermore, the validation measures that have been used in these chapters are less challenging, since they mainly test how well the model is able to replicate average values (see Tables 5.9-5.12 and Table 6.13). While these measures give us a good idea about the predictive performance of the individual facets of a model, a more thorough validation is required in an integrated (i.e. Chapters 5 and 6) model. To this end and consistent with the validation measures that have been used within the Albatross model, the Sequence Alignment Method and the correlation coefficients based on origin-destination matrices were calculated in the integrated data-simulation approach that was proposed in the Chapters 5 and 6.

Unfortunately, even these more advanced validation measures do not enable us to determine whether the lack-of-fit of a model is the result of the remaining noise in the data or whether it is due to the model specification as such. To this end, and to get a better idea about the validation of the model, initial results have been presented that compare the predicted activity patterns of the Simulation model with the predicted activity patterns of the Albatross model (for the facets activity, transport mode, location and time). Despite the fact that both models are based on an entirely different approach, it might be interesting to see how both models perform on the same dataset. Comparisons based on empirical data between other activity scheduling models and (micro-) simulation models are, to the best of our knowledge, non-existing. One of the few studies comparing relative predictive performances between the Albatross model and a linked logit-Poisson model is reported in Arentze and Timmermans (2000) and in Arentze *et al.* (2000). It was shown that Albatross achieved a better predictive performance of approximately 10%.

The purpose of this final chapter is thus to unite the most important contributions of the dissertation that were presented in Chapters 3 and 4 with the research presented in Chapters 5 and 6. Therefore, we will elaborate in the second section of this chapter on the algorithms that have been chosen for comparison within Albatross (Chapters 3 and 4) and on the parameter values that have been used in our simulation framework (Chapters 5 and 6). The third section brings to mind the theoretical differences between both approaches. It is important to take these theoretical differences into account when assessing predictive results. Those results are presented in section 4 at the level of the Sequence Alignment Method and in section 5 at the level of origin-destination matrices. Final conclusions are reported in section 6.

7.2 DEFINING THE VALIDATION CONTEXT

Due to the different character of the Simulation model and the Albatross model, important decisions need to be made at the level of both models before one will be able to produce validation estimates. Let us first proceed with decisions at the level of the algorithms used in the Albatross model.

7.2.1 ALBATROSS MODEL

With respect to the Albatross model, the original CBA algorithm was chosen from Chapter 3 and the BNT technique from Chapter 4 as initial competitors for our Simulation model. The choice for both algorithms is a logical decision, which is based on the solid predictive performance of both algorithms. Parameter setting with respect to CBA and BNT was the same as reported in their respective chapters. This means that for CBA, minimum support value was set at 1% and minimum confidence at 10%, while the maximum number of conditions that can appear in any CAR was restricted to 6 to ensure the comprehensibility of every single rule. For BNT, an entropy reduction of 0.001 bits was used as a threshold to select the most important variables (pruning). In addition to this, C4.5 (Quinlan, 1993), a well-known decision tree induction algorithm is taken as an additional competitor of a supervised learning method in the Albatross model. In order to compare these models on (approximately) the same basis, the second person in the household and the "with whom" facet were removed from the predictive analyses since the proposed Simulation model was not yet able to deal efficiently with household interactions. When compared to the Albatross model, this is a clear disadvantage. A study by Timmermans that dates from (2001) has shown that Albatross is the only model that was able to account for the "with whom" facet. Recently, other models have been developed such as for example the work by Pribyl and Goulias (2004) where interactions among members of the household are explicitly incorporated.

7.2.2 SIMULATION MODEL

Generating activity-travel patterns (Chapter 5) requires decisions about the computation of transition probabilities (i.e. storing activity bundles in approaches 1 or 2, see section 5.4), the inclusion of socio-demographic segmentation (see section 5.5) and the order of the dependency information that is considered (see section 5.4). Given the fact that Approach 2 (see section 5.4.2) took more information into account (i.e. not only the most frequently occurring combinations of elements were stored in activity bundles) and given its higher degree of correspondence with the calculation of original Markov Chain transition probabilities; it is preferred in the integrated Simulation model. The segmentation is also certainly recommended because some of the most powerful explanatory rules that were generated by CBA and BN in the Albatross model, included a significant number of socio-demographic (or other more general) variables (see Table 2.2 and sections 3.7 and 4.6). Based on these decisions (Algorithm 2 and segmentation), we need to rely upon Tables 5.10 and 5.11 to set the appropriate order (lag) that needs to be used for predicting activity-travel patterns. Based on these results, a transition matrix that incorporates relatively low-order dependencies is favoured. For the sake of simplicity, a first-order transition matrix has been used in the integrated Simulation model.

Other parameters need to be set at the level of time and location allocation. First of all, one has to decide about the discounting factor γ, the learning rate α, the exploration rate $P_{explore}$, the time interval that has been used and the maximum duration that an activity can last. For the reasons that were already mentioned in section 6.4.1, the discounting factor is set equal to 0.95, and the learning rate and the exploration rate are set equal to 1. Also in the integrated model, the time interval and the maximum duration were respectively set at 15 minutes and 10 hours. Other parameters are the maximum number of loops and the maximum number of trials (see section 6.3.1). Both were respectively set at 1000000 and

100 to make sure that a solution has been reached that converged to an optimal policy. It was already mentioned in section 6.3.1 that the parameter setting of the algorithm do not influence the optimal learning solution, but there is however an impact on the time that is needed for the algorithm to converge to this optimal solution. Indirectly, the parameter setting thus determines the maximum number of trials and loops that needs to be set. An interesting topic for future research may be to conduct experiments that determine the most optimal parameter setting in an effort to reduce this computation time.

Also in this integrated model, reward tables are important for the final validation. With respect to time allocation, we relied upon section 6.4.2 in order to provide an estimate for time reward. For location allocation, the procedure in the same section (6.4.2) has been used. Parameters were set fixed to W_{car}=0.4; W_{PT}=0.7; W_{bike}=1.4; W_{walk}=1.6; and w_i=1 based on the results that were reported in section 6.4.3.

7.3 Differences between both Approaches

The general theoretical differences between activity-scheduling and simulation models were already briefly mentioned in previous parts of the dissertation. It is important to reformulate both frameworks in terms of possible advantages and disadvantages that can be kept in mind when assessing comparative (quantitative) results.

Most activity-scheduling models rely upon the use of a skeleton of fixed activities. This distinction may not only have an impact on the predictive capabilities and on the complexity of the modelling task, but it also enables and/or facilitates the evaluation of some travel demand management measures. Especially schedule-specific scenarios, such as for instance a shift in terms of starting time in the fixed schedule for work activities, are easier to model in Albatross for instance than in a simulation model. It was repeatedly stated in the dissertation that a distinctive feature of the Albatross model is the ability to disentangle the complex decision making process in separate individual small decision processes (for a more elaborated framework, we refer to Albatross version 2 (Arentze and Timmermans, 2002), which uses 27 decision trees in its model conceptualisation). Also in this case, we believe that this feature facilitates the evaluation of TDM. The reverse side of this is obviously the cost for calibrating the model in another spatial context, which at least will require the conversion of the diaries that are reported by the respondent towards explanatory and dummy variables (see Appendix B) and datasets that link together the individual decision facets. When this is established, a learning algorithm has to be calibrated that determines the outcome of every single decision facet. The burden of this process depends on the number of decision facets that has been used and on the complexity of the learning algorithm.

It is exactly this feature that is probably one of the most important arguments that can be given in favour of our proposed Simulation model. Its ease of application, also in other spatial contexts, is facilitated by the fact that the integrated model is almost completely data-driven and mainly uses full activity diaries and travel times between a pair of destinations as input. The degree of application of the model is therefore mainly determined by the availability and level of detail of both travel time information and activity diary data. The model will rely upon transition matrices that are subdivided in terms of socio-demographic information that is available from the sample. If the sample is assumed to be representative for the population, well-known techniques such as Iterative Proportional Fitting (Beckman *et al.*, 1996) can be used to generate new synthetic populations, and the information that is contained in the correct (in terms of socio-demographic information) transition probability matrix can be used to simulate full activity patterns. Specific (but more limited) TDM scenarios such as for instance a change in socio-demographic or explanatory variables such as household income, car possession, etc. can be evaluated by re-calibrating the full model. Another promising use is the integration with traffic simulation. Other possible applications for simulation modelling have been mentioned before in section 1.1.4-subsection Simulation models.

7.4 Comparisons of Fit Based on SAM Measures

The predicted activity patterns of the Albatross model and the predicted patterns of the Simulation model were compared by means of the Sequence Alignment Method with the observed patterns that are available from the data. Results were shown in Table 7.1.

For reasons of comparability, the patterns that were generated by both type of models, need to be transformed. The predicted patterns were aggregated into discrete time intervals, to ensure a more equal comparison. It can be seen from Table 7.1 that the Simulation model performed worse when compared to the Albatross model. However, it should be kept in mind that the validation measure itself is not completely bias-free due to the use of (non-predicted) fixed activities and location and mode elements that are related to these fixed activities in Albatross and the lack of distinction between fixed and flexible activities in the Simulation model.

Table 7.1: SAM distance measures (full patterns)

SAM distance measure	Albatross			Simulation Model
	CBA	BNT	C4.5	
SAM activity-type	2.710	2.712	2.719	3.113
SAM location	3.111	3.101	3.109	5.135
SAM mode	4.515	4.419	4.439	4.975
UDSAM	16.319	16.313	16.328	17.551

In Table 7.2, the fixed activities were both removed from the predicted patterns of the Albatross model and from the predicted patterns of the Simulation model to try to achieve a more equal comparison. Albatross does predict the mode dimension of the fixed activities, so, theoretically spoken, these elements need not be removed from the mode strings. However, maintaining the mode strings while removing activity and location elements is undesirable. To this end, related mode and location elements were removed, along with the fixed activities. A similar procedure has been proposed in the previous comparison between Albatross and a linked Logit-Poisson model. More information about this procedure can be found in Arentze and Timmermans (2000).

Table 7.2: SAM distance measures (fixed elements removed)

SAM distance measure	Albatross			Simulation Model
	CBA	BNT	C4.5	
SAM activity-type	2.513	2.515	2.519	2.719
SAM location	2.682	2.685	2.690	4.832
SAM mode	2.625	2.622	2.629	2.929
UDSAM	11.356	11.352	11.362	11.921

The Simulation model achieved better results than in Table 7.1 (probably due to the more equal basis for comparison), but still performed somewhat worse when compared to Albatross. Within Albatross, the BNT approach performs slightly better than CBA and C4.5. It should be emphasized however, that the comparison is not yet completely equal because the Simulation model predicted the “fixed” elements in the diary, while they needed to be removed from the sequences that were used for comparison.

7.5 COMPARISONS OF FIT BASED ON OD-MATRICES

The second measure to evaluate the predictive performance of both models is carried out at trip level. The origins and destinations of each trip, derived from the activity patterns, are used to build OD-matrices. The origin locations are represented in the rows of the matrix and the destination locations in the columns. The number of trips that is undertaken from a certain origin to a certain location is used as a matrix entry. In order to determine the degree of correspondence between predicted and observed matrices, a correlation coefficient was calculated. To this end, cells of the matrix are rearranged into one array and the calculation of the correlation is based on comparing the corresponding elements of the predicted and the observed array. It can be seen from Table 7.3 that the correlation coefficient is lower in the Simulation model than in the CBA, BNT (unsupervised) and C4.5 (supervised) learning algorithms (Albatross model).

Table 7.3: Correlation coefficients based on OD matrices

	Albatross			Simulation model
	CBA	BNT	C4.5	
Correlation Coefficient (OD)	0.945	0.942	0.940	0.879

7.6 CONCLUSION, DISCUSSION AND FUTURE RESEARCH

This final chapter joined together the most important contributions of the dissertation that were presented in Chapters 3 and 4 with the research illustrated in Chapters 5 and 6. In order to get a better idea about the predictive performance of both models, an initial comparative study has been conducted. The performance of CBA and BNT (Chapters 3 and 4) and the C4.5 algorithm were used in the Albatross model and were compared with the full Simulation model (Chapters 5 and 6). CBA and BNT were chosen as competitors, based on their solid predictive performance that has been obtained in the dissertation. C4.5 was chosen as an example of a supervised learning algorithm, for the sake of completeness. After a data transformation phase, predicted patterns of both models were compared with the observed patterns that are available from the data.

At the level of SAM distance measures, the simulation model performed somewhat worse when compared to CBA, BNT and C4.5 but differences were minor. Larger differences were observed at the level of SAM location and from the origin-destination correlation coefficients. Based on these results, it can be concluded that the Q-learning time allocation and the simulated activity-travel sequences derived in Chapter 5, performed fairly well, while location allocation achieved worse results. Also important to notice is that the correlation coefficient that was obtained by means of the integrated simulation approach (see Table 7.3), remains very stable in comparison with the correlation coefficients that were calculated before, when sequences where taken from the original data (see Table 6.14). This merely supports the previous remark that Q-learning time allocation and the simulated activity-travel sequences of Chapter 5 performed well.

With respect to location allocation, a couple of additional remarks are needed. Probably, the main reason for the high SAM distance and the somewhat worse correlation coefficient, is the fact that location allocation is only made dependent on the travel time between two locations and on the transport mode that has been chosen to reach these locations. Other dimensions might have determined this location choice in reality, such as for instance the concept of "location orders" as it was defined in Albatross. Location orders can be made dependent on the facilities available at the location (e.g. floor space, number of outlets, etc.). The Q-learning location allocation solution did not account for these available facilities or other explanatory factors and this might have prevented the algorithm for generating better results. A topic for future research is to investigate how these can be incorporated into a possible Q-learning location allocation solution. More research is also needed about the magnitude of the importance between the time and location relationship that was proposed in Chapter 6. It might be desirable to develop an algorithm that is able to automatically determine the most optimal parameters, given the dataset under consideration. It can be expected that such an adaptation and the use of more advanced location reward conversion functions than the one that was proposed in Chapter 6, can result in a more accurate location allocation result. Finally, it might also be particularly interesting to evaluate the performance of the simulation approach against other models, especially because the previously mentioned benchmarking study (which used different empirical assumptions than the one reported in this section, and is therefore not fully representative) indicated that the linked logit-poisson model only achieved a correlation coefficient of 0.687 between observed and predicted origin-destination matrices. In addition to these comparative studies, the developed simulation model also need to be complemented with a population generation module in future research.

APPENDIX A

This appendix gives a general overview of the structure of the most important data files that have been used throughout the dissertation (Chapters 3-6). All data have been collected in the municipalities of Hendrik-Ido-Ambacht and Zwijndrecht in the Netherlands. We are grateful to the Urban Planning Group in The Netherlands for providing these data.

A.1 Activity Classes and Detailed Activities

Table A.1 gives an overview of the activity classes and detailed activities which are used in the diaries. A pre-coded scheme was used for activity reporting. Eighteen different activities classes were distinguished. The activity categories are work or study in-home, bring or get persons or goods, daily shopping, non-daily shopping, service activities, medical visits, eating or drinking, sleeping, out-of-home leisure, in-home leisure, in-home non-leisure (household tasks), out-of-home non-leisure, receive social visit, bring social visit, work or study out-of-home, return home (e.g. drop bags), "other" activities and a missing category (added in post-processing). The transport modes which respondents could report were car (as driver or as car-passenger), walk, bike and public transport.

