DISS. ETH NO. 17238

TRAVEL DEMAND MODELING FOR MULTI-AGENT TRANSPORT SIMULATIONS: ALGORITHMS AND SYSTEMS

A dissertation submitted to

ETH ZURICH

for the degree of

Doctor of Sciences

presented by

MICHAEL BALMER

Master of Science in Computer Science, ETH Zurich

born 01.01.1974

citizen of
Wilderswil, BE

accepted on the recommendation of

Prof. Kay W. Axhausen, examiner
Prof. Kai Nagel, co-examiner

2007

Contents

List of Figures

List of Tables

Abstract

Micro-simulation is becoming increasingly important in traffic simulation, traffic analysis, and traffic forecasting. Some advantages over conventional models are:

- Computational savings in the calculation and storage of large multidimensional probability arrays.
- Larger range of output options, from overall statistics to information about each synthetic traveler in the simulation.
- Explicit modeling of the individuals' decision making processes.

The last point is important since it is not a vehicle which produces traffic; it is the person who drives the vehicle. Persons do not just produce traffic; instead each of them tries to manage his/her day (week, life) in a satisfying way. They go to work to earn money, they go hiking for their health and pleasure, they visit their relatives for pleasure or because they feel obliged to do so, they shop to cook a nice dinner at home, and so on. Since not all of this can be done at the same location, they have to travel, which produces traffic.

To plan an efficient day, many decisions have to be made by each person. He/she has to decide where to perform an activity, which transportation to choose to change location, when and in which order activities have to be performed, with whom an activity should be performed, and so on.

There are more decisions to make; some of them are made hours (days, months) in advance while others are made spontaneously as reactions to specific circumstances. Many decisions induce other decisions. As a result, it is of high importance to describe schedules for each individual in a simulation model, because it is the schedule and the decisions made by the person who adheres to this schedule that produce traffic.

This dissertation presents such a multi-agent simulation for transport planning. The work is embedded in the MATSim-T project (Multi-Agent Transport Simulation Toolkit) and focuses on the design and implementation of the system. On the basis of integrated (daily) individual demand optimization in MATSim-T, we will enlarge the system such that it provides flexible handling of a large variety of input data; extensibility of models and algorithms; a simple interface for new models and algorithms; (dis)aggregation for different spatial resolutions; robust interfaces to third party models, programs, and frameworks; unlimited number of individuals; and an easily usable interface to handle new input data elements.

Zusammenfassung

Die Mikrosimulation gewinnt derzeit immer mehr an Bedeutung in den Bereichen der Verkehrssimulationen, Verkehrsanalyse und Vorhersage. Die wichtigsten Gründe dafür sind:

- Verringerter Rechenaufwand und Speicherbedarf von grossen, mehrdimensionalen Wahrscheinlichkeitsmatrizen
- Variationsreichere Ausgabeoptionen, von aggregierten statistischen Analysen bis hin zu detaillierten Informationen über einzelne Individuen eines Szenarios.
- Explizites Modellieren des Entscheidungsfindungsprozesses jedes einzelnen Individuums.

Insbesondere der letzte Punkt ist sehr wichtig. Denn es sind nicht die Fahrzeuge, die den Verkehr produzieren, sondern es sind die Personen, die sie benutzen. Personen produzieren Verkehr nicht einfach zum Selbstzweck. Vielmehr versuchen sie, ihren Tag (ihre Woche, ihr Leben) auf die bestmögliche Art und Weise zu organisieren. So gehen sie beispielsweise zur Arbeit um Geld zu verdienen. Sie besuchen Freunde und Verwandte zu ihrem Vergnügen oder weil sie sich dazu verpflichtet fühlen. Sie gehen Einkaufen um ihr Abendessen vorbereiten zu können und vieles mehr. Weil viele dieser Aktivitäten nicht am gleichen Ort erledigt werden können, müssen sie reisen, und produzieren damit Verkehr.

Für eine erfolgreiche Tagesplanung muss eine Person viele verschiedene Entscheidungen fällen. Sie muss entscheiden, wo sie eine bestimmte Aktivität ausführt, welches Verkehrsmittel sie benutzt, um von einem Ort zum nächsten zu gelangen, in welcher Reihenfolge und um welche Zeit sie ihren Aktivitäten nachgeht, mit wem sie gewisse Aktivitäten gemeinsam plant, etc.

Darüber hinaus ist noch eine Reihe von weiteren Entscheidungen zu fällen. Manche davon sind spontan und kurzfristig und entstehen typischerweise aufgrund unvorhergesehener Umstände. Andere Entscheidungen bedürfen einer langen Vorlaufzeit und sorgfältiger Planung. Ein weiterer wichtiger Faktor ist, dass viele Entscheidungen andere Entscheidungsfindungsprozesse induzieren.

Daher ist es äusserst wichtig, zur Modellierung des Verkehrsverhaltens den gesamten Planungshorizont einer Person zu betrachten und diesen in einem Simulationsmodell umzusetzen. Denn es sind der gesamte Tagesablauf einer Person und die Entscheidungen, die dahinter stecken, die den Verkehr produzieren.

In dieser Doktorarbeit wird eine solche Agenten-Simulation für die Verkehrsplanung vorgestellt. Die Arbeit ist Teil des MATSim-T Forschungsprojektes (Multi-Agent Transport Simulation Toolkit) und konzentriert sich auf Design- und Implementationsfragen des Systems. Auf der Basis einer vollständig integrierten und individualisierten (täglichen) Nachfrageoptimierung implementiert in MATSim-T wird aufgezeigt, wie das System erweitert wurde, damit es auf verschiedenste Arten von Input-Daten angewendet werden kann. Ausserdem wird dargelegt, wie auf einfache Art und Weise neue, auf die Input Daten angepasste Modelle und Algorithmen hinzugefügt werden können. Weitere Funktionalitäten sind: Daten-Aggregationen und Dissagregationen für beliebige räumlichen Auflösungen, robuste Interfaces zu externen Modellen, Programmen und Systemen, Bearbeitung einer unbegrenzten Anzahl von Individuen und ein einfach benutzbares Interface um neue Datenschnittstellen zu implementieren.

Chapter 1

Introduction

Micro-simulation is becoming increasingly important in traffic simulation, traffic analysis, and traffic forecasting (Vovsha *et al.*, 2002; Bowman *et al.*, 1999; Bhat *et al.*, 2004). Some advantages over conventional models are:

- Computational savings in the calculation and storage of large multidimensional probability arrays.
- Larger range of output options, from overall statistics to information about each synthetic traveler in the simulation.
- Explicit modeling of the individuals' decision making processes.

The last point is important since it is not a vehicle which produces traffic; it is the person who drives the vehicle. Persons do not just produce traffic; instead each of them tries to manage his/her day (week, life) in a satisfying way. They go to work to earn money, they go hiking for their health and pleasure, they visit their relatives for pleasure or because they feel obliged to do so, they shop to cook a nice dinner at home, and so on. Since not all of this can be done at the same location, they have to travel, which produces traffic. To plan an efficient day, many decisions have to be made by each person:

- Which route should I take to get to work? - *Route choice decision*
- Which mode should I use to go to the lake? - *Mode choice decision*

- Should I drink another beer before going home? - *Activity duration choice decision*
- Should I go shopping near my home or at the mall? - *Location choice decision*
- When should I do sports today? - *Activity starting time choice decision*
- Should I go to visit my friend? - *Activity type choice decision*
- Who should I take along? - *Group composition decision*
- Should I go swimming before or after work? - *Activity chain decision*

There are more decisions to make; some of them are made hours (days, months) in advance while others are made spontaneously as reactions to specific circumstances. Many decisions induce other decisions. For example, if I am late for work, I must work longer, leaving no time to go shopping today; therefore, I need time tomorrow to do the shopping. This example shows the importance of describing schedules for each individual in a simulation model, because it is the schedule and the decisions made by the person who adheres to this schedule that produce traffic.

1.1 Requirements of Transport Planning

To describe each resident individually is in fact the desired target in transport planning. The reason for this is quite obvious: If one can synthetically generate and therefore monitor each and every person in a region of interest during a desired time period, one is able to extract any aggregated information about this scenario. Examples are:

- time dependent traffic volumes
- modal split
- time dependent location occupation
- activity chain distributions per population group

- activity duration distributions per activity type
- etc...

It is easy to see that the list of possible information which can be extracted using this idea is almost endless. But only monitoring what a person is actually doing does not give one any information about what the person intended to do. This raises the question of the *decision making process of an individual*. If one would have the complete knowledge about what a person plans to do and how he/she “computes” it, there would be no need to monitor the person anymore, because his actions are then only the result of his/her predictable decisions.

On the other hand, there is no access to all this information because of data privacy, imprecise or aggregated data, costs of the required surveys, and limitation of census data. Therefore, transport planners need models, programs and tools to answer this large variety of questions based on given—typically aggregated—data sources.

The following subsections summarize the goals of transport planning models and what kind of models are usually used to reach these goals.

1.1.1 Transport Planning Problem

Since humans want to be mobile, and transportation of people and goods is a necessary part of our economy, and since transportation itself has negative effects on humans (traffic congestion, safety issues, etc.) and the environment, the process of planning an effective transport system infrastructure involves finding a compromise between these positive and negative aspects. Since building and maintaining transport infrastructure is an expensive long term investment which influences typically a large metropolitan area containing millions of inhabitants, transport planning technologies should be capable of predicting long term impacts of such large-scale scenarios[1]. Even more, infrastructural changes in the system also change the individual needs of each resident in the scenario. Therefore, planning technologies should also reflect changes in the behavior of each individual in the area.

[1] A scenario describes a simplified representation of the region of interest. It defines the area and its constraints to the simulation model.

Even though planning is—at least partially—done to meet the needs of the travelers, people are selfish and are not concerned about the system itself. The travelers use the transport system for their own benefit—to move themselves, or to move goods for economic purposes—without (much) regard for other travelers using the system. The system offers only limited resources (road capacity, space, etc.), so the users end up competing for those resources. Individual travelers do not worry about the functioning of the system as a whole; they only care about the part they use. In fact, they cannot generally perceive any more of it than what they experience. As such, they are not generally aware of the impact their own decisions have on the rest of the system or the other travelers. It is up to the planners, and thus, planning technology, to consider the entire system, how the collective impact of the individual decisions made by its users affect the system as a whole, and how those users will react to the system. It makes sense, then, for the planning technology to model the behavior of individual travelers.

1.1.2 The Four Step Process

The traditional technology for transport planning is the four step process (e.g., Sheffi, 1985; Ortúzar and Willumsen, 2001). This process determines demand on a transport network in terms of the trips taken by travelers between various "zones" of the region being simulated. These trips are then used to find an *equilibrium* solution to the flow of vehicles on the links (i.e. roads) of the network. The four steps as shown in Figure 1.1 are:

1. *Trip generation*: For each possible trip origin, the number of outgoing trips is determined. Similarly, for each possible trip destination, the number of incoming trips is determined.

2. *Trip distribution*: This step connects origins and destinations; that is, for each origin it is determined which fraction of its outgoing trips go to which destination. The result of the trip distribution is a so-called origin-destination (OD) matrix, which specifies the number of trips that go from each origin to each destination.

3. *Mode choice*: These trips can be made by different means of trans-

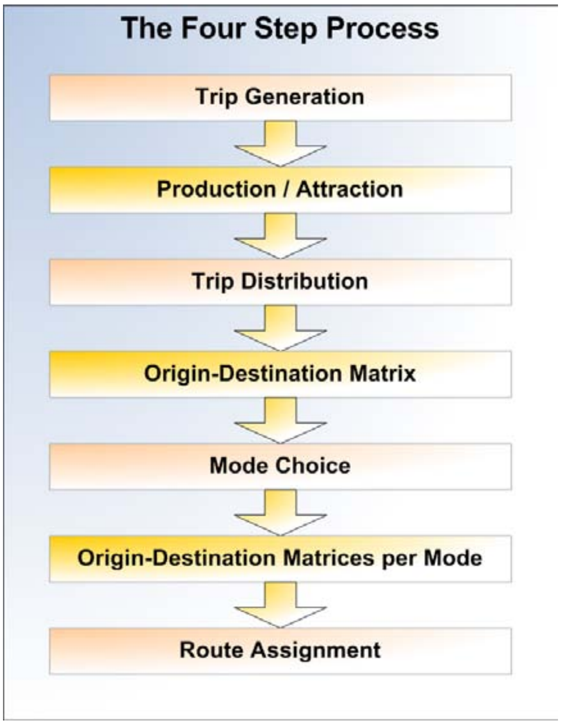


Figure 1.1: The four step process

portation, e.g. by walking, driving, taking the bus, etc. This step computes that choice.

4. *Route assignment*: Each *car* trip is assigned to a path on the network. These paths are sensitive to congestion. Typically, a user equilibrium (UE) is searched for here: All paths used for a given OD pair should have the same travel time; and no unused path should be faster (see also Wardrop, 1952, pp. 325-378).

The static "route assignment" step, under certain conditions, yields a solution to the link flows which is *unique*, meaning one gets the same answer no matter which (mathematically correct) method is used to obtain the result. That is an important feature because it makes it easy to compare the results of different algorithms and software packages, and simplifies the interpretation of various modeling scenarios.

However, the four step process is too simple, and does not offer results that are realistic enough for most "modern" transport planning problems. The four step process is missing:

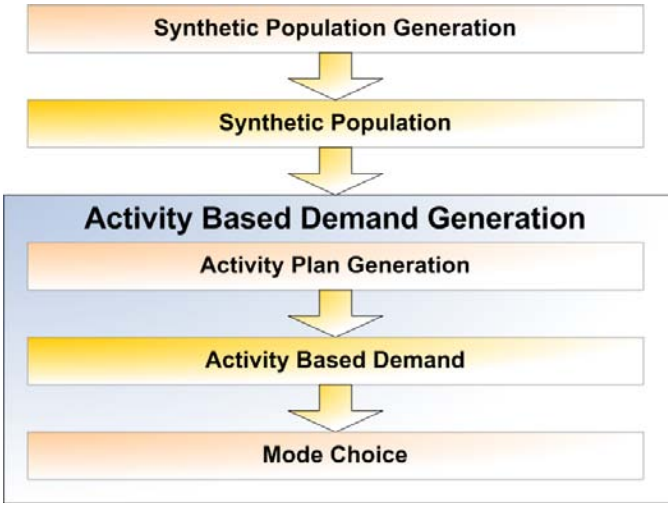


Figure 1.2: Activity based demand generation process

1. *Disaggregation* by individual travelers. The four step process uses traffic streams, without discerning what individual travelers are doing. Distinguishing between travelers in a transport planning application allows planners to connect travelers' decisions (e.g. mode choice) with their specific demographic data, making their choices more realistic and behaviorally motivated.

2. *Temporal dynamics.* The four step process assumes that the vehicles ("particles") in the traffic streams are in a steady state, meaning the stream flows themselves are static, i.e. time independent. Therefore, it ignores time-dependent effects, such as peak traffic spreading; congestion spillback; or vehicle emissions, which depend on engine temperature, which in turn depends on how long the cars have been driving.

Methods to overcome these shortcomings will be discussed in the following three sections.

1.1.3 Activity-Based Demand Generation

Activity-based demand generation differs from the above described four step process since the traffic demand is modeled on individual level.

Travelers in a transport planning system are disaggregated according to the following steps (see Figure 1.2):

1. Generate a *synthetic population* by creating a "random realization" of the census data for the region being simulated. This means creating a set of virtual people to inhabit the region that, while not an exact match for the real-world population, maintains the demographic "structure" of the actual population. In other words, a census taken from the synthetic population would, within statistical limits, return the original census.

   A typical synthetic population is composed of spatially located households which possess certain attributes, such as a street address, household income, or car ownership (Beckman *et al.*, 1996). These households are populated with individuals, who possess additional attributes, such as gender and age.

2. Generate, for each individual of the synthetic population, a complete *daily activity schedule*. The word "activity" refers to actions such as "being at home," "being at school," "working," "shopping," etc. An individual's activity schedule contains the pattern of activities he/she wishes to perform along with the location of each activity (Vaughn *et al.*, 1997). It also contains timing information, such as when activities begin and end. Having activities at different locations motivates individuals to use the transport network, in order to travel from one activity location to the next.

3. Choose, for each individual of the synthetic population, a *mode* of transport for each trip, tour or journey taken by that individual between a pair of activities. This step is similar to the "mode choice" step of the four step process, but does more than distributing the trips based only on the mode characteristics, as is done with the four step process. Here the choice of mode can also be based on the demographic attributes of the traveler that performs the trip.

Hensher (2001), Gärling and Axhausen (2004) and others discuss the concept of activity-based demand generation, while Bowman *et al.* (1999); Vovsha *et al.* (2002); Jonnalagadda *et al.* (2001); Bhat *et al.* (2004); Axhausen (1990b), among others, implement it. Also, the above process steps can vary. For example, the mode of transportation can be chosen before the location choice process, sometimes after it and sometimes simultaneously (examples are Lohse *et al.*, 1997; Kutter, 1983).

Once the network demand (i.e. the trips) has been calculated, all that remains of the four step process is the “route assignment” step. However, the resulting travel demand is now time-dependent. This does not connect well to the time-independent and steady-state route assignment process. Therefore, either both shortcomings of the four step process need to be overcome simultaneously, or one has to make the traffic assignment dynamic instead of static.

1.1.4 Dynamic Traffic Assignment

Dynamic traffic assignment has emerged over the last 20 years (e.g., Kaufman *et al.*, 1991; Astarita *et al.*, 2001; Friedrich *et al.*, 2000) as a technique to go beyond static assignment and produce time-dependent link volumes. Given a time-dependent demand (as described in Section 1.1.3) and a model of the traffic dynamics, which moves vehicles along the network, the dynamic assignment attempts to find a route for each trip of each traveler such that no traveler would be better off by selecting a different route for any given trip. This is just the Nash Equilibrium (Nash, 1951) statement for the dynamic problem.

Although some of the theory of dynamic traffic assignment is known (Bottom, 2000; Cantarella and Cascetta, 1995), it is not as well understood as the four step process, and has fewer mathematically proven properties. In particular, there is no guaranteed unique solution for the dynamics of traffic that include spillback (also called physical queues). For example, Daganzo (1998) shows that one can find more than one Nash Equilibrium solution to the same OD matrix and the same network. This makes it more difficult to compare different techniques and results to one another.

Since an analytical approach to the dynamic traffic assignment is both more difficult and less useful than for static assignment, it makes sense to look at another approach, such as simulation, which is a well known technique where a dynamic model is implemented on a computer, and run forward in time. The simplest version of a simulation for transport planning would involve a representation of the roads, and a way to move

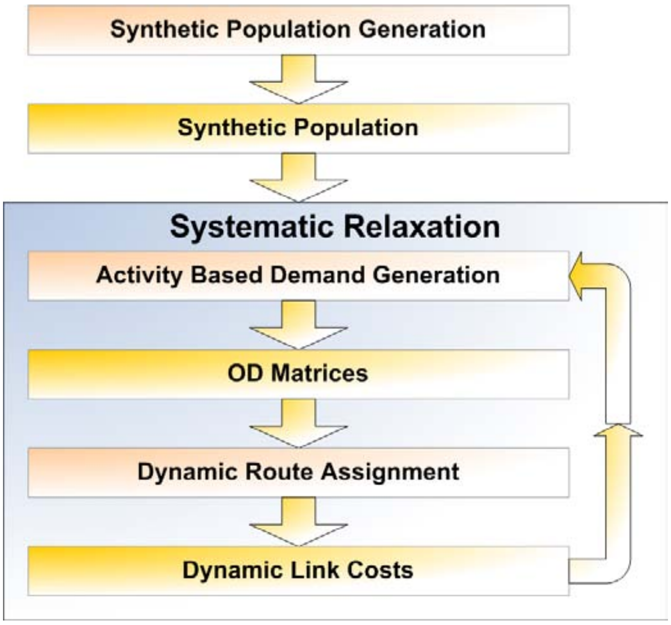


Figure 1.3: Systematic relaxation

traffic forward along the links.

Systematic relaxation (Kaufman *et al.*, 1991; Nagel, 1995; Bottom, 2000) is a common element of simulation-based route assignment, and is implemented by some form of the following procedure (see also Figure 1.3):

1. Start with an initial "guess" for the routes.

2. Perform *network loading* by executing all routes simultaneously in a traffic flow simulation.

3. Re-adjust some or all of the routes using the results of the network loading.

4. Goto 2.

This procedure is sometimes called *feedback*, since it models the feedback of congestion to the route planning. The method is somewhat similar to the Frank-Wolfe-algorithm in static assignment (Frank and Wolfe, 1956), or in more general terms to a standard relaxation technique in numerical analysis.

Many variations of the dynamic traffic assignment are possible, such as varying the fraction of routes which are replanned (Rickert, 1998), using a probabilistic route choice based on, for example, multinomial logit or probit (DynaMIT, 2006), or using different network loading algorithms (Astarita *et al.*, 2001).

1.1.5 Combination of Activity-Based Demand Generation and Dynamic Traffic Assignment

1.1.5.1 Origin-Destination Matrices

So far, it has been shown that demand generation can be disaggregated and made more realistic by basing it on activities, and traffic assignment can be made more realistic by making it time-dependent and then using simulation for the network loading. These were discussed as separate changes, so it is natural to assume that they would be designed to be backwards compatible to the four step process, which means that the activity based approach would produce OD matrices as output, and the dynamic assignment process would take them as input. This also means that activity-based demand could be fed into a traditional static assignment, and the dynamic assignment process could take its input from the traditional demand generation.

However, these OD matrices are usually time dependent, while traditional static assignment works with a single, time-independent OD matrix. Conversely, traditional demand generation produces a single, time-independent OD matrix, but the dynamic assignment process needs time dependent OD matrices to make sense. Therefore, although the use of OD matrices superficially maintains backward compatibility, this backward compatibility cannot be used in any meaningful study. In order to obtain meaningful results, *static* demand generation needs to be fed into a *static* assignment, or *dynamic* demand generation needs to be fed into a *dynamic* assignment.

METROPOLIS (de Palma and Marchal, 2002) approaches this problem by accepting a static OD matrix, but generating a time-dependent solution internally. In consequence, this allows one to feed a *static* OD matrix into a *dynamic* assignment. It does not, however, offer a better solution if the demand generation is already dynamic.

So, now it has been shown that time-*dependent* OD matrices should drive the dynamic assignment. Such matrices are usually derived from "historical" data, i.e. from information that the transport planners of a given region have used for many years. They are then corrected against real-time counts; that is, the OD matrices are modified such that the resulting assignment matches the real-time counts as well as possible (Antoniou *et al.*, 1997). The advantage of this method is that it is only a small step away from the data needs for static traffic assignment.

The combination of activity-based demand with dynamic assignment through an OD matrix has several disadvantages, though. First, it gives up the disaggregation of travelers that was gained by the activity-based approach. Second, it gives up the connection between individuals and their performance in the traffic flow simulation. Any iterative feedback from the traffic system performance could only be based on aggregate measures, such as link travel times, not on individual performance of the travelers. However, an individual's decisions can depend on its attributes: For example, the decision to use a toll road can depend on income; a person planning to catch an airplane may prefer to take a road with lower variability; etc.

Finally, activity-based demand generation produces demand consisting of "chains" of several trips, made between the scheduled activities. If a traveler gets delayed on one trip, this may have impacts on trips later in the day. OD matrices drop time dependencies between scheduled activity chains, making it possible for a person to complete an activity even before he/she has arrived at the destination where the activity will be conducted.

1.1.5.2 Fully Agent-Based Approach

It makes sense, therefore, to bypass OD matrices completely and to feed the complete information from the activity-based demand generation into the assignment process (see Figure 1.4). This means that throughout the whole process, the travelers are maintained as individual entities with individual attributes, and make individual decisions based on these attributes. In other words, the *multi-agent* approach (Ferber, 1999) is applied to the simulation system. In fact, there has already been much work related to using agents in transport (e.g., Wiedemann, 1974; Ar-

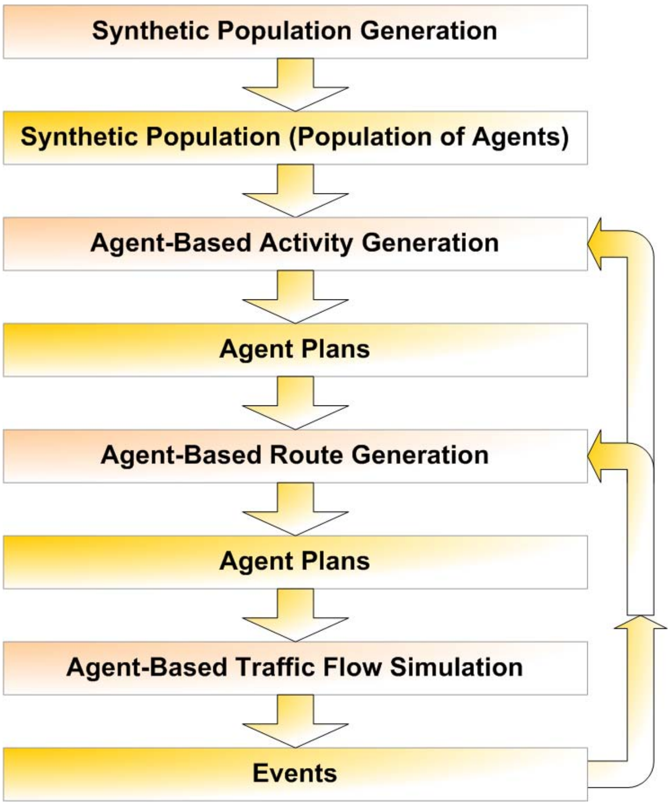

Synthetic Population Generation
Synthetic Population (Population of Agents)
Agent-Based Activity Generation
Agent Plans
Agent-Based Route Generation
Agent Plans
Agent-Based Traffic Flow Simulation
Events

Figure 1.4: Fully agent based approach of dynamic traffic assignment

entze *et al.*, 2000; Wahle *et al.*, 2002).

This may seem like a small change, but in practice it is not, since many implementations store information that should belong to the agent in other ways. A typical example is how route plans are encoded: Most implementations let each individual traveler know only its destination. The path to that destination is then found via “signposts” at each intersection; that is, at each intersection (node) there is such a sign for each possible destination. Such an encoding makes, for example, individual route preferences difficult to model since all travelers of a certain class to the same destination have to take the same set of paths.

This dissertation concentrates on the multi-agent simulation approach as an improvement over the complete four step process. The main differences are:

- The dynamic traffic assignment is made completely agent-based, as discussed above. In particular, it is capable of feeding back agent-based (i.e. individualized) information, not only link-based information.

- The activity-based demand generation is included into the feedback process. Historically, systematic feedback is mostly between the route generation and the network loading; feedback to the demand generation was often done manually by the analysts. However, it has been said for a long time (e.g., Loudon *et al.*, 1997) that this process should be automated.

In particular, the traffic flow simulation assumes the role of a realistic representation of the physical system, including explicit modeling of persons walking to the bus stop, or of a bus being stuck in traffic. Also, in terms of analysis such a system offers enormous advantages. It is, for example, possible to obtain the demographic characteristics of all drivers being stuck in a particular traffic jam. It is also possible to make each traveler react individually to exactly the conditions that this traveler has experienced, rather than to aggregate conditions.

To summarize, the fully agent based approach presents each individual as an autonomous entity during the complete iterative demand modeling and assignment process. Figure 1.5 presents the same idea as Figure 1.4 but highlighting the agent database as the core of the whole system.

1.2 Existing Simulations for Transport Planning

Activity-based demand generation has already been implemented for a number of packages (e.g., Vovsha *et al.*, 2002; Bowman *et al.*, 1999; Bhat *et al.*, 2004; PTV, 2006; Pendyala, 2004; Arentze *et al.*, 2000). Yet, their output is typically expressed in terms of (time-dependent) origin-destination matrices, to be fed into dynamic traffic assignment models.

An important exception is TRANSIMS (TRANSIMS, 2006), which generates individual activity schedules as input to the dynamic assignment. TRANSIMS was difficult to obtain outside the U.S. for a couple of years, thus motivating the toolkit presented here as an alternative. In the

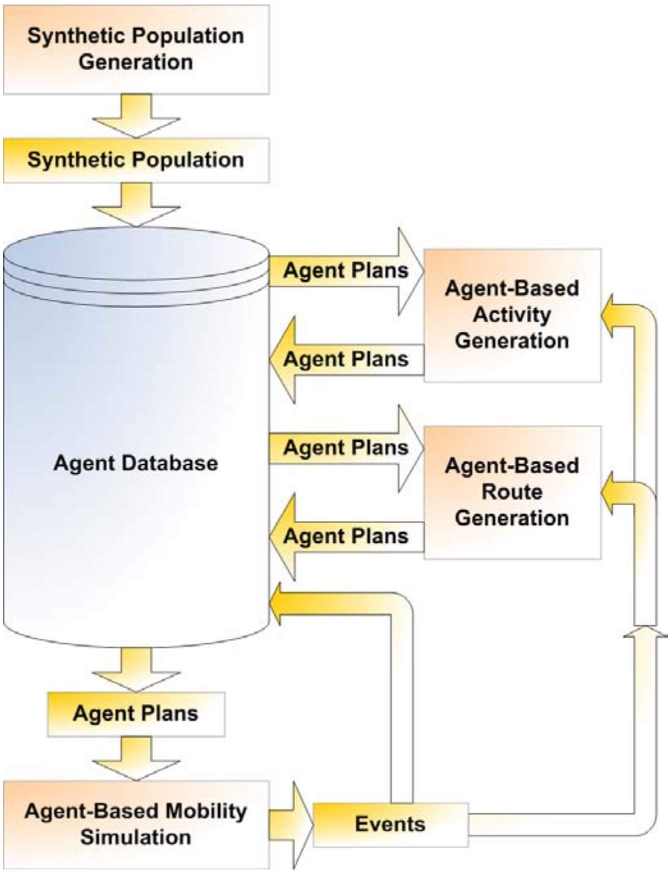

Synthetic Population Generation
Synthetic Population
Agent Database
Agent Plans
Agent Plans
Agent-Based Activity Generation
Agent Plans
Agent Plans
Agent-Based Route Generation
Agent Plans
Agent-Based Mobility Simulation
Events

Figure 1.5: Agent Database

meantime, TRANSIMS has become open source, but the implementation of this dissertation goes beyond TRANSIMS in the following aspects:

- TRANSIMS uses "flat" file formats between modules, while the data format we use in here (see A.1) is defined by the more powerful hierarchical XML format (W3C, 2006).

- With the XML format, it is always possible to do all information exchange between modules with the same file format and the same DTD[2], varying only the detail level of the information included.

[2] Document Type Definition (DTD), defined slightly differently within the XML specifications, is one of several XML schema languages, and is also the term used to describe a document or portion thereof that is authored in the DTD language. A DTD is primarily used for the expression of a schema via a set of declarations that conform to a particular markup syntax and that describe a class, or type of XML documents, in terms of constraints on the structure of those documents. A DTD may also

This means that arbitrary combinations of partial demand generation modules can be used.

- We keep track of multiple schedules per agent generically; for TRANSIMS, this would need considerable changes in the implementation.

- The traffic flow simulation used in this dissertation (see also Cetin, 2005, although simplified when compared to the TRANSIMS traffic flow simulation), runs considerably faster, thus allowing meaningful runs in days instead of weeks.

- In contrast to TRANSIMS, the dynamic assignment process presented in here keeps track of the activity chains' consistency along the time axis—travelers have to spend minimum amounts of time at activities before they can proceed.

In contrast to TRANSIMS, this current work is flexible and universal in terms of the data input requirements (tested with data available in Switzerland and Germany). All other packages besides TRANSIMS would not only have to overcome the data requirements problem, but, to our knowledge, would also need source code changes to allow for traveler-based output.

Other approaches are the agent-based land-use models, for example URBANSIM (Waddell *et al.*, 2003), ILUTE (Salvini and Miller, 2005), or the models by Abraham/Hunt (Hunt *et al.*, 2000). These models face difficulties similar to those in the approach described in here, particularly the need to assemble a consistent agent-based view of the world from diverse data sources (Abraham *et al.*, 2005). In the longer run, it would be useful to have a "plug-and-play" approach between these different models, i.e. where modules of different modeling systems could be interchanged and coupled in various ways. In fact, at least one such effort is under way (Waddell *et al.*, 2005). However, the software engineering challenges for such a project, despite considerable progress in the last decade, are still enormous, particularly for large-scale scenarios (Nagel and Marchal, 2006). Therefore, for the time being, it seems more productive to make progress in smaller steps, which in our case means the

declare constructs that are not always required to establish document structure, but that may affect the interpretation of some documents.

implementation of a general framework demonstrating the overall feasibility and highlighting potential problems.

1.3 Structure of This Dissertation

The following chapter presents a "big picture"—the vision—how a large-scale multi-agent transport simulation system can be built up to fulfill various needs of transport planners. The first implementation is described next. The experiences with it helped to identify various problems and also ways in which they can be resolved.

Chapter 4 then presents two different real-world scenarios which will be used as the data basis for the succeeding chapters. The first—medium size—Kanton Zurich scenario consists of about half a million simulated agents while the large-scale scenario of Berlin-Brandenburg consists of more than three million agents generated on a substantially different input basis.

Chapters 5 and 6 show the evolution from the initial state of the project presented in Section 2.3 to the fully integrated agent-based demand modeling toolkit.

The final discussion highlights how the system has been evolved. The outlook of this dissertation will show the future of this project and which parts of it are already in progress.

Chapter 2

Objectives and Preliminary Implementation

The preceding discussion motivates the use of agent-based simulation for socio-economic problems, transportation in particular. This chapter starts with a formulation of a vision of where the power lies in agent-based micro-simulation. This big picture will be scaled down so that it is still appropriate for transportation issues, followed by a description of the first implementation which is joint work with Bryan Raney published as Balmer *et al.* (2004a,b, 2005b); Raney *et al.* (2003); Raney and Nagel (2005). It simultaneously marks the points discussed in the thesis of Raney (2005) and the starting point of the thesis presented in here.

2.1 Drawing the Big Picture

The basic idea behind a fully agent-based micro-simulation is to describe the world in a more realistic way than aggregated methods.

The world we know is defined by its *physics*. If someone drops an apple it falls to the ground, since the law of gravity take affect. A *person* does not go to the grocery store to buy this apple because some higher force commands him/her to do so, he/she *decides* to do that (i.e. the refrigerator is empty and he/she is hungry). The mode he/she uses to go to the store is also his/her decision based on his/her *environment* and his/her personal *preferences* (i.e. he/she takes the car because the store is 1 km away and he/she is too lazy to walk). On the other hand, there is another shop just around the corner which sells apples but he/she does not *know* that. Last but not least, he/she does all that because his/her *target* is to have a happy life. Therefore, he/she does not want to starve.

This little story shows us what is in principle needed to design a real world simulation:

- We need to define a consistent world defined by unbreakable rules (the physics).
- We need to build an environment in which the agents can act. The environment must respect the rules of the world (i.e. buildings, streets, traffic rules, and so on).
- We need to create agents, representing the individuals of the population, which live in this environment.
- Each agent has some knowledge about the environment.
- Each agent has a target it wants to reach.
- Each agent has some preferred ways to act in the world.
- And each agent is able to decide by itself in which way it wants to reach its target.

As one can see, some of the above aspects seem beyond reach. This is in fact true, otherwise we would be able to rebuild a “Matrix”[1]. But if we are able to reduce the complexity to the level of transport planning issues and therefore simplify the world, the physics, the targets and the agents such that the system is still able to answer questions on this topic, then we should be able to cope with the complexity.

2.2 The Big Picture of MATSim-T

The open-source *Multi-Agent Transport Simulation Toolkit* (MATSim-T, 2006) is presented as a modular and flexible developer platform for

[1] “The Matrix,” directed by the Wachowski brothers

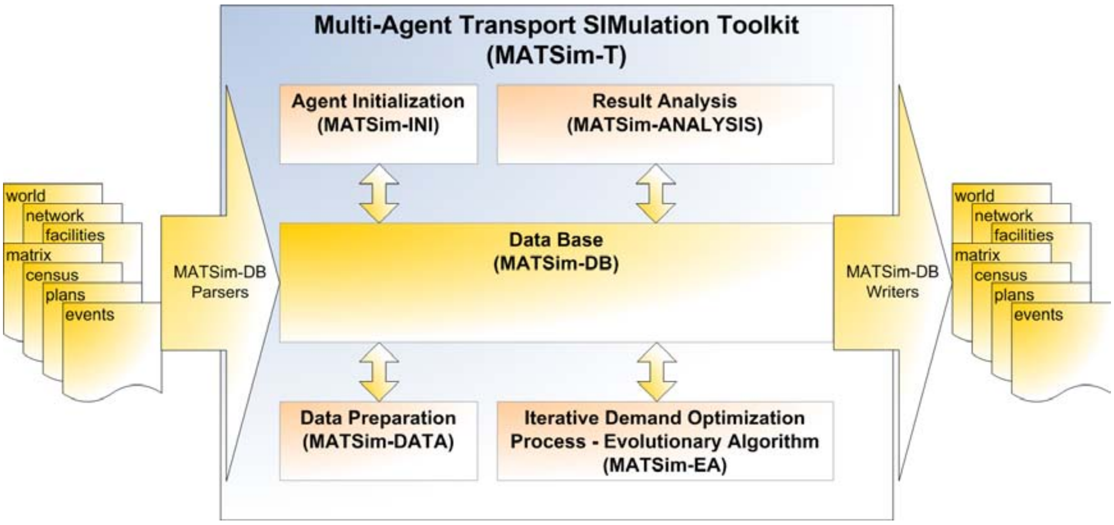


Figure 2.1: Schematic overview of MATSim-T

transport planning software. Figure 2.1 shows a schematic overview of the toolkit. The core is the database (MATSim-DB). It stores the needed information into the memory of the computer. Tests with “off-the-shelf” relational databases like MySQL (Axmark *et al.*, 2006) showed that such databases lead to slow performance accessing specific data points, because of the highly variable amount of information stored for each single agent in the system (see also Raney, 2005, chap. 5 for more details).

Well defined XML (W3C, 2006) data interfaces are provided, which the MATSim-Parsers use to parse the data. The MATSim-Writers allow one to dump any stored data sets into XML data files again, at any point in time during the use of the toolkit.

With the parsers, the database and the writers there is a clean construct to move data from files into memory and vice versa. But there is no functionality defined to adapt, change, enrich or delete data points.

Since the given data structure of MATSim-DB provides defined access to each data point programmers are now able to “plug” an arbitrary number of models, programs and algorithms into the toolkit. They can change, add or delete any data point of MATSim-DB as long as they respect the necessary connections between them.

Each algorithm that a programmer or a transport planner adds to the system fulfills a specific purpose. For example, one algorithm checks if given geographical zones cover the whole area of interest; the next algorithm synthetically generates individuals and places them into the given zones. A third algorithm tries to find the optimal work places for each individual of the system; a fourth algorithm then calculates the traffic produced on the streets defined by a network; and a last one compares the result of the previous one to enterprize census data.

This example indicates the ability to split up the demand modeling process into small pieces. As one probably recognizes, the above mentioned example implicitly defines a sequence in which each of the described algorithms will be used. But the toolkit does not forbid other sequences, e.g. one could also rerun the last three algorithms to minimize the relative error to census data.

This idea of "plug-and-play" gives the user of the toolkit complete freedom to generate demand for a scenario. Therefore, one is no longer forced to follow the 4-step process (see Section 1.1.2) of transport planning in its predefined way. It is possible to iteratively combine any steps.

Nevertheless, also logical parts of MATSim-T are defined to classify the algorithms clearly. As indicated in Figure 2.1 the functional part of MATSim-T are structured into four groups:

MATSim-DATA This part is responsible for checking, fusing and consolidating the various input data such that the database is fully consistent.

MATSim-INI Depending on the level of detail, this part is responsible for generating or enriching the individuals as agents. The resulting population consists of a set of agents with their attribute values. Furthermore, it is also possible to define a partial mental map (the knowledge of each agent) and the initial demand of each agent. Therefore, this part stands for activity-based demand generation as shown in Section 1.1.3. But since MATSim-T leaves complete freedom to the user, the synthetic population generation process could be included there.

MATSim-EA This part is the algorithmic core of the toolkit. It iteratively optimizes each individual's demand based on the following idea: (i) Based on global information of the scenario, each agent plans its day individually. (ii) All agents' plans[2] are then executed in a physical environment. (iii) Each executed plan then gets a score calculated by a scoring function (also called utility function). It reflects how successful the execution of the plan was. (iv) According to the score and some global information received from the last execution, the agents change their plans again. Then the next plans execution process starts, and so on, until a stable state is reached.

MATSim-EA stands for *Evolutionary Algorithm* (Bäck, 1996), since the iterative process described above is based on that concept. A detailed description of MATSim-EA is presented in Section 2.4.

MATSim-ANALYSIS Typically (but not necessarily) at the end of a run of MATSim-EA we analyze the resulting optimized demand and compare it to the input data describing the scenario. Therefore, this part holds a collection of analysis, validation, calibration and visualization tools.

As mentioned above, this order is a typical one, but not a mandatory one. If desired, one can redo parts again according to the analysis, which is in fact another loop of the system.

The following section will present the first implementation of the above described vision of a large-scale multi-agent transport simulation. Furthermore, it shows the use of MATSim-T by introducing a real world scenario placed in the greater Zurich area.

2.3 Preliminary Implementation of MATSim-T

The preliminary state of MATSim-T concentrates on the iterative demand optimization process (MATSim-EA). In fact, the other three parts of MATSim-T are completely separated from the toolkit, implemented as simple pre- and post-processes. As a precondition, it is therefore expected to receive the initial individual demand per agent of a scenario.

[2] The word "plan" will be used now instead of "schedule" to indicate, that it describes what an agent *plans* to do during its day which can differ from what the agent actually *executes* during the day.

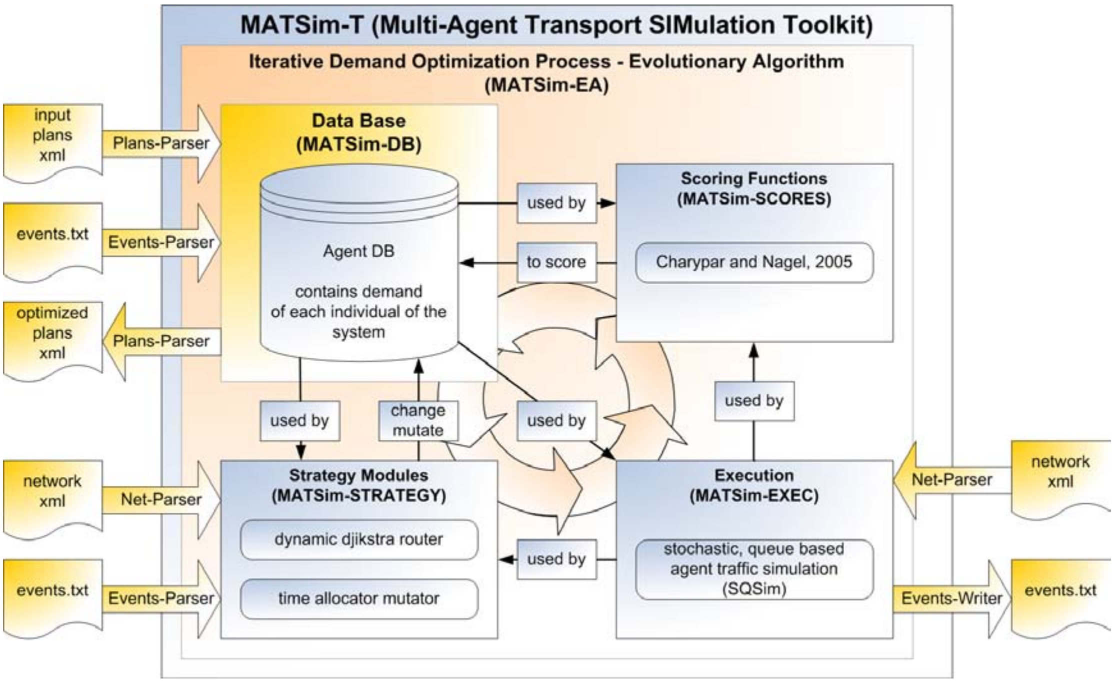


Figure 2.2: Preliminary State of MATSim-T

Even more, it needs to be guaranteed that the demand description is based on a defined network. Based on these two input sources, MATSim-EA now tries to iteratively optimize the individual demand.

Figure 2.2 shows an overview of the optimization process. Again, the main idea is to separate the cycle into modules. Each module is responsible for solving one specific step in the process. MATSim-EA is split up into the following parts:

network.xml The description of a (street) network of the scenario. It is given in XML as a strongly connected geo-coded digraph (see Balmer *et al.*, 2005a, page 8) with a set of required attributes for each link (see Appendix A.2 for a detailed data representation).

plans.xml Defines and describes each individual of the scenario, including their initial demand for a specified time period. An example of the plans file is shown in Figure A.1 of Appendix A.1.

MATSim-DB Consists of the *agent database*. It stores the initial `plans.xml` information in memory. Since the plans of an agent can be modified during an iteration process, it is possible to keep more than one plan per agent in the agent database. It has some additional functionalities, too, which will be explained in the following section in more detail.

MATSim-EXEC The execution is the actual microscopic traffic simulation. Here, the agents of the system interact with each other. The stochastic, queue based, agent traffic simulation *(SQSim)* as described in Cetin *et al.* (2003) simulates the trips agents take during the defined time period. With it, congestion and tailback can be simulated.

events.txt The outcome of MATSim-EXEC gives precise information about where an agent is (for each second) and what it is doing (i.e. driving, arriving at a location, entering a street segment, and so on.). These *events* are stored in the `events.txt` file (see also Appendix B.1). It can be used as an information pool for the other modules in MATSim-EA.

MATSim-SCORES After MATSim-EXEC finishes, the resulting performance of each agent typically differs from what it has expected (as stored in the agent database). Therefore, the quality of each executed plan needs to be calculated. This is done by using a *scoring function* (in evolutionary computation also called "fitness function"). The higher the score of an executed plan, the better it is for an agent. Detailed information about the scoring function is shown in the following section.

MATSim-STRATEGY The strategy part of MATSim-EA can consist of an arbitrary number of *re-planning modules*. The main concept is to take a plan of an agent from the agent database and modify some parts of that plan. In this case, the implementation of the *dynamic Dijkstra router* changes the routes between each pair of activities (the ASCII data of the `<route>` tag of the plan as shown in Figure A.1) while the implementation of the *time allocation mutator* mutates the departure times (`dep_time` of Figure A.1) and the durations (`dur` of Figure A.1).

One probably now recognizes the concept of evolutionary algorithms as typically used in computer science (e.g., Holland, 1992; Palmer *et al.*, 1994). The idea behind this optimization process is to create a population[3] of—let us say—objects. These objects are then assessed against a fitness function (the objective function of the optimization process). The objects which fulfill the fitness function the best (which get the highest score) survive while the others die (also called the "survival of the fittest"). With the remaining objects a new population by mutation and interbreeding (also called "crossover") of the survivors will be generated. These descendants are then the new population which are again compared to the fitness function, and so on.

In this case, the plans of an agent stored in the database are the "population" of objects. By executing (with MATSim-EXEC) and scoring the outcome (MATSim-SCORES) the information about the "fitness" of the executed plan will be given. The database then compares the whole set of plans of each agent against this executed one, deletes the one with the worst score and keeps the others (the survivors). With the modules of MATSim-STRATEGY, one of the survivors will then be changed to create a new plan (mutation and/or crossover). This one again is executed and scored, and so on. As one can see, this is an adaptation of the concept of evolutionary computation, but still respects its concept.

The following section describes MATSim-EA in more detail, followed by a real world scenario which describes the greater area of Zurich, Switzerland. This scenario is also described in Balmer *et al.* (2005b).

2.4 Simulation Structure of MATSim-EA

2.4.1 Overview

As pointed out before, MATSim-EA is constructed around the notion of agents that make independent decisions about their actions. Each traveler of the real system is modeled as an individual agent in the simulation. The overall approach consists of three important pieces:

1. Each agent independently generates a plan, which encodes its in-

[3] In transport planning the population describes all inhabitants of the given scenario while in evolutionary computation the population means the "objects" which will be measured against the fitness function.

tentions during a certain time period, typically a day. A plan contains the itinerary of activities the agent wishes to perform during the day, plus the trips the agent must take to travel between activities. An agent's plan details the order, type, location, duration and other time constraints of each activity, and the mode, route and expected departure and travel times of each leg (see also Figure A.1).

2. All agents' plans are simultaneously executed in the simulation of the physical system (MATSim-EXEC). It is a stochastic, queue-based agent traffic simulation (SQSim) as presented in Cetin (2005).

3. MATSim-STRATEGY is the mechanism that allows agents to learn, implemented as the two indicated modules shown in Figure 2.2. In this implementation, the system iterates between plan generation and traffic flow simulation. The system remembers several plans for each agent, and scores the performance of each plan. Agents normally choose the plan with the highest score, sometimes re-evaluate plans with bad scores, and sometimes obtain new plans.

In the preliminary implementation the focus lies on "home" and "work" as the only activities, and "car" as the only mode. The system does not distinguish between a trip (between two activities) and a stage (a part of a trip which uses exactly one mode), since the "mode change" can also be defined as an activity (which has a specified location, i.e. a train station). Each of the details described in the plan, such as activity duration, is a decision that must be made by the agent. These decisions are mutually dependent, but the decisions made by one agent are independent of those made by another. The task of generating a plan into sets is divided into closely related decisions, and each set is assigned to a separate module. An agent strings together calls to various modules in order to build up a complete plan. To support this "stringing," the input to a given module is a (possibly incomplete) plan, and the output is a plan with some of the decisions updated. Possible modules to be implemented in MATSim-STRATEGY are:

Activity chain generator modules Decides which activities an agent actually wishes to perform during the day, and in what order. This module is not used, but a fixed “home-work-home” pattern for all agents is defined.

Activity location generator modules Determines where the agent will perform a particular activity. Also, this module is not used, but a fixed location for each agent’s “home” and “work” activity is given.

Activity time allocation modules Determines the timing attributes the agent will utilize for each activity in a plan. Activities have two possible timing attributes: “activity duration” and “activity end time.” After starting an activity, an agent performs the activity either for the length of “duration,” or until the “activity end time,” whichever comes first. Activities cannot overlap in time.

Router modules Determines which route and which mode the agent chooses for each trip leg that connects activities at different locations.

A special feature of this approach is that users can choose any number and type of these modules as long as they generate some information that contributes to a plan. For that reason, it is easy to combine for example activity and mode choice into a single module or to add residential or workplace choice. MATSim-EA at this state will employ two modules only: “time allocation mutator” and “dynamic dijkstra router.”

Once the agent’s plan has been constructed, it can be fed into the SQSim. This module executes all agents’ plans simultaneously on the network, allowing agents to interact with one another, and provides output as events describing what happened to the agents during the execution of their plans. The modules produce dependencies. The outcome of SQSim (e.g., congestion) depends on the planning decisions made by the MATSim-STRATEGY modules.

However, those modules can base their decisions on the output of the traffic flow simulation (e.g. knowledge of congestion). This creates an interdependency (“chicken and egg”) problem between the strategy modules and the SQSim. We need these modules to be consistent with one another, and therefore we use feedback in MATSim-EA. This involves an iteration cycle which runs the SQSim with specific plans for the agents, then uses the time allocation mutator and the dynamic dijkstra router to update the plans, and these changed plans are again fed into the SQSim, etc., until consistency between modules is reached.

The feedback cycle is controlled by the agent database, which also keeps track of multiple plans generated by each agent, allowing agents to reuse those plans at will. The repetition of the iteration cycle coupled with the agent database enables the agents to learn how to improve their plans over many iterations. The following sections describes the modules in more detail.

2.4.2 Time Allocation Mutator

This module is called to change the timing of an agent's plan. Here, a simple approach is used which applies a random mutation to the duration and end time of an agent's activities. More precisely, for the first activity, the activity end time is the only attribute that is specified and thus mutated, while for all other activities, the duration is what is specified and mutated. For each such attribute of each activity in an agent's plan, this module picks a random time from the uniform distribution $[-30min, +30min]$ and adds it to the attribute. Any negative duration is reset to zero; any first activity end time before 00:00 AM is reset to 00:00 AM. The entire plan is returned to the agent, with only the time attributes modified.

Although this approach is not sophisticated, it is sufficient to obtain useful results. This is consistent with our overall assumption that, to a certain extent, simple modules can be used in conjunction with a large number of learning iterations (e.g., Nagel *et al.*, 2004). Since each module is implemented as a "plug-in," this module can be replaced by an enhanced implementation if desired.

2.4.3 Dynamic Dijkstra Router

The dynamic dijkstra router is implemented as—the name says it already—a time dependent Dijkstra algorithm. It first calculates link travel times from the events output of the previous traffic flow simulation. The link travel times are aggregated into 15 minute time bins, and then used as the costs of the links in the network graph. Apart from relatively small but essential technical details, the implementation of such an algorithm is straightforward (Jacob *et al.*, 1999). With the knowledge about activity patterns, it computes the fastest path from each activity to the next one in the sequence as a function in time. It returns the entire plan, complete with updated paths, to be used by the agents for the next run of the stochastic, queue based agent traffic simulation.

2.4.4 Stochastic, Queue Based Agent Traffic Simulation (SQSim)

The stochastic, queue based agent traffic simulation simulates the physical world. It is implemented as a queue simulation (Gawron, 1998; Cetin *et al.*, 2003), which means that each street (link) is represented as a FIFO (first-in first-out) queue with three restrictions. First, each agent has to remain for a certain time on the link, corresponding to the free speed travel time. Second, a link storage capacity is defined which limits the number of agents on the link. If this capacity has been reached, no more agents can enter this link. Third, there is a flow capacity, which limits the number of vehicles that can leave the link in any given time step.

Even though this structure is indeed simple, it produces traffic as expected and it can run directly using the data typically available for transport planning purposes. On the other hand, there are some limitations compared to reality, i.e., the number of lanes, weaving lanes, turn connectivities across intersections or signal schedules cannot be included into this model.

2.4.5 Agent Database and Feedback

As mentioned above, the feedback mechanism is important for making the modules consistent with one another, and for enabling agents to learn how to improve their plans. In order to achieve this improvement, agents need to be able to try out different plans and to tell when one plan is “better” than another. The iteration cycle of the feedback mechanism allows agents to try out multiple plans. To compare plans, the agents assign each plan a “score” based on how it performed in the SQSim.

Some traffic-specific learning references are given below.

It is important to note that MATSim-EA always uses actual plan performance for the score. This is in contrast to all other similar approaches that we are aware of which typically feedback some aggregated quantity such as link travel times and reconstruct performance based on those (e.g., UrbanSim, 2006; Ettema *et al.*, 2004). Because of unavoidable aggregation errors, such an approach can fail rather badly in the sense that the performance information derived from the aggregated information may be rather different from the performance that the agent in fact experienced (Raney and Nagel, 2003). The procedure of the feedback and learning mechanism is as follows:

Initial conditions Start with a given plan file that specifies one complete plan for each agent. The agent database loads these plan files into the memory of the agents. Each agent marks its initial plan as the "selected" plan.

Simulate The agent database sends the set of "selected" plans (one for each agent) to the SQSim. The simulation executes the plans simultaneously and outputs events.

Process events The agent database reads the events that are output by the traffic flow simulation and sends each one to the agent identified within it. Each agent uses its events to calculate the score of its "selected" plan—the one it most recently sent to the traffic flow simulation.

Plan pruning The number of plans kept in an agent's memory for reuse can be limited to N plans to conserve memory. If N is defined, each agent that has $P > N$ plans deletes its lowest-scoring $P - N$ plans in this step. Note that when an agent that has N plans generates a new one, it temporarily keeps $N + 1$ plans until the new plan has been scored. Then, in this step, it deletes the worst plan (even if it is the newest one).

Select plans Each agent decides which plan to select for execution by the next traffic flow simulation. It chooses from the following selection options, according to the indicated probabilities:

(10%) New plan, routes only The agent sends an existing plan (chosen with equal probability among all plans in memory) to the dynamic Dijkstra router module of MATSim-STRATEGY. The router calculates new routes in that plan based on the link travel times calculated from the events data from the most recent traffic flow simulation, and returns the updated plan. The new plan is added to the agent's memory and marked as "selected."

(10%) New plan, times and routes The agent sends an existing plan (chosen with equal probability among all plans in memory) to the time allocation mutator module of MATSim-STRATEGY. This module "mutates" the durations and / or end times of all activities in the plan and returns the updated plan. The returned plan is also sent to the router for route re-planning. When it comes back from the route re-planner, it is added to the agent's memory and marked as "selected." (Note that now 20% of agents will have new routes, while only 10% will have new times.)

(10%) Random selection The agent picks an existing plan, chosen with equal probability among all plans in memory, without regard to their scores. This plan is marked as "selected."

(Remaining 70%) Probabilistic selection The agent picks an existing plan from memory, choosing according to probabilities based on the scores of the plans. The probabilities are of the form

$$p \propto e^{\beta \cdot S_j}\,, \tag{2.1}$$

where S_j is the score of plan j, and β is an empirical constant. This is equal to a logit model familiar from discrete choice theory (Ben-Akiva and Lerman, 1985). The chosen plan is marked as "selected."

The cycle returns to step 2 (Simulate), and continues until the system has reached a relaxed state. There is no quantitative measure of when the system is "relaxed"; we allow the cycle to continue until the outcome seems stable. When an agent reuses an existing plan, its previous score is not forgotten, but averaged with its new score:

$$S = (1 - \alpha) \cdot S_{old} + \alpha \cdot S_{new} \, , \tag{2.2}$$

with the blending factor α. This allows the agent to base plan selection on the plans' history and not only on the last iteration. With $\alpha = 0$ no score will be updated and the agents will not learn; the history of a plan is neglected. Score averaging requires all plans to have an S_{old}, so when a new plan is generated, it is optimistically given a preliminary score equal to the score of the agent's best plan. More sophisticated approaches to agent learning are discussed in Timmermans *et al.* (2003).

2.4.6 Scores for plans

In order to compare plans, it is necessary to assign a quantitative score to the performance of each plan. In principle, arbitrary scoring schemes can be used (e.g., prospect theory by Avineri and Prashker, 2003). Here, a simple utility-based approach is used, which is related to the Vickrey bottleneck model (Arnott *et al.*, 1993), but needs to be modified to be consistent with our approach based on complete daily plans (Charypar and Nagel, 2005; Raney and Nagel, 2005). The elements of this approach are as follows:

- The total score of a plan is computed as the sum of individual contributions:

  $$U_{total} = \sum_{i=1}^{n} U_{perf,i} + \sum_{i=1}^{n} U_{late,i} + \sum_{i=1}^{n} U_{travel,i} \, , \tag{2.3}$$

  where U_{total} is the total utility for a given plan; n is the number of activities/trips; $U_{perf,i}$ is the (positive) utility earned for performing activity i; $U_{late,i}$ is the (negative) utility earned for arriving late at activity i; and $U_{travel,i}$ is the (negative) utility earned for traveling during trip i. In order to work in plausible real-world units, utilities are measured in Euro.

- A logarithmic form is used for the positive utility earned by performing an activity (e.g., Axhausen, 1990a):

$$U_{perf,i}(t_{perf,i}) = \max\left[0, \beta_{perf} \cdot t_i^* \cdot \ln\left(\frac{t_{perf,i}}{t_{0,i}}\right)\right] , \tag{2.4}$$

where $t_{perf,i}$ is the actual performed duration of the activity, t_i^* is the "typical" duration of an activity, and β_{perf} is the marginal utility of an activity at its typical duration. β_{perf} is the same for all activities, since in equilibrium all activities at their typical duration need to have the same marginal utility. $t_{0,i}$ is a scaling parameter that is related both to the minimum duration and to the importance of an activity.

If the actual duration falls below $t_{0,i}$, then the utility contribution of the activity is zero, implying that the agent should completely drop that activity. A $t_{0,i}$ only slightly less than t_i^* means that the utility of activity i rapidly decreases with decreasing $t_{perf,i}$, implying that the agent should rather cut short other activities where the utility does not decrease as quickly when reducing their duration. In this application, we use

$$t_{0,i} = t_i^* \cdot e^{-\zeta/(p \cdot t_i^*)} , \tag{2.5}$$

where ζ is a scaling constant and p is a priority indicator. By substitution of $t_{0,i}$ of Equation 2.4 with Equation 2.5 the utility of performing an activity i is

$$U_{perf,i}(t_{perf,i}) = \max\left[0, \beta_{perf} \cdot t_i^* \cdot \left(\ln(\frac{t_{perf,i}}{t_i^*}) + \frac{\zeta}{p \cdot t_i^*}\right)\right] . \tag{2.6}$$

Setting p uniformly to one (all activities have the same priority) the value of $U_{perf,i}(t_i^*) = \beta_{perf} \cdot \zeta$ is independent of the activity type. This consequence is actually the motivation for Equation 2.5, which is used because no better argument was available (Charypar and Nagel, 2005); future research should lead to better versions.

- The (dis)utility of being late is defined as:

  $$U_{late,i} = \beta_{late} \cdot t_{late,i} \,, \tag{2.7}$$

  where $\beta_{late} \leq 0$ is the marginal utility (in Euro/h) for being late, and $t_{late,i}$ is the number of hours late for activity i. To be able to calculate the utility of being late, a starting time window for the activities has to be given.

- The (dis)utility of traveling is defined as:

  $$U_{travel,i} = \beta_{travel} \cdot t_{travel,i} \,, \tag{2.8}$$

  where $\beta_{travel} \leq 0$ is the marginal utility (in Euro/h) for travel, and $t_{travel,i}$ is the number of hours spent traveling during trip i.

At this point, the traffic flow simulation does not differentiate between “being at an activity location” (which potentially includes waiting) and “performing an activity.” In consequence, the simulation makes the agent stay at the activity location for the length of “duration,” no matter whether the agent can perform the activity or not. For example, when work starts at 8 AM but the agent arrives at 7 AM with a duration of 8 hours, then the agent will depart from the activity location at 7 AM plus 8 hours = 3 PM. The utility function, however, differentiates between “arrival time” and “activity start time.” The “work” activity has a particular starting time, and arriving before this time causes the agent to wait until then before actually starting the activity. This means that arriving to an activity before opening time does not gain an agent any activity performance utility.

2.5 Zurich Area Scenario

2.5.1 Scenario Initialization

The full Switzerland scenario demand generation is based on 24 one-hour origin-destination commuter matrices which are described in Section 4.1.5. For the multi-agent simulation, these hourly matrices are then disaggregated into individual trips. That is, individual trips were generated such that summing up the trips would again result in the given

OD matrix. The starting time for each trip is randomly selected between the starting and the ending time of the validity of the OD matrix. The OD matrices are based on the municipality layer as shown in Section 4.1.1.1 while in the simulation trips start on links of the given network (see Section 4.1.3.1). The municipalities are converted to links with the following heuristic. First, the geographic location of the zone is found via the geographical coordinate of its centroid given by the layer. Next, a circle with radius 3 km is drawn around the centroid. Finally, each link starting within this circle is now a possible starting link for the trips. One of these links is randomly selected and the trip start or end is assigned. This led to a list of approximately 5 million trips, or about 1 million trips between 6 AM and 9 AM. Since the origin-destination matrices are given on an hourly basis, these trips reflect the daily dynamics. Intra-zonal trips are not included in those matrices, as by tradition.

Since an agent should keep more than one plan during the iteration process, the memory requirements of one million agents exceeded the available memory. So we restricted our research to the Zurich Area. This was done with the following steps:

1. All trips are routed using free flow travel times.

2. The area of interest is defined as a circle of 26 km radius around the center (“Bellevue”) of the city of Zurich.

3. Each trip that does not cross this area is removed.

This results in 260,275 trips between 6 AM and 9 AM. All trips are now identified with an agent. The “origin” location for the morning trip is assigned to the “home” activity, and the “destination” location is assigned to the “work” activity. The end time of the home activity is set to the departure time of the original trip. The daily patterns “home-work” are then extended to the “home-work-home” pattern, where the two homes are at the same location. The duration of the “work” activity is set to 8 hours, with no fixed activity end time. At the end 260,275 agents that have an initial day plan are produced.

2.5.2 Simulation Parameters

The maximum number of plans that agents are allowed to keep in the agent database, N, is set to 5 plans. This number results from the scenario size in conjunction with computer memory limitations. The value of the empirical constant β used to convert plan scores to selection probabilities is $2.0/Euro$. The following values for the marginal utilities of the utility function are used for calculating scores:

- $\beta_{perf} = +6Euro/h$
- $\beta_{travel} = -6Euro/h$
- $\beta_{late} = -18Euro/h$

Although it is not obvious at first glance, these values mirror the standard values of the Vickrey scenario (Arnott *et al.*, 1993): An agent that arrives early to an activity must wait for the activity to start. During this time, the agent cannot perform any activity and therefore forgoes the $\beta_{perf} = +6Euro/h$ that it could accumulate instead (opportunity cost). An agent that travels fore-goes the same amount, *plus* a loss of $6Euro/h$ for traveling. And finally, an agent that arrives late receives a penalty of $18Euro/h$ late, but is not losing (or gaining) any time elsewhere by being late. Only daily activity chains are considered that consist of one home and one work activity. The "typical" times were set to $t^*_h = 16h$ and $t^*_w = 8h$. With these assumptions, the maximum score is $120Euro$ ($60Euro$ per activity). For the work activity a starting time window is defined between 7:08 AM and 8:52 AM. The blending factor α is set to 0.1. This is a useful compromise between zero learning and overreaction (for more details see Raney, 2005).

2.5.3 Results

2.5.3.1 Overview

The results of four different setups are presented, which result from two different initial conditions and from using time re-planning or not. The two initial conditions are:

Initial departure times given externally Here, the activity end times from the home activity are generated as described earlier. When the home activity ends, agents immediately depart and drive to work, where they stay for 8 hours, and then return. The setup where agents initially use externally defined times will be called *times-routes-initial-times-extern* and *routes-only-initial-times-extern* is named for the setup when times re-planning is enabled and disabled, respectively.