Table A.1: Activity classification and detailed activities that occur in diaries

Activity Class	*Activity*	*Activity Class*	*Activity*
Work/study in home	/	In home leisure	Social activity with household members
			Telephone
			Sports
			Union/Friends
Bring/Get persons or goods	Get/bring child day care Or school		Watching sports
	Get/bring child other		Watch tv
	Get/bring other persons		Reading
	Get/bring goods		Other in home leisure
Daily shopping	/	In home non leisure	Voluntary work
Nondaily/ Window shopping	Non-daily shopping		Tele shopping
	Window shopping		Tele banking

Service activities	Food takeaway		Prepare food
	Rent movie		House keeping
	Personal business		Child care
Medical visits	Medical visit		Pet care
Eating/Drinking	Eat/drink		Administration
	Breakfast		Other in home non-leisure
	Lunch	Out of home non leisure	Voluntary work
	Dinner		Prepare food
Sleeping	/		House keeping
Out of home leisure	Social activity with household members		Child care
	Telephone		Pet care
	Sports		Administration
	Cafe/bar		Other out of home non-leisure
	Restaurant	Social visits receive	/
	Concert	Social visits bring	/
	Library	Work/school out of home	/
	Union	Other	Other
	Swimming		Do nothing
	Watching sports	Return home	/
	Church	Missing	/
	Touring:walk		
	Touring:bike		
	Touring:car		
	Watch tv		
	Reading		
	Other out of home leisure		

A.2 Household Attribute Data

Household attribute data are stored in a separate data file, where each line corresponds with a household and includes the information as shown in Table A.2.

Table A.2: Structure of household attribute data file

Definition	Categories
Household ID	Corresponds with schedule file (See section A.3)
Questionnaire version	Integer number (not used)
Home Location	Four digit zip code
Socio-economic Class	1: minimum, 2: low, 3:medium, 4:high
Age of oldest member	1: < 25; 2: 25-44; 3: 45-64;4:> 64
Household type	1: single, no work; 2: single, work 3: double, one work; 4: double, two work; 5: double, no work
Children	1: none; 2: younger than 6 3: 6-12; 4: older than 12
Number of cars	Integer number
Number of bikes	Integer number
Work time, person 1	Number of official hours work per week
Car availability, person 1	0:no, 1:yes
Bike availability, person 1	0:no, 1:yes
Gender, person 1	1:male, 2:female
Work time, person 2	Number of official hours work per week
Car availability, person 2	0:no, 1:yes
Bike availability, person 2	0:no, 1:yes
Gender, person 2	1:male, 2:female

A.3 Activity Diary Data

Obviously, the main source of information is stored in activity diary data. The structure of the activity diary is shown in Table A.3. In reality, data contain more detailed information such as earliest and latest possible end times, minimum and maximum duration, etc. (not shown here).

Table A.3: Structure of Activity diary data

Header for each case:	**Categories**
Household ID	Corresponds with Household attribute data file (See section A.2)
Day of the week	1: Monday, 2: Tuesday, …, 7: Sunday
Number of activities	Integer number, equals number of lines of a block of activities
Fields for each activity:	**Categories**
Activity class	See section A.1
Activity	See section A.1
Person ID	1: first person in household file, 2: second person in household file

With whom/travel party	1:alone, 2: with others in household, 3: with others outside household, 4: with other in and out
Activity start time	24 hour notation: 300: 3AM – 2700: 3 AM next day
Activity end time	24 hour notation: 300: 3AM – 2700: 3 AM next day
Location	Out-of-home: four digit zip code, in-home:0
Travel Mode	1: car driver, 2: Walk; 3: bike, 4: Public Transport, 6: Car Passenger. Missing if no travel
Travel Time	Minutes (missing if no travel)
Waiting time component of travel time (for instance for public transport)	Minutes (missing if no travel)

A.4 Travel Time and Travel Distance Data

The travel time and travel distance data are contained in 13 mode-specific 104x104 matrices. The matrices report 104 different four-digit codes that can be aggregated into broader location zones (see section A.5). All travel times are measured in minutes travel on the shortest route under free-floating conditions. Table A.4 gives a description of the information that is stored per matrix.

Table A.4: Travel time and travel distance information per matrix

Matrix number	**Contents**
1	Travel times: walk
2	Travel times: bike
3	Travel times: car
4	Travel times: public transport, fastest connection
5	Travel times: public transport, average across all connections
6	Travel times: walking-time component of public transport travel time
7	Travel times: in-vehicle time component of public transport travel time
8	Travel times: waiting component of public transport travel time
9	Number of transfers in public transport connection
10	Travel distance: walk
11	Travel distance: bike
12	Travel distance: car
13	Travel distance: public transport

A.5 Location Data

Location data that are reported in activity diaries can be grouped into the following 20 zones (see Table A.5).

Table A.5: Postal codes and corresponding zones

Zone description	Zone number	Postal codes (4 digit zip code). (Wildcard * denotes that any number can follow)
Rotterdam-Noord	1	302*,303*,304*,305*
	2	306*
	3	301*
Rotterdam-Zuid	4	308*
	5	307*
	6	298*
Hendrik-Ido-Ambacht	7	3341
	8	3342
	9	3343
Zwijndrecht	10	3331
	11	3332
	12	3333
	13	3334
	14	3335
	15	3336
Paependrecht, Sliedrecht, Dordrecht	16	335*,336*
	17	3311,3312
	18	3314,3316,3317
	19	3313,3315,3318,3319,3328,3329
Elsewhere	20	Other

APPENDIX B

This appendix gives an overview of the independent variables that are used for predicting the nine different choice facets within the Albatross system (Chapters 3-4). The appendix is largely based upon the description of the variables as it has been illustrated by Arentze and Timmermans (2000) and more recently by Moons (2005).

B.1 Mode for Work

The first decision choice facet that is considered in the Albatross system is the "Mode for Work" dimension. The independent variables for the "Mode for work" dimension are shown in Table B.1. The first variable, *group* is included to allow the system to distinguish between cases where there is no partner or in case there is, whether the partner's schedule for that day is either unknown or known. The next group of variables describe the activity program at the level of the schedule skeleton (S). These include the total time engaged in *Work1* and in *Work1* and *Work2* together. *Work1* includes work/school activities and *Work2* voluntary work activities. The number of mandatory, out-of-home activities other than work and the presence of a bring/get activity are also incorporated at the level of the schedule skeleton. Other variables are related to the partner, where the variables are equal to zero if there is no partner, or if the partner's schedule is unknown. The succeeding variables describe the chain of work episodes for which a mode choice is to be made (Work-chain, W). These include work time and travel time information. Bike travel time is taken as an indicator for travel distances. Furthermore, travel time ratios between modes are included as indicators for the relative speed of each mode on the (shortest) route between locations. Also included at the level of the work chain, are a number of descriptors such as the start time of the first work episode and end time of the last work episode of the chain. The number of different work locations involved, serve as a measure of the amount of travel involved apart from the first and last commute. Activities that are included in the skeleton and that are closely related in time to the start of the first work episode or the end of the last work episode, are recorded as possible conditions for trip-chaining during the first and the last commute. Finally, the last set of variables cover travel demands of the partner during the work-chain. These include the number of out-of-home activities in the schedule skeleton, maximum travel time across locations and the presence of a bring/get activity.

Table B.1:Explanatory variables in the “Mode for work” choice facet (Albatross)

Variable Name	Description	Categories
Group	Partner status	1: no partner; 2: partner schedule unknown; 3: partner schedule known
Two	Total time of Work1 in minutes in S	0: 0; 1: ≤240; 2: 241-360; 3: 361-480; 4: > 480
Ttot	Total time of Work1 and Work2 in S	1: ≤240; 2: 241-360; 3: 361-480; 4: > 480
Nsec	Number of mandatory, out-of-home activities other than work in S	0: 0; 1: 1; 2: 2; 3: 3-4; 4: 4-5; 5: > 5
yBget	There is a bring/get activity in S	0: yes; 1: no
Pwo	Total time of Work1 in minutes in S of partner	0: 0; 1: ≤240; 2: 241-360; 3: 361-480; 4: > 480
PTtot	Total time of Work1 and Work2 in S of partner	1: ≤240; 2: 241-360; 3: 361-480; 4: > 480
PNsec	Number of fixed out-of-home activities other than work in S of partner	0: 0; 1: 1; 2: 2; 3: 3-4; 4: 4-5; 5: > 5
PyBget	There is a bring/get activity in S of partner	0: yes; 1: no
Tbike	Objective travel time by bike to location of W in minutes	1: ≤10; 2: 11-20; 3: 21-30; 4: 31-50; 5: 51-100; 6: > 100
Rcabi	Ratio car/bike travel time in%	1: ≤25, 2: 26-50; 3: 51-75; 4: > 75
Rpubi	Ratio public transport/bike travel time in %	1: ≤100; 2: 101-150; 3: 151-200; 4: > 200
Rpuca	Ratio public transport/car travel time in %	1: ≤300; 2: 301-500; 3: 501-700; 4: > 700
Peak1	Start time of W falls in 7:30-9:00 AM	0: yes; 1: no
Peakn	End time of W falls in 17:00-18:00	0: yes; 1: no
Two2	Total time of W in minutes	1: ≤300; 2: 301-500; 3: 501-700; 4: >700
Nloc	Number of different locations in W	1: one; 2: more than one
Avo	Activity in S with end time within 1-hour interval before first work episode in W	1: none, 2:bring/get; 3: other

Ana	Activity in S with start time within 1-hour after last work episode in W	1: none; 2:bring/get; 3: other
Pywork	Partner has work activity during work time	0: no; 1: yes
Pybget2	There is a bring/get activity in S of partner during W	0: no; 1: yes
PNfix	Number of out-of-home activities in S of partner during W	0: none; 1: one; 2: more than one
PTTmax	Maximum bike travel time across activities in S of partner during W (minutes)	0: none; 1: 1-15; 2: 16-30; 3: > 30

B.2 Activity Selection, Travel Party and Duration

Activity selection, travel party and duration decision facets are considered after the mode for work decision facet, as illustrated in Figure 2.2.

In Table B.2, a distinction can be made between explanatory variables for fixed and flexible activities. The fixed activities belong to the skeleton of the schedule, are considered as given and remain constant during the process. The variables Two and Ttot belong to the fixed activities and are respectively defined as the total time scheduled for Work1 and for Work1 and Work2 together. Twincl is added to take observed travel time as well as activity time related to Work1 activities into account. In the present step, the travel time information is considered known, given our assumption that transport mode choice for primary work activity is made in the previous step.

The other variables (except for yBget) belong to the flexible activities and they define for each flexible activity the total time scheduled (T-variables) or, simply, the presence of the activity (y-variables) in the current schedule. The variable values are initially equal to zero and they are updated each time an activity is added. The Nsec variable is a summary variable which represents the number of flexible or fixed out-of-home activities other than work in the current schedule. The A1dur variable represents an alternative way of encoding activity time. The variable defines a short, average and long time relative to the activity type under concern, such that for example the long category of one type may still be shorter than the average of another type. The definition of the duration categories corresponds with the alternatives considered for the duration choice (see Table B.3).

The next set of variables describe cases at the schedule level. First, the Tmax(*t*) variables represent the maximum time available across available time slots in the schedule skeleton. The index *t* defines a particular time period among six distinguished time periods: before 10 AM; 10-12 AM; 12-2 PM; 2-4 PM; 4-6 PM and after 6 PM. The time for each time slot and each time period is determined by the overlap between time ranges given by opening hours of available facilities for the activity type, the time between fixed activities and the time period *t*. Second, the yCar(*t*) variables represent the availability of the car in each time period *t*, as a function of the number of available cars in the household, the number of adult members of the household and the mode used by the partner for work. For example, the car is considered not available if the car is in use for work by the partner and there is less than one car per adult available in the household. The equal-frequency method was used to discretise continuous variables.

Besides activity type (Atype), the activity-level variables are specific for each of the three considered choice facets. The variables at this level represent feasibility conditions for choice alternatives. Selecting an activity is considered infeasible if the maximum available time across the time slots that are available within opening hours of available facilities for the activity is shorter than the minimum duration for the activity type ($\text{Yavail}_{\text{Select}}$). For travel party decisions, the options 'alone' and others outside the household' are considered to be always available. The 'other(s) inside the household' option, however, is considered available only in multi-person households ($\text{Yavail}_{\text{party}}$). Given the positive selection decision, the shortest duration class is available by definition. The average and long duration alternatives, however, are available only if the minimum duration defined for the concerned class fits in the schedule.

Table B.2: Explanatory variables in the “Activity selection”, “Travel party” and “Duration” choice facets (Albatross)

Variable Name	**Description**	**Categories**
Iact	Number of instances of the current activity type in S	0: 0; 1: 1; 2: > 1
Two	Total time of Work1 in S (in minutes)	0: 0; 1: ≤240; 2: 241-360; 3: 361-480; 4: > 480
Twincl	Total time of Work1 incl. travel in S	0: 0; 1: ≤260; 2: 261-380; 3: 381-500; 4: > 500
Ttot	Total time of Work1 and Work2 in S	0: 0; 1: ≤240; 2: 241-360; 3: 361-480; 4: > 480
Nsec	Number of out-of-home activities	0: 0; 1: 1; 2: 2; 3: 3; 4: 4;

	other than work in S	5: > 4
yBget	There is a bring/get activity in S	0: no; 1: yes
yDshop	There is a daily shopping activity in S	0: no; 1: yes
yServ	There is a service activity in S	0: no; 1: yes
yNDshop	There is a non-daily shopping activity in S	0: no; 1: yes
ySoc	There is an out-of-home social activity in S	0: no; 1: yes
yLeis	There is an out-of-home leisure activity in S	0: no; 1: yes
Tsoc	Total time of social activities (in-home and out-of-home) in S	0: 0; 1: ≤30; 2: 31-60; 3: 61-120; 4: > 120
Tleis	Total time of out-of-home leisure activities in S	0: 0; 1: ≤30; 2: 31-60; 3: 61-120; 4: > 120
Td-shop	Total time of daily shopping activities in S	0: 0; 1: ≤20; 2: 21-40; 3: 41-60; 4: > 60
Tserv	Total time of service activities in S	0: 0; 1: ≤20; 2: 21-40; 3: 41-60; 4: > 60
Tnd-shop	Total time of non-daily shopping activities in S	0: 0; 1: ≤30; 2: 31-60; 3: 61-120; 4: > 120
A1dur	Total relative time of current activity in S	0: none; 1: short; 2: average; 3: long
Tmax(t)	Maximum available time in t-th time interval in S^{fix} (in minutes)	0: 0; 1: 1-30; 2: 31-60; 3: > 60
yCar(t)	Availability of car in t-th time interval in S^{fix}	0: no; 1: yes; 2: schedule partner is unknown
Atype	Activity type	1: daily shopping; 2: service; 3: non-daily; shopping; 4: social; 5: leisure
$\text{YAvail}_{\text{selection}}$	Selection of activity is feasible given S and minimum duration for the activity type	0: no; 1: yes
$\text{YAvail}_{\text{party}}$	'Others in the household' option is available given the household composition	0: no; 1: yes
$\text{yAvail2}_{\text{duration}}$	The 'average' duration class is feasible given S and the minimum duration for that class	0: no; 1: yes
$\text{yAvail3}_{\text{duration}}$	The 'long' duration class is feasible given S and the minimum duration for that class	0: no; 1: yes
$\text{Awith}_{\text{duration}}$	Travel party	0: none; 1: only others inside hh; 2: others outside hh involved

Short, average and long durations depend upon on the type of flexible activity. The classification that is used within the Albatross model has been shown in Table B.3.