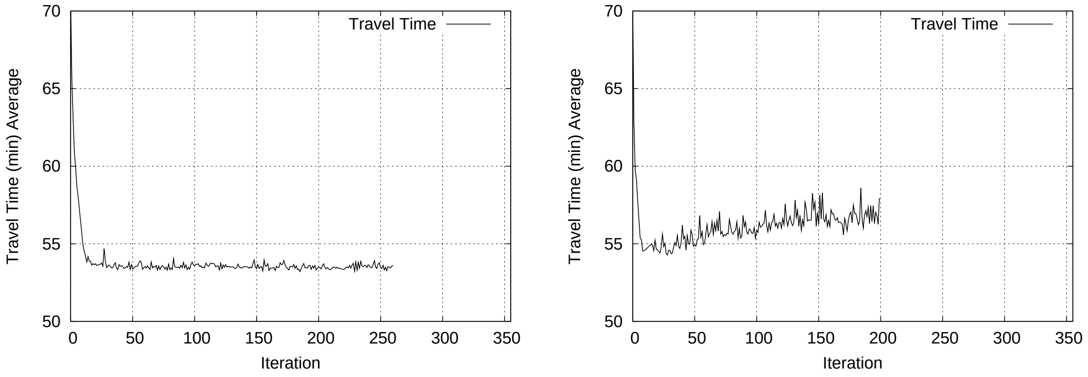
All agents depart home at 6 AM Once departed, agents drive to work, where they work for 8 hours, and then return. These initial conditions are used to have a scenario where the simulation starts with a clearly implausible situation. The question that is tested is whether it will recover to a realistic solution by itself. The setup where all agents depart at 6 AM will be called *times-routes-initial-times-all6am* and *routes-only-initial-times-all6am* defines the setup when times re-planning is enabled and disabled, respectively.

Note that when times re-planning is disabled, only 10% of agents perform route re-planning, but when it is enabled, a total of 20% of agents perform route re-planning, with half of those also performing times re-planning. The results are compared with the following indicators:

1. *Average travel time*: The average travel-time across all agents' plans for each iteration.
2. *Average score*: The average score across all agents for each iteration.
3. *Departure and arrival time histograms*: The number of agents that arrive/depart from an activity over time during a certain iteration.
4. *Traffic count data comparison*: Mean bias and error of the simulations compared to counting data.

2.5.3.2 Initial Plans with Externally Defined Departure Times

This setup tests whether or not the learning, once time re-planning is switched on, drifts away from the time structure given by the external



(a) Average travel times by iteration for routes-only-initial-times-extern setup.

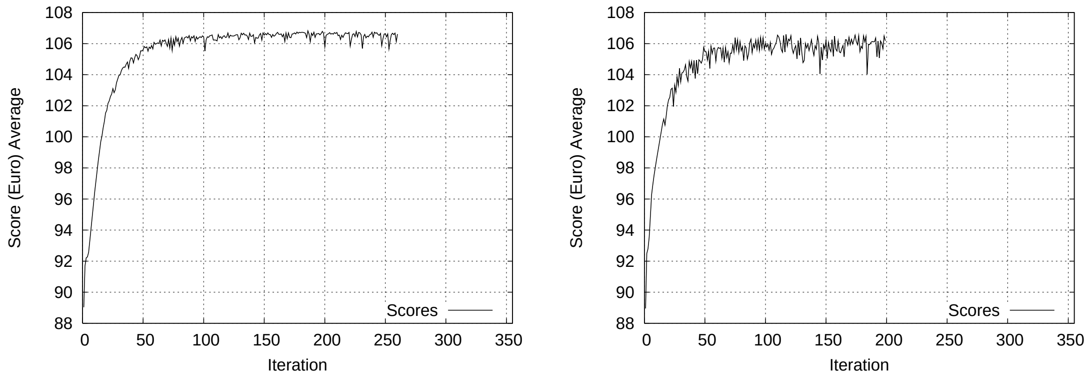
(b) Average travel times by iteration for times-routes-initial-times-extern setup.

Figure 2.3: Average travel times of routes-only-initial-times-extern and times-routes-initial-times-extern.

data. Since these initial plans are based on realistic time distributions, one would assume that the time re-planning will not affect the result that much. Re-routing alone should decrease the average travel time and congestion. Figure 2.3 compares the average travel times over the iterations. The routes-only iteration (Figure 2.3(a)) quickly gets to a stable result because re-routing is the only part which has to be optimized. The small fluctuations are due to the fact that some percentage of the agents always re-plan, and that the traffic flow simulation is stochastic.

The iterations where time re-planning is switched on (Figure 2.3(b)) behave in a similar way, but the average travel time is slightly higher than routes-only, and it fluctuates more. However, the scores of the times-routes setup are not worse than the scores of the routes-only setup. This indicates that the agents are “trading off” travel time for other parts of their utility. In other words, by adjusting their activity times (i.e., the times they make their trips) they make up for the fact that trips are longer by arriving at a more suitable time to work. The higher fluctuations can be attributed to the fact that there are now two re-planning elements which have to be optimized.

Figure 2.4 shows the scores for each iteration of both setups. They are once more similar to each other, and once more the routes-only setup (Figure 2.4(a)) shows less fluctuation than the setup with time re-planning (Figure 2.4(b)). The reason is the same as described above.



(a) Average scores by iteration for routes-only-initial-times-extern setup.

(b) Average scores by iteration for times-routes-initial-times-extern setup.

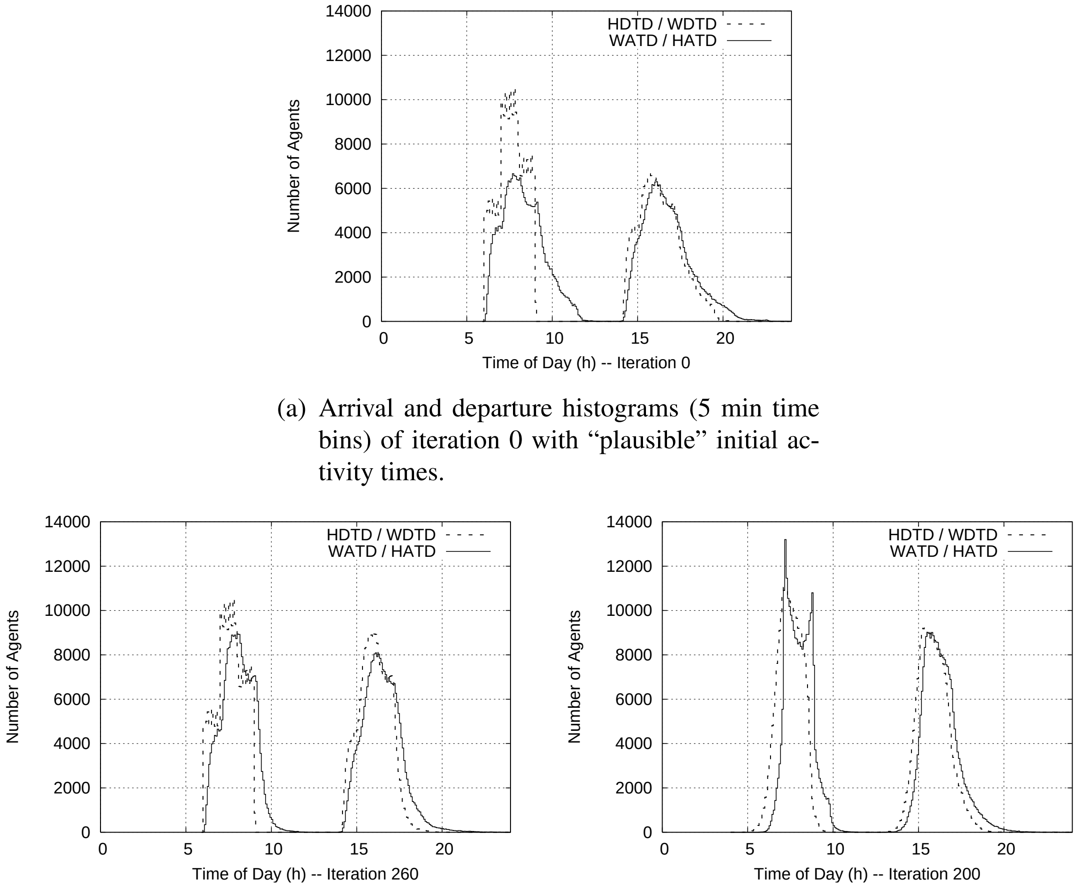
Figure 2.4: Average scores of routes-only-initial-times-extern and times-routes-initial-times-extern.

Comparing to Figure 2.3, one can see that in both setups, the average scores relax considerably more slowly than the average travel times. This is due to the score averaging in the agent database.

The histograms in Figure 2.5 show how the re-planning affects the agents. Starting with the same configuration (Figure 2.5(a)), the routes-only iteration only tries to minimize travel times, so that the arrival period decreases (see bold graph of Figure 2.5(b)), while departure from home stays the same (see dotted graph of Figure 2.5(b)).

Switching on time re-planning changes also the dotted graph (see Figure 2.5(c)). The two peaks of the arrival (bold) graph are at 7:08 AM and 8:52 AM, which are the borders of the time window we defined for these scenarios. The reason for that is the fact that agents, which arrive too late or too early to work try to “squeeze” into this time window. Once they are inside the time window they will more or less stay at this plan if they succeeded.

Finally, we look at the traffic count data. Figure 2.6 shows the comparison of the two setups and the real data given by the 12 links as presented in Section 4.1.7. As expected, the two results do not differ that much, and they are comparable to reality (see also Raney *et al.*, 2003). Also the quantitative measures of bias and errors are similar (Table 2.1).



(a) Arrival and departure histograms (5 min time bins) of iteration 0 with “plausible” initial activity times.

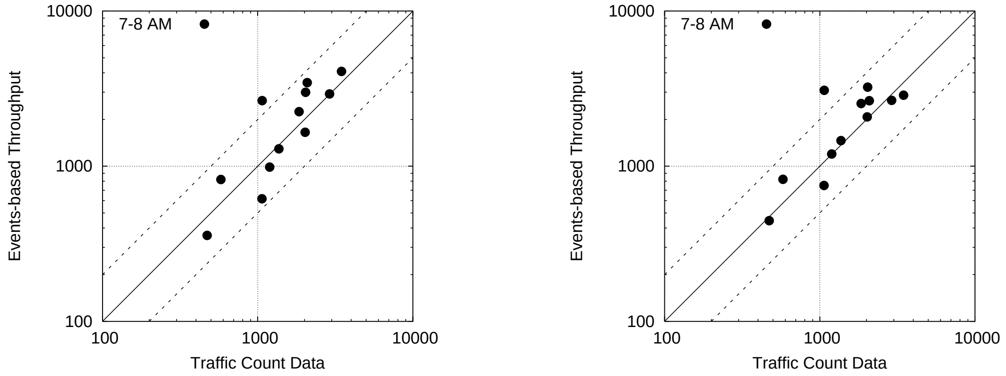
(b) Arrival and departure histograms (5 min time bins) of iteration 260 with time re-planning switched off.

(c) Arrival and departure histograms (5 min time bins) of iteration 200 with time re-planning switched on.

Figure 2.5: Arrival and departure histograms when the initial plans have “plausible” departure times.

Bias / Error	Routes Only	Times and Routes
Mean Abs. Bias:	+331.40	+306.32
Mean Rel. Bias:	+19.62%	+25.27%
Mean Abs. Error:	533.55	503.77
Mean Rel. Error:	37.50%	35.38%

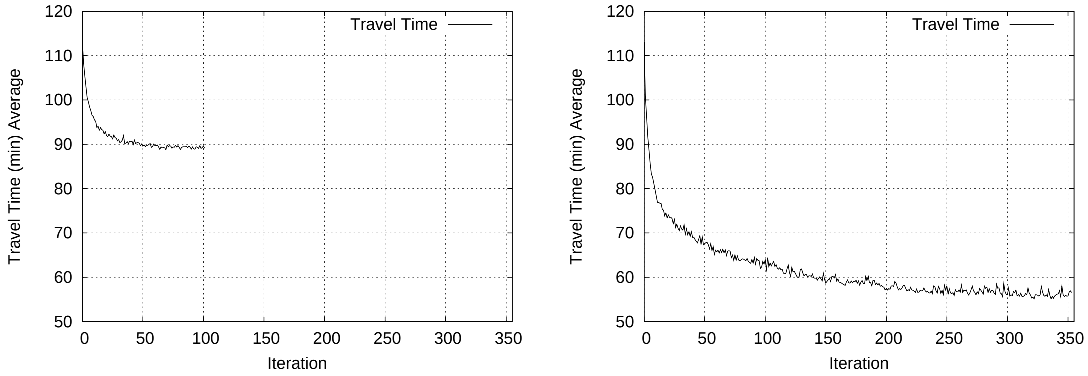
Table 2.1: Bias and Error of routes-only-initial-times-extern and times-routes-initial-times-extern compared to field data at 7-8 AM.



(a) Traffic count data vs. 260th iteration of routes-only-initial-times-extern setup at 7-8 AM.

(b) Traffic count data vs. 200th iteration of times-routes-initial-times-extern setup at 7-8 AM

Figure 2.6: Initial plans with externally defined departure times: comparison to traffic count data.



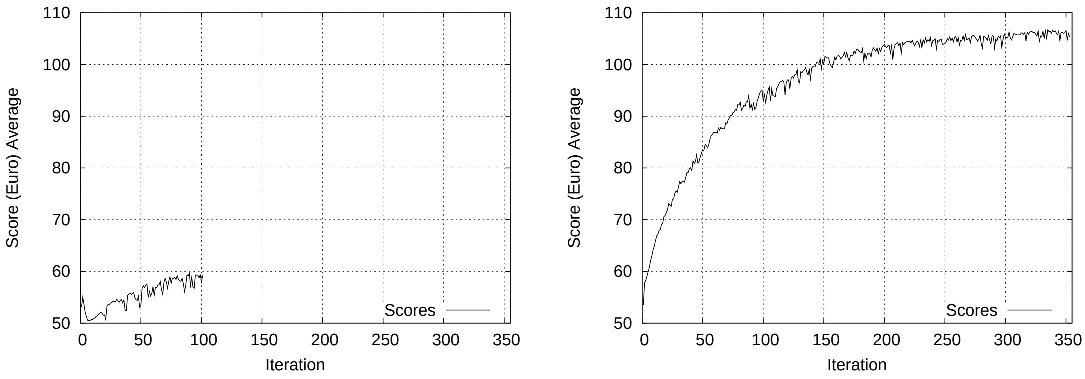
(a) Average travel times by iteration for routes-only-initial-times-all6am setup.

(b) Average travel times by iteration for times-routes-initial-times-all6am setup.

Figure 2.7: Average travel times of routes-only-initial-times-all6am and times-routes-initial-times-all6am.

2.5.3.3 Initial Plans with Departure Time at 6 AM for all Agents

The previous section demonstrated that the results both with respect to the time structure and with respect to validation do not (at least) become worse when time re-planning is switched on. However, the initial condition was still based on the externally given time structure. The experiments in this section will test in how far a realistic time structure can



(a) Average scores by iteration for routes-only-initial-times-all6am setup.

(b) Average scores by iteration for times-routes-initial-times-all6am setup.

Figure 2.8: Average scores of routes-only-initial-times-all6am and times-routes-initial-times-all6am.

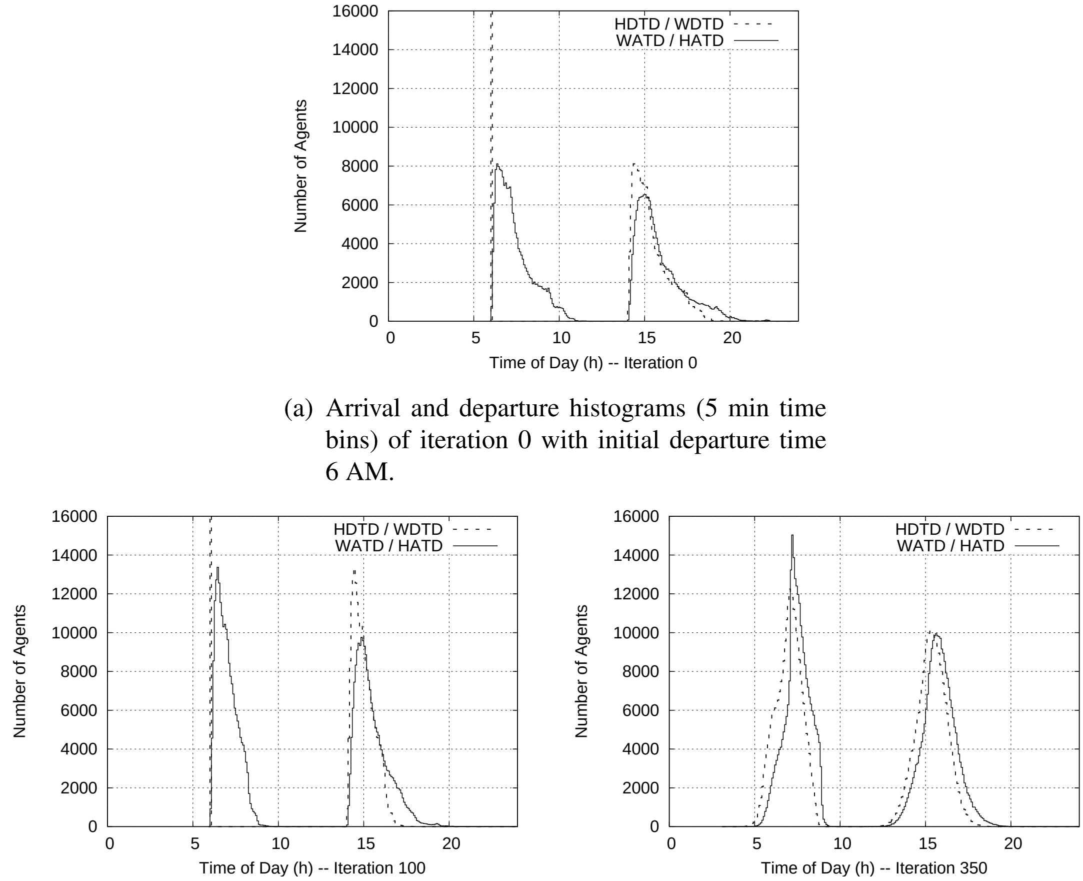
be generated even when starting from a clearly implausible initial condition. For this purpose, all initial plans will be modified so that all agents initially depart at 6 AM. Apart from that, the initial plans are the same as before.

Figure 2.7 shows again the average of travel times for both setups. We see that this time, the routes-only setup decreases travel time more slowly than before because it is harder to avoid congestion when all agents start traveling at the same time. Of course, at the end the average travel time will be higher. With time re-planning switched on, average travel times decrease rather quickly, because agents are now allowed to change their departure time, too.

Also, average scores without time re-planning (Figure 2.8(a)) show only little improvement. Only optimizing routes does not help that much because a major part of the agents will then arrive at work too early which does not increase scores (Figure 2.9(b)).

When the time re-planning module is also switched on, agents are now able to have short travel times and still arrive at work within the given time window. Figure 2.8(b) shows that the average score slowly increases to the same level as in Figure 2.4(b).

The histograms (Figure 2.9) also show those facts. There are many more people who arrive between 6 and 7 AM in the routes-only setup (Figure 2.9(b)) than in the times-routes setup (Figure 2.9(c)). The peak



(a) Arrival and departure histograms (5 min time bins) of iteration 0 with initial departure time 6 AM.

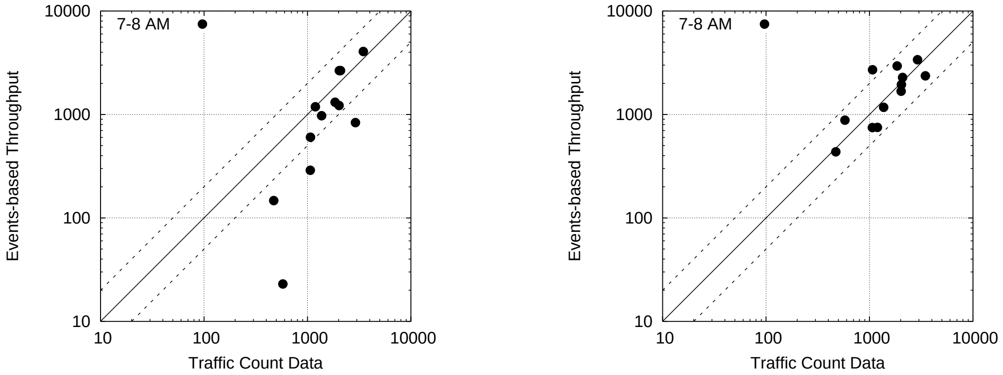
(b) Arrival and departure histograms (5 min time bins) of iteration 100 with time re-planning switched off.

(c) Arrival and departure histograms (5 min time bins) of iteration 350 with time re-planning switched on.

Figure 2.9: Arrival and departure histograms when in the initial plans everybody departs at 6 AM.

Bias / Error	Routes Only	Times and Routes
Mean Abs. Bias:	−344.76	+99.24
Mean Rel. Bias:	−31.30%	+12.40%
Mean Abs. Error:	644.11	520.26
Mean Rel. Error:	43.80%	36.10%

Table 2.2: Bias and Error of routes-only-initial-times-all6am and times-routes-initial-times-all6am compared to field data at 7-8 AM.



(a) Traffic count data vs. 100th iteration of routes-only-initial-times-all6am setup at 7-8 AM.

(b) Traffic count data vs. 350th iteration of times-routes-initial-times-all6am setup at 7-8 AM.

Figure 2.10: Departure time 6 AM plans: comparison to traffic count data.

of the departure time (dotted) graph of Figure 2.9(c) has moved toward the same time as shown in Figure 2.5(c) of the previous section.

Comparing the results with real word data shows a high discrepancy between the two setups. In the routes-only setup almost everybody starts too early. So it underestimates the throughput between 7 and 8 AM (Figure 2.10(a)). In the times-routes setup (Figure 2.10(b)), agents slowly move to more appropriate departure times which—at the end—will converge to similar results as obtained before. Of course, the calculation of the bias and the error (Table 2.2) now produces completely different results for the routes-only setup.

2.6 Discussion

The preliminary implementation of MATSim-EA presented above proves that the fully agent-based approach is useful for transport planning. It also shows that the model is efficient enough for large-scale scenarios. But there are still many open questions to answer. In the next chapter, issues in design and concept are stated which still need to be solved.

Chapter 3

Design Issues

The previous Chapter 2 showed an overview about the functionalities of MATSim-T implemented and tested. The strategy modules *dynamic djikstra router* and *time allocation mutator* allow one to find a stable state of the system. But it also shows that this is a solution for specific type of scenario. Many replanning possibilities were excluded. In other words, the agents' search space for optimizing their daily plan is small. I.e., they are not allowed to choose other locations than the given ones. Also, the daily activity chain is unchangeable. Last but not least, all agents in the system do have the same activity chain (the "home-work-home" pattern).

Compared to the overview of Section 2.2, only MATSim-EA is implemented. But to produce the demand for a real-world scenario, we need to create the agents in a way such that they reflect the real world in a more realistic way.

In here the questions are formulated concerning the needs for transport planning that have to be added to the toolkit. It is divided into three subsections:

- Structural issues: Formulates problems with respect to implementation of the system and computational performance.
- Functional issues: Formulates problems with respect to the replanning of agent's daily plans, measurement of plan performance (scoring of plans) and functional enrichment of the system.
- Initial condition issues: Which are—in fact—the missing parts of MATSim-T, as presented in Section 2.3, to generate agents and

initial individual demand from the given input data of a scenario.

Some of the following issues go beyond the scope of this work. But still it is important to summarize all of them, since it gives one a nearly complete overview in which directions the MATSim-T project will be enhanced.

3.1 Structural issues

It will be shown that it is necessary to build up a system such that it is feasible to enhance it with new features without breaking up the programming concept. The implementation should be as flexible as possible without losing too much computational speed. The problems arising during the development phase will be formulated and discussed next.

3.1.1 Implementation

MATSim-EA is logically split into independent parts as indicated in Figure 2.2 (Agent database, MATSim-EXEC, the modules in MATSim-STRATEGY and MATSim-SCORES). The preliminary implementation does in fact work a little different, as we show here.

3.1.1.1 MATSim-SCORES

The scoring function by Charypar and Nagel (2005) as described in Section 2.4.6 has in fact implemented as a part of the Agent database (see Raney, 2005, for more details) which actually calculates the scores of the executed plans. Technically, that seems quite obvious, since the whole state of an agent is stored in the agent-DB. But thinking about more enhanced strategy modules than the ones we were using (i.e. the time allocation mutator), it is necessary to calculate scores also in the strategy modules for more enhanced replanning. Otherwise, replanning modules do not know anything about the direction in which they have to optimize agents' plans.

3.1.1.2 Plan Selection

This is an issue which is easily forgotten. Before MATSim-EXEC actually executes the agent's plan in a physical environment, one plan has to be selected for each agent. As described in Section 2.4.5, the plan selection was also a feature of the agent database. This does make sense since selection must take place where all information of an agent is stored.

But the way to select a plan is also part of the model of MATSim-EA which should be easily exchangeable. At the moment, this interface is not easily accessible. Therefore, it is also necessary to build "plug-and-play" functionality into the system.

3.1.2 Computational Performance

Computational performance is a crucial issue in agent based microsimulation models. Since MATSim-T is a support tool for transport planners, we need to be concerned about the computing time for a whole run of a given scenario.

If one needs several weeks to obtain results for one complete run, it does not satisfy the needs of planners. A fairly acceptable upper time limit lies somewhere between one and two days. This target must be always kept in mind while enriching the functionality of the system.

In the following, we give some indications about where the system loses computational speed. On one hand, much computation time is spent with information passing. On the other hand, each module also takes time. The better an implementation is done (i.e. faster algorithms), the faster the whole optimization process runs. But we need to distinguish between modules and algorithms that run only one time (algorithms of the data preparation, agent initialization and result analysis) and algorithms which run several times during the whole optimization process of MATSim-EA. The second ones are crucial to computational performance. Therefore, the focus lies on MATSim-EXEC, MATSim-STRATEGY and MATSim-SCORES.

3.1.2.1 Information Passing between Modules

Information about agents is passed through well defined XML files. This is an advantage when adding new modules to the system that are written in an arbitrary programming language. Each module is a separate program which reads in a plans file, modifies it according to its task, and writes the file out again. Therefore, such a program does—in principle—not need to know anything about the MATSim-EA iteration process.

But this has a disadvantage for the computational performance. Each module—in fact—wastes time because it needs to read and write files. The more replanning modules are used, the more I/O will be produced. This problem is even more crucial in parallel computation. In this case, reading and writing happens through the network of the computer cluster, which consumes a fair amount of bandwidth of the network (see Cetin (2005) for more details).

A simple and still flexible and robust solution to this performance issue is to add replanning modules as subroutines to the database, since all information needed is already stored there in a well defined data structure. Therefore, if the agent database provides clean interfaces to the data points, we are able to "plug-in" the modules such that they do not have to read or write anything from/to files. The concept of the modular approach of MATSim-T is not disregarded by implementing the modules as subroutines, only the interfaces are redefined. The disadvantage compared to the file based information passing is that it is no longer possible to write modules in an arbitrary programming language.

3.1.2.2 Stochastic, Queue Based Agent Traffic Simulation (SQSim)

Cetin (2005) shows that the physical execution of the agents' plans can be done with a RTR[1] between 100 and 600, highly dependent on the simulation setup, the message passing, the network speed and bandwidth, the amount of information output by the simulation, and the number of computers used in the cluster.

This shows that it is possible to run agent based micro-simulation models for large scenarios. The disadvantage about this implementation

[1] RTR means Real Time Ratio, which shows how much faster a simulation runs compared to real life. A RTR of 1 says that one second in the simulation is equal to one second in the real world. The higher the RTR is, the faster the system runs.

is that it scales according to the size of the network. This means that the simulation has more or less the same RTR during the traffic peak hours as in the middle of the night, where there is almost no traffic.

Charypar *et al.* (2007) is developing a deterministic, event driven, queue based agent traffic simulation (DEQSim) which scales according to the number of agents. Compared to SQSim it has the advantage that it uses computation time only when something actually happens (when an agent produces an event). With it, the simulation runs significantly faster during off-peak time periods. For testing purposes, we usually reduce the population of a scenario to a one or ten percent sample. In this case, the speedup of the DEQSim is remarkable while the SQSim still needs about the same time as when run for the whole scenario.

Because of the modular approach of MATSim-T, we are able to exchange MATSim-EXEC modules with others. The promising first results of the DEQSim show that it is desirable to integrate it in the iterative optimization process.

3.1.2.3 Time Allocation Mutator

Because the time allocation mutator module is an extremely simple minded module, it does not spend much computation time. But it is clear that because of its random nature (arbitrary variation of departure time and duration planning) it is the reason that we need to do many hundreds of iteration steps to reach a stable state of the system as described in Section 2.4. Therefore, it is essential to implement a much more enhanced time-planning module to reduce the number of iterations.

But a “better” module typically means also a module which runs slower than before. We need to keep track of the performance gain by iterating less compared to the performance loss by using an enhanced time-planning module.

3.1.2.4 Dynamic Dijkstra Router

The implemented dynamic Dijksta router can be split up into two parts considering computation time: First, the calculation of the time dependent link travel times for each link of the given network is done using the output events (see Section B.1) of MATSim-EXEC. Therefore, the computation time scales with the amount of events produced by the physical simulation. The number of events again scales with the number of agent in the system. A typical one day scenario has about a million agents. Each agent produces about 50 to 100 events. Therefore, the router needs to process ca. 50 to 100 million events before it can start to re-route. One possibility to gain computational speed is to compute time dependent link travel time already in MATSim-EXEC to reduce the amount of information passed to the router.

Second, the re-routing itself also takes a fair amount of time. In a typical day an agent produces about 2 to 4 trips. If—for example—every tenth agent's plan will be re-routed, the router needs to calculate about 200 to 400 thousand routes. So, it also would make sense to speed up the algorithm itself.

3.2 Functional Issues

The previous Section addressed more the computer scientist while the issues discussed here are more interesting for transport planners. The foci are shown where to enrich and improve the functionalities of MATSim-T.

3.2.1 Strategy Modules

The implemented strategy modules define the search space in which an agent is able to optimize its plan. The two current modules (route-replanning and time-replanning) can still be improved in their functionalities.

3.2.1.1 Dynamic Dijkstra Router

As already described above, the router calculates the fastest routes between an origin and a destination location based on the *full* information about the outcome of MATSim-EXEC. In other words, each agent does have *full* knowledge of the time dependent link travel times of *all* links. To find an "optimal" state of the system, this global knowledge helps us to reduce the number of iterations needed to find this state. On the other hand, no individual does have complete traffic knowledge.

This leads to the idea of introducing a *mental map* (Axhausen, 2006) of the network for each agent. They can only re-route based on a subset of the given information. There are many different approaches which handle incomplete knowledge, typically used in the field of research of reinforcement learning (Sutton and Barto, 1998). But there are still many open questions. A mental map of an individual is changing over time based on new experiences, information exchange, exploration, exploitation, etc., which are all behavioral aspects of an individual. If the agents in the system do not behave in—at least—a similar way, the outcome would be substantially biased.

Still, one is able to justify the use of global knowledge. For a typical day of an urban area, the approximation of using all information gives acceptable results, since the mental map of a person is quite similar to the global knowledge (at least for the parts of the scenario on which he acts). A commuter does in fact know when and where congestion typically happens and when it dissolves.

Nevertheless, the importance of mental maps should not be minimized. Especially if one wants to simulate "untypical" days, e.g. the "Street-Parade" day of Zurich or the final match of World Soccer Championship in Berlin. In such situations, a fair number of persons who want to visit the event actually do not re-route, because they do not have any information about the system except one route directly to the event. Therefore, their mental maps are quite small compared to the whole scenario.

3.2.1.2 Time Allocation Mutator

The time allocation mutator module used in Section 2.4.2 produces—after a fair amount of iterations—feasible departure times of activities as shown in Section 2.5.3. But we have to be careful with this interpretation. The scenario is set up such that all agents need to fulfil the same home-work-home pattern with the same opening time windows for the work activity.

To show the problems arising, let us do a small thought experiment by adding only one additional person with the following initial day-plan:

- He wants to execute a home-leisure-home pattern with desired 2 hours of leisure.
- His leisure-location lies on the other side of the scenario and has a special opening time window from 8 PM to midnight (i.e. a pub).
- His initial departure time from home is—let us say—3 AM (while the others in the setup shown in Section 2.5.3.3 initially leave at 6 AM).

Since the time allocation mutator randomly sets a new departure time between $\pm 30\ min$ of the given one, this person needs several iterations of time-replanning until he even reaches a departure time around 7 AM. Because at this time the other agents in the system already produce a high amount of traffic, this special person is also stuck in congestion. The resulting score of the plan departing at 7 AM is therefore lower than a plan which starts at 3 AM (because of longer travel times and not performing the leisure activity). As a result, he will throw away the 7 AM plan and keep the one which he started earlier. Ergo, he will never find out that it would make sense to depart from home sometime in the evening, even though the congestion is not that high and he actually can perform his leisure activity.

This thought experiment shows that the time allocation mutator module only produces feasible results for a subset of possible scenarios, since the search space is too limited. Therefore, it is clearly necessary to implement a generally usable time replanning module, especially if one want to run scenarios with complete daily plans including shopping, education and leisure activities and with different activity chains.

3.2.2 Functional Enrichment of the System

Since the system held time and route replanning strategies as the only two strategies, all the other parts of a complete daily plan will—per definition—stay fixed during the whole iterative optimization process. But since there are (at least) eight different choices (as described in Chapter 1) an agent could follow, it is desirable to include those additional strategies into the optimization process, because of the high dependency between all decision processes of an agent.

Short and medium targets are the *short term location choice* and the *mode choice* strategies, which are actually choices people typically make daily. Other decision processes, such as *activity chain* decision processes, usually produce dependencies which are based on longer time periods, like a week (i.e., people plan to visit a friend some days or weeks in advance).

3.2.3 Plan Performance Measurement

Plan performance measurement as described in Section 2.4.6 calculates the score of an executed plan as a function of durations. This is sufficient as long as the agents are allowed to change only routes and times. But if the system also includes other strategies, the function needs to extended to capture also time independent (dis)utilities. For example, the choice of using public transport does not only depend on the travel time difference between public transport modes and individual transport modes. It is also highly dependent on the number of modal interchanges (Wardman and Hine, 2000) a person has to make to go from one location to another.

3.2.4 Analysis

Up to this point, analysis of the results is done by writing small and inflexible scripts to extract aggregated information to compare with real world data (i.e. traffic counts). This is highly undesirable since for the reuse of such scripts they typically need to be adapted for each newly built scenario.

The target is to produce standardized—and therefore reusable—analysis tools that make it easy to extract any kind of desired information based on the results of a demand optimization run. It is necessary to define a clean and simple mechanism to add more and more analysis tools to the system such that the existing ones are still usable.

3.3 Initial Condition Issues

The system as it was presented in Section 2.4 is completely based on a given street network as the only static description of the environment.

In principle, there are no restrictions on which links agents perform activities. They were implicitly given by the commuter matrix (see Section 2.5). It is easy to see that there are many gaps to close if we want to go from simple "home-work-home" scenarios towards realistic daily demand generation. The following formulates the missing parts of the system.

3.3.1 Scenario Descriptions and Constraints

In agent-based systems, the agent acts in an environment which is sufficiently defined. In other words, a *closed and consistent world* has to be defined in which the agents are able to "live." In this case, agents want to travel and perform activities. But in fact, there is no description of where agents can actually perform specific activities. Therefore, the system needs to be extended with this information, such that it can be guaranteed that, for instance, no one can go shopping on a link where no shopping facility exists. This is of high interest if the system allows the agents to change activity location.

Since MATSim-T is completely dynamic, also the times when agents are allowed to perform an activity at a facility needs to be defined as *opening and closing times* for each activity of the facility. At the moment, these constraints are given as global scenario parameters, which means that all agents in the system have to respect the same constraints. This is of course not true in the real world.

Another constraint which the system has to be concerned with is the limited capacity of a facility. For example, we need do be assure that a company with one hundred work places does not allow that more than one hundred agents are working there, at any one time.

If public transport is included to the system, also the public transport infrastructure needs to be added to the system, e.g. to calculate access times to the the public transport system for agents that want to use it. Of course, also this part of the system defines constraints like time tables. Therefore, agents can access the public transport at a specific place only at predefined times.

3.3.2 Initial Demand Modeling

Based on the above mentioned closed and consistent world the system allows one to create an initial demand for the iterative optimization process, such that it respects the physical environment. Based on what dimension of the demand will be optimized, and which dimension will stay fixed during optimization, the initial demand modeling process has to make sure to model the fixed part such that it reflects reality. For example, if the system does not allow agents to optimize their choice of mode of transport, the initial demand of the scenario needs to produce a realistic modal split on the basis of available survey data. But if the agents can choose the modes by themselves, the modal split process step can be left out. Therefore, the relaxed state of the optimization process should reflect reality by comparing its result with measurements.

The work presented focuses on this topic, presenting a consistent description of the physical environment and a flexible toolkit for modeling the initial individual demand of a scenario.

Chapter 4

Into Practice: Scenario Description

The proceedings described in the following chapters will be applied to two different scenarios. One—medium size scenario—is located in the greater Zurich area (Kanton Zurich, Switzerland) and consists of about 1.3 million individuals. The second—large scale scenario—is defined for the states of Berlin-Brandenburg in Germany with about 7 million inhabitants. The two scenarios differ in the amount of available information, in spatial resolution, and in depth of the data.

4.1 Kanton Zurich

The Kanton Zurich is part of the Swiss "Mittelland". It reaches the Rhine in the north and the Alps in the south. The size is 1,729 km^2 and covers about 4% of Switzerland. With more than 1.2 million inhabitants (ca. 16% of Switzerland) the Kanton Zurich is the most populous regions in the country. Figure 4.1 shows where the Kanton Zurich is located in Central Europe. The following sections describe the available input data.

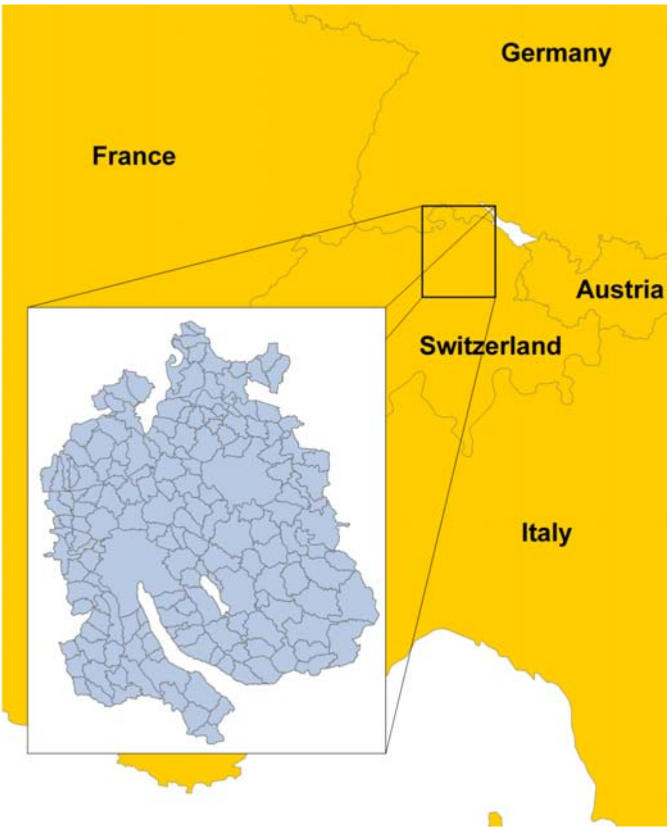


Figure 4.1: Kanton Zurich and its 171 municipalities

4.1.1 Spatial Levels of Detail

4.1.1.1 Municipality Layer

The Kanton Zurich contains 171 political municipalities, of which the City of Zurich is one. This resolution is quite appropriate for suburban regions but not for the City itself. Therefore, it is spit up into 12 zones (called “Kreise”) resulting in 182 geographical zones. In the following, we will denote this as the *municipality layer*.

4.1.1.2 Raster Layer

Based on the *Swiss map projection* (see Swisstopo, 2006) the hectare[1] raster level of detail is also available. Since we are only interested in

[1] A hectare is an area of $100 * 100m$.

regions accessible by land, 168’167 cells for the Kanton Zurich are included in this layer, excluding cells which are located completely in a lake. The raster level of detail will be denoted as *raster layer*.

4.1.1.3 Layer Mapping

For (dis)aggregation purposes it is necessary to define an appropriate mapping rule between the above defined layers. Since both layers are based on the same coordinate system, an unambiguous mapping between the cells of the raster layer and the 182 municipalities can be provided: Each municipality holds at least one raster cell, while each cell belongs to exactly one municipality. The mapping is calculated with GIS “intersect”-operation: the intersection of each hectare with the borders of the given municipalities shows whether a hectare is—at least—part of one municipality. Otherwise, this cell is erased from the raster layer. All remaining cells are assigned to the municipality which covers the largest area of that cell.

4.1.2 Facilities

The land-use information is crucial for the activity based demand generation processes. It has to be guaranteed that certain activities can only be performed at appropriate places. Therefore, land-use information needs to describe which types of activity are available at a defined location. Further, it is of interest to define typical times when such activities can be performed. With respect to those requirements, the focus lies more on a description of a *facility* then of some vague aggregated land-use information.

Nevertheless, aggregated land-use information is typically the data available, which then can be interpreted as one facility representing the union of all existing facilities of a given spatial unit.

4.1.2.1 Hectare-Facilities

The hectare-based land-use information is provided by the Swiss Federal Statistical Office (BfS, 2006), describing one aggregated facility per cell of the raster layer described in Section 4.1.1.2. Figure 4.2 shows an

```
<cell x100="641500" y100="241400">
  <activity type="home" capacity="12"/>
  <activity type="work" capacity="84" start_time="07:00"
    end_time="19:00"/>
  <activity type="shop" start_time="18:30" end_time="22:30"/>
</cell>
```

Figure 4.2: XML example of an aggregated facility for a raster cell in the Kanton Zurich

example of such an aggregated facility. The data points per raster cell are:

- Activity types (home, work, education, shopping and/or leisure)
- Capacity for each given activity type (defines the maximal number of persons performing that activity at the same time)
- Opening hours per activity (if no opening time is given, the activity can always be performed)

In the following this land-use information will be denoted as *Hectare-Facilities*.

4.1.3 Network

For any agent-based micro-simulation purposes, appropriate descriptions of transport networks for each transport mode are necessary. In this case, a street network is needed, represented as a *strongly connected geo-coded digraph* (see Balmer *et al.*, 2005a, page 8) with the following required attributes for each directed link:

- The length of the link (must be at least the Euclidean distance between its start and end nodes)
- The speed limit of the link
- The number of lanes of this link
- The maximum flow capacity of the link

The last item is crucial since it defines the upper limit of the throughput of a link, but it is not that easy to obtain. One method is to extract the peak hour throughput of the link. But usually this information is only available for a few links of a given network. A typical way to obtain the link flow capacities from a network is to use calculated flow capacities from assignment models. Since the outcome typically described the modeled average volumes per link, some heuristics are need to extract the "operational link flow capacities"[2]. This method produces feasible upper limits but are not completely error resistent (since the results are already based on a model). Another—quite robust—solution is to use the general classification by street type, i.e., motor ways, major roads, minor roads, and so on. Such classifications are usually available for street networks.

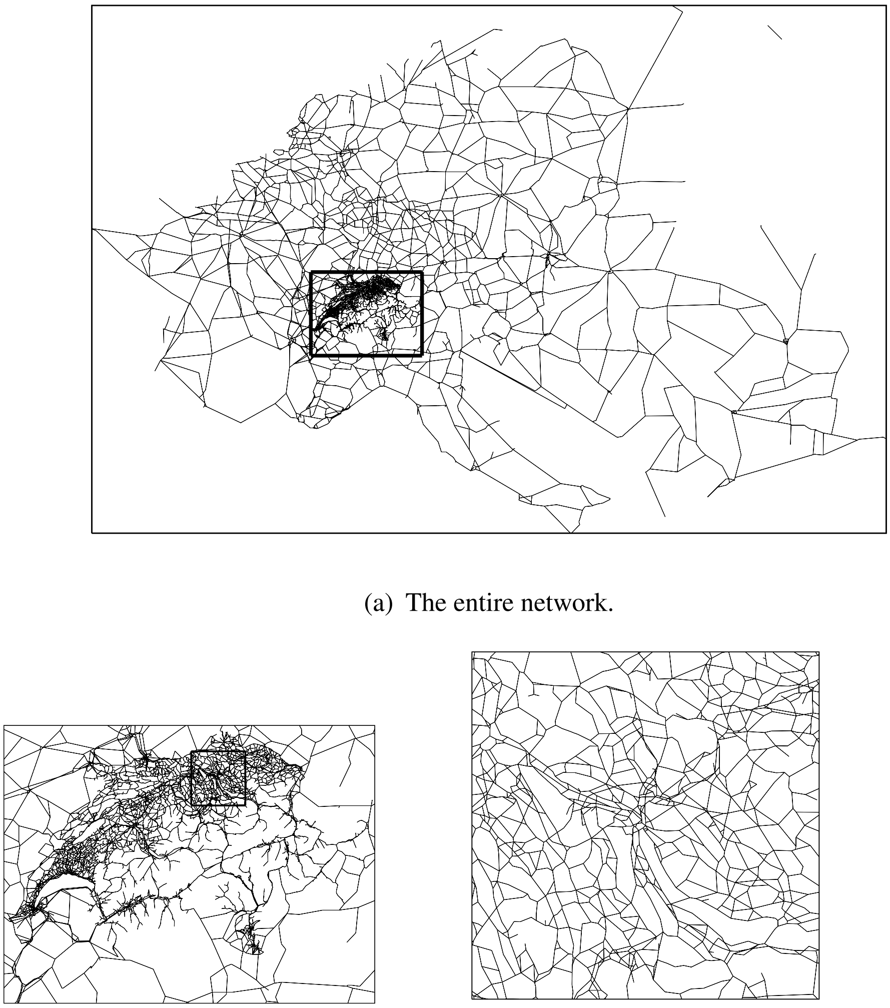
4.1.3.1 The Switzerland Network

The network of streets and intersections used in the Switzerland scenarios was originally developed for the Federal Office for Spatial Development (ARE), and covered only Switzerland, but has been extended with the major European transit corridors for a railway-related study (Vrtic *et al.*, 2003). See Fig. 4.3(a) for the complete network and Fig. 4.3(b) for the higher-resolution Switzerland portion.

The network supposedly contains the status for 1999, but contains at least one major error (a high capacity tunnel in Zurich is missing). Our initial simulations resulted in traffic gridlock in Zurich, which was also reflected in the VISUM assignment displaying V/C ratios significantly above 100%. A manual comparison with a higher resolution network of Zurich led to the conclusion that capacity in Zurich was in general significantly underestimated; in consequence, the corresponding road capacities had to be increased for transit corridors through Zurich.

After these modifications, the network includes 10,564 nodes and 28,624 links. Also fairly typical, the major attributes of these links are type, length, speed, and capacity.

[2] The *physically possible* throughput of a street link



(a) The entire network.

(b) Switzerland area.

(c) Zurich area.

Figure 4.3: The network used for the Switzerland-based scenarios, at various scales. The entire network (a) includes major European transit corridors. The box in (a) indicates the Switzerland area, seen in (b). The box in (b) indicates the Zurich area, seen in (c).

Source: Raney (2005)

4.1.4 Synthetic Population

Based on "Census 2000" and "Microcensus 1989 and 2000" (GS EVED and BfS, 1992; ARE and BfS, 2001) a synthetic population for the Kanton Zurich is created by Frick and Axhausen (2004). It consists of

1,147,906 individuals with the following attributes:

- Person ID
- Age (given in 5 year time bins, from 0-4 years to 105-109 years)
- Sex (`male`, `female`)
- Driving licence ownership (`yes`, `no`)
- Car availability (`always`, `sometimes`, `never`)
- Employee (`yes`, `no`)
- Half-fare ticket ownership[3] (`yes`, `no`)
- General season ticket ownership[4] (`yes`, `no`)
- Income (in Swiss Francs)
- Home location based on raster layer (see Section 4.1.1.2)

As an example, Figure 4.4 shows the male persons between 35 and 39 years of age.

4.1.5 Pendlermatrix 2000

The full Switzerland scenario demand generation is based on the 24-hour origin-destination commuter-matrices of the Swiss spatial planning authority (ARE, 2006), based on the municipality level of detail described in Section 4.1.1.1. The original 24-hour matrix was converted into 24 one-hour matrices using a three step heuristic (Vrtic and Axhausen, 2003). The first step employed departure time probabilities by population size of origin zone, population size of destination zone, and network distance. These were calculated using the 1994 Swiss National Travel Survey (BfS, 1996). The resulting 24 initial matrices were then

[3] In Switzerland it is called “Halb-Tax”. It costs about 150 SFr. per year and allows one to buy almost any public transport ticket for half price.

[4] In Switzerland it is called “General Abonnement”. It costs about 3000 SFr. per year and allows one to use any public transport with no additional charge.

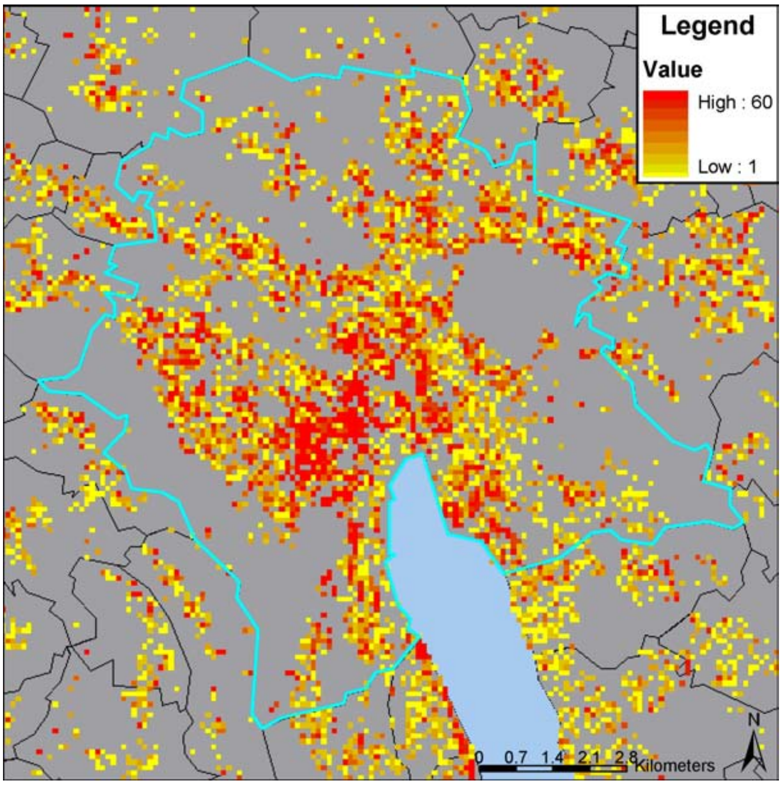


Figure 4.4: Synthetic Population of Kanton Zurich: Occurrence of males of age between 35 and 39 years

Source: Frick and Axhausen (2004)

corrected (calibrated) against available hourly counts using the OD matrix estimation module of VISUM (PTV, 2006). Hourly traffic count data is available from the counting stations on the national highway system (ASTRA, 2006). Finally, the hourly matrices were re-scaled so that the totals over 24 hours match the original 24 hour matrix. VISUM assignment of the matrices showed that the patterns of congestion over time are realistic and consistent with the known patterns.

4.1.6 Activity Chains and their Distribution

In the Microcensus 2000 (ARE and BfS, 2001), carried out by the Federal Office for Spatial Development (ARE, 2006) in cooperation with the Swiss Federal Statistical Office (BfS, 2006), about 1,670 different activity chains can be found. Most of them appear rarely, therefore only

Pattern	Percentage	trips per pattern
h-l-h	27.668%	2
h-w-h	26.342%	2
h-s-h	16.593%	2
h-e-h	12.147%	2
h-w-l-w-h	3.066%	4
h-l-l-h	2.438%	3
h-w-s-w-h	1.751%	4
h-s-l-h	1.581%	3
h-l-s-l-h	1.092%	4
h-w-w-h	1.760%	3
h-s-s-h	0.884%	3
h-l-s-h	0.803%	3
h-l-w-h	0.723%	3
h-w-l-h	0.992%	3
h-w-s-h	0.794%	3
h-e-l-h	0.406%	3
h-s-w-h	0.474%	3
h-e-e-h	0.214%	3
h-l-e-h	0.116%	3
h-w-e-h	0.087%	3
h-e-s-h	0.068%	3
total	100.000%	weighted avg = 2.23

Table 4.1: Activity chain distribution of Switzerland after reduction from about 1,670 occurring activity chains in the Microcensus 2000. The characters are defined as: ‘h’ = activity performed at a “home” location, ‘w’ = “work” activity, ‘e’ = “education” activity, ‘s’ = “shopping” activity and ‘l’ = “leisure” activity.

the 100 most frequently occurring activity chains are considered, which cover more than 90% of all days.

In a next step, the given activity chains are reduced again, since they include special activity types: “Begleitung” (escort), “Service” (service), “Geschäftsreise” (business trip) and “Dienstreise” (traveling on company business). They are redefined as `work` activities. Therefore, the activity type set is the same as it is defined by the input facilities as described above.

At last, the activity chains should match at least one of the following requirements:

- Activity chains consist of 5 or less activities.
- Number of occurrences is at least 1% or more of all chains.

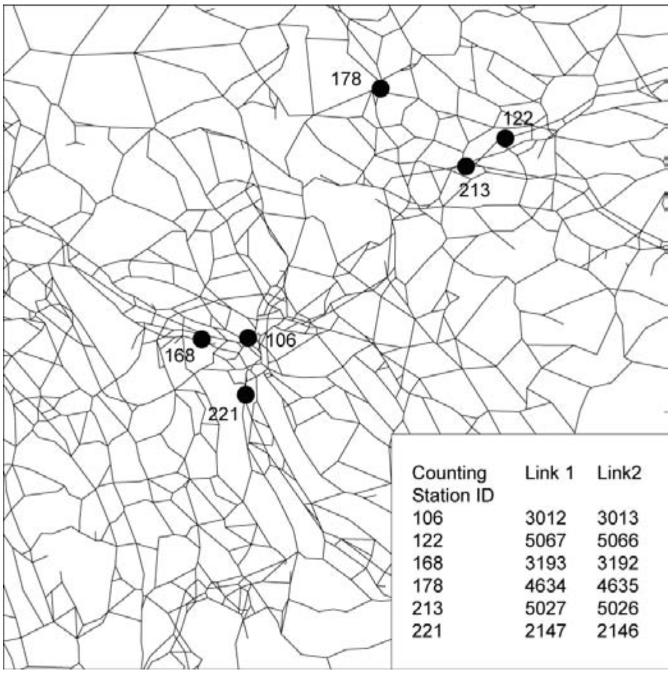
This way, 21 activity chains remained, representing nearly 93% of the 100 given activity chains. Table 4.1 shows the different chains with their frequencies. This step reduces the number of different activity chains to 21, so that forthcoming analysis is still manageable.

It is to mention that the chains do not adequately reflect the majority of the chains found in the micro census. Since in a later chapter comparisons of MATSim-T results with a VISEM/VISUM (PTV, 2006) assignment will be done, the constraints given by these two PTV products needs to be fulfilled. Since VISEM does not handle—at least in the version available at that time—“home” activities which occur sometime during the day[5], it was decided to ignore those activity chains found in the micro census. As a result, the weighted average number of trips performed per day is 2.23, whereas the actual number for Switzerland lies somewhere between 3 and 4.

4.1.7 Traffic Count Data

There are about 230 automatic counting stations registered with the Swiss Federal Roads Authority (ASTRA, 2006). Of those, hourly traffic count data for 75 stations are available. Unequivocally locations can be located on the network for only 33 counting stations. Unfortunately there are only 6 useful counting stations left in the study area. Since they are bidirectional, this means that it is possible to compare volumes for 12 links to reality as shown in Figure 4.5.

[5] Please note, that MATSim-T is not restricted to such tour-based activity chains. In fact, any kind of activity chains can be handled equally.



Counting Station ID	Link 1	Link2
106	3012	3013
122	5067	5066
168	3193	3192
178	4634	4635
213	5027	5026
221	2147	2146

Figure 4.5: Six counting stations (12 links) in the Zurich area.

4.2 Berlin-Brandenburg

Berlin is the capital of Germany with about 3.4 million inhabitants. It is also the smallest state in the country with 890 km^2 surrounded by the state of Brandenburg. Its size is about 29,500 km^2 but has only 2.6 million people. Compared to the captial, Brandenburg is weakly urbanized. Figure 4.6 shows the geographic location of the scenario.

The scenario of the German states Berlin-Brandenburg is somewhat different to that of the Kanton Zurich scenario because less precise information is available compared to the Kanton Zurich scenario.

4.2.1 Spatial Levels of Detail

Berlin-Brandenburg consist of counties (in Germany called “Kreise”) but for our purpose this resolution is not adequate enough, since the whole of Berlin is only one of them (see also Figure 4.7(a)). Therefore, a more detailed layer is needed.

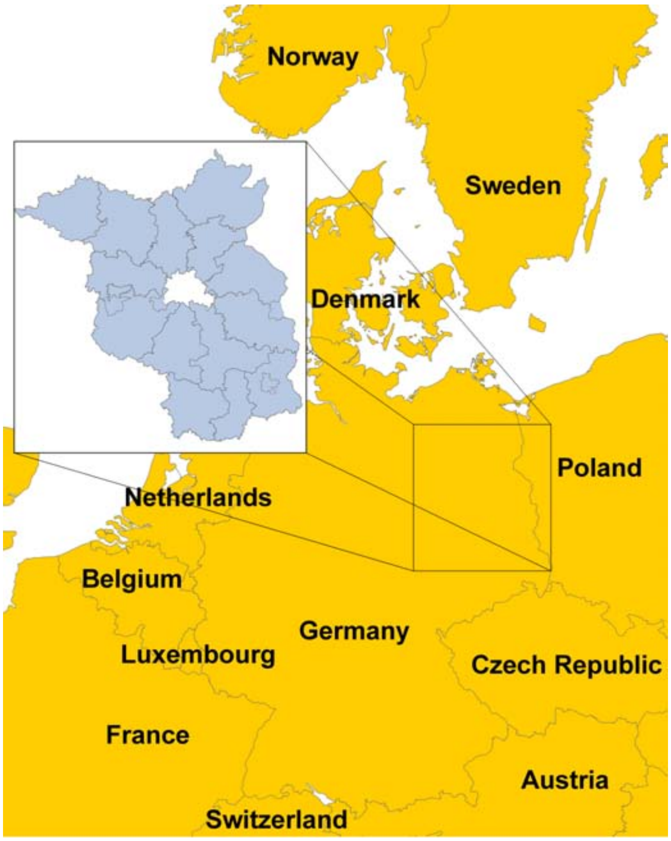


Figure 4.6: Europe, Germany and the states Berlin-Brandenburg

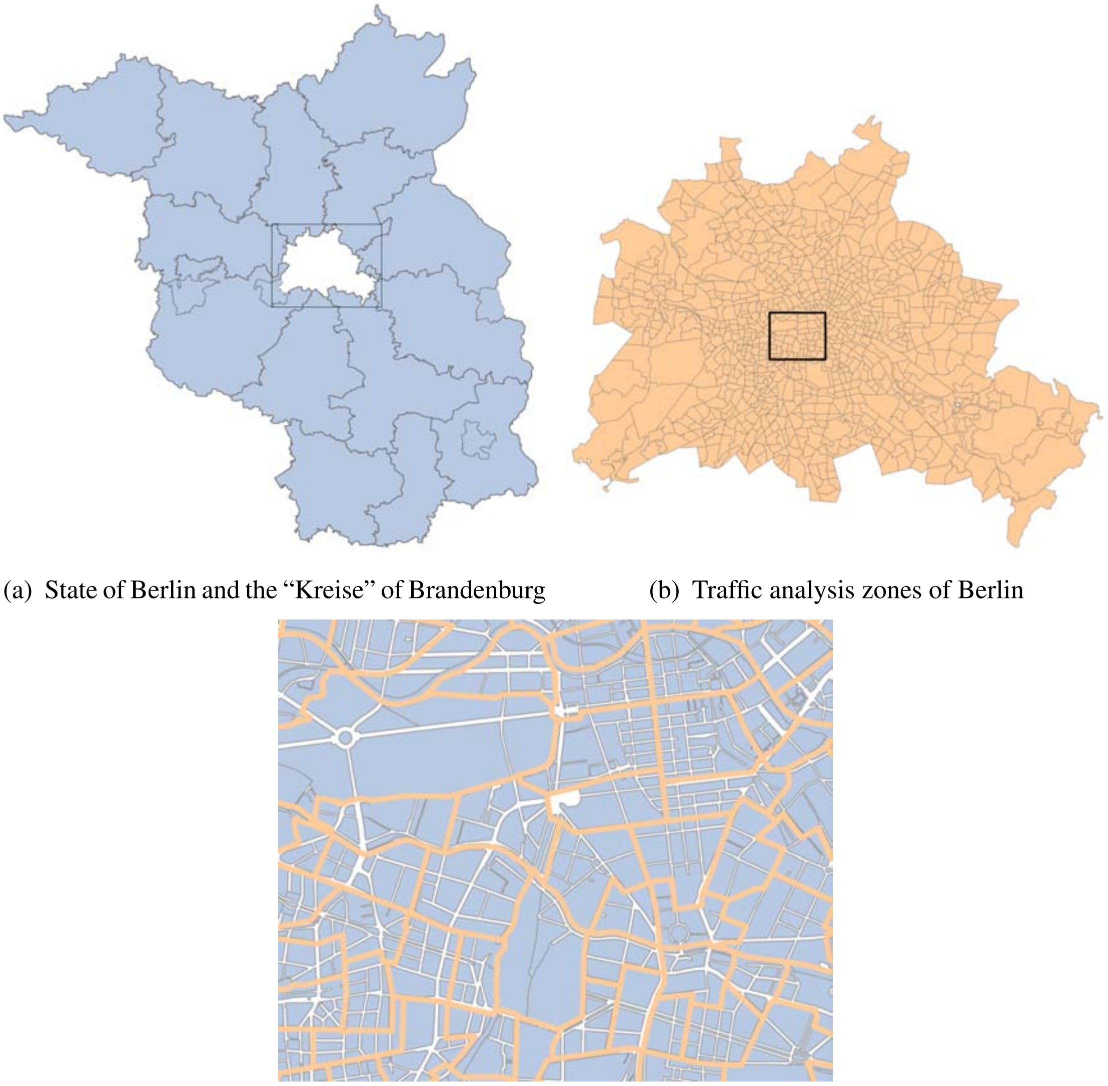
4.2.1.1 Traffic Analysis Zone Layer

Berlin-Brandenburg area is divided into 1,008 traffic analysis zones (RBS, 2001). The advantage of them over political units like municipalities is that traffic analysis zones are created according to demographic information. In the following chapters this layer will be referred as the *taz layer*. Figure 4.7(b) shows the traffic analysis zones of Berlin.

4.2.1.2 Block Layer

For the traffic analysis zones of Berlin there is access to an even more detailed resolution layer called *blocks*. Figure 4.7(c) shows the blocks of a part of the city of Berlin. 12,260 blocks are available for this scenario. This layer will be referred as the *block layer*.

Since there is no access to the same resolution in the state of Bran-



(a) State of Berlin and the “Kreise” of Brandenburg

(b) Traffic analysis zones of Berlin

(c) Blocks of a part of the Berlin City

Figure 4.7: The states of Berlin-Brandenburg, at various scales. (a) shows Berlin state (white) and the “Kreise” of Brandenburg. The box in (a) Berlin state is seen in (b), including the traffic analysis zones. (c) is a zoom of the box in (b). It shows the blocks of the area. The orange lines are the borders of the traffic analysis zones.

denburg, exactly one block is assigned to each traffic analysis zone in Brandenburg.

4.2.1.3 Raster Layer

Some demand modeling steps, specific algorithms need layers which are rasters. Neither the *taz layer* nor the *block layer* fulfils that requirement. So, a synthetically generated a third level of detail is generated which will be denotes as *raster layer* (similar to the high resolution layer of the Kanton Zurich shown in Section 4.1.1.2). On the basis of the rectangular extend of the *block layer*, a raster is defined, for which each cell has a size of 500 x 500 meters. The produced raster consists of 169,420 cells.

4.2.1.4 Mapping

Between the *taz layer* and the *block layer* an unambiguous mapping is generated: Each traffic analysis zone consists of at least one block. Each block belongs to only one traffic analysis zone. The additional blocks for the state of Brandenburg respect this rule by definition.

Since the *raster layer* is generated synthetically, also the mapping between the *block layer* and the *raster layer* is generated via a mapping algorithm. The mapping rule is the following: A cell of the *raster layer* refers to none, one, or many blocks while each block refers to at least one cell.

4.2.2 Facilities

Similar to the Kanton Zurich scenario, a land-use data set is available. For each given traffic analysis zone of Berlin-Brandenburg, it is possible to extract the capacities for the activity types “home”, “work”, “education”, “shopping” and “leisure”. Therefore, one aggregated facility per traffic analysis zone is available.

4.2.3 Network

The network of Berlin-Brandenburg is given as a VISUM network (PTV, 2006). It includes 5,408 nodes and 13,239 links. As one recognizes, this is an even lower resolution network than the Switzerland Network described in Section 4.1.3.1, while the spatial extent of the two networks are comparable. For the City of Berlin almost all major streets are included, while the street information for Brandenburg is poor.