Table B.3:Activity duration classification

	Short		Average		Long	
type	range	mean	range	mean	range	mean
daily shopping	[10-20]	15	[21-45]	35	[46-90]	50
service	[5-10]	5	[11-20]	15	[21-40]	30
non-daily shopping	[10-30]	20	[31-80]	60	[81-160]	90
social	[10-75]	60	[76-150]	120	[151-300]	180
leisure	[10-60]	40	[61-120]	90	[121-240]	150

B.3 Activity Start Time

Table B.4 shows the independent variables for the "start time" choice facet. The characters A and S^{all} in Table B.4 respectively denote the concerned activity and the complete observed schedule. There are quite some variables which are similar to the independent variables that were used for the "Activity selection", "Travel party" and "Duration" facets.

However, one of the variables which is different is Tmax (*t*), which represents for each distinguished time interval *t* (before 10 AM, 10-12 AM, 12-2 PM, 2-4 PM, 4-6 PM and after 6 PM).) the available time in the current schedule given start and end time times of the fixed activities, the opening hours of available facilities for the concerned activity and estimated travel times for the free as well as for the fixed activities in the current schedule. Tmax represents the maximum time across feasible positions in the current schedule. Because the location, mode and trip chains are not yet known in this stage, the travel time estimates are based on activity-type specific ratios between activity type derived from the entire data set. These ratios are shown in Table B.5.

The Tmax variables are updated after each start time decision. Initially, only the schedule skeleton is given and the fixed start and end times determine the available time in each position. The levels for Tmax are defined dependent on the duration class of the activity under concern. The zero level means that there is no feasible schedule position for the *t*-th start-time range even if the minimum duration of the activity is taken. The levels 1 and 2 denote respectively, that there is a feasible position for implementing an average and long duration type of activity. Hence, the Tmax variable has two functions. First, it defines the feasibility condition for each start-time option and second, it indicates the extent to which each time period allows flexible choice of activity duration.

The next set of variables allows the system to anticipate on possibilities to establish connections with other out-of-home activities. A number of various indicators has been included. First, Btwo(*t*) indicates whether the current schedule includes a work activity with a start time falling in the *t*-th time period. Second, the ETx(*t*) denotes the same for the end time of any out-of-home activity. For existing flexible activities possible end times given duration and start-time constraints are taken. Note that for other than work activities only the end times are taken into account. This is done to reduce redundancy in the set of variables. Other-than-work activities tend to be short so that start and end times

Table B.4: Explanatory variables in the “Start time” choice facet (Albatross)

Variable Name	**Description**	**Categories**
Nsec	Number of mandatory out-of-home activities other than work in S^{all}	0: 0; 1: 1; 2: 2; 3: 3-4; 4: > 4
Two	Total time of Work1 in S^{all} (in minutes)	0: 0; 1: ≤240; 2: 241-360; 3: 361-480; 4: > 480
Twincl	Total time of Work1 incl. travel in S^{all}	0: 0; 1: ≤260; 2: 261-380; 3: 381-500; 4: > 500
Ttot	Total time of Work1 and Work2 in S^{all}	0: 0; 1: ≤60; 2: 61-120; 3: 121-240; 4: > 240
yBget	There is a bring/get activity in S^{all}	0: no; 1: yes
yDshop	There is a daily shopping activity in S^{all}	0: no; 1: yes
yServ	There is a service activity in S^{all}	0: no; 1: yes
yNDshop	There is a non-daily shopping activity in S^{all}	0: no; 1: yes
ySoc	There is an out-of-home social activity in S^{all}	0: no; 1: yes
yLeis	There is an out-of-home leisure activity in S^{all}	0: no; 1: yes
Tsoc	Total time of social activities (in-home and Out-of-home) in S^{all}	0: 0; 1: ≤30; 2: 31-60; 3: 61-120; 4: 121-240; 5: > 240
Tleis	Total time of out-of-home leisure activities in S^{all}	0: 0; 1: ≤30; 2: 31-60; 3: 61-120; 4: 121-240; 5: > 240
Td-shop	Total time of daily shopping activities in S^{all}	0: 0; 1: ≤20; 2: 21-40; 3: 41-60; 4: > 60

Tserv	Total time of service activities in S^{all}	0: 0; 1: ≤20; 2: 21-40; 3: 41-60; 4: > 60
Tnd-shop	Total time of non-daily shopping activities in S^{all}	0: 0; 1: ≤30; 2: 31-60; 3: 61-120; 4: > 120
Tmax(t)	Maximum available time in t-th time interval in S^{all} (possible duration for A)	0: < minimum; 1: minimum-average; 2: average-maximum; 3: > maximum
Btwo(t)	There is a Work1 activity with start time in t=1, ..., 3 in S	0: no; 1: yes
Etx(t)	There is an out-of-home activity with end time in t=1, ..., 6 in S	0: no; 1: yes
DBT(t)	Saved bike travel time if A is linked with out-of-home activity with start time in t=1, ..., 3	0: 0 or no such activity; 1: ≤10; 2: 11-30; 3: > 30
DET(t)	Saved bike travel time if A is linked with out-of-home activity with end time in t=1, ..., 6	0: 0 or no such activity; 1: ≤10; 2: 11-30; 3: > 30
yCar(t)	Availability of car in t-th time interval in S	0: no; 1: yes; 2: schedule partner is unknown

Atype	Activity type of A	1: daily shopping; 2: service;
		3: non-daily shopping;
		4: social; 5: leisure

Awith	Travel party of A	0: none; 1: only others inside household; 2: others outside household involved
Iact	Number of the current activity type of A	1: 1; 2: > 1
Adur	Duration of A	1: short; 2: average; 3: long
Ad1	Shortest bike travel time across possible locations for A (minutes)	0: 0; 1: ≤10; 2: 11-30; 3: > 30

often fall within the same time period and only one value can serve as an indicator for both.

Second, the DBT(*t*) and DET(*t*) variables more specifically indicate the travelled distance that could be saved by establishing a travel connection. Hereby, DBT refers to the work activity with start time falling into time period *t*, if any, DET, relates to the out-of-home activity of any type with the end time falling in the *t*-th interval, if any. As in previous models, bike travel time is taken as indicator of distance. Let O denote the existing out-of-home activity, A the activity for which the start time choice is made and H the home location, then the saved time is determined by comparing the sum of travel time across H-O-H and H-A-H tours with the travel time of H-O-A-H or H-A-O-H trip. In all trip types, the location

Table B.5: Travel time/duration ratios uses to estimate travel times based on activity duration

Activity	Ratio
Daily shopping	0.33
Service	0.65
Non-daily shopping	0.28
Social	0.14
Leisure	0.14
Unknown	0.30

that minimises travel time across location alternatives for A is taken as the location for A.

The final set of schedule-level variables is given by yCar(*t*). As explained before, this variable represents the availability of the car in the *t*-th time period, given the number of cars present per adult member of the household and the mode used for the work activity in the partner's schedule (if any). Finally, the remaining variables all relate to dimensions of the activity for which the start-time decision is currently made. These are restricted to the dimensions considered known at this stage, i.e. the activity type, travel party, duration and shortest home-based distance.

B.4 Trip Chaining

Table B.6 shows the independent variables for the “Trip Chaining” choice facet. The notation of symbols and the description of some variables is similar to previous decision facets. In this section, we only consider variables that are specific for trip-chaining step.

First, the yAstop, yBstop and yIBstop variables denote the feasibility of trip-chaining options. The feasibility of trip-chaining is determined by spatial, temporal and institutional constraints. The next set of variables (y-variables), describes the concerned flexible activity regarding the dimensions that are considered known in this step. The Awith, Adur and Astart variables describe the travel-party, duration and start-time dimensions in terms of the choice alternatives of the choice facets in previous steps. Finally, Ad1 measures the shortest distance from the home location across the possible locations for the activity. Note that in the case of social activities every zone in the area is by definition zero. For the other activities, the shortest distance depends on locations of available facilities.

The final set of variables describe the (uniquely) identified feasible activities, if any, for making a before or an after connection respectively. Note that if A can be positioned before as well as after an certain activity, the variables refer to the same activity. First, the Otime variables represent the available time for completing A in that position. In fact, the maximum available time is calculated if there is flexibility in determining the start time and duration for existing activities in the current schedule. The Otype, Owith and Odu describe the activity type, the travel party and duration of the activity, again, in terms of the same categories that are used throughout the model. Finally, the next variables describe the spatial context in terms of the (shortest) distances to O from home, the distance between A and O and the saved travel distance when the connection would be made (H-A-O-H or H-O-A-H) compared to the single stop option (H-A-H).

Table B.6: Explanatory variables in the "Trip Chaining" choice facet (Albatross)

Variable Name	**Description**	**Categories**
Nsec	Number of mandatory out-of-home activities other than work in S^{all}	0: 0; 1: 1; 2: 2; 3: 3-4; 4: > 4
Two	Total time of Work1 in S^{all} (in minutes)	0: 0; 1: ≤240; 2: 241-360; 3: 361-480; 4: > 480
Twincl	Total time of Work1 incl. travel in S^{all}	0: 0; 1: ≤260; 2: 261-380; 3: 381-500; 4: > 500
Ttot	Total time of Work1 and Work2 in S^{all}	0: 0; 1: ≤60; 2: 61-120; 3: 121-240; 4: > 240
yBget	There is a bring/get activity in S^{all}	0: no; 1: yes
yDshop	There is a daily shopping activity in S^{all}	0: no; 1: yes
yServ	There is a service activity in S^{all}	0: no; 1: yes
yNDshop	There is a non-daily shopping activity in S^{all}	0: no; 1: yes
ySoc	There is an out-of-home social activity in S^{all}	0: no; 1: yes
yLeis	There is an out-of-home leisure activity in S^{all}	0: no; 1: yes
Tsoc	Total time of social activities (in-home and out-of-home) in S^{all}	0: 0; 1: ≤30; 2: 31-60; 3: 61-120; 4: 121-240; 5: > 240

Tleis	Total time of out-of-home leisure activities in S^{all}	0: 0; 1: ≤30; 2: 31-60; 3: 61-120; 4: 121-240; 5: > 240
Td-shop	Total time of daily shopping activities in S^{all}	0: 0; 1: ≤20; 2: 21-40; 3: 41-60; 4: > 60
Tserv	Total time of service activities in S^{all}	0: 0; 1: ≤20; 2: 21-40; 3: 41-60; 4: > 60
Tnd-shop	Total time of non-daily shopping activities in S^{all}	0: 0; 1: ≤30; 2: 31-60; 3: 61-120; 4: > 120
yCar	There is a car available in the selected time-of-day, given work activity of partner	0: no; 1: yes; 2: schedule partner is unknown
yBstop	Feasibility of a Before Stop, given space-time constraints	0: no; 1: yes
yAstop	Feasibility of an After Stop, given space-time constraints	0: no; 1: yes
yIBstop	Feasibility of a Between Stop, given space-time constraints	0: no; 1: yes
Atype	Activity type of A	1: daily shopping; 2: service; 3: non-daily shopping; 4: social; 5: leisure
yAd-shop	A is a grocery activity	0: no; 1: yes
yAserv	A is a service activity	0: no; 1: yes
yAnd-shop	A is a non-grocery activity	0: no; 1: yes
yAsoc	A is a social activity	0: no; 1: yes
yAleis	A is a leisure activity	0: no; 1: yes
Awith	Travel party of A	0: none; 1: only others inside household; 2: others outside household involved
Adur	Duration of A	1: short; 2: average; 3: long
Astart	Start time of A	1: < 10 AM; 2: 10-0 AM; 3: 0-2 PM; 4: 2-4 PM; 5: 4-6 PM; 6: > 6 PM
Ad1	Shortest bike travel time across possible locations for A (minutes)	0: 0; 1: ≤10; 2: 11-30; 3: > 30
Ontime	Available time for A before (On) or after	0: <minimum; 1: minimum - average;
Optime	O (Op), given the timing of fixed activities	2: average - maximum; 3: > maximum
On-/Optype	Activity type of O	1: bring/get; 2: work1; 3: other
Onwith	Travel party of O	0: none; 1: only others

Opwith		inside the household; 2: others outside the household involved
On-/Opdu	Duration of O	1: ≤10; 2: 11-40; 3: 41-120; 3: > 120
Ondu1	Bike travel time to (nearest) location	0: 0; 1: ≤10; 2: 11-20; 3:> 20
Opd1	of O from home	
Ond3	Shortest bike travel time between location	0: 0; 1: ≤15; 2: 16-30; 3:> 30
Opd3	of O and possible locations for A	
On-/Opd13	Saved bike travel time of A is linked with O	0: 0; 1: ≤10; 2: 11-30; 3:> 30

B.5 Activity Transport Mode

Table B.7 shows the independent variables for the "Transport mode for other than work activities" (Mode Other) choice facet. The notation of symbols and the description of some variables is similar to previous decision facets. The symbol *C* represents the concerned tour.

The tour-level variables cover various dimensions. First, the time-of-day when the tour is undertaken is potentially relevant as it may determine the degree of congestion on the road network during travelling. However, at this stage the start time of the tour is not exactly known. The exact departure time will dependent upon the mode used for the tour. For example, a fast mode allows one to delay the departure time, while keeping the time engaged in the activities itself constant. Moreover, the start time and duration of flexible activities are flexible. To account for the freedom of choice on all these dimensions, a variable that determines the earliest possible start time of the tour was included.

A second potentially important aspect is the tour's purpose. The dimensions of the first activity in the tour, such as the activity type, travel party and duration, are included as descriptors for the tour's purpose.

Third, the required travel distance on the tour is a potential moderator of mode choice. We use the shortest-route bike time as an indicator of distance. Because locations of flexible activities are still unknown in this stage, the shortest-travel time across possible locations for the activity is taken as an index here. If the tour involves more than one flexible activity, the shortest travel time is calculated based on home-based distances. This was done to avoid the computational complexity of optimising the choice of multiple locations simultaneously. Additionally, the tour-specific relative speed of each mode is measured in terms of travel-time ratios between car/bike, public transport/bike and public transport/car.

Mode choice may further depend on the required travel distance to reach locations of higher order. Fast modes probably reduce the disutility of travel and therefore may be preferred in cases where the individual wishes to visit a higher-location at a relative long marginal distance. The fourth set of tour-level variables, therefore, defines the extra bike-travel time required to reach locations for each higher-order location. The bike times calculated relate to the first activity only and assume a home-based trip. In case of fixed activities (no location choice) and social activities (no higher-order locations), the marginal distances are set equal to zero. Note that these variables describe location choice options and, consequently, allow the system to anticipate on location choices in choosing a mode.

Furthermore, the activity schedule of the partner, if any, may compete with the use of car in households where there is only one car available. The fifth set of tour-level variables describe the presence of a bring/get activity, the presence of a shopping or service activity and the maximum bike-travel time across the partner's tour that necessarily overlap in time with the tour concerned. Overlapping tours are identified by comparing latest possible start times and earliest possible end times. If there is no partner, the partner's schedule is unknown or there are no overlapping tours, the variables are set to zero.

The final independent variable defines the availability of the car-driver mode. Three categories can be distinguished: there is no car available (Avcar = 0), there is a car available (Avcar = 1) and there is no partner or the schedule of the partner is unknown (Avcar =2). The other mode alternatives - car passenger, public transport and slow mode - are considered always available.