4.2.4 Synthetic Population

Since this scenario describes each and every individual of Berlin *and* Brandenburg, it is necessary to create a synthetic population for the whole region. Unfortunately, at this time there is access to population information of Berlin only, provided by SLAB (2006). The spatial resolution of the 3.3 million individuals is high (based on the block level of detail as shown in Section 4.2.1.2), while the person description is not as precise as for the Kanton Zurich (Section 4.1.4). It consists only of the following items per individual:

- Person ID
- Age (split up into 8 bins)
  - 5 years old or younger
  - from 6 to 14
  - from 15 to 17
  - from 18 to 26
  - from 27 to 44
  - from 45 to 54
  - from 55 to 64
  - and 65 years old or older
- Home location based on block layer (see Section 4.2.1.2)

4.2.5 24-Hours Origin-Destination Matrix

The data resources of Kanton Zurich allow us to answer specific questions using specific data. For the Berlin-Brandenburg scenario, the number of useful data sets is much more limited compared to the Kanton Zurich scenario. While a commuter matrices is given for whole of Switzerland; at here, a 24-hour OD matrix, delivered by SfSB (1998), is available only. It describes—aggregated over 24 hours—all trips in Berlin-Brandenburg. That information is based on the traffic analysis zones described in Section 4.2.1.1. Note that there is no information about the purposes of the trips.

4.3 Summary

The two scenarios exemplify one important fact: Data sources of different scenarios differ in many aspects. One cannot assume to always get the same predefined specific data sets on which a transport model can be built up. Hence, it is mandatory that simulation systems for transport planning can handle this large variety of given input data.

Chapter 5

Complete Daily Plan

The results as presented in Section 2.5 show the ability to optimize an agent's route and time choice for a given daily plan by using the iterative optimization process of MATSim-EA. The resulting relaxed state of the scenario gives us the expected traffic volumes for the morning peak hours.

On the other hand, it is obvious that the predefined activity chains of that scenario are far from reality: An individual does not perform only work and home activities. In fact, the average number of trips made by a person lies around three to four location changes per day (ARE and BfS, 2001). Therefore, the artificial generation of a daily home-work-home activity chain does not produce a feasible solution for the whole simulated day, but only the morning peak.

This raises the question of modeling more realistic initial daily plans for the individuals of the given scenario. Since—at least for the moment—we optimize only routes and times of the given plans and use only the private transport mode, we need to be sure that the other, unchangeable parts of the plans are modeled as realistically as possible. The following list shows the points on which we need to focus when generating initial daily plans for a given scenario:

- Synthetic population generation (inclusive home location of each individual)
- Activity chain generation for each agent in the system
- Primary activity location choice

- Secondary activity location choice
- Modal split
- Initial activity timing generation
- Initial route generation.

Since the target is to model initial individual demand which reflects reality as well as possible, the above steps are based on input given by census, micro census and additional surveys.

The last two items are necessary because MATSim-EA needs initial timing and routing information to start the optimization process. The results of Section 2.5.3 show that—apart from computational performance issues and the limitation of the time mutation module mentioned in Chapter 3.2.1.2—the initialization of activity timing and the initial routes can be chosen randomly.

5.1 Initial Day Plan Generation Based on Time Dependent Origin-Destination Matrices

In a first approach, the combination of the demand generation of VISEM implemented by the PTV company (PTV, 2006) with MATSim-EA is desired. On one hand, the motivation is based on practical issues. Demand generation models used for many years typically produce traffic demand in the form of static or time dependent origin-destination matrices. To validate or compare the outcome of agent based micro-simulations like MATSim-EA against existing macroscopic models, one needs to be sure that both assignment models are based on similar demand. On the other hand, the work presented in this chapter is also motivated by the fact that time dependent OD matrices should be consistent in respect to the description of complete daily demand given for each individual of the scenario. Therefore, it should be possible to *construct each person's plan from the given time dependent origin-destination matrices*. This is in fact the core target of this chapter.

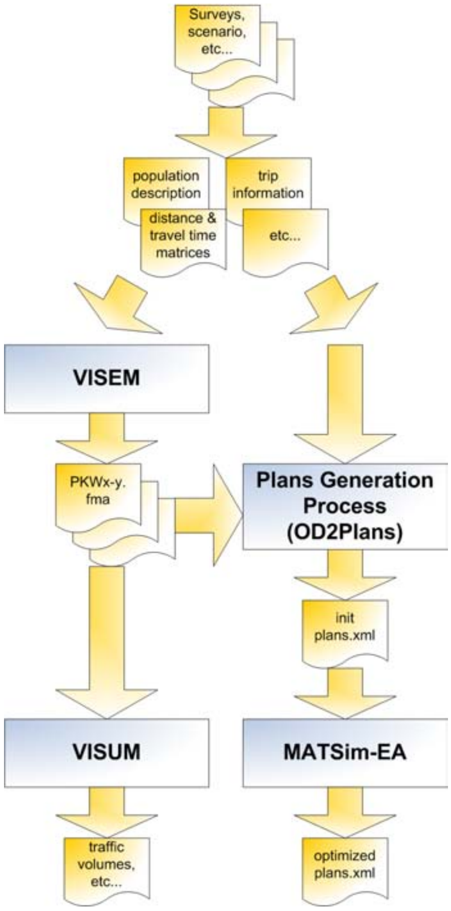


Figure 5.1: A schematic overview of the initial plan generation process using time dependent origin-destination matrices

Figure 5.1 shows a schematic overview of the data flow when the *Plans Generation Process* (in the following also called *OD2Plans*) is the process to reconstruct daily plans from given OD matrices. The production of the time-dependent OD matrices (indicated as *PKWx-y.fma*[1] in Figure 5.1) is described in detail by Rieser (2004a).

5.1.1 Input Data

In order to generate day plans out of OD matrices, additional data is required. The geographical location of zones is needed, as well as demo-

[1] Each one of the generated OD matrices represents the demand for a specific hour of the day for private transport. “x-y” indicates the hour, i.e. 8-9 means from 8 AM to 9 AM.

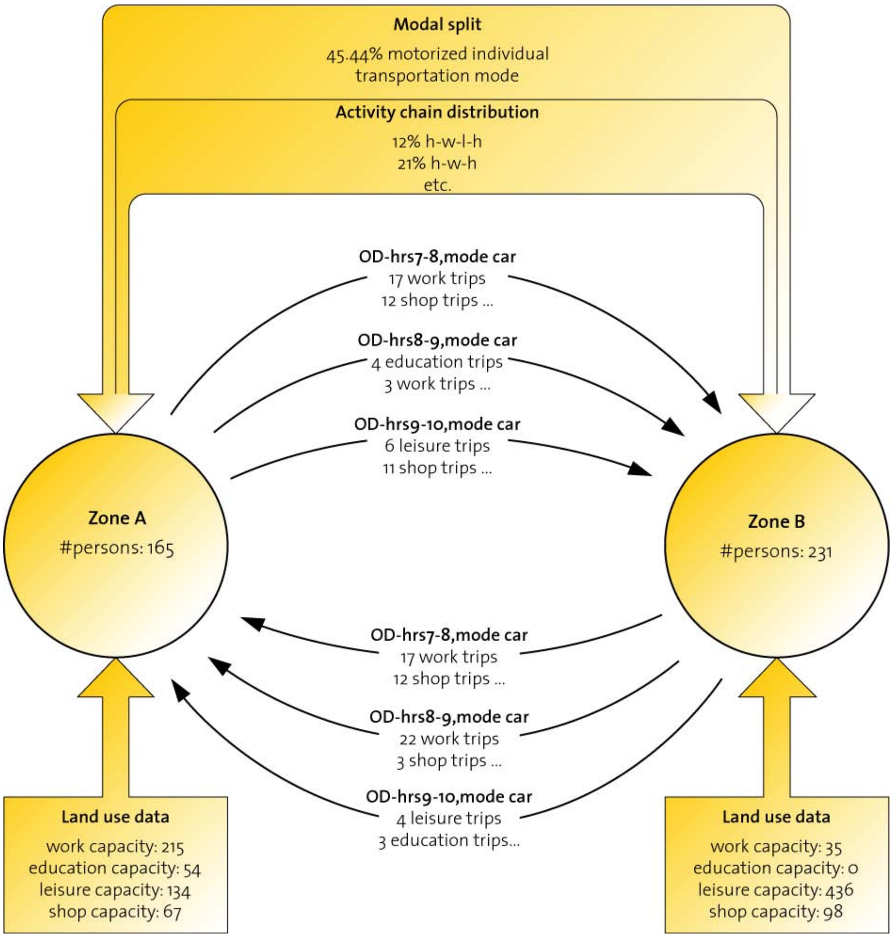


Figure 5.2: Summary of the used input data.
Source: Rieser (2004b)

graphical information and behavioral data of the population. Figure 5.2 shows a simple example of required (and available) data.

The input data is stored in multiple text files. The format of the data is either compatible with VISEM or consists of space-delimited values. The results of VISEM can be used by VISUM to do a static assignment of the traffic in the OD matrices. As the input for MATSim-EA is essentially based on the input and output of VISEM, meaningful comparisons between the outcome of the two programs can be done.

5.1.2 Target

MATSim-EA accepts different kinds of plans: from simple one-trip-plans (corresponds to OD trips) to full daily activity chains. But using the advantage of enhanced re-planning rules for individuals' daily chains is favored. The first activity should include an end-time which defines when the first trip of the plan starts. In the case of the daily chain, the last activity does not need any timing information because it defines the end of the day performed. All other activities must have a defined duration. The location (given as x and y coordinates) and the activity type has to be included for each activity.

5.1.3 The Process

The process creates the MATSim-EA input plans for each user-defined mode separately (at the current state of MATSim-EA, private transport mode is the only one implemented). Figure 5.3 shows the process used to create the activity chains for the specified mode. Each step uses specific input files and also produces well defined output files. Each step can be started independently, which makes it easier to extend the process with additional steps, methods, or information. This section describes the different files first, followed by descriptions of each step:

villages.txt Holds the 182 municipalities / districts, their names and the coordinates of their centroids as described in Section 4.1.1.1.

population.txt Holds the number of persons for each population group and each municipality. In contrast to Section 4.1.4, this information is extracted from the VISEM input[2] (see Rieser, 2004a, for more details).

patterns.txt Holds the activity chain distribution as presented in Section 4.1.6.

translation.txt correspondence list between German and English activity codes.

[2] Since the results generated with the process described in here will be compared against the outcome of VISUM, the population description should be the same for both processes.

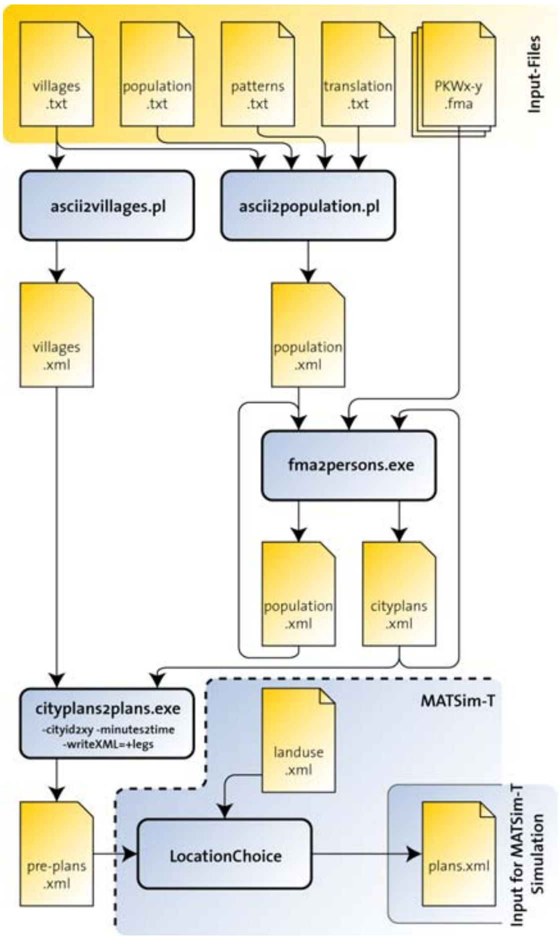


Figure 5.3: The plan generation process in detail.

villages.xml The same information as in villages.txt but in XML format.

fma files The hourly OD matrices for a specified mode generated from VISEM.

population.xml Contains the persons by municipality who do not have an activity chain assigned yet.

cityplans.xml Chains for persons described in XML-plans format, but the activities do not have a location assigned yet. However, each person retains the information about their home location and the specified mode.

pre-plans.xml Chains for all persons described in XML-plans format, but only the primary activities have a location assigned.

landuse.xml This file contains land use data as shown in Section 4.1.2.1. It essentially describes the attractiveness of each area. Since for Switzerland land use information is available for hectares (100 meters times 100 meters), the land use information also refers to that resolution. Attractiveness is described as the number of opportunities by activity type.

Furthermore, the modal split is given to reduce the number of trips by the specified mode. But since trips are tightly bounded to the given activity chains and therefore to the population, the modal split is used to reduce the population rather then the trips.

5.1.4 Steps of the Process

This section describes each step of the process shown in Figure 5.3 in detail:

ascii2xml.pl This is a pre-process to convert the villages.txt file to XML format.

poppatterns.pl This is also a pre-process which calculates the number of persons by municipality with a particular activity chain and by the specified mode. The given distribution of the activity chains is used to assign a chain to the persons. Notice that minor rounding errors of the number of people can occur because of distributing the population to the municipality, reducing the population according the given modal split and assigning the activity chains.

peopleGenerator This is the main step of the process. It sequentially goes through each hour of the OD matrices for the specified mode and generates partially completed chains.

The peopleGenerator process reads (sequentially, hour by hour) one line after the other from the OD matrix. Every cell of the OD matrix represents a number of persons which travel from one location to another. The algorithm selects those persons from the population.xml file of the origin of the OD trip such that their primary activity matches the purpose of the OD trip. The match also defines the start-time (given by the current hour of the OD matrix). Since each matrix represents a full hour of the day, the chosen start-time is calculated as a uniform distribution over that hour. Also, the number of people in the population list is decreased by the number of assigned persons. Recall that every person already has an activity pattern assigned from the pre-processing stage.

This module is used only to assign locations to the primary activities. In this simplified approach, there is only one primary activity per chain. If there is more than one out-of-home activity (peopleGenerator recognizes only 'work', 'education', 'leisure' and 'shop' out-of-home activities at the moment), then one of them is declared as the primary activity. The idea here is to select the location of the primary activity based on the VISEM output, and then use some other module to select secondary activity locations. This leaves the question of defining the primary activity when there are multiple out-of-home activities in a row. The following simplified approach was adopted:

The first occurrence of "work" is set as primary. If "work" is not part of the chain, the first occurrence of "education" is set as primary. If "education" is not part of the chain, the first occurrence of "leisure" is set as primary. If "leisure" is not part of the chain, the first occurrence of "shopping" is set as primary.

As the count of available people is decreased while reading one OD matrix entry after the other, it can happen that there are no more people available in a location to fulfil the travel predicted by VISEM. In this case, no new plans are generated.

Every chosen person together with their plan is written to the population.xml file. When writing, each activity gets a "standard" duration assigned. If an activity type occurs more than once in a plan, the duration for each single activity is divided by the number of occurrences of the activity type. This way, the sum of durations of all the activities of one type in the plans matches the assumed default durations.

Even though those assumptions are sensible, sometimes the durations can be quite wrong. For example, the duration of the shopping activity in the activity chain 'home-shop-home' could be just some minutes (i.e. buying a bread at the bakery next door) or it could also be about 10 hours (i.e. a shopping day at the mall). The same question applies for the leisure activities. On the other hand, the average duration of "work" and "education" usually has less variance. Nevertheless, for lack of more detailed assumptions the generation process will be employing these assumptions for the time being.

cityid2xy Until now, all plans have the attribute "cityid", which MATSim-T does not recognize. Instead, it expects the x- and y-coordinates for a location. *Cityid2xy* achieves this by choosing a random point within the given municipality, thus disaggregating the locations.

To choose random points, the algorithm simplifies the shape of cities into circles. For each municipality, the algorithm searches for the nearest neighbor and takes half of that distance as the radius for a circle around the respective center. Using this procedure, the areas of cities will not overlap each other.

The *cityid2xy* step is a simplified approach to define coordinates within a municipality. The random choice within the actual borders of a municipality (typically a polygon) would give a much more precise distribution.

LocationChoice MATSim-T contains an external module (see Marchal and Nagel, 2006) which chooses locations for secondary activities in such a way that each agent improves its daily chain. The home-location and the location of the primary activity remain unchanged.

The application uses a variant of a genetic algorithm to determine the locations for each agent. Each agent (based on the previously created chain) has limited knowledge about places in the area of interest and searches within these places for a path, such that his chain improves its utility. Additionally, each agent knows some other agents and exchanges information with them about places after every iteration. This way, a social network is built that helps the agents improve the utility of their chains.

The original application as described in Marchal and Nagel (2006) assumed that all plans contain the activity “work”, which was interpreted as primary activity. Additionally, the application expected every chain to have one or two secondary activities. As the activity chains as shown in Section 4.1.6 do not fulfil these restrictive requirements, the application was modified. In a first step, it was ensured that short plans with no secondary activities (i.e. home-work-home) were processed correctly. Such chains will now be ignored for the iterative calculations, and left unchanged.

In a second step, the code was modified to respect the primary tag for activities instead of checking whether an activity is of the type “work”. The primary tag is set by the process described above.

After these modifications, the *LocationChoice* module is able to generate the missing locations for the secondary activities for the plans given by the *cityid2xy* process.

It needs to be pointed out one important fact about this location choice process: Even though it does use land use data, it does not respect the constraints given by that. In other words, it can happen that more people are doing their shopping activity at a given raster element at the same time than the capacity of the shop allows.

Also, it must be mentioned that this whole plan generation process often does not reach a perfect match with the given OD matrices. As an example, the output matrix for a whole day generated by VISEM as explained above prevents a perfect match: for many municipalities, the number of trip starts differs from the number of trip ends making it impossible for persons to be at home at the end of a day. On the other hand, the activity chain generation process generates only round trips, so the number of trip starts and trip ends will be the same in each village.

5.1.5 Comparison

In this section, the results of an assignment procedure done with VISUM (PTV, 2006) using the above described OD matrices are compared to a simulation run with MATSim-EA sa described in Section 2.4 using plans generated by the above described plan reconstruction process. In addition, a direct conversion of the OD matrices into one-trip-plans was performed and simulated with MATSim-EA. The results of this simulation run will also be compared to the VISUM results. The comparison between VISUM and MATSim-EA will be performed only for the motorized private transport since MATSim-EA is only able to handle this mode at the moment. Both simulations are using the same underlying street network shown in Section 4.1.3.1. The comparison of the outcome of the simulations will be done for twelve links (see Figure 4.5) of the given network where also count data is available (ASTRA, 2006).

5.1.5.1 Data Used

The case study uses the population described in Section 4.1.4. Using a modal share of 45.44% for motorized individual transport, more than 500,000 persons are simulated. The activity chain distribution shown in Section 4.1.6 is used.

5.1.5.2 Traffic Count Comparison between MATSIM-EA, VISUM and Count Data

The VISUM output and the traffic counts are the references to which the volumes produced by the different MATSim-EA runs will be compared. MATSim-EA is known to produce results similar to VISUM. Nevertheless, a similarity check is also done by a one-to-one conversion of each trip of the OD matrices into "one-trip-plans", meaning each trip corresponds to a separate person. Of course, those are not "real" plans but the traffic produced should be similar to the one VISUM generated. We ran the chain generation process that uses the OD matrices three times with the different average duration assumptions. The three generated plan files are used as initial plan files for MATSim-EA iterating 50 times by using only the route re-planning[3] module (Raney and Nagel, 2003). Therefore the following results will be compared to the VISUM reference output:

[3] Since the plan reconstruction process described in Sec 5.1.3 also reconstructs time aspects of daily plans, the agents of the system are not allowed anymore to adapt departure times and durations of activities.

OD2Trips Direct conversion from trips to one-trip-plans.

OD2PlansW8 Activity chain generation with average durations: 8 hours work, 4 hours education, 4 hours leisure, 1 hour shopping.

OD2PlansW9 Activity chain generation with average durations: 9 hours work, 6 hours education, 5 hours leisure, 2 hours shopping.

OD2PlansW10 Activity chain generation with average durations: 10 hours work, 8 hours education, 6 hours leisure, 3 hours shopping.

To calculate adequate scores (utilities) for each executed plan (Raney and Nagel, 2005), MATSim-EA requires constraints on each type of activity. In this case we define them as follows:

- The work and education opening time is set to 7 AM.
- The work and education closing time is set to 6 PM.
- The latest work arrival and education arrival time for the first work activity, education activity resp. is set to 9am.
- The shop opening time is set to 8 AM.
- The shop closing time is set to 8 PM.
- The leisure opening time is set to 1 AM.
- The leisure closing time is set to 11 PM.

5.1.5.3 Hourly Volumes

Figure 5.4 and Figure 5.5 show the hourly volumes of four of the twelve links for which counts are available. All four curves of the VISUM outcome (bold red curves of Figures 5.4 and Figure 5.5) show three peaks. The volumes differ substantially from the hourly volumes recorded at the count stations (grey area of Figure 5.4 and Figure 5.5). This obvious bias during noon time is the result of the modeling process of VISEM. It produces more trips than depicted by other research studies of that area (i.e.

from the "Pendlermatrix" as mentioned in Section 4.1.5). A detailed discussion about that outcome can be found in Rieser (2004a). Therefore, a comparison to real world data does not make sense until the calibration of VISEM has been improved. Nevertheless, we are still able to compare VISEM/VISUM with the four resulting traffic volumes produced by MATSim-EA.

The comparison between VISUM and *OD2Trips* (bold blue curves) shows similarities comparable to those reported in Raney *et al.* (2003). That again confirms that VISUM and MATSim-EA produce comparable results, since the input is—in fact—identical. At a first sight, the hourly volumes of OD2PlansW8 (light grey curves of Figure 5.4 and Figure 5.5) differ strongly from the VISUM volumes. This is not really surprising since the activity durations are only assumptions. Anyways, some similarities are still observable:

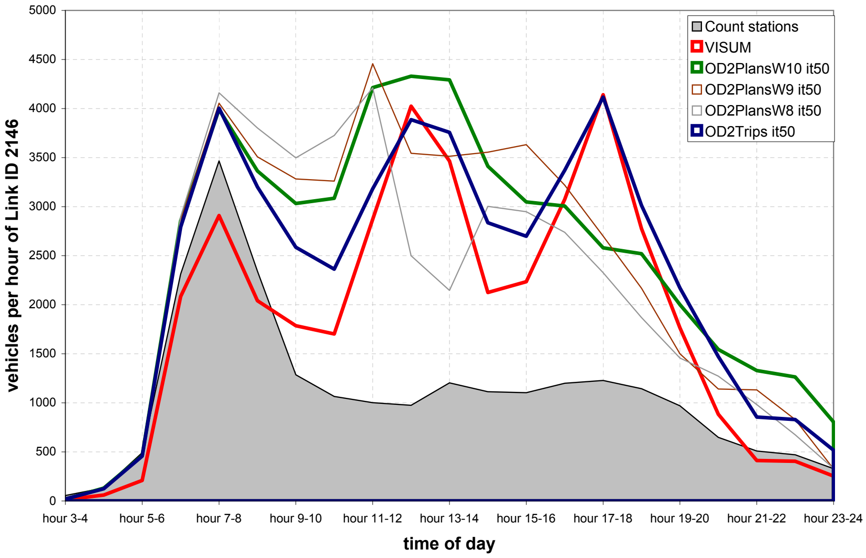
- The three peaks are present in both curves.
- The morning peak matches pretty well.
- The other peaks appear too early in the day (noon peak around 11am, evening peak around 4pm).

This leads us to the other two duration definitions (*OD2PlansW9* and *OD2PlansW10*). Especially *OD2PlansW10* (bold green curves of Figure 5.4 and Figure 5.5) matches the first two peaks quite well. Also, the third peak appears at the right time, but with too low volumes, while later in the night, the volumes are too large.

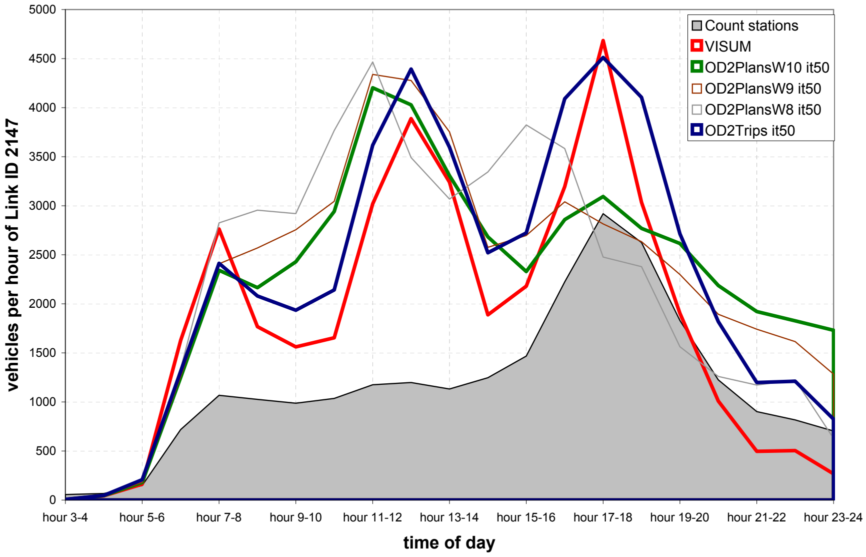
The reason for this difference is again in the duration definition. The duration of a chain can differ between 3 hours (h-s-h chain) to 16 hours (h-w-l-w-h). If more activity chains were used, the duration of a chain could go up to 27 hours (h-w-l-e-s-h)! In one sentence, short chains usually produce too short out-of-home durations while the reverse is true for longer ones. The results are still surprisingly good for the crudeness of the assumptions made.

5.1.5.4 Comparison to Land-Use Data

The plan generation process described in Section 5.1.3 uses the location choice module of Marchal and Nagel (2005). As mentioned, it does not



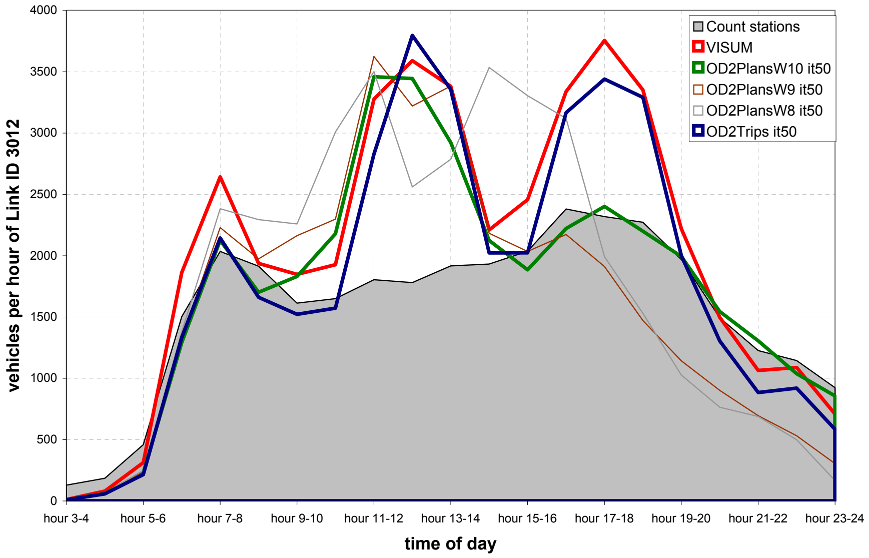
(a) Hourly traffic volumes of link-id 2146



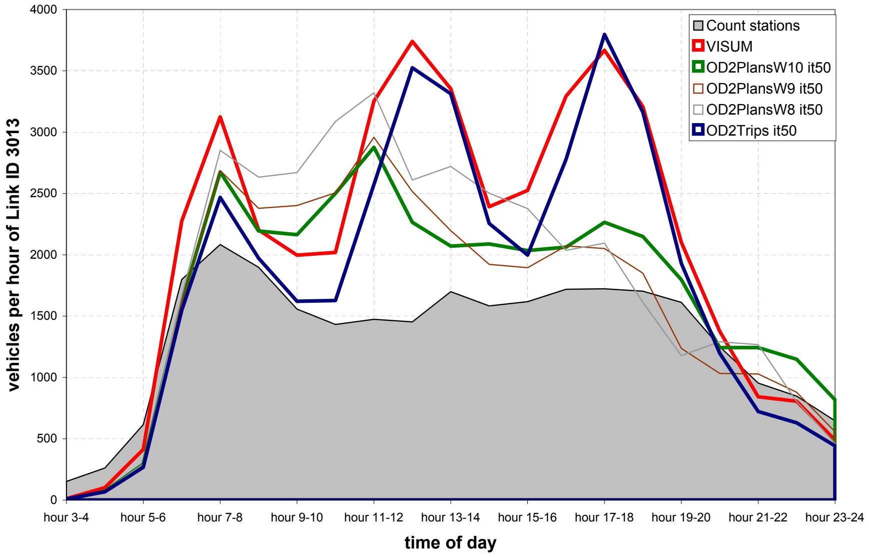
(b) Hourly traffic volumes of link-id 2147

Figure 5.4: Comparison of hourly traffic volumes produced by VISUM, the four defined MATSim-EA runs and traffic count data.

respect the capacity constraints of work places, shopping areas, etc. This section analyzes what the differences are between the given land use data and the occupancy generated by the processes. Figure 5.6 shows the labor



(a) Hourly traffic volumes of link-id 3012



(b) Hourly traffic volumes of link-id 3013

Figure 5.5: Comparison of hourly traffic volumes produced by VISUM, the four defined MATSim-EA runs and traffic count data.

force density for each municipality of the Zurich area. One can clearly see that Zurich itself and also the region of Winterthur have a higher density. The square indicates the area in which comparison of occupancy

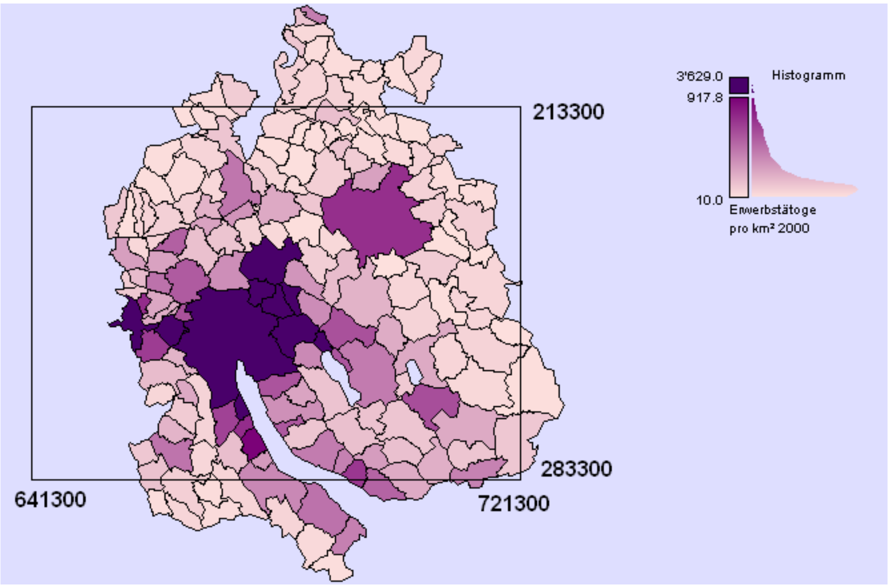


Figure 5.6: Labor force density of the municipalities of the Kanton Zurich

Source: Stochastisches Amt Kanton Zurich (http://www.statistik.zh.ch)

against land use data will be done.

Since both demand generation processes use only 45.44$\%$ of the whole population, we need to scale up the resulting occupancies to 100$\%$ again, otherwise a direct comparison to land use is not complete. Figure 5.7 shows the difference in occupancy generated by the process and land use, split up into four of the five activity types, namely “home”, “work”, “shop” and “leisure”. The resolution is of $2km \cdot 2km$ square. The light blue areas indicate that the capacity given by the land use is not reached by the process while the dark red area shows that the process produced occupancy is higher than the given capacity.

The land-use data compared with the *OD2PlansW10* run shows that in all four activity types, the location choice produces higher occupancy than there is available space for the City of Zurich and partially also for Winterthur (see Figure 5.7). The other regions are underestimated (especially for ’work’ activity).