Table B.7: Explanatory variables in the "Transport Mode for other than work activities" choice facet (Albatross)

Variable Name	Description	Categories
Nsec	Number of mandatory out-of-home activities other than work in S	0: 0; 1: 1; 2: 2; 3: 3-4; 4: > 4
Two	Total time of Work1 in S (in minutes)	0: 0; 1: ≤240; 2: 241-360; 3: 361-480; 4: > 480
Ttot	Total time of Work1 and Work2 in S	1: ≤120; 2: 121-240; 3: 241-360; 4: 361-480; 5: > 480
CBT	Earliest possible begin time of C	1: < 10 AM; 2: 10-0 AM; 3: 0-2 PM; 4: 2-4 PM; 5: 4-6 PM; 6: > 6 PM
Aty1	Type of the first activity in C	1: work; 2: bring/get; 3: grocery; 4: service; 5: non-grocery; 6: leisure; 7: social; 8: other
Aty2	Type of the second activity in C	0: home; 1: work; 2: bring/get; 3: (non-) grocery or service; 4: leisure or social;5:other
Adur1	Duration of the first activity in C	1: short; 2: average; 3: long
Awith1	Travel party of the first activity in C	0: none; 1: only others inside the household; 2: others outside the household involved
Cbrget	Bring or get activity is part of C	0: no; 1: yes
Cgroc	Grocery activity is part of C	0: no; 1: yes
Cserv	Service activity is part of C	0: no; 1: yes
Cshop	Non-daily shopping activity is part of C	0: no; 1: yes
Csoco	Social activity is part of C	0: no; 1: yes
Cleiso	Leisure activity is part of C	0: no; 1: yes
Cnlout	Non-leisure activity is part of C	0: no; 1: yes
TTbike	Shortest travel time by bike for tour C(in minutes)	0: 0; 1: ≤5; 2: 6-15; 3: 16-25; 4: 26-35; 5: 36-60; 6: > 60
Rcabi	Travel time ratio between car and bike (%)	1: ≤25, 2: 26-33; 3: 34-85; 4: > 85
Rpubi	Travel time ratio between public transport and bike (%)	1: ≤100; 2: 101-200; 3: 201-260; 4: > 260
Rpuca	Travel time ratio between public transport and car (%)	1: ≤100; 2: 101-700; 3: 701-900; 4: > 900
Textra2	Extra bike travel time to reach	0: 0; 1: ≤10; 2: 11-15;

	location of order 2 (minutes)	3: 16-30; 4: > 30
Textra3	Same for order 3	0: 0; 1: 1-15; 2: 16-20; 3: 21-35; 4: > 35
Textra4	Same for order 4	0: 0; 1: 1-20; 2: 21-30; 3: 31-40; 4: > 40
Pbrget	Partner has a bring/get activity during tour C	0: no; 1: yes
Pserv	Partner has a shopping or service during tour C	0: no; 1: yes
PTmax	Partner's maximum bike travel time across activities during tour C (minutes)	0: 0; 1: 1-10; 2: 11-20; 3: 21-40; 4: > 40
Avcar	Car is available given the work activity of the partner	0: no; 1: yes; 2: there is no partner or schedule partner is unknown

B.6 Locations

The independent variables for the “Location” choice facets have been shown in Table B.8. Most of the variables in Table B.8 are analogous to variables that have been discussed in previous choice facets.

The only difference is a set of variables, AvCmin - AvCmax, which has a special status. They determine, e.g., that Cmin is not available in cases where the nearest home location (Hmin) is identical with the nearest tour-based location (Cmin).

For the location2 choice facet, we consider only the “other” locations from the previous choice facet (i.e. location1). This facet now selects a travel-time band comprising the location where the activity is to be performed. Travel times are evaluated exclusively in the context of the concerned tour (as opposed to home-based).

Table B.8: Explanatory variables in the “Location1” and “Location2” choice facets (Albatross)

Variable Name	**Description**	**Categories**
Twincl	Total time of Work1 inclusive travel in S (in minutes)	0: 0; 1: ≤260; 2: 261-380; 3: 381-500; 4: > 500
Ttot	Total time of Work1 and Work2 in S	1: ≤240; 2: 241-360; 3: 361-480; 4: > 480
Nsec	Number of mandatory out-of-home activities other than work in S	0: 0; 1: 1; 2: 2; 3: 3-4; 4: 4-5; 5: > 5

Atype	Activity type	1: daily shopping; 2: service; 3: non-daily shopping; 4: social; 5: leisure
Mode	Transport mode	1: car (driver or passenger); 2: slow; 3: public
Adur	Activity duration	1: short; 2: average; 3: long
Awith	Travel party of A	0: none; 1: only others inside the household; 2: others outside the household involved
Tiday	Start time of A	1: < 10 AM; 2: 10-0 AM; 3: 0-2 PM; 4: 2-4 PM; 5: 4-6 PM; 6: > 6 PM
Tmax	Maximum available time in the schedule position of the activity (inclusive travel times)	0: 0; 1: 1-30; 2: 31-60; 3:> 60
Nout	Number of out-of-home activities in C	1: 1; 2: 2; 3: > 2
fromH	Trip to A starts from home	0: no; 1: yes
toH	Trip from A ends at home	0: no; 1: yes
Aprev	Type of previous activity	0: home; 1: work; 2: other mandatory; 3: social or leisure
Anext	Type of next activity	0: home; 1: work; 2: other mandatory; 3: social or leisure
AvCmin	Cmin location is available in choice set	0: no; 1: yes
AvCext5	Cext5 location is available in choice set	0: no; 1: yes
AvCext10	Cext10 location is available in choice set	0: no; 1: yes
AvCext20	Cext20 location is available in choice set	0: no; 1: yes
AvCmax	Cmax location is available in choice set	0: no; 1: yes

APPENDIX C

In this appendix, the adapted CBA algorithm is evaluated on 16 popular UCI datasets. In order to get a more comprehensive evaluation, these datasets are also classified by original CBA, the classical decision tree technique C4.5 (both on original and discretized datasets) and Naïve Bayes. The continuous attributes are discretized by means of an entropy-based discretization method if needed. Ten-fold cross validation is used to test the performance of these classifiers. The benchmarking results are described in Table C.1.

Adapted CBA-1 and CBA-2, which respectively correspond to the new algorithms that incorporate intensity of implication and dilated χ^2, perform better than any of the other classifiers in comparison to the average error rate. The average error rate of adapted CBA-1 on these 16 datasets is 13.21%, and that of adapted CBA-2 is only 12.81% if the best parameter is selected for each dataset. Furthermore, adapted CBA-1 generates 17.925 rules in average, which is almost one third of those rules generated by original CBA. The classifiers built by adapted CBA-2 are more compact and averagely contain 11.82 rules. The original CBA also has a better performance than C4.5 and Naïve Bayes. Although Naïve Bayes performs excellent on several datasets such as breast, heart and labor, its behaviour is unstable since it assumes attributes are independent, which is a very fragile assumption in real life datasets. The performance of C4.5 on discretized datasets is better than on original datasets, so only the former result is presented.

Although it is difficult to compare two classifiers based on datasets from different domains, Wilcoxon signed-rank test is applied because the number of datasets under evaluation is sufficiently large to give a rough statistical comparison.

Table C.1: Benchmark experiments on 16 UCI Datasets

Dataset		**Original CBA**		**Adapted CBA-1**		**Adapted CBA-2**		**C45**	**NB**
		error rate(%)	no. of rules	error rate(%)	num. of rules	error rate(%)	no. of rules	error rate(%)	error rate(%)
1	austra	14.35	130.5	13.48	26.4	13.04	12.4	13.48	18.70
2	breast	3.86	42.2	4.72	28.4	3.58	28.3	4.43	2.58
3	cleve	17.16	63.8	15.47	16.9	16.13	9.6	20.79	16.17
4	crx	14.93	138.2	12.90	34.2	13.04	12.4	12.75	18.99
5	diabetes	22.26	38.5	24.21	10.4	21.74	10.7	22.92	24.22
6	german	26.70	134	25.60	56.5	26.80	19.7	27.60	25.30
7	heart	17.78	37.6	16.30	13.6	16.67	7.4	18.89	14.81
8	hepati	16.21	25.2	18.67	18.4	16.83	11.3	16.77	15.48
9	horse	19.03	87.9	14.12	1	14.12	1	15.22	20.92
10	hypo	1.64	30	1.23	24.4	0.85	10.9	0.85	1.90
11	iono	8.25	44.8	9.10	21.7	6.55	18.5	9.69	8.26
12	labor	10.00	12.5	11.67	4.2	8.33	4.4	15.79	8.77
13	pima	23.43	38.3	23.17	11	22.00	10.7	22.66	25
14	sick	2.64	47.4	2.43	10.7	3.25	1	2.07	4.32
15	sonar	22.60	41	18.31	27.4	18.74	21.8	18.75	25.48
16	ti-tac	0.00	8	0.00	8	3.34	9	14.20	29.65
average		13.80	49.34	13.21	17.925	12.81	11.82	14.80	16.28

As shown in Table C.2, significant improvement is achieved by adapted CBA-2 at a 5% confidence interval. Although the performance of adapted CBA-1 could not generate the same good result as CBA-2, it performs best in several cases and requires no parameter selection. Based on these experiments, it is safe to conclude that intensity of implication and dilated χ^2 are both appropriate measures for associative classification.

Table C.2: Performance comparison

p-values for one tail test	**Original CBA**	**C4.5**	**Naïve Bayes**
Adapted CBA-1	0.1652	0.0844	0.1057
Adapted CBA-2	0.0107	0.0035	0.0125

APPENDIX D

It was mentioned in Chapter 3, section 3.5.2 and 3.6.1 that Intensity of implication followed the hypergeometric law. In the remainder of that section, a Poisson approximation was used that has been advanced by Suzuki and Kodratoff (1998), which significantly reduced the computational effort. The results of the original hypergeometric law formula have been added in this appendix in Table D.1 for the sake of completeness. They were not reported in Chapter 3 because of the high computational burden and because we did not want to overload the chapter. The results are similar and even somewhat better at an average scale than CBA-1 with respect to the performance on the test set, but the size of the ruleset is also somewhat higher.

Table D.1: The performance of the Adapted CBA-1 algorithm, using the hypergeometrical law formula

Dataset	Adapted CBA-1, hypergeometical law formula		
	Train (%)	Test (%)	Num. of rules
Duration	41.2	40.4	13
Location1	63.4	65.9	67
Location2	42.4	40.3	19
Mode for work	74.8	75.4	28
Mode other	60.4	55.9	94
Selection	79.1	79.2	1
Start time	34.3	33.8	109
Trip chain	82.8	81.8	27
With whom	59.0	54.0	92
Average	**59.7**	**58.5**	**50**

APPENDIX E

The dilated chi-square measure that has been used in the CBA-2 algorithm has been heuristically derived in section 3.5.3 as:

$dia\left(\chi^2\right) = \left(\frac{|D|}{\text{lmax}\left(\chi^2\right)}\right)^{\alpha} \chi^2$. The formula uses the parameter α and depending on the value that is used for α, a different accuracy could be obtained. It was mentioned in section 3.6.1 that only the best accuracy result has been reported in Table 3.7. For the sake of completeness, Figure E.1 reports the results of a sensitivity analysis that has been carried out for different values of the α-parameter. It can be seen from this figure that while there is some variation in predictive accuracy, it remains relatively low for all the datasets under consideration.

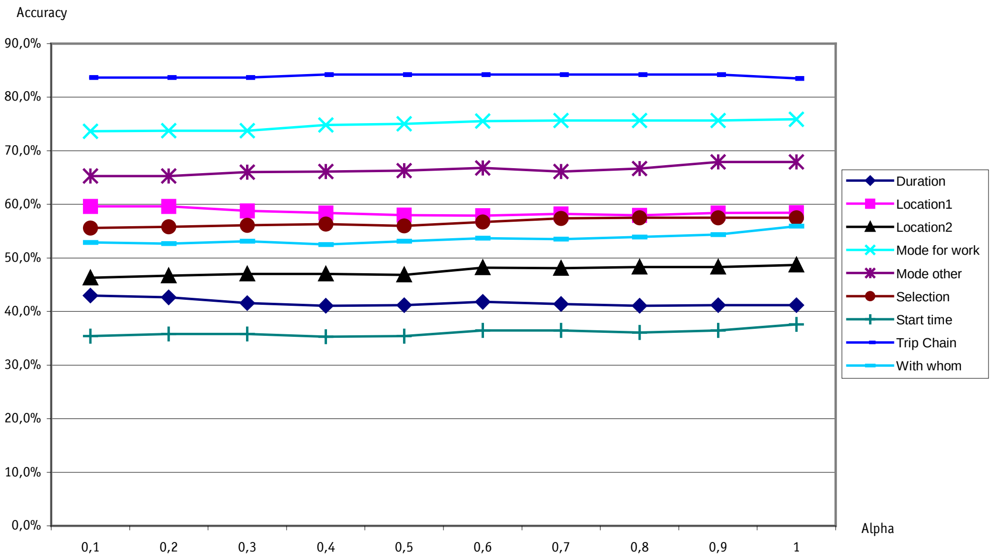


Figure E.1: Sensitivity analysis in terms of predictive accuracy for different values of α (CBA-2).

NEDERLANDSE SAMENVATTING

De nood aan wetenschappelijk onderzoek op het vlak van transport komt wellicht het meest tot zijn recht door het belang dat aan de sector wordt gehecht door verschillende supranationale instellingen zoals de Verenigde Naties en het Internationale Energie Agentschap. De sector is dan ook niet enkel een grote energieverbruiker maar is tevens verantwoordelijk voor een aantal neveneffecten op het economische en sociale vlak, alsook op het vlak van milieu. De toenemende ongerustheid over hoe deze steeds groter wordende neveneffecten kunnen worden aangepakt in een context van mondialisering en globalisering, maakt dat beleidsmaatregelen die door transportplanners worden uitgestippeld, onder een groter wordende druk komen te staan om aan een aantal van deze problemen het hoofd te kunnen bieden.

Vooral in de Verenigde Staten, maar recent ook in Europa, is er een duidelijke trend merkbaar waarbij beleidsmaatregelen, al dan niet in een juridisch kader, worden onderbouwd met behulp van ondersteunende transportmodellen. Deze transportmodellen zijn in staat om het effect van een nog te implementeren beleidsmaatregel door te rekenen en kunnen op deze manier ondersteuning bieden bij het beslissings- en uitvoeringsproces dat door beleidsinstanties wordt uitgestippeld. Uiteraard is de modelspecificatie en –calibratie in deze essentieel om te komen tot een betrouwbare voorspelling en ondersteuning.

Het wetenschappelijk onderzoeksdomein dat zich bezig houdt met transportmodellering heeft de laatste decennia reeds een belangrijke metamorfose doorgemaakt. Waarschijnlijk de meest ingrijpende verandering is de introductie en de doorbraak geweest van activiteitengebaseerde transportmodellen in het domein. Deze modellen beschouwen de vraag naar transport als afgeleide vraag, i.e. een vraag die bepaald wordt door de activiteiten die gezinnen en personen wensen uit te voeren of waar zij behoefte aan hebben. Het modelleren van transport wordt dus bekeken in een ruimer globaal kader, waarbij verplaatsingen en transport worden beschouwd om activiteiten en bijgevolg dus doelstellingen en noden te kunnen realiseren. Het ruimere kader komt het best tot uiting in de verschillende facetten (welke activiteiten, waar, wanneer, op welke locatie, met wie, voor hoe lang en met welk vervoermiddel) die door activiteitengebaseerde transportmodellen worden gemodelleerd. Het hoeft geen betoog dat tengevolge van deze ruimere context, de modelspecificatie en –calibratie ook een toenemende mate van complexiteit hebben gekend.

Transportmodellen zijn ruwweg in te delen in "scheduling modellen" enerzijds en in simulatiemodellen anderzijds. Beide type modellen hebben hetzelfde doel, i.e. de best mogelijke predictie van activiteiten- en reispatronen (dikwijls gekwantificeerd in de vorm van herkomst- en bestemmingsmatrices of door andere afgeleide transportgerelateerde variabelen), maar gebruiken hiervoor een verschillende insteek. Scheduling modellen gebruiken concepten die van oorsprong afstammen uit de geografie, micro-economie en psychologie en gebruiken een aantal voorgedefinieerde regels en beperkingen om het plannings(schedulings)proces van personen en huishoudens zo goed mogelijk te capteren. Ze omvatten echter ook een aantal leeralgoritmes die het mogelijk maken om kennis die bevat is in de data (dikwijls onder de vorm van dagboekjes) te extraheren en te gebruiken voor hun voorspelling. Met deze laatste eigenschap is de overstap naar simulatiemodellen vlug gemaakt. Globaal gesproken hebben simulatiemodellen immers niet tot doel om het planningsproces volledig te capteren, maar ze zullen de verschillende facetten (zie supra) van activiteitengebaseerde modellen simuleren, dikwijls louter op basis van patronen en structuren die vervat zitten in de data. Echter, ondanks deze verschillende insteek komen beide modellen wel tot dezelfde finale doelstelling.

Scheduling- en simulatiemodellen vormen het uitgangspunt en achtergrond van dit proefschrift. Zoals hierboven reeds vermeld, ligt de mate van overlap tussen beide modellen vooral in de extractie van kennis dewelke vervat zit in de data, alsook in de outputindicatoren van beide type van modellen. Het doel van dit proefschrift is dan ook duaal. Enerzijds wordt in het proefschrift het gebruik van alternatieve leeralgoritmes onderzocht die kunnen worden aangewend om kennis uit data te extraheren in beide types van modellen. Anderzijds worden de finale outputindicatoren van beide types van modellen met elkaar vergeleken wanneer de volledige modelcalibratie is voltooid.

Om aan beide doelstellingen tegemoet te kunnen komen is het onderzoeksdomein van "Unsupervised Machine Learning" als vertrekpunt genomen. Er zijn een aantal goede redenen aan te halen voor deze keuze. Machine Learning is een multidisciplinair onderzoeksveld dat een veelvoud aan inductie-algoritmes aanreikt om kennis uit data te extraheren. Het domein is reeds door verscheidene onderzoekers in verschillende onderzoeksdisciplines aangewend, maar is tot op heden slechts vrij beperkt in het domein van activiteitengebaseerde modellen gebruikt. De zogenoemde vorm van "Supervised" leren begint langzamerhand opgang te kennen, maar toepassingen van "Unsupervised" leren zijn tot op heden eerder uitzonderingen. In tegenstelling tot "Supervised" leren, heeft "Unsupervised" leren niet tot doel om één specifieke doelfunctie af te schatten om tot een voorspelling te komen van één bepaalde te verklaren variabele, maar is het in staat om een set van associaties en verbanden die in de data vervat zitten te identificeren, zonder één bepaald vooraf vooropgesteld doel (verklarende variabele). Omwille van het feit dat unsupervised leren nog in haar kinderschoenen staat voor toepassingen binnen activiteitengebaseerde transportmodellen, is een aanpassing van methodes zoals ze tot op heden bestaan en worden toegepast in andere domeinen, een absolute noodzaak. De aanpassingen en fundamentele bijdragen van het proefschrift starten vanaf hoofdstuk 3, en zullen in het vervolg van deze samenvatting verder worden toegelicht.

Vooreerst start het proefschrift in hoofdstuk 1 met een overzicht van het hierboven geschetste conceptuele kader. Dit kader wordt o.a. uitvoerig aangevuld door middel van een gedetailleerd literatuuroverzicht en met een situering van de belangrijkste wetenschappelijke bijdragen van het proefschrift. Hierbij wordt enerzijds de nadruk gelegd op de aanpassingen aan bestaande algoritmes binnen een context van Unsupervised Machine Learning en anderzijds wordt het belang van een vergelijking van de hogervernoemde types van modellen verder aangetoond.

Vervolgens wordt in een tweede hoofdstuk, het Albatross ("A Learning-Based Transportation Oriented Simulation System") model geïntroduceerd. Albatross is een typisch voorbeeld van een "scheduling model". Albatross is het eerste operationele model en is uitgegroeid tot één van de meest belangrijke modellen in de literatuur. Om het transportgedrag van personen te modelleren, gebruikt Albatross een set van beslissingsregels dewelke idealiter uit data worden geëxtraheerd. In het oorspronkelijk model, dat ontwikkeld werd door Arentze en Timmermans (2000) wordt hiervoor een standaard algoritme gebruikt (CHAID), wat gebaseerd is op beslissingsbomen. Het algoritme heeft vooral grote bekendheid verworven vanuit een statistische hoek. Wanneer dit echter in een ruimer kader wordt bekeken, zijn er sinds de ontwikkeling van CHAID vele andere alternatieven beschikbaar die voornamelijk uit het domein van Machine Learning zijn ontstaan. De volgende 2 hoofdstukken doen in een eerste fase een gedetailleerd voorstel over hoe deze technieken kunnen worden aangewend binnen Albatross en stellen in een tweede fase meerdere aanpassingen voor aan deze bestaande algoritmes. Net als de oorspronkelijke algoritmes werden de voorgestelde aanpassingen allen uitvoerig geëvalueerd door middel van verschillende kwantitatieve outputindicatoren. Omwille van het feit dat de algoritmes die binnen Albatross worden gebruikt eigenlijk vooral ten behoeve van Supervised Learning zijn, diende in deze opzet naar een integratie te worden gestreefd met de voorgestelde Unsupervised Learning methodieken.

Het algoritme dat in een derde hoofdstuk wordt voorgesteld, is het standaard CBA-algoritme (Classification Based on Associations), en is een goed voorbeeld over hoe Unsupervised Learning en Supervised Learning kunnen worden geïntegreerd. Het algoritme geniet uiteraard minder bekendheid dan CHAID of andere meer traditionele inductie-algoritmes, maar heeft in het verleden tot een aantal goede resultaten kunnen leiden in andere domeinen en vormde hierdoor een uitdagend vertrekpunt voor onderzoek binnen een activiteitengebaseerd model. De resultaten van het onderzoek werden op 3 niveaus geëvalueerd. Vooral op het eerste niveau (choice facet level) waren de resultaten erg bevredigend, doch op de 2 andere niveaus (pattern level en trip level) kon een zekere mate van "overfitting" van het model worden vastgesteld. Omwille van deze reden werden 2 additionele aanpassingen aan het oorspronkelijke CBA-algoritme geïmplementeerd en verder geëvalueerd. De aanpassingen situeerden zich vooral op het vlak van het gebruik van een complexere sorteermaatstaf, waarbij enerzijds gebruik werd gemaakt van Intensity of Implication, een bestaande sorteermaatstaf en anderzijds op basis van Dilated Chi-square, een eigen geïmplementeerde heuristiek. De resultaten van beide algoritmes waren matig bevredigend, vooral de mate van overfitting op pattern en trip level kon hierdoor significant worden teruggedrongen en een erg compacte "classifier" (i.e. een belangrijke reductie in het aantal beslissingsregels) kon worden bekomen. De belangrijkste doelstelling kon hierdoor worden bereikt, doch was het algoritme ook onderhevig aan een kleine terugval op choice facet level. Het hoofdstuk bespreekt naast deze gedetailleerde kwantitatieve benchmark ook een uitvoerige kwalitatieve vergelijking van de verschillende geteste algoritmes.

De onderzoeksopzet in hoofdstuk 4 is vrij analoog. Het uitgangspunt in dit hoofdstuk is de analyse en het gebruik van Bayesiaanse netwerken. Bayesiaanse netwerken zijn eveneens technieken die een duidelijke oorsprong vinden in Machine Learning en zijn tevens eerder unsupervised en beschrijvend van nature. Het beschrijvende karakter van de techniek is erg krachtig, vooral omwille van het feit dat verschillende relaties en complexe conditionele afhankelijkheden kunnen worden gekwantificeerd en tevens gevisualiseerd in een netwerkstructuur. In dit hoofdstuk wordt een methodiek uitgewerkt over hoe de techniek kan worden aangewend voor classificatie binnen het Albatross model. De resultaten werden eveneens op de 3 hoger vermelde niveaus geëvalueerd en zijn vrij gelijkaardig aan de resultaten van hoofdstuk 3. Eveneens werd vervolgens een aanpassing aan de bestaande methodiek voorgesteld door de ontwikkeling van een nieuwe heuristiek dewelke de integratie van Bayesiaanse netwerken en beslissingsbomen nastreeft. De heuristiek gebruikt de informatie die vervat zit in de netwerkstructuur om te komen tot de ontwikkeling van een beslissingsboom, een benadering die niet eerder in andere onderzoeken op dezelfde manier werd geëvalueerd. Door deze toepassing worden conditionele afhankelijkheden en verbanden niet meer onmiddellijk uit de data geëxtraheerd maar dient het Bayesiaanse netwerk als het ware als een filter om een zekere mate van redundantie op te vangen. De nieuwe heuristiek werd tevens geëvalueerd binnen Albatross op de 3 benchmarking-niveaus, met erg bevredigende resultaten tot gevolg. Analoog aan hoofdstuk 3, sluit het hoofdstuk af met een kwalitatieve analyse van de geteste algoritmes.

Terwijl de hoofdstukken 3 en 4 een evaluatie van Unsupervised Machine Learning algoritmes beoogden binnen een "scheduling model", wordt in hoofdstuk 5 gestart met de toepassing en ontwikkeling van een aantal technieken binnen een simulatiecontext. In tegenstelling tot hoofdstukken 3 en 4 wordt geen bestaand model als vertrekbasis genomen, maar is het doel eerder de eigen ontwikkeling van een model op basis van een aantal inzichten die het domein van Unsupervised Learning ons kunnen bieden. Zo worden in hoofdstuk 5 Markov ketens gebruikt omdat zij het mogelijk maken om sequentiële patronen dewelke vervat zitten in data te capteren en aan te wenden voor predictie. De expliciete identificatie van sequentiële informatie uit data is erg beperkt binnen het onderzoeksdomein. Omwille van deze reden is er een belangrijke bijdrage en vormt het onderzoek een goede startbasis voor de ontwikkeling van een innovatief simulatiemodel. Binnen hoofdstuk 5 worden diverse aanpassingen voorgesteld om de sequentiële informatie beter te kunnen capteren. De belangrijkste aanpassingen en bijdragen tot de literatuur kunnen in 3 grote delen worden samengevat. Ten eerste worden 2 alternatieve methodes voorgesteld om te komen tot een betere en efficiëntere berekening van transitie-probabiliteiten zoals ze in traditionele Markov Chains worden gebruikt. De voorgestelde methodes maken het mogelijk om de verschillende sequenties onafhankelijk van elkaar te beschouwen en zijn daarom beter geschikt voor de identificatie van sequentiële patronen op persoonsniveau. Bovendien gebeurt de calculatie efficiënter door de voorgestelde benaderingen waardoor de identificatie van sequentiële informatie niet enkel beperkt blijft tot eerste-orde effecten maar tevens kunnen hierdoor hogere-orde effecten efficiënt worden opgespoord. Ten tweede wordt een methodiek voorgesteld die het mogelijk maakt om de geïdentificeerde informatie te clusteren op basis van socio-demografische of tijdsgerelateerde kenmerken waardoor een accurate en meer gedetailleerde benadering van diverse clusters mogelijk wordt. Om dit te bereiken werd o.a. een aanpassing voorgesteld en geïmplementeerd op basis van een beslissingsboom, met deze belangrijke verschilpunten dat enerzijds sequentiële informatie kan worden opgenomen in de boomstructuur en anderzijds deze sequentiële informatie het resultaat is van een inductie-algoritme en dus niet rechtstreeks uit de data wordt geobserveerd zoals het geval is bij traditionele beslissingsbomen. Ten derde wordt de geïdentificeerde sequentiële informatie aangewend voor de predictie van sequenties van activiteiten en transportmodes in een aparte heuristisch opgebouwde module. De predictieresultaten werden op basis van gemiddeld aantal tours en trip rates geëvalueerd.

De voorspelde sequenties van activiteiten en transportmodes uit hoofdstuk 5 worden in hoofdstuk 6 aangewend om te komen tot een allocatie van tijds- en locatie-informatie. Hiervoor wordt de techniek van Reinforcement Learning als uitgangspunt genomen. Reinforcement Learning is een vrij innovatieve techniek die in eerder technische toepassingsdomeinen zoals bvb. robotica, of lift dispatching wordt gebruikt. De techniek leert in belangrijke mate door interactie met de omgeving en is gebaseerd op het “trial en error” leerprincipe. In hoofdstuk 6 wordt een gedetailleerde aanzet gegeven tot de manier waarop deze techniek kan worden gebruikt binnen een simulatiemodel om locatie- en tijdsinformatie toe te wijzen aan een bestaande sequentie van activiteiten en transportmodes. Ook in dit hoofdstuk wordt de reeds bestaande literatuur uitgebreid door enerzijds een meer geavanceerde toepassing voor wat betreft de allocatie van tijdsinformatie op basis van een bestaand algoritme (Q-learning), en anderzijds door een nieuwe ontwikkeling en aanwending van de techniek voor wat betreft de allocatie van locatie-informatie. Ook wordt een nieuw geïntegreerd framework voorgesteld waardoor tijds- en locatie-informatie voortaan simultaan kunnen worden gealloceerd.

Op deze manier leidt de synergie van hoofdstukken 5 en 6 tot een modulair simulatiemodel dat kan worden aangewend voor een evaluatie met de bestaande schedulingsmodellen uit hoofdstukken 3 en 4. Hoofdstuk 7 brengt op deze manier de verschillende wetenschappelijke bijdragen van het proefschrift samen en rapporteert de resultaten in een vergelijkende studie van de beste algoritmes die werden geselecteerd uit de hoofdstukken 3 en 4 voor wat betreft het schedulingsmodel enerzijds en anderzijds van de hoofdstukken 5 en 6 voor wat betreft het ontwikkelde simulatiemodel.

REFERENCES

Abbott, A. (1995) Sequence analysis: new methods for old ideas. *Annual Review of Sociology*, **21**, 93-113.

Abbott, A. and Hrycak, A. (1990) Measuring resemblance in sequence data: an optimal matching analysis of musicians' careers. *Journal of Sociology*, **96**, 144-185.

Adler, T. and Ben-Akiva, M.E. (1979) A theoretical and empirical model of trip-chaining behaviour. *Transportation Research B*, **13**, 243-257.

Agrawal, R., Imielinski, T. and Swami, A. (1993) Mining association rules between sets of Items in large databases. *Proceedings of the ACM SIGMOD Conference on Management of Data*, Washington, D.C., USA, 207-216.