It is to mention, that there is a remarkable bias between person’s home activities and the land-use data. It indicates that the synthetic population does not fit well with the given land use information. One reason for that could be, that the population is generated with other data re-

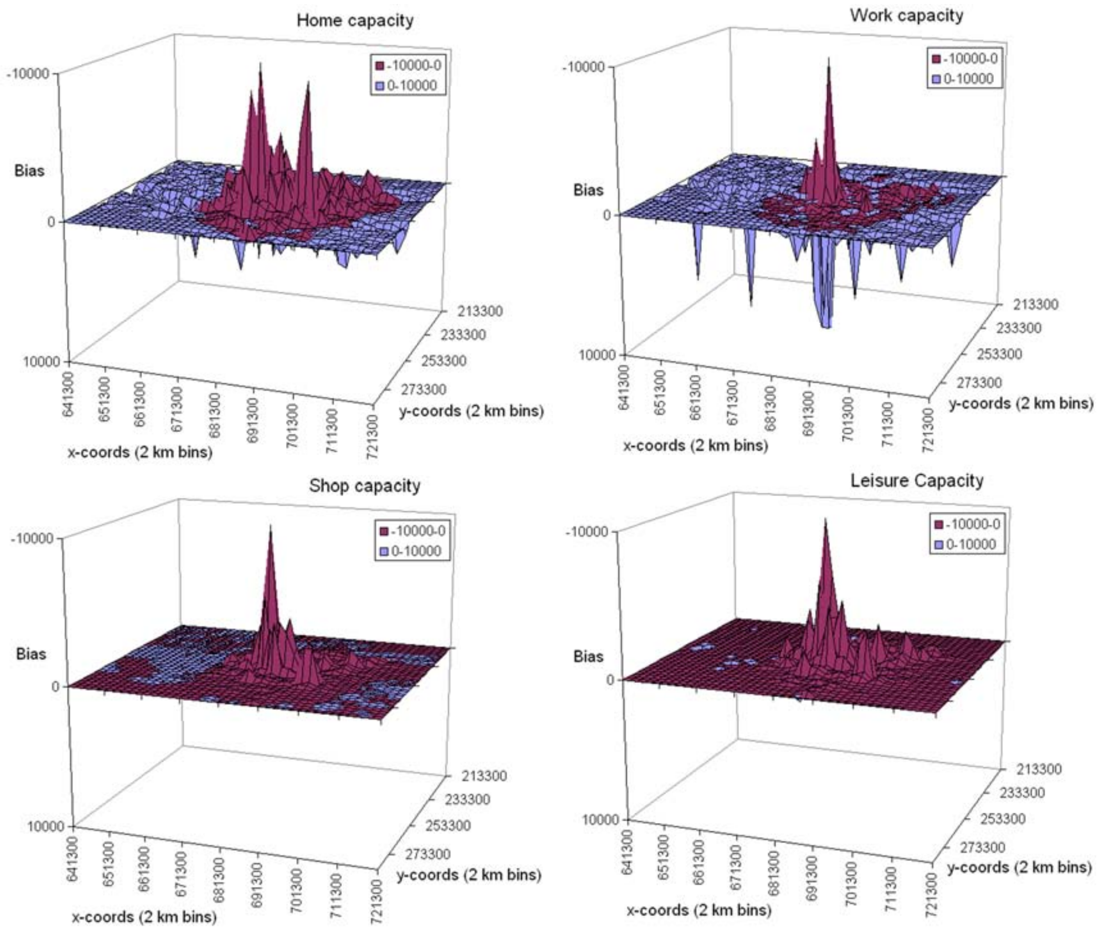


Figure 5.7: Difference between given land use data and *OD2PlansW10* setup.

sources (e.g. land use information on municipality level of detail).

Also of interest is that in downtown Zurich there are much fewer work activities than expected. The reason for that could be the high peak in the north-eastern part of the city of Zurich. The location choice process of VISEM more often places work activities there because the Zurich industry is found in this area. The type 'leisure' is almost only chosen at the lakeside of Zurich, meaning that almost everybody travels to the city of Zurich for recreation, which is not true in reality, since the land use clearly shows that there are other attractive recreation areas in the Kanton of Zurich.

To summarize, it is unfortunate that location choice processes are placed on different parts of the demand modeling. The bias for “home” activity probably comes from the discrepancy of the given input, differences of “work” occupancy happen during the VISEM processes, while the bias for “leisure” and “shop” activities are results of the secondary location choice module.

5.2 Discussion

This chapter shows an algorithmic reconstruction of daily traffic demand of each individual in a given area. Given origin-destination matrices on an hourly based resolution and some additional demographic data, this process is able to reconstruct a daily demand for each individual in the examined region. To a certain degree, it can be shown that traffic produced by the generated daily demand reflects the traffic produced by the OD matrices with the advantage that the daily demand is based on individual plans for each agent.

Nevertheless, the process uses some assumptions, in particular typical durations for the generated activities. As shown in Section 5.1.5, several runs have to be done to find out which assumptions produce the best match compared to the given matrices. And it is difficult to see if the above chosen durations could be further optimized. The location choice process by Marchal and Nagel (2005) which is used to calculate the locations for the secondary activities has the disadvantage that it does not respect the constraints given by the land use data. It should be modified or substituted by a better version of that process step. Since the general activity chain generation process consists of several independent components, it is quite easy to replace parts of the process by enhanced models.

Some computational issues have to be mentioned, too. The activity chain generation process needs about one hour to generate daily demand for the given population on a single 1.8 GHz CPU. Additionally, about 4 hours is needed to generate the secondary activity locations. MATSim-EA needs about 18 hours for 20 iterations (reaching a relaxed state using only route re-planning described in Balmer *et al.*, 2004b) on a Beowulf cluster (Beowulf.org, 2006) using 8 CPUs. Therefore, the whole computation can be done in about 1 day.

5.2.1 Open Questions

If we now separate each individual aspect of the initial individual demand modeling process presented in this chapter, the following questions still need to be answered:

MATSim-EA optimization process As shown in Section 2.4, the optimization process including time-choice should be used instead of only route-choice. But this leads to computing time issues, since the process just takes too long.

Activity duration definition of MATSim-EA Section 2.4.6 requires t_i^* for each given type of activity. As the results show, there are still open questions about setting these parameters.

Since in the last years further information has been extracted about actual behavior of individuals (i.e. person diary surveys as presented in Rai *et al.*, 2007), we are able to set these "typical duration" parameters better. But this leads us directly to the next item.

Available input data The data which can be used to generate initial individual demand varies strongly from scenario to scenario. Therefore, it should be possible to vary the process according to the given input data. But the process above is strongly connected to VISEM input and output data. Even more, it is not that clear what mechanism has actually been used to produce the resulting demand matrices.

To be more independent—and therefore more flexible in aspects of data available—it would make much more sense to directly use all given information of the region of interest.

MATSim-T process integration Finally, the process of initial individual demand modeling is actually not part of MATSim-T yet, but rather a pre-process. A closer integration is preferable, since both parts are based on the same information.

The following two chapters will show a way how the above mentioned questions can be answered.

Chapter 6

Activity Based Individual Demand Modeling

In contrast to the previous chapter, in which the focus was the reconstruction of individual daily demand via OD matrices, the target here is to model *activity based individual demand* as a completely integrated procedure that fulfills the following tasks:

- It should be flexible enough to handle various, qualitatively different input data for the scenario of interest. Therefore, the system must be able to integrate, fuse and validate the given data, resulting in a well defined data representation of the scenario.
- In order to have full flexibility to vary the fusion process of the input data, a simple interface to the data points has to be given, such that the fusion process can be easily adapted.
- To provide the same flexibility to the individual demand optimization process, again a simple and well defined interface to the data must be provided to vary the optimization process itself.
- The order in which the process steps will be used should be variable, too. Furthermore, the system should allow one to reuse any process steps again.
- At any point of the demand modeling process, the state of the whole scenario can be exported into well defined data files, such that third party modules can be used for the modeling process.

If one compares the above mentioned items to MATSim-EA as presented in Section 2.4, it already fulfills the last three requirements. The agent database (Section 2.4.5) reads, stores, and writes well defined XML files (see Section A.1) and keeps them consistent; the two replanning modules (Section 2.4.2 and Section 2.4.3) read, modify, and write them, too; and the SQSim (Cetin, 2005) reads them again, including the required network, and produces events, which then can be used to adapt agents' plans. Everything here is completely file-based. It has the advantage that any other module can be added to the system, written in any preferred programming language, as long as it respects the input and output definition.

But this system also has some disadvantages. First, basic functionalities, like file parsers and readers, basic calculation routines, etc., must be—in principle—re-implemented for a module. Second, the system produces an enormous amount of file I/O, which slows down the process. Third, the integration of modules into other ones is difficult (sometimes impossible). Last but not least, there is completely no guaranty that the modules act on the same data basis.

The version of MATSim-T presented in this chapter is written in the JAVA programming language, in contrast to the version as shown in Section 2.4, which is written in C++. Apart from improved robustness of code, there are two main reasons for this change. First, because the target is to build a fully integrated system, the freedom of choosing the programming language for implementing a module, as mentioned before, will be lost. Therefore, a language which is widely known in the transport planning community is preferable. Second, JAVA is platform independent, which means that it can be implemented on one operating system and used on another, which is—in fact—much more difficult to do in C++. As a result, the user does not need to change his preferred operating system to make use of MATSim-T.

This chapter presents the fully integrated approach of MATSim-T, fulfilling all above mentioned issues. This chapter focuses on the data preparation aspects and the generation of the *initial individual demand* for each agent of a scenario (the same target as presented in Chapter 5). With respect to Figure 2.1, we therefore will present the conceptual structure of MATSim-DB, MATSim-Parsers/Writers, MATSim-INI and MATSim-DATA, while we will use MATSim-EA (see Figure 2.2) as an

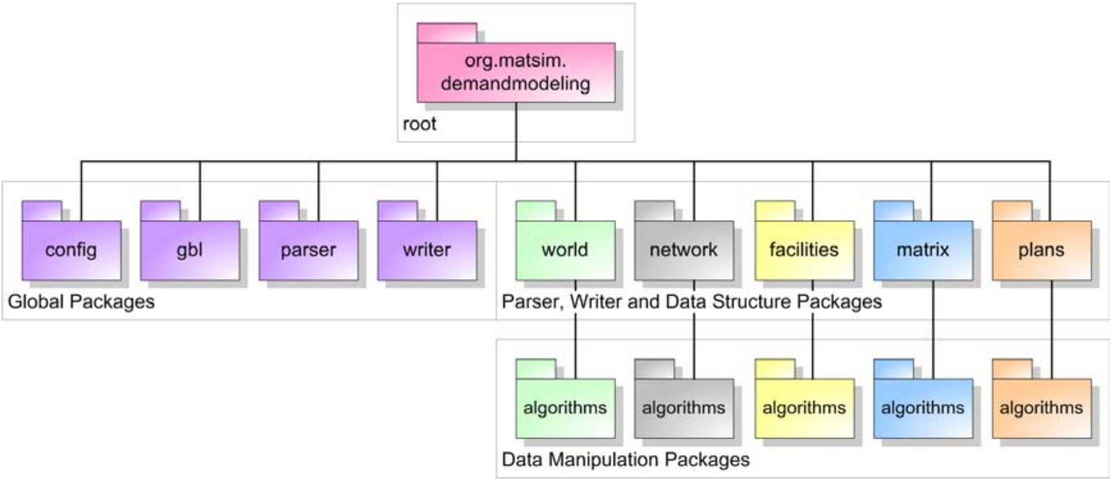


Figure 6.1: The JAVA package structure of MATSim-T.

external module in the same way as described in Section 2.4.

6.1 General Structure

The general structure is based on the concept of JAVA packages as shown in Figure 6.1. Each package fulfills a specific function; some of them provide basic functionalities, while others keep global information. The packages *world*, *network*, *facilities*, *matrix* and *plans* actually define a well defined data structure where each data point represents one aspect of the scenario of interest. For each of those packages, an *algorithms* subpackage is defined. It keeps a collection of subroutines which manipulate the appropriate data structure.

6.1.1 The MATSim-T Packages in Detail

The parser, writer and gbl package The parser package provides a general base class for parsing XML data via the SAX parser (SAX, 2006). The writer package is the base for writing XML files. The purpose of the gbl package is to hold global constants and globally accessible functions.

The config package In the whole framework, exactly one configuration data structure exists which holds all required input parameters for a specific demand-modeling process. Typical parameters are locations of input data, different file formats, special function parameters, etc. The information stored in this package can be accessed from every part of the framework (global singleton design pattern, see Gamma *et al.*, 1995).

The world package The world package has a special functionality. It describes the region for which the demand is to be modeled. Therefore, it must guarantee that only one world exists during the demand-modeling process. This package holds all information about cells, blocks, zones, municipalities, etc. which are modeled as primitive shapes. If the other optional packages refer to such shapes, they have to point to them in the world. With this constraint, uniqueness of each shape object is guaranteed. Even more, it is also a control mechanism to verify that other input data is consistent with the given scenario.

Another important functionality of the world package has to be mentioned. During the demand-modeling process, the resolution of the world can differ. For example, land-use information might be based on a raster of $100 \cdot 100m$ cells, while commuter matrices are based on traffic analysis zones. Therefore, it is necessary to (dis)aggregate one data set into another spatial resolution layer. The world holds this mapping for an arbitrary amount of resolution layers of the same region. With it, on the fly (dis)aggregation is provided. The mapping between two resolution layers can be generated either by the toolkit, according to a specified mapping rule, or given as predefined input data. The mappings must produce a *tree relation* between the different layers. Otherwise, the (dis)aggregation can produce ambiguity (i.e. in a circular mapping relation between layers).

Therefore, this package plays a central role in the toolkit, since it is the “glue” which holds the different data resources together.

The network package MATSim-EA is based on a (street) network. Agents travel on a network and they perform activities at a link of the network. Therefore, the network is also a spatial resolution layer which can be added to the *world*. The mapping will be always produced by the system, since it is desired to keep the flexibility to exchange one network by another to produce the same demand on different networks of the same scenario.

The facilities package It stores land-use information. Either it represents actual facilities (i.e. a shopping mall) with a specific coordinate, or each facility represents the union of all facilities for each zone of one of the given layers defined by the *world*. Therefore, the system either maps "coordinate-facilities" to one of the resolution layers, or uses the mappings stored in the *world* to (dis)aggregate "layer-facilites" to another one (including the network resolution layer).

The matrix package The matrix package stores directed relations between zones of a layer defined in the *world*. Each matrix entry consists of a source and sink zone and a weight. The semantics behind a matrix is intentionally left out, so the user (his own implemented demand modeling step) can decide the purpose of that information. A typical example is a commuter matrix, describing the relation between home and work of the commuters of the region. But the matrix package could also hold information defining relations between facilities located in the zones (i.e. a company with their subsidiaries). The system allows one to store an arbitrary number of matrices in the data structure, accessible by user defined unique identification strings.

The plans package This is—in principle—the core of MATSim-T. It is used as a "working file", utilizing the extreme flexibility of the MATSim-T XML plans[1] file (Appendix A.1). In the minimum version, the file holds only the identity number (id) of all simulated persons. On the other hand, it is possible to add a large amount of information about each person to the file, such as age, sex, car ownership, home and work location (at different world resolutions), etc. Additionally, each person can access one or many different individual plans, describing when a person wants to start

[1] The word "plans" is misleading, since it describes the population of the scenario and not only their plans. The reason we call it "plans" is historical and should be changed into "population file".

> an activity, where it will be performed, which route and mode to take to go from one location to the next, etc. The internal data structure of the plans package provides exactly the same flexibility as the XML file format. Therefore, it is possible to sequentially add additional plan details to a given incomplete MATSim-T plans file.

All packages (apart from the *parser*, *writer* and *gbl* package) are based on an XML data file. Each of them provides a unique data structure similar to the defined input XML file. They also provide a specific parser and writer (inherited from the *parser* and *writer* package) to store the data in memory or dump it to a file again.

Furthermore, the system provides uniqueness of the data points. This is an important feature, as redundancies can cause inconsistencies. If a demand modeling process step wants to manipulate a specific data point, the concept of uniqueness guarantees that the steps following it will automatically receive the changed information and not a copy of the data.

This concept causes clear dependencies of the data structure. For example, it is not possible to store matrix information without defining a resolution layer of the *world* package, since a matrix entry always refers to a zone of a layer. On the other hand, it is not mandatory to use the matrix package. If the demand modeling process can be done without such matrices, that package can be ignored.

Of course, MATSim-T in this implementation phase does not provide data interfaces for all information which could be available for describing a scenario. If one wants to use, e.g., information about social networks (i.e., Hackney and Axhausen, 2006) the system allows one to add more and more packages in the same manner as already implemented for facilities and matrices.

As one can see, these packages are actually the MATSim-DB, MATSim-DB Parsers and the MATSim-DB Writers as shown in Figure 2.1. Next, an interface for each package is defined such that models, programs and algorithms are able to manipulate any data point of MATSim-DB.

6.1.2 The Algorithms Packages

Algorithms can be added to each package to verify, manipulate, add, or delete data items according to the purpose of the algorithm. Since different algorithms have to be used or implemented for each new scenario, it is critical that algorithms should be clearly separated from the data structure. They also should be easily exchangeable. The order in which algorithms are called should be flexible as well. The algorithms are collected into the *algorithms* sub-package of that data structure which they manipulate. Therefore, an algorithm at `commuters.algorithms` manipulates data in the commuters data structure.

The interface provided is quite simple, but is powerful for our purpose: Each algorithm needs to implement one single method called `run(`*`DataStructure`*`)`. As input, it receives a reference to the complete data structure it refers to, and therefore, it has complete access to add, modify or even delete data points of the data structure. This `run` method automatically defines the entry point of the algorithm. So, complex models can be implemented and called via this single method.

The use of such an algorithm is again simple: The user can add an arbitrary number of algorithms to the system, which then automatically will be executed in the same order in which they were added.

6.1.3 External Models

At any point in time during the demand-modeling process, MATSim-T allows one to store all data into well defined XML data files. They do—by definition—respect the format as described in the underlying DTDs. Therefore, a clean and stable interface to third party programs and models is available even when those models are not part of MATSim-T. For the two case studies described in the following two Sections 6.2 and 6.3, we will again use the external *Secondary Location Choice Module* (see Section 5.1.4). The module uses the same DTDs, allowing simple information interchange.

As mentioned at the beginning of this chapter, MATSim-EA will be used as an external module as the last step of the demand modeling process.

6.1.4 Computational Issues

One important issue for demand-modeling is the amount of input information needed. Because of the variety of possible demand-modeling algorithms and input sources, it is necessary to have fast access to that data. One simple and rapid solution is to load all information into memory and provide a hierarchical data structure (tree structure) to access any item from any other location in $O(2 \cdot \log(n))$, where n is the depth of the data tree.

The hierarchical data structure is already provided by the input data (XML format), but the available space in memory might not be sufficient. While the description of the world, the land-use data information, commuter matrices, etc. typically hold a relatively small amount of data, the amount of information for the individuals goes far beyond the size of a typical memory capacity (on average around 1–2 GB of memory).

To handle this problem, MATSim-T uses the idea of sequential individual demand modeling (streaming of individuals). In other words, the toolkit reads one individual at a time, runs the defined algorithms for it, writes the results to file, and frees the memory. In this way, the number of individuals in the given scenario is unlimited. This idea will still work if—instead of single persons—demand modeling will be done at the household level for a small number of persons. But the limit of this approach will be reached if one also wants to add the concept of social networks (Hackney and Axhausen, 2006). In this case, demand-modeling of one individual can—in principle—depend on all other individuals in the scenario and therefore the whole population must be stored in memory.

Nevertheless, the plans package still allows one to store all individuals in memory, if the amount of data is not too large or if one has access to machines with a sufficient amount of memory. The user of the toolkit can switch between "streaming" and "no-streaming" by setting a defined flag.

The concept of "streaming individuals" has one conceptual disadvantage: some demand-modeling processes need summary information about the given population, which can not be calculated before the last individual is parsed. If the whole population does not fit simultaneously into computer memory, there is no other choice than to parse the population twice; first, to obtain aggregated information, and second, to distribute it again to each individual in the desired way. If a problem is "doubly constrained" and an iterative procedure is necessary, this process may have to be repeated several times until convergence is reached. This is a typical example of a "trade-off" between computational speed and available memory.

6.2 Demand Modeling Process for the Kanton Zurich, Switzerland

Using the toolkit described in the previous section, the section presents the steps taken to model daily demand for the Kanton Zurich, Switzerland. The following subsections describe briefly which input data were used and which algorithms were employed to model the daily demand.

The algorithms used are really not that sophisticated. They demonstrate the use of MATSim-T rather than delivering "state-of-the-art" demand modeling processes.

6.2.1 Data Resources

The world describes the region (Kanton Zurich of Switzerland) at two different resolutions:

- 170 municipalities and 12 additional districts inside the city of Zurich (*municipality layer* as described in Section 4.1.1.1, denoted as *M*)

- Raster of $100 \cdot 100m$ cell resolution (*raster layer* as described in Section 4.1.1.2, denoted as *R*).

The mapping between those two levels is also available (see Section 4.1.1.3). Furthermore, the network layer for Switzerland will be used as described in Section 4.1.3.1. The information about population distribution in the municipalities is shown in Section 4.1.4. The Swiss Federal Statistical Office (BfS, 2006) provides land-use data, holding information about capacities of different activity types like "work", "shopping", "education", etc. That information is also based on raster level

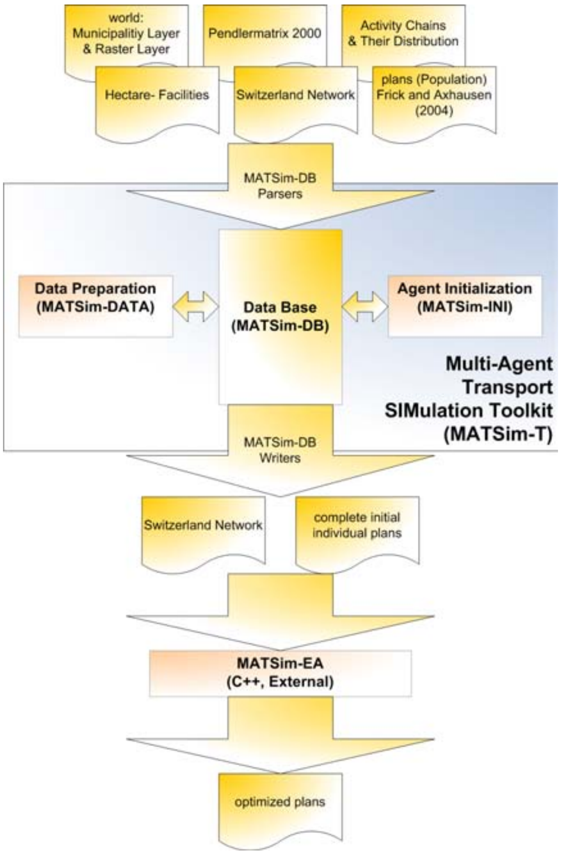

world: Municipalitiy Layer & Raster Layer
Pendlermatrix 2000
Activity Chains & Their Distribution
Hectare- Facilities
Switzerland Network
plans (Population) Frick and Axhausen (2004)
MATSim-DB Parsers
Data Preparation (MATSim-DATA)
Data Base (MATSim-DB)
Agent Initialization (MATSim-INI)
Multi-Agent Transport SIMulation Toolkit (MATSim-T)
MATSim-DB Writers
Switzerland Network
complete initial individual plans
MATSim-EA (C++, External)
optimized plans

Figure 6.2: The general demand modeling process for the Kanton Zurich scenario.

R (see Section 4.1.2.1). The "Pendlermatrix 2000" (Section 4.1.5) holds information about work and education commuters at municipality level *M*. The same 21 given activity chains and their distribution will be used as described in Section 4.1.6 and already used in Chapter 5.

In respect to Figure 5.1, we re-draw the demand modeling process as shown in Figure 6.2. It shows the demand modeling process directly based on the available input data of the Zurich scenario. The process does not use VISEM input or output data anymore.

6.2.2 Demand-Modeling Process Steps

The demand-modeling process itself is split into six sequential steps (see Figure 6.3 for an overview). Each process step (except the first) uses one

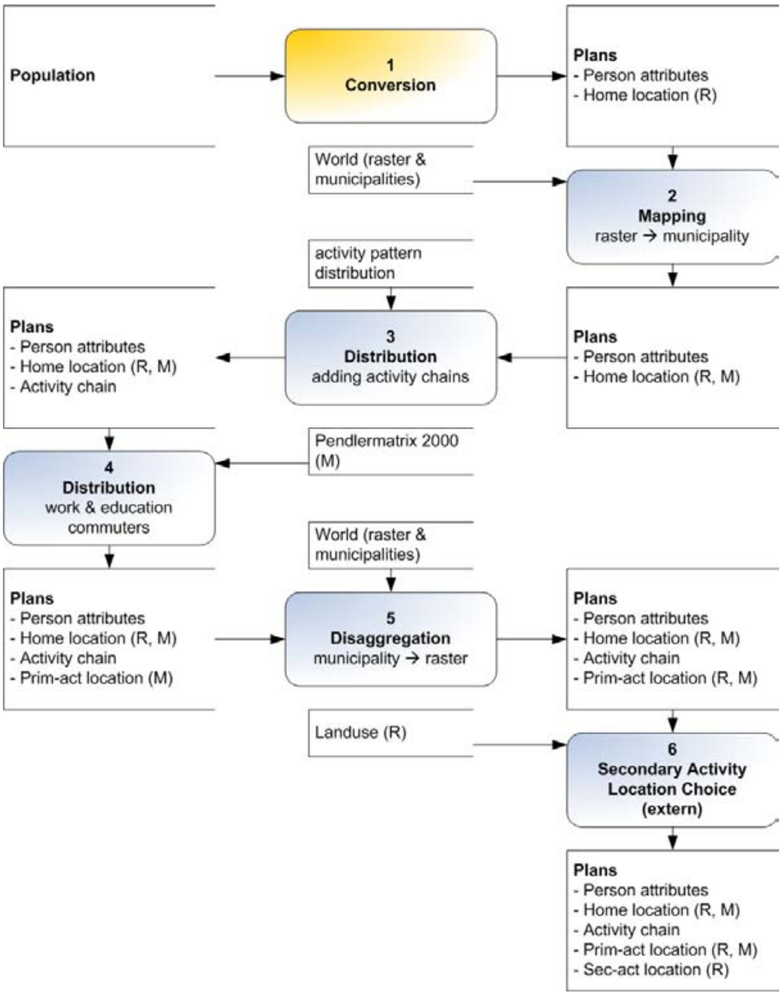


Figure 6.3: The demand modeling process steps of the Kanton Zurich scenario.

specific data resource to add details to each individual plan:

1. *Conversion:* This process step converts the input population file into the XML person description file. None of the person tags holds a plan yet, but additional attributes like age, sex, etc. are included. It also defines in which land-use raster element (*R*) this person lives.

2. *Mapping:* Each raster element of the land-use data belongs exactly to one municipality. By using this mapping, each agent can be assigned to the municipality of his home location.

3. *Distribution:* Given the distribution of the activity chains described above, this step assigns one of the chains to each person according to the given distribution. It also also respect the fact that e.g. children do not go to work, therefore persons of young age are not allowed to hold an activity chain including a "work" activity. Note that this independent random sampling from aggregate distributions can cause a lack of consistency with the given distribution. To avoid this, it is necessary to parse the population twice to obtain aggregate information (see Section 6.1.4).

4. *Distribution:* The "Pendlermatrix 2000" (see Section 4.1.5) holds information about work and education commuters at the municipality level (*M*). With the assumption about primary activities (see Section 5.1.4), one is able to add locations for the primary activities "work" and "education". Unfortunately, there is no similar data for the primary activities "shop" and "leisure". As long as no better data (or no appropriate algorithm) is available, the assumption is made that those activities are undertaken in the same municipality (district) in which the person lives.

5. *Disaggregation:* The process step 6 uses the secondary location choice module already used in Chapter 5 working at the raster level of detail (*R*). Therefore, disaggregation of the locations of the secondary activities to that level has to be done. This is accomplished by uniformly picking one of the raster elements of the given municipality (district).

6. *Secondary activity location choice (extern):* The module of Marchal and Nagel (2006)—adapted as described in Section 5.1.4—is again used as an external program example to add missing locations of secondary activities.

The final plan of each person in the scenario now describes the following:

- Person attributes
- Which activities will be performed

- In which order activities will be performed
- Where activities will be performed

There are some final algorithmic steps missing in that modeling process: there is no information about the times when an activity should be performed or how long it should take. We also still do not know which mode of transport will be used or which route to take to go from one location to the next. These steps are left out purposely because we will use the MATSim-EA iterative optimization process as shown in Figure 6.2 to create that information. A short overview about this process will be given later in this chapter.

6.3 Demand Modeling Process for the Länder Berlin-Brandenburg, Germany

The Berlin-Brandenburg scenario is somewhat different from the Kanton Zurich scenario, because only dissimilar, less precise information is available.

Again, no sophisticated algorithms are used. It is more interesting to show that some algorithms from the Zurich scenario can be reused while others have to be added.

6.3.1 Data Resources

The world describes the region of Berlin-Brandenburg. In this scenario, three different spatial resolutions are used:

- The traffic analysis zones “TAZ” as described in Section 4.2.1.1 (zone level denoted as *Z*)
- Each zone consists of several blocks (Section 4.2.1.2). 12,260 blocks are available for this scenario (block level denoted as *B*).
- A raster of 169,420 cells is also available. Each cell has an extent of 500x500 meters (raster level denoted as *R*).

The mapping rule between the zone and the block layer is also available, described in Section 4.2.1.4. The raster is synthetically generated; therefore, no additional information is encoded. The raster and the mapping are also created by the framework described above using a special world algorithm in the `world.algorithms` package. A cell refers to none, one, or many blocks while each block refers to at least one cell.

Only the population of Berlin itself (about 3.3 million individuals) was available at that time (Section 4.2.4). Less detail was obtainable than for Kanton Zurich's population. The only available items were:

- Home locations (*B*)
- Age

The land-use of Berlin-Brandenburg is based on the traffic analysis zone level *Z* (Section 4.2.2). For this scenario, there is no commuter matrix (and also no production-attraction matrix) available; only a 24-hour origin-destination matrix is given (Section 4.2.5). This matrix will be used in two different ways. First, for all trips starting in Berlin, this information is taken to guess where individuals of the corresponding zone are performing their primary activity. In other words, it is used in a way similar to the commuter matrix of the Kanton Zurich scenario. The remaining trips starting in the traffic analysis zones of Brandenburg are used to create the missing part of the population and also to define the location of their primary activity.

This is—of course—an approximate way to interpret an origin-destination matrix, but still a step to close the information gap. It will be one of the highest priorities of our future work to design a better primary activity location choice method for the Berlin-Brandenburg scenario.

Again, the activity chain distribution of the Kanton Zurich scenario are used. Those patterns reflect average Swiss behavior and thus do not necessarily reflect those of Berlin-Brandenburg, but it is a feasible start as long as no other data is available.

6.3.2 Demand-Modeling Process Steps

In comparison to the process of the Kanton Zurich scenario, there is one step more because three different levels of spatial resolution are used

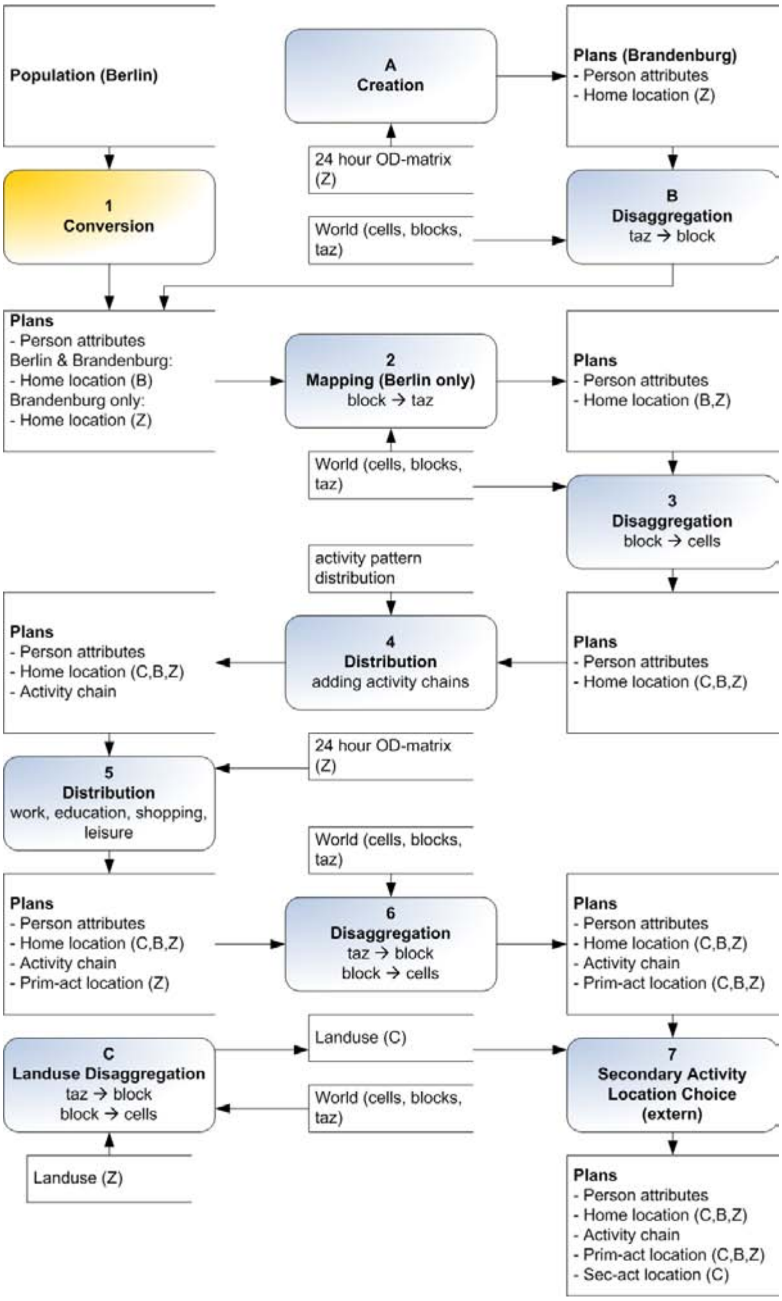


Figure 6.4: The demand modeling process steps of Berlin-Brandenburg

instead of two. Furthermore, three additional steps are included (see Figure 6.4):

A. *Creation:* As already mentioned above, there is no information about the population of Brandenburg. Because the number of trips of the origin-destination matrix starting in zones of Brandenburg fits with its total population, we assigned one individual for each of those trips to that start zone. This is—of course—a rough approximation of the distribution of the population of Brandenburg.