Agrawal, R. and Srikant, R. (1994) Fast algorithms for mining association rules. *Proceedings of the 20th VLDB Conference*, Santiago, Chile.

Agrawal, R. and Srikant, R. (1995) Mining sequential patterns. *Proceedings of the 11th International Conference on Data Engineering*, Taipei, Taiwan, 3-14.

Algers, S., Eliasson, J. and Mattsson, L.-G. (2005) Is it time to use activity-based models? A discussion of planning needs and modelling possibilities. *Forthcoming in The Annals of Regional Science*.

Ali, K., Manganaris, S. and Srikant, R. (1997) Partial classification using association rules. *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, 115-118.

Anand, S.S., Hughes, J.G., Bell, D.A. and Patrick, A.R. (1997) Tackling the cross-sales problem using data mining. *Proceedings of the First Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 331-343.

Andreassen, S., Jensen, F.V., Andersen, K., Falck, B., Kjaerulff, U., Woldbye, M., Sorensen, A.R., Rosenfalck, A. and Jensen, F. (1989) MUNIN-and expert EMG assistant. In Desmedt, J.E. (Ed.): *Computer-Aided Electromyography and Expert Systems*, Elsevier Science Publishers, Amsterdam, 255-277.

Arentze, T.A., Borgers, A.W.J., Hofman, F., Fujii, S., Joh, C.-H., Kikuchi, A., Kitamura, R., Timmermans, H.J.P. and van der Waerden, P. (2000) Rule-based versus utility-maximizing models of activity-travel patterns. *Proceedings of the 9th International Association for Travel Behaviour Research Conference*, Gold Coast, Queensland, Australia.

Arentze, T.A., Hofman, F., Kalfs, N. and Timmermans, H.J.P. (1999) (SYLVIA) System for logical verification and inference of activity diaries. *Transportation Research Record*, **1660**, 156-163.

Arentze, T.A., Hofman, F. and Timmermans, H.J.P. (2003) Reintroduction of Albatross' decision rules using pooled activity-travel diary data and extended set of land use and cost-related condition states. *Proceedings of the 82nd Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Arentze, T.A., Katoshevski, R. and Timmermans, H.J.P. (2001) A micro-simulation model of activity-travel behavior of individuals in urban environments. *Paper presented at the 7th International CUPUM Conference*, Honolulu, USA.

Arentze, T.A. and Timmermans, H.J.P. (2000) *Albatross: A Learning-Based Transportation Oriented Simulation System*. European Institute of Retailing and Services Studies, Eindhoven.

Arentze, T.A. and Timmermans, H.J.P. (2002) *Albatross 2.0*. European Institute of Retailing and Services Studies, Eindhoven.

Arentze, T.A. and Timmermans, H.J.P. (2003) Modelling learning and adaptation processes in activity-travel choice: A framework and numerical experiment. *Transportation*, **30**, 37-62.

Attneave, F. (1959) *Applications of Information Theory to Psychology: A Summary of Basic Concepts, Methods, and Results*. Holt, Rinehart & Winston, New York.

Axhausen, K. (2000) Activity-based modelling: research directions and possibilities. Internal paper n. 48. IVT ETH Zurich.

Bakeman, R. and Gottman, J.M. (1986) *Observing Interaction: An Introduction to Sequential Analysis*. Cambridge University Press, New York.

Barret, G. (1996) The transport dimension. In Jenks, M., Burton, E., Williams, K. (Eds.): *The Compact City: A Sustainable Urban Form?*, Chapman & Hall, London, 171-180.

Barto, A.G. and Sutton, R.S. (1981) Associative search network: a reinforcement learning associative memory. *Biological Cybernetics*, **40**, 201-211.

Barton, S. (1994) Chaos, self-organization, and psychology. *American Psychologist*, **49**, 5-14.

Beckman, R., Baggerly, K.A. and McKay, M.D. (1996) Creating synthetic baseline populations. *Transportation Research A*, **30**, 415-429.

Ben-Akiva, M.E. and Bowman, J.L. (1995) Activity based disaggregate travel-demand system with daily activity schedules. *Paper presented at the Workshop on Activity-Based Analysis*, Eindhoven, The Netherlands.

Ben-Akiva, M.E. and Bowman, J.L. (1998) Integration of an activity-based model system and a residential location model. *Urban Studies*, **35 (7)**, 1231-1253.

Ben-Akiva, M.E., Bowman, J.L. and Gopinath, D. (1996) Travel demand model system for the information area. *Transportation*, **25**, 241-266.

Berchtold, A. and Raftery, A.E. (2002) The mixture transition distribution model for high-order Markov chains and non-Gaussian time series. *Statistical Science*, **17**, 328-356.

Bharucha-Reid, A.T. (1960) *Elements of the Theory of Markov Processes and Their Applications*. McGraw-Hill Series in Probability and Statistics, New York.

Bhat, C.R. (1996) A hazard-based duration model of shopping activity with nonparametric baseline specification and nonparametric control for unobserved heterogeneity. *Transportation Research B*, **30**, 189-207.

Bhat, C.R. (1999) A comprehensive and operational analysis framework for generating the daily activity-travel pattern of workers. *Paper presented at the 78th Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Bhat, C.R., Guo, J.Y., Srinivasan, S. and Sivakumar, A. (2004) Comprehensive econometric microsimulator for daily activity-travel patterns. *Transportation Research Record*, **1894**, 57-66.

Bhat, C.R. and Misra, R. (2001) A comprehensive activity-travel pattern modelling system for non-workers with empirical focus on the organization of activity episodes. *Paper presented at the 80th Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Bhat, C.R. and Misra, R. (2002) Comprehensive activity-travel pattern modeling system for non-workers with empirical focus on the organization of activity episodes. *Transportation Research Record*, **1777**, 16-24.

Bhat, C.R. and Singh, S. (1997) A joint model of work mode choice, evening commute stops and post-home arrival stops. Final report, submitted to U.S. DOT Region 1, MIT.

Bhat, C.R. and Singh, S. (2000) A comprehensive daily activity-travel generation model system for workers. *Transportation Research A*, **34**, 1-22.

Bishop, Y.M., Fienberg, S.E. and Holland, P.W. (1977) *Discrete Multivariate Analysis: Theory and Practice*. The MIT Press, Cambridge.

Blake, C.L. and Merz, C.J. (1998) UCI Repository of machine learning databases. [http://www.ics.uci.edu/~mlearn/MLRepository.html], Department of Information and Computer Science, University of California, Irvine, CA, USA.

Bloemer, J.M.M., Brijs, T., Swinnen, G. and Vanhoof, K. (2003) Comparing complete and partial classification for identifying customers at risk. *International Journal of Research in Marketing*, **20**, 117-131.

Bollerslev, T., Chou, R.Y. and Kroner, K.F. (1992) ARCH modeling in finance: a review of the theory and empirical evidence. *Journal of Econometrics*, **52**, 5-59.

Bowman, J.L. (1995) *Activity Based Travel Demand Model System with Daily Activity Schedules*. Master of Science Thesis in Transportation, Massachusetts Institute of Technology.

Bowman, J.L. and Ben-Akiva, M.E. (1999) The day activity schedule approach to travel demand analysis. *Paper presented at the 78th Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Bowman, J.L., Bradley, M., Shiftan, Y., Lawton, T.K. and Ben-Akiva, M.E. (1998) Demonstration of an activity-based model system for Portland. *Paper presented at the 8th World Conference on Transport Research*, Antwerp, Belgium.

Brail, R. (1969) *Activity System Investigations*. Ph.D. Dissertation.

Breiman, L. (1996) Heuristics of instability and stabilization in model selection. *The Annals of Statistics*, **24(6)**, 2350-2383.

Brijs, T. (2002) *Retail Market Basket Analysis: A Quantitative Modelling Approach*. Ph.D. Dissertation, Faculty of Applied Economic Sciences, Limburg University, Diepenbeek.

Brin, S., Motwani, R., Ullman, J.D. and Tsur, S. (1997) Dynamic itemset counting and implication rules for market basket data. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Tucson, Arizona (USA), 255-264.

Chapin, F.S. (1974) *Human Activity Pattern in the City*. Wiley, New York.

Charypar, D., Graf, P. and Nagel, K. (2004) Q-learning for flexible learning of daily activity plans. *Proceedings of the Swiss Transport Research Conference (STRC)*, Monte Verita, Czechoslovakia.

Charypar, D. and Nagel, K. (2005) Q-learning for flexible learning of daily activity plans. *Paper presented at the 84th Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Chen, L.H.Y. (1975) Poisson approximation for dependent trials. *Annals of Probability*, **3**, 534-545.

Cheng, J., Bell, D. and Liu, W. (2002) Learning Bayesian networks from data: an efficient approach based on information theory. *Artificial Intelligence*, **137**, 43-90.

Chow, C.K. and Liu, C.N. (1968) Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, **14**, 462-467.

Cohen, A., Ivry, R.I. and Keele, S.W. (1990) Attention and structure in sequence learning. *Journal of Experimental Psychology: Learning, Memory & Cognition*, **16**, 17-30.

Cooper, G.F. and Herskovits, E. (1992) A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, **9**, 309-347.

Crites, R.H. and Barto, A.G. (1996) Improving elevator performance using reinforcement learning. In Touretzky, D.S.; Mozer, M.C.; Hasselmo, M.E. (Eds.): *Advances in Neural Information Processing Systems 8*, The MIT Press, 1017-1023.

Damm, D. (1980) Interdependencies in activity behaviour. *Transportation Research Record*, **750**, 33-40.

Davis, G. and Pei, J. (2004) Applying Bayesian methods to the clinical investigation of speed as a cause of road accidents. *Proceedings of the 83rd Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

De Jong, G.C. (1997) A microeconomic model of the joint decision on car ownership and car use. In Stopher, P., Lee-Gosselin, M. (Eds.): *Understanding Travel Behaviour in an Era of Change*, Pergamon, Oxford.

Dijkstra, E. (1959) A note on two problems in connection with graphs. *Numerical Mathematics*, **1**, 269-271.

Dijst, M. (1995) *Het Elliptisch Leven*. Ph.D dissertation, KNAG, Utrecht University, Utrecht.

Dijst, M. (1997) Spatial policy and passenger transportation. *Netherlands Journal of Housing and the Built Environment*, **12**, 91-111.

Dijst, M. and Vidakovic, V. (1997) Individual action space in the city. In Ettema, D.F., Timmermans, H.J.P. (Eds.): *Activity-Based Approaches to Activity Analysis*, Pergamon Press, Oxford, 73-88.

Dong, G., Zhang, X., Wong, L. and Li, J. (1999) CAEP: classification by aggregating emerging patterns. *Proceedings of the Second International Conference on Discovery Science*, Tokyo, Japan, 30-42.

Doob, J.L. (1990) *Stochastic Processes*. John Wiley & Sons, New York.

Dougherty, J., Kohavi, R. and Sahami, M. (1995) Supervised and unsupervised discretization of continous features. *Proceedings of the Proceedings of the Twelfth International Conference on Machine Learning*, San Francisco, USA, 194-202.

DYNAMIT (2000) Massachusetts Institute of Technology. Cambridge, Massachusetts.

Dynkin, E.B. (1965) *Markov Processes, Vols. 1-2*. Springer-Verlag, Berlin-Göttingen-Heidelberg.

Esser, J. (1998) *Simulation von Stadtverkehr auf der Basis Zellularer Automaten*. Ph.D. Dissertation, University of Duisburg, Germany.

Esser, J. and Nagel, K. (2001) Iterative demand generation for transportation simulations. In Hensher, D. (Ed.): *Travel Behaviour Research, The Leading Edge*, Elsevier, 689-709.

Ettema, D.F., Borgers, A.W.J. and Timmermans, H.J.P. (1994) Using interactive computer experiments for identifying activity scheduling heuristics. *Paper presented at the 7th International Conference on Travel Behavior*, Santiago, Chile.

Ettema, D.F., Borgers, A.W.J. and Timmermans, H.J.P. (2000) A simulation model of activity scheduling heuristics: an empirical test. *Geographical and Environmental Modelling*, **4**, 175-187.

European Commission (2001). White paper: European transport policy for 2010: time to decide. Luxembourg: Office for Official Publications of the European Communities. Available: http://europa.eu.int.

Everitt, B.S. (1992) *The Analysis of Contingency Tables (Second Edition)*. Chapman and Hall, London.

Fayyad, U.M. and Irani, K. (1993) Multi-interval discretization of continuous-valued attributes for classification learning. *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, San Francisco, USA, 1022-1027.

Fayyad, U.M., Piatetsky-Shapiro, G. and Smyth, P. (1996) From data mining to knowlegde discovery: an overview. In Fayyad, U.M.; Piatetsky-Shapiro, G.; Smyth, P.; Uthurusamy, R. (Eds.): *Advances in Knowledge Discovery and Data Mining*, AAAI Press / The MIT Press, 1-34.

Fosgerau, M. (1998) PETRA: an activity based approach to travel demand analysis. *Paper presented at the 8th World Conference on Transport Research*, Antwerp, Belgium.

Fried, M., Havens, J. and Thall, M. (1977) Travel behavior - a synthesized theory. Final Report, NCHRP. *Transportation Research Bord*, Washington, D.C.

Gärling, T., Brännäs, K., Garvill, J., Golledge, R.G., Gopal, S., Holm, E. and Lindberg, E. (1989) Household activity scheduling. *Transport Policy, Management and Technology Towards 2001, Selected Proceedings of the 5th World Conference on Transport Research*, **4**, Ventura, CA, 235-248.

Gawron, C. (1998) *Simulation-Based Traffic Assignment*. Ph.D. Dissertation, University of Cologne, Germany.

Geiger, D. and Pearl, J. (1988) Logical and algorithmic properties of conditional independence. Technical Report R-97, Cognitive Systems Laboratory, UCLA.

Golledge, R.G. (1975) On reality and reality. *Paper presented at the Association of American Geographers*, Milwaukee, United States.

Golledge, R.G. (1978) Learning about urban environments. In Carlstein, T.; Parkes, D.N.; Thrift, N.J. (Eds.): *Time Space and Spacing Time I: Making Sense of Time*, Edward Arnold, London, 76-98.

Good, I.J. (1965) *The Estimation of Probabilities: an Essay on Modern Bayesian Methods*. M.I.T. Press.

Goodwin, P.B., Kitamura, R. and Meurs, H. (1990) Some principles of dynamic analysis of travel demand. In: *Developments in dynamic and activity-based approaches to travel analysis*, Gower, London.

Gottman, J.M. (1981) *Time-Series Analysis. A Comprehensive Introduction for Social Scientists*. Cambridge University Press, Cambridge.

Gottman, J.M. and Roy, A.K. (1990) *Sequential Analysis. A Guide for Behavioral Researchers*. Cambridge University Press, New York.

Goulias, K.G. (1999) Longitudinal analysis of activity and travel pattern dynamics using generalized mixed Markov latent class models. *Transportation Research B*, **33(8)**, 535-557.

Goulias, K.G. and Kitamura, R. (1992) Travel demand forecasting with dynamic microsimulation. *Transportation Research Record*, **1357**, 8-17.

Goulias, K.G. and Kitamura, R. (1996) A dynamic model system for regional travel demand forecasting. In Golob, T., Kitamura R. (Eds.): *Panels for Transportation Planning: Methods and Applications*, Kluwer Academic Publishers.

Goulias, K.G. and Kitamura, R. (1997) Regional travel demand forecasting with dynamic microsimulation models. In Golob, T., Kitamura R. (Eds.): *Panels for Transportation Planning: Methods and Applications*, Kluwer Academic Publishers, 321-348.