B. *Disaggregation:* Since the individuals are placed at the traffic analysis zone level (Z), a mapping needs to be done to the block level (B). A uniform distribution over all blocks of a given zone is assumed.

C. *Land-use disaggregation:* To use the secondary activity location choice module, the land-use information has to be given at the raster level. Therefore, the TAZ based land-use data also has to be disaggregated into the raster level (R). It is done in a first step by uniform distribution over all blocks (B) of each TAZ. In a second step, another uniform distribution is used to propagate the land-use data from block level (B) to raster level (R).

1. *Conversion:* This is the same conversion described above. Together with the two special steps A and B, the whole population is now given in the data structure.

2. *Mapping:* To add the corresponding TAZ level to the population of Berlin, a mapping from block level to TAZ level has to be done. This mapping is non-ambiguous.

3. *Disaggregation:* The last remaining spatial level (raster level R) is mapped in this step. The method is again an additive uniform distribution over each cell of the given block.

4. *Distribution:* Same as for the Zurich scenario

5. *Distribution:* In contrast to the same step of the Kanton Zurich process, all different primary activity types are assigned according to the given matrix. Because there is no information about the purpose of a given trip of the matrix, one cannot distinguish it from others. The location of the primary activity is chosen according to destination distribution of those trips starting at the home location of the current individual.

6. *Disaggregation:* This step again provides a disaggregation of the location of the secondary activities to the raster level (R). It is done in the same way as described for the land-use (see above).

7. *Secondary activity location choice (extern):* The secondary location choice module is again used to find the missing locations as described in process step 6 of the Kanton Zurich scenario.

The final demand of that scenario holds the same data items as described in the Zurich scenario.

6.4 MATSim-EA Iterative Demand Optimization Process

As shown in Figure 6.2, the generated demand is used as input for the MATSim-EA iterative optimization process. Section 2.4 shows—apart from the limitation of the time mutation mentioned in Section 3.2.1.2—that feasible timing of given plans can be generated starting with random initial departure times and durations of activities. Therefore, it is not necessary to include an appropriate algorithm into the demand modeling process, but one can use MATSim-EA as a final external module for that.

The second part of this Section will then present MATSim-EA with an enhanced time choice module replacing the time mutator module, which optimizes activity departure times and activity durations using the whole search space.

6.4.1 MIT Run: MATSim-EA using *Time Allocation Mutator*

For the first MATSim-EA setup, we are using the same strategy modules as shown in Section 2.4, but the input is now the initial individual demand for Kanton Zurich as modeled above. The generated results—denoted as *MIT* run to indicate that the 45.44% share of the population using motorized individual transport is used as already done in Section 5.1.5—will be compared to the VISUM results and the outcome of the MATSim-EA run presented in Chapter 5. Furthermore, again comparison of the resulting volumes of the four links against the count data are shown in the same manner as already done in Figure 5.4 and Figure 5.5.

6.4.1.1 Results

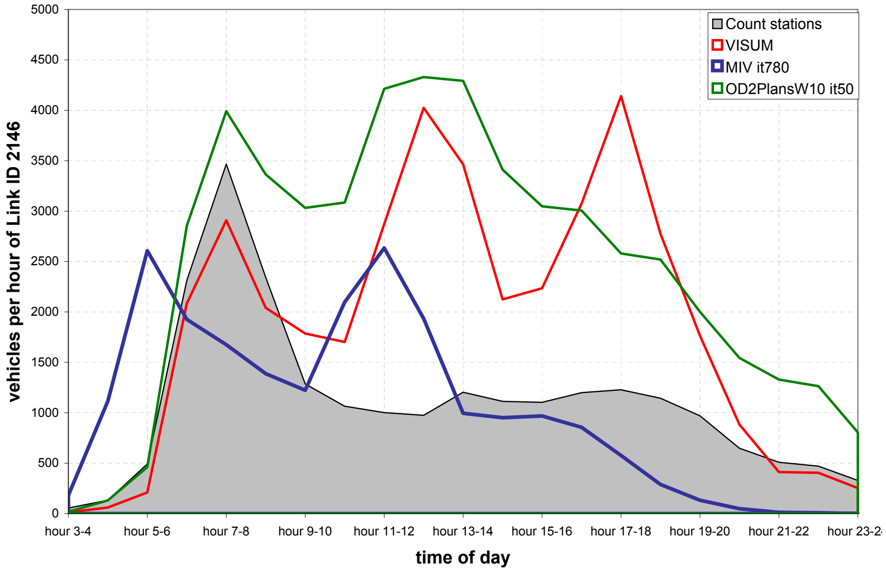
The parameter configuration is done in the same manner as presented in Section 5.1.5. Calculating 780(!) iterations, the resulting traffic volumes are compared against the result of the previous Chapter 5, the VISUM outcome, and the given count data.

It is not surprising that the volumes produced by MATSim-EA using the demand from the initial individual demand modeling process (bold blue curves of Figure 6.5 and Figure 6.6) are quite different from VISUM, since they are not based on the same origin-destination matrices. Nevertheless, it is quite questionable why the *MIT* run also produces a peak around noon (Figure 6.5(a) and Figure 6.6(b)). Since the network links on which the volumes are measured are located in an urban area (the city of Zurich), the noon peak could be produced by agents who are performing shop and leisure activities between two work or education activities. Since the simulation does not allow mode change, all agents are using their vehicle to reach such an intermediate activity. On the other hand, it is more typical that individuals who work in the city are having e.g. a lunch break next to their offices.

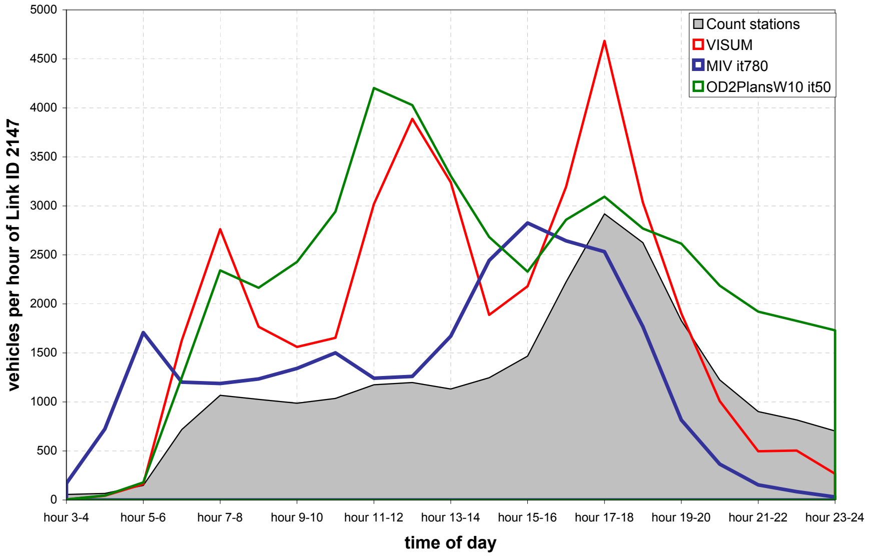
It should also be mentioned that in three of the four volume measurements, the *MIT* run produces much lower volumes than the VISUM run. The reason for that lies in the fact that the agents are not that strongly bound to a fixed time for leaving an activity. They are able to rearrange the times when they travel and also which route they choose.

Apart from the above mentioned noon peak, the volumes produced by the *MIT* model (bold blue curves of Figure 6.5 and Figure 6.6) are quite similar to the field data (grey areas of Figure 6.5 and Figure 6.6), but are showing one difference: The *MIT* curve has a shift of about two hours compared to the field data. The cause of this shift could be the fixed time windows described in Section 5.1.5. Instead of defining the work activity opening time between 7 AM and 9 AM, we could shift it more towards noon.

It has to be mentioned that the MATSim-EA simulation process had not yet reached a stable state when it was producing the outcome of iter-

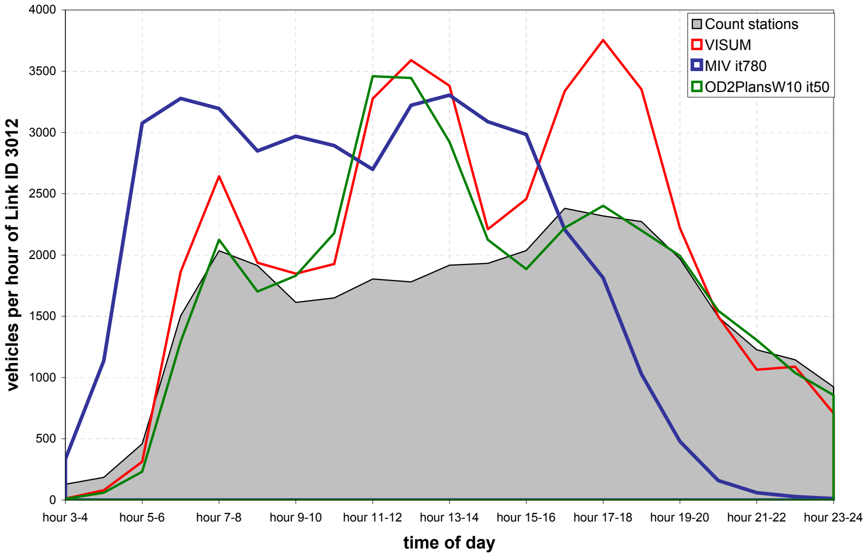


(a) Hourly traffic volumes of link-id 2146

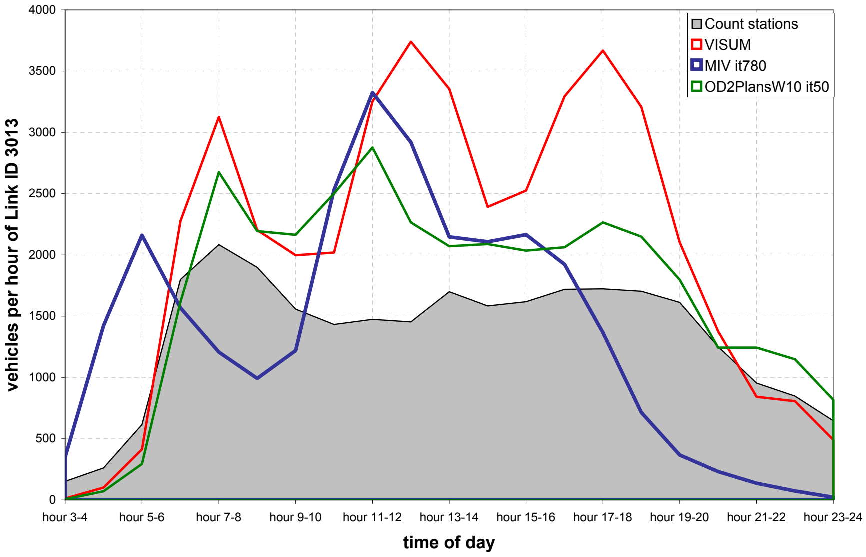


(b) Hourly traffic volumes of link-id 2147

Figure 6.5: Comparison of hourly traffic volumes produced by VISUM, *OD2PlansW10* run as presented in Chapter 5, the *MIT* run based on the initial demand modeling process of Kanton Zurich as shown in this chapter, and traffic count data.



(a) Hourly traffic volumes of link-id 3012



(b) Hourly traffic volumes of link-id 3013

Figure 6.6: Comparison of hourly traffic volumes produced by VISUM, *OD2PlansW10* run as presented in Chapter 5, the *MIT* run based on the initial demand modeling process of Kanton Zurich as shown in this chapter, and traffic count data.

ation 780. With the defined initial end time of the first activity between 6am and 8am, the simulation process of MATSim-EA is still attempting to reach the 7 AM to 9 AM time window. Therefore, a more appropriate setup will reduce the observed bias.

Comparing the volumes produced by VISUM, the *MIT* run, and available count data on other links in the city of Zurich (i.e. link ID 3192 and 3193 of Figure 4.5), similar plots can be generated as described above. But the more we get to the border of the Zurich area (count station ID 178, 213, and 221 of Figure 4.5) the less traffic is produced by MATSim-EA and VISUM compared to the traffic counts. This is expected, since only intra-kantonal demand is modeled. Therefore, all transit traffic passing through the Kanton Zurich is missing. This hints at the importance of the right definition of the study area for the outcome of a scenario analysis.

6.4.2 1% Run: MATSim-EA using *planomat*

In contrast to Section 2.4, Section 5.1.5, and Section 6.4.1, we use an enhanced time allocation module called *planomat* (see Meister *et al.*, 2005, for more detail). Using a genetic algorithm, this module can be used for time scheduling as well as for location or mode choice. Since activity chain choice and location choice are already performed by the demand modeling process, and the traffic flow simulation (SQSim) of MATSim-EXEC (Cetin, 2005) handles only car traffic, the functionality of the *planomat* module is reduced to time choice.

Compared to the *Time Allocation Mutator* module used above, the *planomat* module allows time replanning on the complete time axis of the search space. Therefore, this strategy allows one to find optimal time-replanning according to the information received from the last execution of the SQSim.

MATSim-EA optimizes daily plans and not single trips. Therefore, the consistency at the individual level is guaranteed (an agent cannot leave a location before he arrives there). This is—in fact—one of the most important issues in describing demand on the basis of individuals, instead of losing important information by using origin destination matrices.

6.4.2.1 Time Scheduling Results

Using the same MATSim-EA parameter configuration as in Section 5.1.5, one percent sample of the Kanton Zurich scenario population is optimized. By artificially reducing each link's capacities within the given network by a factor of 50[2], the congestion patterns produced are similar to those obtained by using all persons.

Figure 6.7 shows the differences between the initial departure times of the agents (iteration 0) and the departure times after 400 iterations. The figure highlights (bold lines) two groups of individuals:

Bold grey line Individuals with plans containing "work" as the second and second to last activity in the chain.

Bold black line Individuals with plans containing "leisure" as the second and second to last activity in the chain

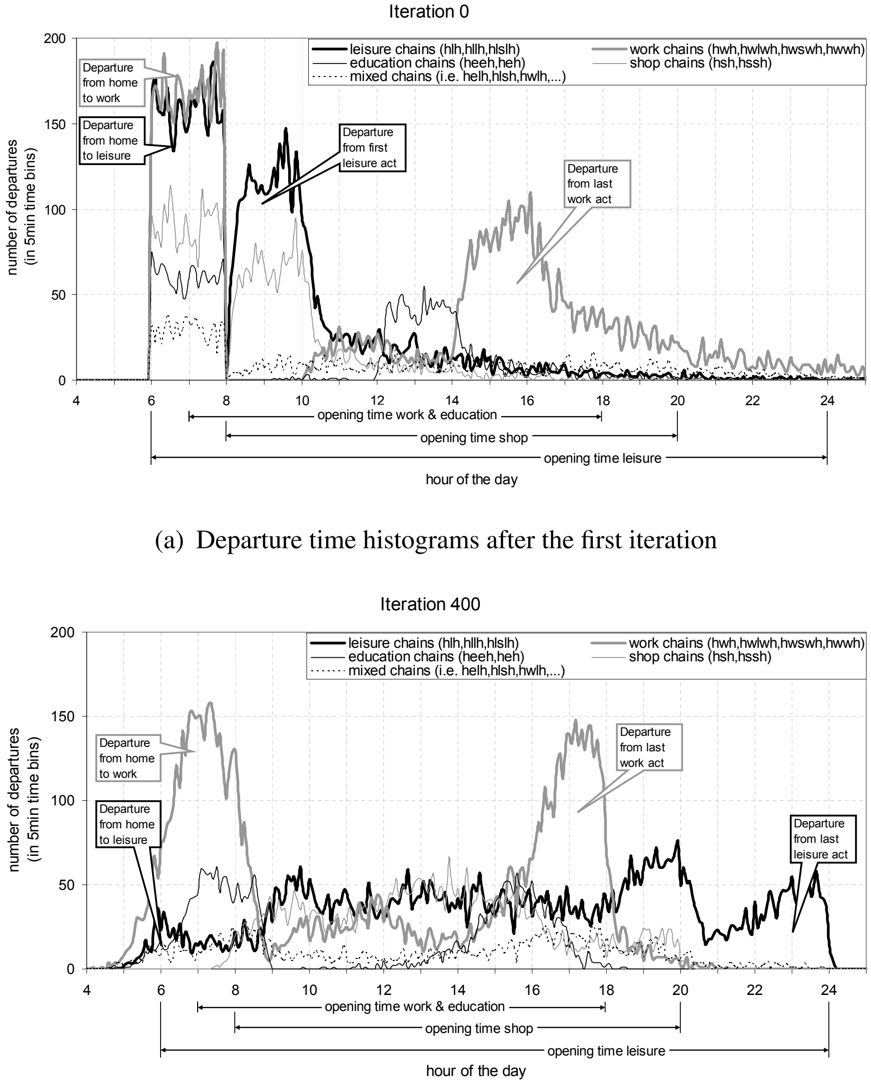
Since the "work" activities should be performed between 7 AM and 6 PM and the duration of performing "work" should be around 8 hours (in Charypar and Nagel, 2005, described as the "operating point" t^*), they produce the expected morning and evening peak (rush hour) after 400 iterations.

The leisure activities do not have such a rigid constraint (the setup defines it between 6 AM and midnight). The "operating point" of performing one leisure activity is set to 2 hours. Therefore, there is much more time scheduling flexibility for leisure activities. Figure 6.7 reflects that fact. While the time choice at iteration 0 is unsatisfying, the improvement after 400 iterations is immense. Only a few persons choose the morning rush hour to travel while the majority performs their leisure activity in the evening (finishing their last leisure activity just before midnight, when almost no traffic exists).

6.5 Discussion

This chapter presented MATSim-T, a fully integrated approach to modeling individual (daily) demand for large scale scenarios. It fulfills all

[2] A 1% sample of the *whole* population is in fact ca. a 2% sample of the *MIT* part of the population (with a 45.44% share using motorized individual transport)



(a) Departure time histograms after the first iteration

(b) Departure time histograms after 400 iterations

Figure 6.7: Departure time histograms of a 1% sample population of Kanton Zurich.

functionalities mentioned at the beginning of this chapter. It provides

- flexible handling of a large variety of input data
- extensibility of models and algorithms
- simple interface for new models and algorithms

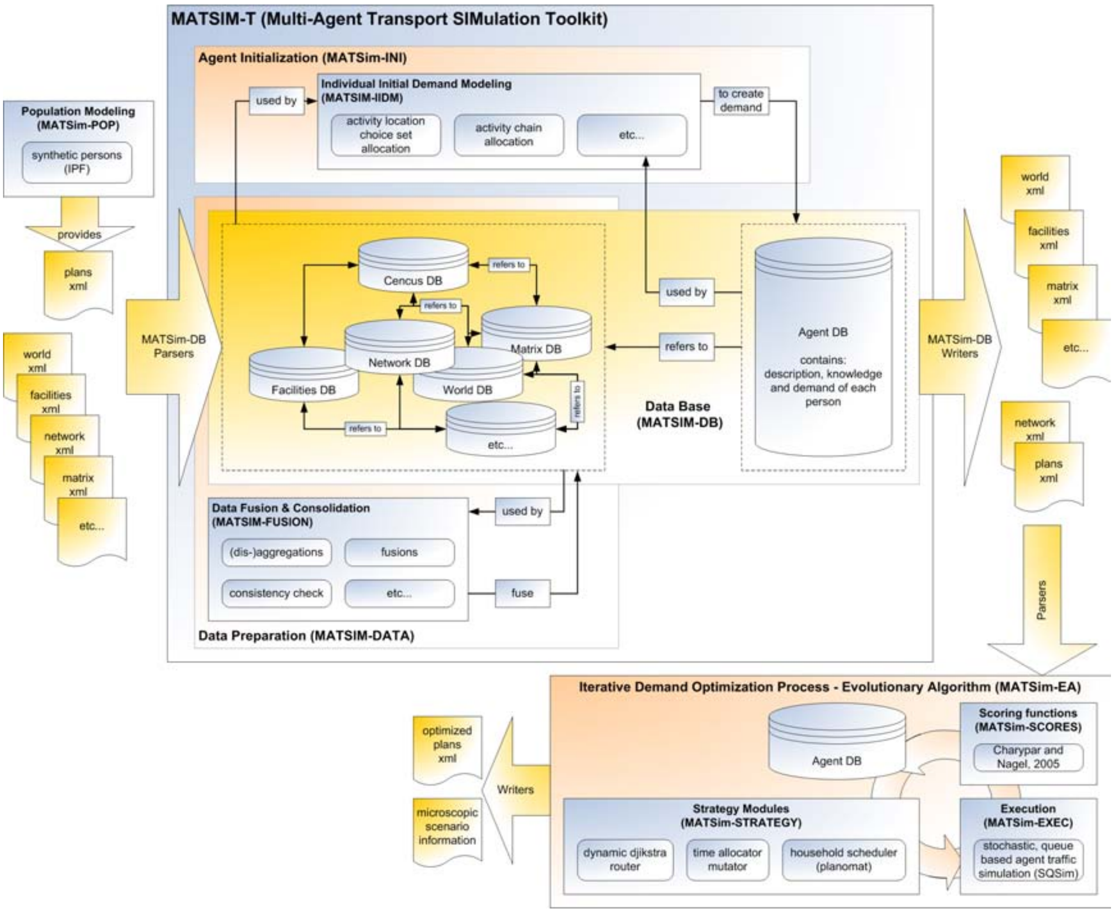

MATSIM-T (Multi-Agent Transport SIMulation Toolkit)
Agent Initialization (MATSim-INI)
used by
Individual Initial Demand Modeling (MATSIM-IIDM)
activity location choice set allocation
activity chain allocation
etc...
to create demand
Population Modeling (MATSim-POP)
synthetic persons (IPF)
provides
plans xml
world xml
facilities xml
network xml
matrix xml
etc...
MATSim-DB Parsers
Cencus DB
refers to
refers to
Matrix DB
Network DB
Facilities DB
World DB
refers to
refers to
etc...
used by
refers to
Agent DB
contains: description, knowledge and demand of each person
Data Base (MATSIM-DB)
MATSim-DB Writers
world xml
facilities xml
matrix xml
etc...
network xml
plans xml
Data Fusion & Consolidation (MATSIM-FUSION)
(dis-)aggregations
fusions
consistency check
etc...
used by
fuse
Data Preparation (MATSIM-DATA)
Parsers
Iterative Demand Optimization Process - Evolutionary Algorithm (MATSim-EA)
Agent DB
Scoring functions (MATSim-SCORES)
Charypar and Nagel, 2005
Strategy Modules (MATSim-STRATEGY)
dynamic djikstra router
time allocator mutator
household scheduler (planomat)
Execution (MATSim-EXEC)
stochastic, queue based agent traffic simulation (SQSim)
Writers
optimized plans xml
microscopic scenario information

Figure 6.8: MATSim-T integrated demand modeling approach

- simple (dis)aggregation for different spatial resolutions
- robust interface to third party models, programs and frameworks
- unlimited number of individuals, and
- simple interface to handle new input data elements.

If we now redraw (Figure 6.8) the target presented in Figure 2.1 and structure all the above mentioned process steps, one can see that the fully integrated approach gives one the complete freedom to generate and/or optimize demand, and therefore have full access to any microscopic result of the process.

The core is MATSim-DB, consisting of several data structures to store specified parts of the scenario. MATSim-Parsers and MATSim-Writers define the interfaces to the XML files. Systematical fusion and consolidation processes guaranty the uniqueness of each data point and the correct connections between them. All this is part of MATSim-DATA, which provides the following processes easy data access.

While the population generation is not yet part of MATSim-T, it is necessary to integrate it, since that process must be based on the same data as the demand modeling itself. It therefore will be part of MATSim-POP, indicating the process to generate the agent database.

The two demand modeling processes to model demand for Zurich and Berlin-Brandenburg generate the initial (not yet optimized) individual demand (denoted as MATSim-IIDM) and is—together with the population generation—an integral part of the initialization process (MATSim-INI).

Replacing the time-replanning module by the *planomat* indicates the importance of the flexible "plug-and-play" functionality of MATSim-EA. It this case, the planomat helps MATSim-EA to find the relaxed state after fewer iterations (in fact, after about 60 iterations the system is already in a stable state).

The actual optimization process for the modeled demand is provided by MATSim-EA, which is still used as an external module. Since also this process is based on exactly the same input data, it is mandatory to integrate it also into MATSim-T.

In the following chapter, we will stretch the final target of the toolkit and why it helps one to enhance the system in each and every aspect of research without breaking MATSim-T apart.

Chapter 7

Summary and Outlook

7.1 Summary

This thesis presented the evolution of the multi-agent transport simulation toolkit (MATSim-T) from a iterative optimization process to a complete demand modeling system for transport planners and researchers. While the research of Raney (2005) and the joint work of myself and Raney presented in Section 2.3 focuses on the systematic relaxation process of a multi-agent system implemented as an evolutionary algorithm, this dissertation focuses on the functional enrichment of the system. The major improvements are:

- Defining and implementing a well defined memory database providing full description of a given scenario, including flexible definition of different resolution layers.

- Providing flexible data structures which allow the users to add a large variety of input data to the system. They can differ in their characteristics as well as in resolution and quality.

- Providing a clean and consistent interface which allows one to add an arbitrary amount of algorithms, modules and programs to the toolkit, manipulating any data point of the scenario. Therefore, the demand modeling process can be easily adapted to the given input data.

- The system also provides well defined file interfaces to third party programs and algorithms at any step of the modeling process.
- The “plug-and-play” functionality of the memory database allows one to exchange information of a process step with another one without writing and reading the information of the database into files. This is in fact a great improvement for computational performance.
- The implementation of the initial individual demand modeling processes of the scenarios of Kanton Zurich and Berlin-Brandenburg as presented in Chapter 4 already produced a fair number of demand modeling process steps, which can be reused, enhanced and/or adapted to future scenarios.

The integrated demand modeling approach as shown in Figure 6.8 already indicates that the agent database of MATSim-EA is the C++ version of the MATSim-T Java agent database. Since MATSim-STRATEGY MATSim-EXEC and MATSim-SCORES are again algorithms using, adding, removing or manipulating data points of the memory database, it is easy to see that those parts can be included into the JAVA toolkit. Therefore, we are now able to redraw the MATSim-T system (Figure 7.1) in its current state.

In fact, the work presented in here is now the basis on which several developers are implementing new modules or re-implementing existing C++ modules. For example, the deterministic queue simulation (DQSim) is already ported into MATSim-T as well as the dynamic Dijkstra router and the scoring function. Rieser *et al.* (2007) has already presented first results with the fully integrated system (MATSim-EA included).

On the other hand, we still want to have the possibilities to use third party products or implementations of modules written with other programming languages. As an example, the deterministic, event-driven queue simulation (DEQSim) implemented in C++ by (Charypar *et al.*, 2007) has advantages in speed and functionality compared to the ones already used. Since one of the goals of this simulation is to accelerate computational speed, a parallelized version of it is in the development phase. Therefore, we do not plan to migrate the module to JAVA, but we still are able to use it via the defined file interfaces.

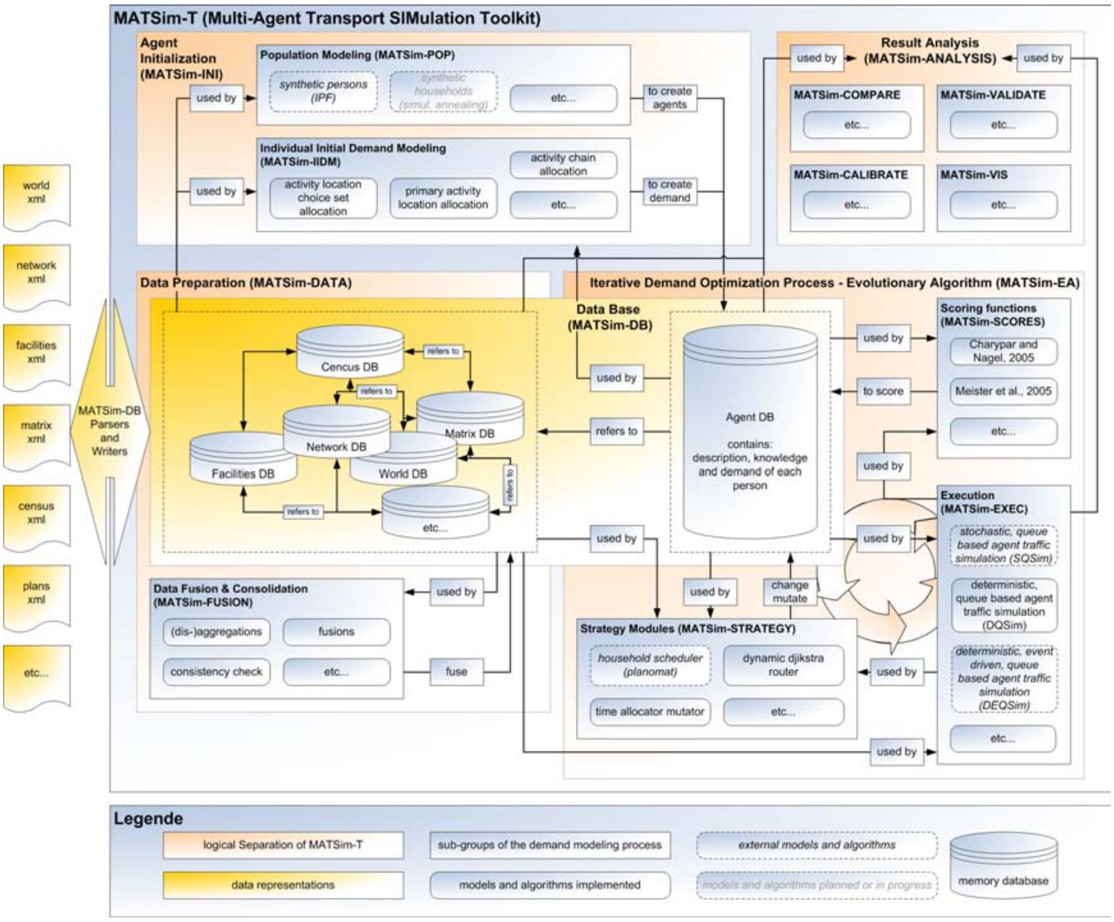

MATSim-T (Multi-Agent Transport SIMulation Toolkit)
Agent Initialization (MATSim-INI)
Population Modeling (MATSim-POP)
synthetic persons (IPF)
synthetic households (simul. annealing)
etc...
used by
to create agents
Individual Initial Demand Modeling (MATSim-IIDM)
activity location choice set allocation
primary activity location allocation
activity chain allocation
etc...
used by
to create demand
Result Analysis (MATSim-ANALYSIS)
used by
used by
MATSim-COMPARE
etc...
MATSim-VALIDATE
etc...
MATSim-CALIBRATE
etc...
MATSim-VIS
etc...
world xml
network xml
facilities xml
matrix xml
census xml
plans xml
etc...
MATSim-DB Parsers and Writers
Data Preparation (MATSim-DATA)
Cencus DB
refers to
refers to
Network DB
Matrix DB
Facilities DB
World DB
refers to
refers to
etc...
Data Base (MATSim-DB)
used by
refers to
used by
Agent DB
contains: description, knowledge and demand of each person
Iterative Demand Optimization Process - Evolutionary Algorithm (MATSim-EA)
Scoring functions (MATSim-SCORES)
used by
Charypar and Nagel, 2005
to score
Meister et al., 2005
etc...
used by
Execution (MATSim-EXEC)
used by
stochastic, queue based agent traffic simulation (SQSim)
deterministic, queue based agent traffic simulation (DQSim)
deterministic, event driven, queue based agent traffic simulation (DEQSim)
etc...
Data Fusion & Consolidation (MATSim-FUSION)
used by
(dis-)aggregations
fusions
consistency check
etc...
fuse
used by
change mutate
Strategy Modules (MATSim-STRATEGY)
household scheduler (planomat)
dynamic djikstra router
time allocator mutator
etc...
used by
used by
Legende
logical Separation of MATSim-T
data representations
sub-groups of the demand modeling process
models and algorithms implemented
external models and algorithms
models and algorithms planned or in progress
memory database

Figure 7.1: The final state of MATSim-T

7.2 Outlook

Figure 7.1 shows the current state of MATSim-T. The system is designed to enhance demand modeling in various respects. The following gives a brief overview about topics which are worth investigating to enhance the system.