Gras, R. and Lahrer, A. (1993) L'implication statistique: une nouvelle méthode d'analyse des données. *Mathématique, Informatique et Sciences Humaines*, **120**, 5-31.

Greaves, S.P. and Stopher, P.R. (2000) Creating a synthetic household travel and activity survey: rationale and feasibility analysis. *Transportation Research Record*, **1706**, 82 - 91.

Grieco, M. and Turner, J. (1997) Gender, poverty and transport: a call for policy attention. *Paper presented at the International Forum of Urban Poverty*, Florence, Italy.

Guillaume, S., Guillet, F. and Philippé, J. (1998) Improving the discovery of association rules with intensity of implication. *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, New York, USA, 318-327.

Guo, J.Y. and Bhat, C.R. (2001) Activity-based travel-demand modeling for metropolitan areas in texas: representation and analysis plan and data needs analysis for the activity-travel system. Research Report 4080-1. *Center for Transportation Research*, Austin, Texas.

Hägerstrand, T. (1970) What about people in regional science? *Regional Science Association*, **24**, 7-21.

Haken, H. (1983) *Synergetics. An Introduction. Nonequilibrium Phase Transitions and Self-Organization in Physics, Chemistry and Biology*. Springer, Berlin.

Hamed, M.M. and Mannering, F.L. (1993) Modeling travelers' postwork activity involvement: Toward a new methodology. *Transportation Science*, **27**, 381-394.

Han, J., Pei, J., Mortazavi-Asl, B., Chen, Q., Dayal, U. and Hsu, M.-C. (2000) FreeSpan: frequent pattern-projected sequential pattern mining. *Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining (KDD-00)*, Boston, USA.

Hastie, T., Tibshirani, R. and Friedman, J. (2001) *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Spinger-Verlag, New York.

Heckerman, D. (1986) Probabilistic interpretation for MYCIN's certainty factors. In Kanal, L.N.; Lemmer, J.F. (Eds.): *Uncertainty in Artificial Intelligence*, Elsevier/North Holland, Amsterdam, 167-196.

Heckerman, D., Geiger, D. and Chickering, D.M. (1995) Learning Bayesian networks: the combination of knowledge and statistical data. *Machine Learning*, **20**, 197-243.

Heckerman, D., Horvitz, E.J. and Nathwani, B.N. (1992) Toward normative expert systems part I: The Pathfinder Project. KSL Report 92-66.

Hemmens, G.C. (1970) Analysis and simulation of urban activity patterns. *Socio-Economic Planning Sciences*, **4**, 53-66.

Herskovits, E. (1991) *Computer-Based Probabilistic Network Construction*. Ph.D. Dissertation, Medical information sciences, Stanford University, Stanford, CA.

Holte, R.C. (1993) Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, **11**, 63-90.

Hopp, W. (1987) A sequential model of R&D investment over an unbounded time horizon. *Management Science*, **33**, 500-508.

Huigen, P. (1986) *Binnen of Buiten Bereik?: een Sociaal-Geografisch Onderzoek in Zuidwest-Friesland*. Nederlandse Geografische Studies 7, Utrecht.

Janssens, D., Brijs, T., Vanhoof, K. and Wets, G. (2005a) Evaluating the performance of cost-based discretization versus entropy- and error-based discretization. *Forthcoming in Computers and Operations Research*, Available online 12 February 2005.

Janssens, D., Lan, Y., Wets, G., Chen, G. and Brijs, T. (2004a) Empirically validating an adapted classification based on associations algorithm on UCI data. *Proceedings of the Fuzzy Information Processing Conference on Applied Computational Intelligence*, Blankenberge, Belgium, 167-178,ISBN 981-238-873-7.

Janssens, D. and Wets, G. (2005) The presentation of an activity-based approach for surveying and modelling travel behaviour. *Paper to be presented at the 32nd "Colloquium Vervoersplanologisch Speurwerk*", Antwerp, Belgium.

Janssens, D., Wets, G., Brijs, T. and Vanhoof, K. (2004b) Evaluating the use of Bayesian networks in a sequential rule-based model for activity scheduling behaviour. *Proceedings of the 10th World Conference on Transport Research*, Istanbul, Turkey.

Janssens, D., Wets, G., Brijs, T. and Vanhoof, K. (2004c) Simulating activity diary data by means of sequential probability information: development and evaluation of an initial framework. *Proceedings of the 83rd Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Janssens, D., Wets, G., Brijs, T. and Vanhoof, K. (2004d) The simulation of activity diary data using sequential probability distributions. *Proceedings of the ISCTSC Triennial International Conference on Transport Survey Methods*, Costa Rica.

Janssens, D., Wets, G., Brijs, T. and Vanhoof, K. (2005b) Adapting the CBA Algorithm by means of intensity of implication. *Information Sciences*, **173(4)**, 305-318.

Janssens, D., Wets, G., Brijs, T. and Vanhoof, K. (2005c) The development of an adapted Markov chain modelling heuristic and simulation framework in the context of transportation research. *Expert systems with applications*, **28**, 105-117.

Janssens, D., Wets, G., Brijs, T. and Vanhoof, K. (2005d) Simulating daily activity patterns throught the identification of sequential dependencies. In Timmermans, H.J.P. (Ed.): *Progress in Activity-based analysis*, Elsevier, Amsterdam, 67-90. Also in *Proceedings of the Conference on Progress in Activity-Based Analysis,* Maastricht, The Netherlands.

Janssens, D., Wets, G., Brijs, T. and Vanhoof, K. (2005e) Using an adapted classification based on associations algorithm in an activity-based transportation system. In Ruan, D., Chen, G., Kerre, E.E., Wets, G. (Eds.): *Intelligent Data Mining: Techniques and Applications*, 275-292.

Janssens, D., Wets, G., Brijs, T., Vanhoof, K., Arentze, T.A. and Timmermans, H.J.P. (2005f) Integrating Bayesian networks and decision trees in a sequential rule-based transportation model. *Forthcoming in European Journal of Operational Research*, Available online 16 June 2005.

Janssens, D., Wets, G., Brijs, T., Vanhoof, K. and Chen, G. (2003a) Adapting the CBA-algorithm by means of intensity of implication. *Proceedings of the First International Conference on Fuzzy Information Processing Theories and Applications*, Beijing, China, 397-403, ISBN:7-302-062994.

Janssens, D., Wets, G., Brijs, T., Vanhoof, K. and Timmermans, H.J.P. (2003b) Identifying behavioral principles underlying activity patterns by means of Bayesian networks. *Proceedings of the 82nd Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Janssens, D., Wets, G., Brijs, T., Vanhoof, K., Timmermans, H.J.P. and Arentze, T.A. (2004e) Improving the performance of a multi-agent rule-based model for activity pattern decisions using Bayesian networks. *Proceedings of the 83rd Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Janssens, D., Wets, G., Brijs, T., Vanhoof, K., Timmermans, H.J.P. and Arentze, T.A. (2004f) Improving the performance of a multi-agent rule-based model for activity pattern decisions using Bayesian networks. *Journal of the Transportation Research Board*, **1894**, 75-83.

Janssens, D., Wets, G. and Timmermans, H.J.P. (2005g) The presentation of an activity-based approach for surveying and modelling travel behaviour. *Paper to be presented at the 2nd "Belgische Geografendag"*, Gent, Belgium.

Jensen, F.V., Lauritzen, S.L. and Olesen, K.G. (1990) Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quarterly*, **4**, 269-282.

Joh, C.-H. (2004) *Measuring and Predicting Adaptation in Multidimensional Activity-Travel Patterns*. Ph.D Dissertation, Urban Planning Group, University of Eindhoven, Eindhoven.

Joh, C.-H., Arentze, T.A., Hofman, F. and Timmermans, H.J.P. (2001a) Activity pattern similarity: a multidimensional sequence alignment method. *Transportation Research B*, **36**, 385-403.

Joh, C.-H., Arentze, T.A. and Timmermans, H.J.P. (2001b) A position-sensitive sequence alignment method illustrated for space-time activity diary data. *Environment and Planning A*, **33**, 313-338.

Joh, C.-H., Arentze, T.A. and Timmermans, H.J.P. (2003) Estimating non-linear utility functions of time use in the context of an activity schedule adaptation model. *Proceedings of the 10th International Conference on Travel Behaviour Research*, Lucerne, Switzerland.

Joh, C.-H., Arentze, T.A. and Timmermans, H.J.P. (2004) Activity-travel (re)scheduling decision processes: empirical estimation of the Aurora model. *Proceedings of the 83rd Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Johansson, O. and Schipper, L. (1997) Measuring long-run automobile fuel demand; Separate estimations of vehicle stock, mean fuel intensity and mean annual driving distance. *Journal of Transport Economics and Policy*, **31**, 277-292.

Johansson-Stenman, O. (2002) Estimating individual driving distance by car and public transport use in Sweden. *Applied Economics*, **34**, 959-967.

Jones, P.M., Dix, M.C., Clarke, M.I. and Heggie, I.G. (1983) *Understanding Travel Behaviour*. Gower, Aldershot.

Jonz, J. (1989) Textual sequence and 2nd-language comprehension. *Language Learning*, **39**, 207-249.

Kaelbling, L.P. (1993) *Learning in Embedded Systems*. The MIT Press/Bradford Books.

Kaelbling, L.P., Littman, M.L. and Moore, A. (1996) Reinforcement learning: a survey. *Journal of Artificial Intelligence Research*, **4**, 237-285.

Kass, G.V. (1980) An exploratory technique for investigating large quantities of categorical data. *Applied Statistics*, **29**, 119-127.

Kasturirangan, K., Pendyala, R.M. and Koppelman, F.S. (2002) On the role of history in modelling activity type choice and activity duration for commuters. *Transportation Research Record*, **1807**, 129-136.

Katz, L. and Proctor, C.H. (1959) The concept of configuration of interpersonal relations in a group as a time-dependent stochastic process. *Psychometrika*, **24**, 317-327.

Kawakami, S. and Isobe, T. (1982) Development of a one-day travel-activity scheduling model for workers. In Carpenter, S. and Jones, P. (Eds.): *Recent Advances in Travel Demand Analysis*, Gower, Aldershot, 314-344.

Kawakami, S. and Isobe, T. (1988) Development of travel-activity scheduling model incorporating choice process of the number, location and time allocation of activities. *Paper presented at the Oxford Conference on Travel and Transportation*, Oxford, United Kingdom.

Kawakami, S. and Isobe, T. (1989) Development of a travel-activity scheduling model considering time constraint and temporal transferability test of the model. *Selected Proceedings of the 5th World Conference on Transport Research*, **4**, 221-233.

Kemeny, J.G. and Snell, J.L. (1976) *Finite Markov Chains*. Springer-Verlag, New York.

Kemeny, J.G., Snell, J.L. and Knapp, A.W. (1976) *Denumerable Markov Chains*. Springer-Verlag, New York.

Kendall, S.M. and Ord, J.K. (1990) *Time Series*. Edward Arnold, London.

Keuleers, B., Wets, G., Arentze, T.A. and Timmermans, H.J.P. (2001) Association rules in identification of spatial-temporal patterns in multiday activity diary data. *Transportation Research Record*, **1752**, 32-37.

Keuleers, B., Wets, G., Timmermans, H.J.P., Arentze, T.A. and Vanhoof, K. (2002) Stationary and time-varying patterns in activity diary panel data: explorative analysis with association rules. *Transportation Research Record*, **1807**, 9-15.

Kim, T. and Goulias, K.G. (2004) Dynamic analysis of time use and frequency of activity and travel using Latent Class Clustering and Structural Equation Modeling. *Paper presented at the Conference on Progress in Activity-Based Analysis*, Maastricht, The Netherlands.

Kitamura, R. (2000) Longitudinal Methods. In Hensher, D.; Button, K. (Eds.): *Handbook of Transport Modelling*, Pergamon, Kidlington, Oxford, 113-128.

Kitamura, R., Chen, C. and Pendyala, R.M. (1997) Generation of synthetic daily activity-travel patterns. *Transportation Research Record*, **1607**, 154–162.

Kitamura, R., Chen, C., Pendyala, R.M. and Narayanan, R. (2000) Microsimulation of daily activity-travel patterns for travel demand forecasting. *Transportation*, **27(1)**, 25-51.

Kitamura, R. and Fujii, S. (1998) Two computational process models of activity-travel choice. In Gärling, T., Laitila, T. and Westin, K. (Eds.): *Theoretical Foundations of Travel Choice Modelling*, Elsevier, Oxford, 251-279.

Kitamura, R. and Goulias, K.G. (1991) *MIDAS: A Travel Demand Forecasting Tool Based on a Dynamic Model System of Household Demographics and Mobility*. Project Bureau Integrale Verkeer- en Vervoerstudies, Ministerie van Verkeer en Waterstaat, The Netherlands.

Kitamura, R. and Kermanshah, M. (1983) Identifying Time and History Dependencies of Activity Choice. *Transportation Research Record*, **944**, 22-30.

Kitamura, R., Pendyala, R.M., Pas, E.I. and Reddy, D.V. (1995) Application of AMOS: an activity-based TCM evaluation tool applied to Washington D.C. metropolitan area. *Proceedings of the 23rd Summer Annual Meeting*, London, United Kingdom, 177-190.

Krygsman, S. (2004) *Activity and Travel Choice in Multimodal Public Transport Systems*. PhD. Thesis, Faculty of Geographical Sciences, Utrecht University, Utrecht.

Kulkarni, A. and McNally, M.G. (2001) A microsimulation of daily activity patterns. *Paper presented at the 80th Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Kwan, M.P. (1997) GISICAS: an activity-based travel decision support system using a GIS interfaced computational process model. In Ettema, D.F. and Timmermans, H.J.P. (Eds.): *Activity Based Approaches to Activity Analysis*, Pergamon Press, Oxford, 263-282.

Lam, W. and Bacchus, F. (1994) Learning Bayesian belief networks: an approach based on the MDL principle. *Computational Intelligence*, **10**, 269-293.

Lan, Y., Chen, G., Janssens, D. and Wets, G. (2004) Dilated Chi-square: a novel interestingness measure to build accurate and compact decision list. *Proceedings of the International Conference on Intelligent Information Processing*, Beijing, China, 233-238, ISBN 0-387-23151-X (HC).

Lan, Y., Janssens, D., Chen, G. and Wets, G. (2006) Improving associative classification by incorporating novel interestingness measures. *Forthcoming in Expert systems with applications*.

Langeheine, R. and van de Pol, F. (1990) A unifying framework for Markov modelling in discrete space and discrete time. *Sociological Methods and Research*, **18(4)**, 416-441.

Lemay, P. (1999) *The Statistical Analysis of Dynamics and Complexity in Psychology: a Configural Approach*. Ph.D. Dissertation, Social and Political Sciences, University of Lausanne, Lausanne.

Lenntorp, B. (1976) *Paths in Space-Time Environment: a Time Geographic Study of Possibilities of Individuals*. Lund Studies in Geography B: Human Geography, 44, Lund: Gleerup.

Lipman, D. and Pearson, W. (1984) Rapid and sensitive protein similarity searches. *Science*, **22**, 1435-1441.

Littman, M.L. (1994) Markov games as a framework for multi-agent reinforcement learning. *Proceedings of the Eleventh International Conference on Machine Learning*, San Francisco, CA, USA, 157-163.

Liu, B., Hsu, W. and Ma, Y. (1998) Integrating classification and association rule mining. *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98)*, New York, USA, 80-86.

Liu, B., Ma, Y. and Wong, C. (2001a) Classification using association rules: weaknesses and enhancements. In Kumar, V. (Ed.): *Data Mining for Scientific and Engineering Applications*, 591.

Liu, W., Han, J. and Pei, J. (2001b) CMAR: accurate and efficient classification based on multiple class-association rules. *Proceedings of the International Conference on Data Mining (ICDM-01)*, San Jose, CA, USA.

Logan, J.A. (1981) A structural model of the high-order Markov process incorporating reversion effects. *Journal of Mathematical Sociology*, **8**, 75-89.

Lucardie, G.L. (1994) *Functional Object-Types as a Foundation of Complex Knowledge-Based Systems*. Ph.D. Dissertation, Eindhoven University of Technology, Eindhoven.

MacDonald, I.L. and Zucchini, W. (1997) *Hidden Markov and Other Models for Discrete-Valued Time Series*. Chapman and Hall, London.

Mahadevan, S. and Connell, J. (1992) Automatic programming of behavior-based robots using reinforcement learning. *Artificial Intelligence*, **55**, 311-365.

Mahmassani, H., Hu, T. and Jayakrishnan, R. (1995) Dynamic traffic assignment and simulation for advanced network informatics (DYNASMART). In Gartner, N., Improta, G. (Eds.): *Urban Traffic Networks: Dynamic Flow Modeling and Control*, Springer, Berlin/New York.

Mannila, H., Toivonen, H. and Verkamo, A.I. (1994) Efficient algorithms for discovering association rules. In Fayyad, U.M., Uthurusamy, R. (Eds.): *Knowledge Discovery in Databases*, 181-192.

McBrearty, S. (1988) The Sagoan–Lupemban and middle stone-age sequence at the Muguruk site. *World Archaeology*, **19**, 388-420.

McNally, M.G. (1995) An activity-based microsimulation model for travel demand forecasting. In Ettema, D.F. and Timmermans, H.J.P. (Eds.): *Activity-Based Approaches to Travel Analysis*, Pergamon Press, Oxford, 37-54.

McNally, M.G. (1999) Activity-based forecasting model incorporating GIS. *Geographical Systems*, **5**, 163-187.

McNally, M.G. (2000) The activity-based approach. *Center for Activity Systems Analysis*, Paper UCI-ITS-AS-WP-00-4.

Miller, H.J. (2003) What about people in geographic information science. *Computers, Environment and Urban Systems*, **27**, 447-453.

Mitchell, T. (1997) *Machine Learning*. McGraw Hill, New York.

Moons, E. (2005) *Modelling Activity-Diary Data: Complexity or Parsimony?* Ph.D. Dissertation, Faculty of Sciences, Limburg University, Diepenbeek.

Moriarty, D. and Langley, P. (1998) Learning cooperative lane selection strategies for highways. *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, Madison, Wisconsin, USA, 684-691.

Nagel, K. and Marchal, F. (2003) Computational methods for multi-agent simulations of travel behavior. *Proceedings of the 10th International Conference on Travel Behaviour Research*, Lucerne, Switzerland.

Ortúzar, J.d.D. and Willumsen, L.G. (2002) *Modelling Transport*. 3rd Edition John Wilson & Sons, West Sussex.

Ozbay, K. and Noyan, N. (2005) Application of Bayesian networks to analyze in analyzing incidents and decision-making. *Proceedings of the 84th Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Pearl, J. (1988) *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Palo Alto.

Pearman, A.D. and Button, K.J. (1976) Regional variations in car ownership. *Applied Economics*, **8**, 231-233.

Pegram, G.G.S. (1980) An autoregressive model for multilag Markov Chains. *Journal of Applied Probability*, **17**, 350-362.

Pei, J., Han, B., Pinto, H., Chen, Q., Dayal, U. and Hsu, M.-C. (2001) Pre xSpan: mining sequential patterns efficiently by pre x-projected pattern growth. *Proceedings of the Seventeenth International Conference on Data Engineering (ICDE '01)*, Heidelberg, Germany, 215-224.

Pendyala, R.M. (2004) FAMOS: application in Florida. *Paper presented at the 83rd Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Pendyala, R.M., Kitamura, R., Chen, C. and Pas, E.I. (1997) An activity-based microsimulation analysis of transportation control measures. *Transport Policy*, **4**, 183-192.

Pendyala, R.M., Kitamura, R. and Reddy, D.V. (1995) A rule-based activity-travel scheduling algorithm integrating neural networks of behavioural adaptation. *Paper presented at the EIRASS Conference on Activity-Based Approaches*, Eindhoven, The Netherlands.

Pendyala, R.M., Kitamura, R. and Reddy, D.V. (1998) Application of an activity-based travel demand model incorporating a rule-based algorithm. *Environment and Planning B*, **25**, 753-772.

Perlich, C., Provost, F. and Simonoff, J.S. (2004) Tree induction vs. logistic regression: a learning-curve analysis. *Journal of Machine Learning Research*, **4(2)**, 211-255.

Plach, M. (1997) Using Bayesian networks to model probabilistic inferences about the likelihood of traffic congestions. In Harris, D. (Ed.): *Engineering Psychology and Cognitive Ergonomics*, Aldershot: Ashgate, 363-371.

Pribyl, O. and Goulias, K.G. (2004) Simulation of daily activity patterns. *Paper presented at the Conference on Progress in Activity-Based Analysis*, Maastricht, The Netherlands.

Prinzie, A. and Van den Poel, D. (2005) Investigating purchasing-sequence patterns for financial services using Markov, MTD and MTDg models. *European Journal of Operational Research*, Forthcoming.

Quinlan, J.R. (1993) *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo.

Raftery, A.E. (1985) A model for high-order Markov chains. *Journal of the Royal Statistical Society*, **47B**, 528-539.

Raftery, A.E. and Tavaré, S. (1994) Estimation and modelling repeated patterns in high order Markov chains with the mixture transition distribution model. *Applied Statistics*, **43**, 179-199.

Rakha, H.A. and Van Aerde, M.W. (1996) Comparison of simulation modules of TRANSYT and INTEGRATION models. *Transportation Research Record*, **1566**, 1-7.

Raney, B., Balmer, M., Axhausen, K. and Nagel, K. (2003a) Agent-based activities planning for an iterative traffic simulation of Switzerland. *Paper presented at the 10th International Conference on Travel Behaviour Research*, Lucerne, Switzerland.

Raney, B., Cetin, N., Vollmy, A., Milenko, V., Axhausen, K. and Nagel, K. (2003b) Toward activity-based microscopic simulation of all of Switzerland. *Paper presented at the 82nd Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Recker, W.W. (1995) The household activity pattern problem: general formulation and solution. *Transportation Research B*, **29**, 61-77.

Recker, W.W., McNally, M.G. and Root, G.S. (1986a) A model of complex travel behavior: Part 1: theoretical development. *Transportation Research A*, **20**, 307-318.

Recker, W.W., McNally, M.G. and Root, G.S. (1986b) A model of complex travel behavior: Part 2: an operational model. *Transportation Research A*, **20**, 319-330.

Rickert, M. (1998) *Traffic Simulation on Distributed Memory Computers*. Ph.D. Dissertation, University of Cologne, Germany.

Rietveld, P. (1994) Spatial economic impacts of transport infrastructure supply. *Transportation Research A*, **28 A**, 329-341.

Ruiter, E.R. and Ben-Akiva, M.E. (1978) Disaggregate travel demand models for the San Francisco bay area. *Transportation Research Record*, **673**, 121-128.

Sabherwal, R. and Robey, D. (1995) Reconciling variance and process strategies for studying information system development. *Information Systems Research*, **6**, 303-327.

Salomon, I., Bovy, P.H.L. and Orfeuil, J.P. (1993) *A Billion Trips a Day: Tradition and Transition in European Travel Patterns*. Kluwer Academic Publishers, Transportation Research, Economic and Policy, The Netherlands.

Schaal, S. and Atkeson, C. (1994) Robot juggling: an implementation of memory-based learning. *Control Systems Magazine*, **14**, 57-71.

Schneider, J., Boyan, J. and Moore, A. (1998) Value function based production scheduling. *Proceedings of the Fifteenth International Conference on Machine Learning*, Madison, Wisconsin, USA, 522-530.

Scholtes, J.C. (1991) Unsupervised learning and the information retrieval problem. *Proceedings of the International Joint Conference on Neural Networks*, New York, USA, 95-100.

Smith, L., Beckman, R., Anson, D., Nagel, K. and Williams, M. (1995) TRANSIMS: transportation analysis and simulation system. *Proceedings of the 5th National Transportation Planning Methods Applications Conference*, Seatlle, Washington, USA.

Sparmann, U. (1980) *ORIENT: Ein Verhaltensorientiertes Simulationsmodell zur Verkehrsprognose*. Institutes für Verkehrswesen, Universität Karlsruhe, Karlsruhe.

Spiegel, M.R. (1980) *Probability and Statistics*. McGraw-Hill, New York.

Srikant, R. and Agrawal, R. (1996a) Mining quantitative association rules in large relational tables. *Proceedings of the the ACM SIGMOD International Conference on Management of Data*, 1-12.

Srikant, R. and Agrawal, R. (1996b) Mining sequential patterns: generalizations and performance improvements. *Proceedings of the Fifth International Conference on Extending Database Technology*, Avignon, France, 3-17.

Stärk, K.D.C. and Pfeiffer, D.U. (1999) The application of non-parametric techniques to solve classification problems in complex data sets in veterinary epidemiology – An example. *Intelligent Data Analysis*, **3**, 23-35.

Stopher, P.R., Greaves, S.P. and Bullock, P. (2003) Simulating household travel survey data: application to two urban areas. *Proceedings of the 82nd Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Stopher, P.R., Hartgen, D.T. and Li, Y. (1996) SMART: simulation model for activities, resources and travel. *Transportation*, **23**, 293-312.

Suen, C.Y. (1979) N-gram statistics for natural language understanding and text processing. *IEEE Transactions on Pattern Analysis and Machine Intelligence Conference*, **1**, 164-172.

SUT Partnership (2002) Smart Urban Transport. Transportation biggest energy user by 2002.

Sutton, R.S. and Barto, A.G. (1998) *Reinforcement Learning: An Introduction*. MIT Press, Cambridge.

Suzuki, E. and Kodratoff, Y. (1998) Discovery of surprising exception rules based on intensity of implication. *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, New York, USA, 10-18.

Swiderski, D. (1982) A model for simulating spatially and temporally coordinated activity sequences on the basis of mental maps. In Carpenter, S., Jones P. (Eds.): *Recent Advances in Travel Demand Analysis*, Gower, Aldershot, 314-344.

Tesauro, G. (1992) Practical issues in temporal difference learning. *Machine Learning*, **8**, 257-277.

Tesauro, G. (1994) TD-Gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation*, **6**, 215-219.

Thrun, S. (1995) Learning to play to game of chess. In Tesauro, G.; Touretzky, D.; Leen, T. (Eds.): *Advances in Neural Information Processing Systems 7*, Morgan Kaufmann, San Francisco.

Timmermans, H.J.P. (2001) Models of activity scheduling behaviour. *Stadt Regional Land*, **71**, 63-78.

Timmermans, H.J.P., Arentze, T.A. and Joh, C.-H. (2002) Analyzing space-time behavior: new approaches to old problems. *Progress in Human Geography*, **26**, 175-190.

Torres, F.J. and Huber, M. (2003) Learning a causal model from household survey data using a Bayesian belief network. *Paper presented at the 82nd Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

TRANSIMS (2003) TRansportation ANalysis and SIMulation System. [http://transims.tsasa.lanl.gov], Los Alamos National Laboratory, Los Alamos, NM.

Transport and Social Exclusion Workshop (1998) *Summary Report*, University of Manchester.

United Nations Economic and Social Council, Commission on Sustainable Development (2001) Transport. *Report of the Secretary-General, Document E/CN.17/2001/3*.

Vaughn, K.M., Speckman, P. and Pas, E.I. (1997) Generating household activity-travel patterns (HATPs) for synthetic populations. *Proceedings of the 76th Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Vaughn, K.M., Speckman, P. and Sun, D. (1999) Identifying relevant socio-demographics for distinguishing household activity-travel Patterns: A multivariate regression tree approach. Paper prepared for The National Institute of Statistical Sciences (NISS).

Veldhuisen, K., Timmermans, H.J.P. and Kapoen, L.L. (2000a) Ramblas: a regional planning model based on the micro-simulation of daily activity travel patterns. *Environment and Planning A*, **32**, 427-443.

Veldhuisen, K., Timmermans, H.J.P. and Kapoen, L.L. (2000b) Micro-simulation of activity travel patterns and traffic flows: validation tests and an investigation of Monte Carlo error. *Transportation Research Record*, **1706**, 126-135.

Verhoeven, M., Arentze, T.A., Timmermans, H.J.P. and van der Waerden, P. (2005) Modelling the impact of key events on long-term transport mode choice decisions: a decision network approach using event history data. *Proceedings of the 84th Annual Meeting of the Transportation Research Board*, Washington, D.C., USA.

Viveros, M.S., Nearhos, J.P. and Rothman, M.J. (1996) Applying data mining techniques to a health insurance information system. *Proceedings of the 22nd Conference on Very Large Databases*, India, 286-294.

Vovsha, P., Petersen, E. and Donnelly, R. (2002) Microsimulation in travel demand modeling: lessons learned from the New York best practice model. *Transportation Research Record*, **1805**, 68-77.

Wang, K., Tay, W. and Liu, B. (1998) An interestingness-based interval merger for numeric association rules. *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, New York, USA, 121-128.

Wang, K. and Zhou, S. (2000) Growing decision trees on support-less association rules. *Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining (KDD-00)*, Boston, USA, 265 - 269.

Watkins, C. (1989) *Learning from Delayed Rewards*. Ph.D. Dissertation, Psychology Department, University of Cambridge, Cambridge.

Watkins, C. and Dayan, P. (1992) Technical note: Q-learning. *Machine Learning*, **8**, 279-292.

Wen, C.H. and Koppelman, F.S. (1999) An integrated system of stop generation and tour formation for the analysis of activity and travel patterns. *Transportation Research Record*, **1676**, 136-144.

Wets, G. (1998) *Decision Tables in Knowledge-Based Systems: Adding Knowledge Discovery and Fuzzy Concepts to the Decision Table Formalism*. Ph.D. Dissertation, Eindhoven University of Technology, Eindhoven.

Wilson, C. (1998) Activity pattern analysis by means of sequence-alignment methods. *Environment and Planning A*, **30**, 1017-1038.

Wilson, F.R. (1967) *Journey to Work - Modal Split*. Maclaren and Sons LTD, London.

Xu, J.G. and Agrawal, R. (1996) Membrane separation process analysis and design strategies based on thermodynamic efficiency of permeation. *Chemical Engineering Science*, **51**, 365-385.

Yin, X. and Han, J. (2003) CPAR: classification based on predictive association rules. *Proceedings of the SIAM International Conference on Data Mining (SDM'03)*, San Fransisco, CA, USA.

Yoda, K., Fukuda, T., Morimoto, Y., Morishita, S. and Tokuyama, T. (1997) Computing optimized rectilinear regions for association rules. *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, Newport Beach, USA, 96-103.

Zaki, M.J. (2001) SPADE: an efficient algorithm for mining frequent sequences. *Machine Learning*, **42**, 31-60.