The process of producing optimized demand for a scenario can be split up into the two main parts:

MATSim-IIDM the initial individual demand modeling process for each person of a given population, and

MATSim-EA the iterative individual demand optimization process for each agent of the scenario.

Until this point, the optimization process calculates appropriate routes, departure times and activity durations for each agent. Therefore, MATSim-IIDM has to define the remaining parts of a complete (daily) plan based on given spatial information and survey data. But MATSim-IIDM does not need to pre-calculate appropriate routes and times since MATSim-EA takes care of that.

It is—of course—desirable to add more and more functionalities into MATSim-EA such that the amount of input data can be reduced and therefore, MATSim-IIDM process steps can be minimized. The given data of a scenario then can be used to validate and calibrate the iterative optimization process.

The following items show which functional enrichments are in development phase or planned.

Secondary Location Choice As presented in Chapter 5 and Chapter 6, an external module for calculating locations for secondary locations is already used as a module for MATSim-IIDM. Since each single part of a (daily) plan is highly dependent on the other parts, it is planned to include this module into the optimization process. This induces an extension to the scoring function. For example, the capacity of a facility restricts the number of agents performing an activity at this place. Accessing overfull locations by an agent should therefore result in a penalty (negative utility) for that agent.

Mode Choice Instead of predefining the share of agents traveling by car using a modal split parameter, agents should have the possibility to decide which mode they choose. Therefore, an additional data matrix must be added to MATSim-T to provide information on access times and travel times to, e.g. public transport and other modes.

A long term target is—of course—microscopic simulation of the different transport modes. We therefore need to enrich MATSim-EXEC modules with other transport modes.

Intra Day Mode Choice In contrast to the previous item, agents should also have the possibility to change their mode of transport *during* the day. I.e., a “mode choice” strategy module then should take care about “park-and-ride” planning. A car left at a train station should therefore be fetched again at the evening, before the agent will go home again.

Parking Another restriction of planning a day is the amount of available parking slots in the area of interest. Park-search traffic and distance to the actual facility are aspects that affect the planning.

Personalized Planning Different population groups have different preferences which are reflected by their behavior. As a result, the scoring function should be personalized. I.e., the value of travel time saving for a business man is typically higher than for a homemaker (see for example Axhausen *et al.*, 2007).

Road Pricing The topic of road pricing goes hand in hand with the previously mentioned item. Different person groups react differentlt to changes of travel cost. Introducing different road pricing systems into MATSim-T is a desired target, not only because it is a crucial topic for transport planners nowadays, but also because it shows the flexibility of introducing new transportation concepts into the system.

Enhanced MATSim-EXEC The physical execution of the agents' plans is a simplified representation of the scenario. A trade-off between computational performance and realism had to be made. Nevertheless, it is desirable to add more realism to MATSim-EXEC to gain a better representation about what is happening during the day and—last but not least—to calculate a more and more precise performance of an executed plan. Shortcomings are *turn penalties*, *turning lanes*, *traffic lights* and so on.

Activity Chain Optimization In the longer term, an agent should also have the ability to re-plan any part of his plan, including the activities themselves. It should be possible to add, drop and shuffle the activity chain to optimize his day.

As one can see, there are many topics which we can investigate and the above list is far from complete. But one important fact has to be mentioned: Since large-scale multi-agent systems are highly complex, each single functionality added to the optimization process has to be tested for different scenarios and setups. It is almost impossible to identify the effects of a change to the system if we include several functionalities at the same time.

The above mentioned note directly leads us to the topic of result analysis, validation and calibration techniques. Since the outcome of MATSim-T is completely microscopic, it is possible to answer an enormous number of different questions—not only in the area of transport planning, but also questions of environmental impact of noise, pollution, social interactions, dispersal of diseases, and so on. Therefore, we need to provide a large, standardized and extensible set of analysis and validation tools in the system to produce appropriate information at the outcome.

The intention of MATSim-T, the multi-agent transport simulation toolkit, is that it should grow into a powerful collection of data bases, population and demand modeling algorithms and modules, integrated optimization procedures, and flexible analysis and validation tools.

Acknowledgement

I would like to thank my family and friends, who have always supported me throughout my academic career. A very special thank you belongs to my wife Nga, who was always patient with me during the last years, when I spent many nights and weekends on my research. In addition, this dissertation would not have been possible without the following important contributions.

Funding This dissertation was funded in part by the Federal Institute of Technology Zurich (ETHZ) core founding and by ETH projects.

Input Data Prof. Dr. Kay W. Axhausen's group at the Institute for Transport Planning and Systems (IVT) at ETHZ, the Swiss Federal Office for Spatial Development (*Bundesamt für Raumentwicklung*; ARE) and the Swiss Federal Statistical Office (*Bundesamt für Statistik*; BfS) provided the large amount of data for Switzerland. Count data of Switzerland are provided by the Swiss Federal Roads Authority (FEDRO). Marcel Rieser's semester thesis produced time dependent OD matrices with VISEM and Philipp Fröhlich delivered data exports from VISUM. Figures about spatial representations are made by Michael Löchl from IVT at ETHZ.

The Institute for Transport Systems Planning and Transport Telematics (*Verkehrssystemplanung und Verkehrstelematik*; VSP) of the Technical University of Berlin (TU-Berlin) made it possible to use data of Berlin-Brandenburg. Ulrike Beuck at VSP provided the ArcGIS data set to produce Figures of Germany. The title figure is created by Andreas Horni while the background picture is provided by the U.S. National Archive.

Computational Resources The major part of the computation for this work was performed on the 192-CPU Beowulf Cluster "Xibalba," made available by ETHZ and the department of Computer Science. Marc Schmitt did a great job maintaining the cluster and other computational resources. Konrad Meister was organizing computational resources at IVT to produce the results of the *planomat* run.

Assistance and Guidance Prof. Dr. Kay W. Axhausen, my thesis advisor, gave me great guidance during my research. Especially, he helped me to see the research topic from the viewpoint of transport planning. He gave my a lot of freedom during my research, but always showed me the direction when it was necessary. Prof. Dr. Kai Nagel, my former thesis advisor and my co-examiner, actually introduced me into the exiting topic of agent-based micro-simulation for transport planning. Many interesting discussions with him about this research, related topics and implementation details, at coffee breaks, in front of a computer or at dinner, always gave me good feedback and a lot of ideas to extend my work.

Bryan Raney, my colleague and a great friend, helped me to get to know MATSim-EA. It was a great pleasure to work with him, because he always had an open mind about new ideas and concepts. Nurhan Cetin, who provided the stochastic and queue based traffic flow simulation, and Christian Gloor often gave me good advice regarding working at the ETH.

A special thank to David Charypar. He and I followed the same career, starting with the study of computer science at the ETH. After getting our diploma, we began our PhD study at Prof. Kai Nagel's former simulation group at the Institute of Computational Science (ICOS) at ETHZ. And now, we are both researchers at IVT. He is a great friend and it is always a pleasure to discuss with him specific research topics in computer science, transport planning and aerodynamics.

I would also like to thank Dr. Fabrice Marchal for his assistance in using METROPOLIS for result analysis, Konrad Meister for advancing the *planomat*, Michael Bernard and Arnd Vogel for paper reviews and Nadine Schüssler for data exchange between VISUM and MATSim-T. Finally, special thanks to Marcel Rieser. He and I had many informative discussions about MATSim-T, programming, agent, interfaces, XML-files and many other topics.

Bibliography

Abraham, J. E., T. Weidner, J. P. Gliebe, C. Willison and J. D. Hunt (2005) Three methods for synthesizing baseyear built form for use in integrated land use-transport models, *Transportation Research Record*, **1902**, 114–123.

Antoniou, C., M. E. Ben-Akiva, M. Bierlaire and R. Mishalani (1997) Demand simulation for dynamic traffic assignment, in M. Papageorgiou (ed.) *8th IFAC Symposium on Transportation Systems*, Chania, Jun. 1997.

ARE (2006) Federal Office for Spatial Development, webpage, `http://www.are.admin.ch`.

ARE and BfS (2001) Mobilität in der Schweiz, Ergebnisse des Mikrozensus 2000 zum Verkehrsverhalten, *Final Report*, **6/91**, Federal Office for Spatial Development and Swiss Federal Statistical Office, Bern, `http://www.are.admin.ch/are/de/verkehr/mobilitaetskennziffern/unterseite26/index.html`.

Arentze, T. A., F. Hofman, H. Mourik and H. J. P. Timmermans (2000) Albatross: A multi-agent rule-based model of activity pattern decisions, *Transportation Research Record*, **1706**, 136–144.

Arnott, R., A. de Palma and R. Lindsey (1993) A structural model of peak-period congestion: A traffic bottleneck with elastic demand, *The American Economic Review*, **83** (1) 161–179.

Astarita, V., K. Er-Rafia, M. Florian, M. Mahut and S. Velan (2001) Comparison of three methods for dynamic network loading, *Transportation Research Record*, **1771**, 179–190.

ASTRA (2006) Swiss Federal Roads Authority, webpage, `http://www.astra.admin.ch/`.

Avineri, E. and J. N. Prashker (2003) Sensitivity to uncertainty: Need for a paradigm shift, *Transportation Research Record*, **1854**, 90–98.

Axhausen, K. W. (1990a) Judging the day: A synthesis of the literature on measuring the utility of activity patterns, *Working Paper*, **561**, Transport Studies Unit, University of Oxford, Oxford.

Axhausen, K. W. (1990b) A simultaneous simulation of activity chains, in P. M. Jones (ed.) *Developments in Dynamic and Activity-Based Approaches to Travel Analysis*, 206–225, Avebury, Aldershot.

Axhausen, K. W. (2006) Moving through nets: An introduction, in K. W. Axhausen (ed.) *Moving Through Nets: The Physical and Social Dimensions of Travel*, 1–7, Elsevier, Oxford.

Axhausen, K. W., S. Hess, A. Knig, J. J. Bates and M. Bierlaire (2007) State-of-the-art estimates of swiss value of travel time savings, in TRB (ed.) *the 86th Annual Meeting of the Transportation Research Board*, Washington, D.C., Jan. 2007.

Axmark, D., A. Larsson and M. Widenius (2006) *MySQL 3.23, 4.0, 4.1 Reference Manual*, MySQL AB, Dec. 2006, `http://dev.mysql.com/doc/refman/4.1/en/index.html`.

Balmer, M., M. Bernard and K. W. Axhausen (2005a) Matching geocoded graphs, in STRC (ed.) *the 5th Swiss Transport Research Conference (STRC)*, Ascona, Mar. 2005, `http://www.strc.ch/2005.html`.

Balmer, M., N. Cetin, K. Nagel and B. Raney (2004a) Towards truly agent-based traffic and mobility simulations, in AAMAS (ed.) *the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, New York, Jul. 2004, `http://www.aamas2004.org/`.

Balmer, M., K. Nagel and B. Raney (2004b) Large-scale multi-agent simulations for transportation applications, *Journal of Intelligent Transportation Systems*, **8** (4) 205–223.

Balmer, M., B. Raney and K. Nagel (2005b) Adjustments of activity timing and duration in an agent-based traffic flow simulation, in H. J. P. Timmermans (ed.) *Progress in Activity-Based Analysis*, 91–114, Elsevier, Oxford.

Bäck, T. (1996) *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*, Oxford University Press, Oxford.

Beckman, R. J., K. A. Baggerly and M. D. McKay (1996) Creating synthetic baseline populations, *Transportation Research Part A: Policy and Practice*, **30** (6) 415–429.

Ben-Akiva, M. E. and S. R. Lerman (1985) *Discrete Choice Analysis: Theory and Application to Travel Demand*, MIT Press, Cambridge.

Beowulf.org (2006) *Beowulf: Frequently Asked Questions*, Beowulf.org, `http://www.beowulf.org/overview/faq.html`.

BfS (1996) Verkehrsverhalten in der Schweiz 1994: Mikrozensus Verkehr 1994, *Final Report*, Swiss Federal Statistical Office, Bern, `http://www.bfs.admin.ch/bfs/portal/de/index/themen/verkehr_und_nachrichtenwesen/nutz_verk_inf/verkehrsverhalten/publikationen.html?publicationID=1237`.

BfS (2006) Swiss Federal Statistical Office, webpage, `http://www.bfs.admin.ch`.

Bhat, C. R., J. Y. Guo, S. Srinivasan and A. Sivakumar (2004) A comprehensive econometric microsimulator for daily activity-travel patterns, *Transportation Research Record*, **1894**, 57–66.

Bottom, J. A. (2000) Consistent anticipatory route guidance, Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge.

Bowman, J. L., M. Bradley, Y. Shiftan, T. K. Lawton and M. E. Ben-Akiva (1999) Demonstration of an activity-based model for Portland, *World Transport Research*, **3**, 171–184.

Cantarella, C. and E. Cascetta (1995) Dynamic process and equilibrium in transportation network: Towards a unifying theory, *Transportation Science A*, **25** (4) 305–329.

Cetin, N. (2005) Large-scale parallel graph-based simulations, Ph.D. Thesis, ETH Zurich, Zurich.

Cetin, N., A. Burri and K. Nagel (2003) A large-scale multi-agent traffic microsimulation based on queue model, in STRC (ed.) *the 3th Swiss Transport Research Conference (STRC)*, Ascona, Mar. 2003, `http://www.strc.ch/2003.html`.

Charypar, D., K. W. Axhausen and K. Nagel (2007) Event-driven queue-based traffic flow microsimulation, in TRB (ed.) *the 86th Annual Meeting of the Transportation Research Board*, Washington, D.C., Jan. 2007.

Charypar, D. and K. Nagel (2005) Generating complete all-day activity plans with genetic algorithms, *Transportation*, **32** (4) 369–397.

Daganzo, C. F. (1998) Queue spillovers in transportation networks with a route choice, *Transportation Science*, **32** (1) 3–11.

de Palma, A. and F. Marchal (2002) Real cases applications of the fully dynamic METROPOLIS tool-box: An advocacy for large-scale mesoscopic transportation systems, *Networks and Spatial Economics*, **2** (4) 347–369.

DynaMIT (2006) Intelligent transportation system program, webpage, `http://mit.edu/its/dynamit.html`.

Ettema, D. F., H. J. P. Timmermans and T. A. Arentze (2004) Modelling perception updating of travel times in the context of departure time choice under ITS, *Journal of Intelligent Transportation Systems*, **8** (1) 33–44.

Ferber, J. (1999) *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*, Addison-Wesley, Boston.

Flynn, P. (2006) *The XML FAQ: Frequently-Asked Questions about the Extensible Markup Language*, Cork, `http://xml.silmaril.ie/`.

Frank, M. and P. Wolfe (1956) An algorithm for quadratic programming, *Naval Research Logistics Quarterly*, **3**, 95–110.

Frick, M. and K. W. Axhausen (2004) Generating synthetic populations using ipf and monte carlo techniques: Some new results, in STRC (ed.) *the 4th Swiss Transport Research Conference (STRC)*, Ascona, Mar. 2004, `http://www.strc.ch/2004.html`.

Friedrich, M., I. Hofsäss, K. Nökel and P. Vortisch (2000) A dynamic traffic assignment method for planning and telematic applications, in AET (ed.) *European Transport Conference (ETC)*, Cambridge, `http://www.aetransport.co.uk`.

Gamma, E., R. Helm, R. Johnson and J. Vlissides (1995) *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, Boston.

Gawron, C. (1998) An iterative algorithm to determine the dynamic user equilibrium in a traffic simulation model, *International Journal of Modern Physics C (IJMPC)*, **9** (3) 393–408.

Gärling, T. and K. W. Axhausen (2004) Introduction: Habitual travel choice, *Transportation*, **30** (1) 1–11.

GS EVED and BfS (1992) Verkehrsverhalten in der Schweiz 1989, Mikrozensus 1989, Berichtband, *Final Report*, **6/91**, Dienst für Gesamtverkehrsfragen (GS EVED) and Swiss Federal Statistical Office, Bern, `http://www.are.admin.ch/imperia/md/content/are/gesamtverkehr/personenverkehr/mz89.pdf`.

Hackney, J. K. and K. W. Axhausen (2006) An agent model of social network and travel behavior interdependence, in IATBR (ed.) *the 11th International Conference on Travel Behaviour Research (IATBR)*, Kyoto, Aug. 2006, `http://term.kuciv.kyoto-u.ac.jp/iatbr06/`.

Hensher, D. A. (ed.) (2001) *Travel Behaviour Research: The Leading Edge*, Pergamon, Oxford.

Holland, J. H. (ed.) (1992) *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, MIT Press, Cambridge.

Hunt, J. D., R. Johnston, J. E. Abraham, C. J. Rodier, G. R. Garry, S. H. Putman and T. de la Barra (2000) Comparisons from Sacramento model test bed, *Transportation Research Record*, **1780**, 53–63.

Jacob, R., M. Marathe and K. Nagel (1999) A computational study of routing algorithms for realistic transportation networks, *ACM Journal of Experimental Algorithmics (JEA)*, **4** (6) 197–218.

Jonnalagadda, J., N. Freedman, W. A. Davidson and J. D. Hunt (2001) Development of microsimulation activity-based model for San Francisco: Destination and mode choice models, *Transportation Research Record*, **1777**, 25–35.

Kaufman, D. E., K. E. Wunderlich and R. L. Smith (1991) An iterative routing/assignment method for anticipatory real-time route guidance, *Working Paper*, **91-02**, Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor.

Kutter, E. (1983) Modellierung der Verkehrsnachfrage auf der Basis verhaltensorientierter Kategorien, *DVWG-Schriftenreihe*, **B** (57) 189–212.

Lohse, D., G. Bachner, B. Dugge and H. Teichert (1997) Ermittlung von Verkehrsströmen mit n-linearen Gleichungssystemen unter Beachtung von Nebenbedingungen einschliessich Parameterschätzung, *Final Report*, **5**, Institute for Transportation Planning and Traffic, Technical University Dresden, Dresden, Dec. 1997, `http://vplno1.vkw.tu-dresden.de/vpl/viseva/Download/Heft-5.pdf`.

Loudon, W. R., J. Parameswaran and B. Gardner (1997) Incorporating feedback in travel forecasting, *Transportation Research Record*, **1607**, 185–195.

Marchal, F. and K. Nagel (2005) Modeling location choice of secondary activities with a social network of cooperative agents, in TRB (ed.) *the 84th Annual Meeting of the Transportation Research Board*, Washington, D.C., Jan. 2005.

Marchal, F. and K. Nagel (2006) Modeling location choice of secondary activities with a social network of cooperative agents, *Transportation Research Record*, **1935**, 141–146.

MATSim-T (2006) Multi Agent Transportation SIMulation Toolkit, software, `http://www.matsim.org`.

Meister, K., M. Balmer and K. W. Axhausen (2005) An improved replanning module for agent-based micro simulations of travel behavior, *Working Paper*, **303**, IVT, ETH Zurich, Zurich, `http://www.ivt.ethz.ch/vpl/publications/reports/ab303.pdf`.

Nagel, K. (1995) High-speed microsimulations of traffic flow, Ph.D. Thesis, University of Cologne, Cologne.

Nagel, K. and F. Marchal (2006) Computational methods for multi-agent simulations of travel behavior, in K. W. Axhausen (ed.) *Moving Through Nets: The Physical and Social Dimensions of Travel*, 131–188, Elsevier, Oxford.

Nagel, K., M. Strauss and M. Shubik (2004) The importance of timescales: Simple models for economic markets, *Physica A: Statistical Mechanics and its Applications*, **340** (4) 668–677.

Nash, J. (1951) Non-cooperative games, *The Annals of Mathematics*, **54** (2) 286–295.

Ortúzar, J. d. D. and L. G. Willumsen (2001) *Modelling Transport*, 3. edn., John Wiley & Sons, Chichester.

Palmer, R. G., B. W. Arthur, J. H. Holland, B. Lebaron and P. Tayler (1994) Artificial economic life: A simple model of a stockmarket, *Physica D: Nonlinear Phenomena*, **75** (1-3) 264–274.

Pendyala, R. M. (2004) Phased implementation of a multimodal activity-based travel demand modeling system in florida. volume II: FAMOS users guide, *Final Report*, Florida Department of Transportation (FDOT), Tallahassee, `http://www.eng.usf.edu/\~pendyala/publications/report/FAMOS\%20Users\%20Guide.pdf`.

PTV (2006) Traffic Mobility Logistics (PTV), webpage, `http://www.ptv.de`.

Rai, R. K., M. Balmer, M. Rieser, V. S. Vaze, S. Schönfelder and K. W. Axhausen (2007) Capturing human activity spaces: New geometries, in TRB (ed.) *the 86th Annual Meeting of the Transportation Research Board*, Washington, D.C., Jan. 2007.

Raney, B. (2005) Learning framework for large-scale multi-agent simulations, Ph.D. Thesis, ETH Zurich, Zurich.

Raney, B., M. Balmer, K. W. Axhausen and K. Nagel (2003) Agent-based activities planning for an iterative traffic simulation of Switzerland, in IATBR (ed.) *the 10th International Conference on Travel Behaviour Research (IATBR)*, Lucerne, Aug. 2003.

Raney, B. and K. Nagel (2003) Truly agent-based strategy selection for transportation simulations, in TRB (ed.) *the 82nd Annual Meeting of the Transportation Research Board*, Washington, D.C., Jan. 2003.

Raney, B. and K. Nagel (2005) An improved framework for large-scale multi-agent simulations of travel behavior, in TRB (ed.) *the 84th Annual Meeting of the Transportation Research Board*, Washington, D.C., Jan. 2005.

RBS (2001) *Regionales Bezugssystem (RBS)*, Senatsverwaltung für Stadtentwicklung Berlin, Berlin, `http://www.statistik-berlin.de/statistiken/rbs/start1.htm`.

Rickert, M. (1998) Traffic simulation on distributed memory computers, Ph.D. Thesis, University of Cologne, Cologne.

Rieser, M. (2004a) Berechnung von Nachfragematrizen mit VISEM, *Semester Project*, **23**, IVT, ETH Zurich, Zurich, `http://www.ivt.ethz.ch/docs/students/sem23.pdf`.

Rieser, M. (2004b) Generating day plans from origin-destination matrices, *Semester Project*, **29**, IVT, ETH Zurich, Zurich, `http://www.ivt.ethz.ch/docs/students/sem29.pdf`.

Rieser, M., K. Nagel, U. Beuck, M. Balmer and J. Rümenapp (2007) Truly agent-oriented coupling of an activity-based demand generation with a multi-agent traffic simulation, in TRB (ed.) *the 86th Annual Meeting of the Transportation Research Board*, Washington, D.C., Jan. 2007.

Salvini, P. A. and E. J. Miller (2005) ILUTE: An operational prototype of a comprehensive microsimulation model of urban systems, *Networks and Spatial Economics*, **5** (2) 217–234.

SAX (2006) Simple API for XML, webpage, `http://www.saxproject.org`.

SfSB (1998) Datengrundlagen Stadtentwicklungsplan Verkehr, unpublished, Senatsverwaltung für Stadtentwicklung Berlin, Berlin, `http://www.stadtentwicklung.berlin.de/verkehr/`.

Sheffi, Y. (1985) *Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods*, Prentice-Hall, Englewood Cliffs.

SLAB (2006) Statistisches Landesamt Berlin, webpage, `http://www.statistik-berlin.de`.

Sutton, R. S. and A. G. Barto (1998) *Reinforcement Learning: An Introduction*, MIT Press, Cambridge.

Swisstopo (2006) *Swiss map projections*, `http://www.swisstopo.ch/en/basics/geo/system/projectionCH`.

Timmermans, H. J. P., T. A. Arentze and D. F. Ettema (2003) Learning and adaptation behaviour: Empirical evidence and modelling issues, in EIRASS (ed.) *the Euro-Conference Workshop on Behavioral Responses to ITS*, Eindhoven, Apr. 2003.

TRANSIMS (2006) TRansportation ANalysis and SIMulation System, webpage, Dec. 2006, `http://transims.tsasa.lanl.gov`.

UrbanSim (2006) A software-based simulation model for integrated planning and analysis of urban development, software, `http://www.urbansim.org`.

Vaughn, K. M., P. Speckman and E. I. Pas (1997) Generating household activity-travel patterns (HATPs) for synthetic populations, in TRB (ed.) *the 76th Annual Meeting of the Transportation Research Board*, Washington, D.C., Jan. 1997.

Vovsha, P., E. Petersen and R. Donnelly (2002) Microsimulation in travel demand modeling: Lessons learned from the New York best practice model, *Transportation Research Record*, **1805**, 68–77.

Vrtic, M. and K. W. Axhausen (2003) Experiment mit einem dynamischen Umlegungs-Verfahren, *Strassenverkehrstechnik*, **47** (3) 121–126.

Vrtic, M., P. Fröhlich and K. W. Axhausen (2003) Schweizerische Netzmodelle für Strassen- und Schienenverkehr, in T. Bieger, C. Lässer and R. Maggi (eds.) *Jahrbuch 2002/2003 Schweizerische Verkehrswirtschaft*, 119–140, SVWG Schweizerische Verkehrswissenschaftliche Gesellschaft, St. Gallen.

W3C (2006) *eXtensible Markup Language (XML)*, World Wide Web Consortium (W3C), `http://www.w3.org/XML`.

Waddell, P., A. Borning, M. Noth, N. Freier, M. Becke and G. Ulfarsson (2003) Microsimulation of urban development and location choices: Design and implementation of urbansim, *Networks and Spatial Economics*, **3** (1) 43–67.

Waddell, P., H. S. Sevcikova, D. Socha, E. J. Miller and K. Nagel (2005) OPUS: An international collaboration to develop an open platform for urban simulation, in CUPUM (ed.) *9th Conference on Computers in*

Urban Planning and Urban Management (CUPUM), London, Jun. 2005.

Wahle, J., A. L. C. Bazzan, F. Klügl and M. Schreckenberg (2002) The impact of real-time information in a two-route scenario using agent-based simulation, *Transportation Research Part C: Emerging Technologies*, **10** (5-6) 399–417.

Wardman, M. and J. P. Hine (2000) Costs of interchange: A review of the literature, *Working Paper*, **546**, Institute for Transport Studies, University of Leeds.

Wardrop, J. G. (1952) *Some Theoretical Aspects of Road Traffic Research*, vol. 1, Institute of Civil Engineers.

Wiedemann, R. (1974) Simulation des Verkehrsflusses, Ph.D. Thesis, University of Karlsruhe, Karlsruhe.

Appendix A

XML Formats

A.1 XML Plans Format

The purpose of this input format is to keep all the information for an agent together in one place, including information about the agent itself, such as socio-demographic information (age, income, etc.), its personal knowledge of the system and the activities and routes information that describe the agent's plans. This format should be flexible, so adding new information for one module does not interfere with parsing done by other modules.

XML (eXtensible Markup Language; see W3C, 2006) is a technology that has seen increasing popularity in recent years for many tasks, including data exchange. It is a "metalanguage" which is used to describe other languages (Flynn, 2006, answer A.1). XML is used to describe the "language" of agents, plans, activities, and routes—all the information that needs to be available to the modules in the strategic layer of the framework. Figure A.1 shows an example of an agent and a plan in this new data format.

As one can see from the figure, the format is rather intuitive. The format combines all demographic data about the person with the plan information into one place. With this format, modules have access to the entire agent when generating plans.

However, the main advantage of XML is its extensibility (the "X" in its name). One can add additional information to the format without breaking existing parsers. In particular, one can add information to only a subset of agents, for example a format to describe a mental map for

```
<plans name="example plans file" xml:lang="de-CH"> ...
  <person id="393241" age="37" income="50000">
    <knowledge>
      <activity type="work">
        <location type="link" id="844">
          <capacity value="180" />
          <opentime day="wkday" start_time="07:00:00"
            end_time="20:00:00" />
          <opentime day="sat" start_time="07:00:00"
            end_time="16:00:00" />
        </location>
        ...
      </activity>
      ...
    </knowledge>
    <plan>
      <act type="home" link="58" start_time="00:00" dur="07:00"
        end_time="07:00" />
      <leg mode="car" dept_time="07:00" trav_time="00:25"
        arr_time="07:25">
         <route>1932 1933 1934 1947</route>
      </leg>
      <act type="work" link="844" start_time="07:25" dur="09:00"
        end_time="16:25" />
      <leg mode="car" dept_time="16:25" trav_time="00:14"
        arr_time="16:39">
         <route>1934 1933</route>
      </leg>
      <act type="home" link="58" start_time="16:39" dur="07:21"
        end_time="24:00" />
    </plan>
  </person>
  ...
</plans>
```

Figure A.1: A typical plan in XML. This agent, id 393241, leaves home (on link id 5834 of the given network) at 7:00 AM (performing home activity for 7 hours), and drives to work via a four-node route (five links) which it expects to take 25 minutes to traverse. The agent stays at work for 9 hours, then drives home again via a two-node route and stays at home until midnight. Therefore, it describes a complete day-plan for person number 393241.

each agent.

It is important to note that the principal units of description are the "person" (i.e. agent) and the "plan." Any other module using the same

```
<network name="example network file" xml:lang="de-CH">
  <nodes>
    <node id="1" x="0.0" y="0.0" />
    <node id="2" x="80.0" y="0.0" />
    <node id="3" x="160.0" y="0.0" />
    ...
  </nodes>
  <links capperiod="01:00:00">
    <link id="100" from="1" to="2" length="90.0" freespeed="7.5"
      capacity="1800.0" permlanes="1"/>
    <link id="101" from="2" to="3" length="90.0" freespeed="7.5"
      capacity="1800.0" permlanes="1"/>
    ...
  </links>
</network>
```

Figure A.2: A typical network in XML. Defined as a strongly connected geo-coded di-graph, it first contains the set of nodes including their id's and coordinates, followed by the links of the network. Each link is connected to its start and end node (`from` and `to`-node) and consists of the attributes: id, length in meters, freespeed in meter per second, maximum flow capacity (`capacity`) in number of vehicles per time period (defined as attribute `capperiod`) and the number of lanes (`permlanes`).

principal units will be able to communicate with the rest of the system. This means adding new modules to the system becomes a simple task, since all modules will read and write the exact same file format.

A.2 XML Network Format

We use XML also to describe the network of a scenario. At this state of research the network defines only the streets used by cars, since it is the only mode which is actually microscopic simulated. In future, this network will be extended to capture links used by other modes, e.g. public transport. For that, an extension of the network description is required. Since XML is an extensible, additional information can be added without breaking up the already existing parts of the network. Figure A.2 shows an example of a street network.

Appendix B

ASCII Formats

B.1 ASCII Events Format

MATSim-EXEC microscopically simulates the execution of the (daily) plans of each agent simultaneously. To gather any information about what happen during the simulation MATSim-EXEC produces events for each agent in the system at a specific situation. Each single line of the events file precisely describes one event sorted according to the time when the event occures. An event contains:

- *t_gbl*; the time when an event was produced,
- *veh_id*; which agent the event produced,
- *leg_nr*; which leg the agent executes,
- *link_id*; on which link the event happened,
- *from_node_id*; the node from which the agent was coming,
- *event_flag*; which type of event occurred, and
- *description*; the textual description of the event type.

Each event describes a specific type given as the *EVENT_FLAG* and as its *DESCRIPTION*. An agent can produce the following types of events:

t_ gbl	veh_ id	leg_ nr	link_ id	from_ node_ id	event_ flag	description
0	106	0	110	0	7	actstart home
28800	106	0	110	0	8	actend home
28800	106	0	110	0	6	departure
29223	106	0	110	0	4	wait2link
29225	106	0	110	0	2	left link
29225	106	0	111	0	5	entered link
29253	106	0	111	0	2	left link
29253	106	0	100	0	5	entered link
29270	106	0	100	0	0	arrival
29270	106	1	100	0	7	actstart leisure
43200	106	1	100	0	8	actend leisure
43200	106	1	100	0	6	departure
.....	...	.	...	.	.	...
43286	106	1	108	0	0	arrival
43286	106	2	108	0	7	actstart education
57600	106	2	108	0	8	actend education
57600	106	2	108	0	6	departure
57634	106	2	108	0	4	wait2link
.....	...	.	...	.	.	...
57674	106	3	110	0	0	arrival
57674	106	4	110	0	7	actstart home
86400	106	4	110	0	8	actend home

Table B.1: Example of the produced events by a specific agent. It precisely describes when, where and what the agent was doing. Here: Agent number 106 performed “home” activity from 00:00:00 to 08:00:00 on link id 110. After waiting for 423 seconds to get on the network, it was driving along link id 110 and 111 to reach the next activity “leisure” on link id 100 at time 08:07:50. The “leisure” activity is then performed for almost 4 hours. It ends exactly at noon and the agent drives on to its next activity until he reaches home again at the evening.

- *actstart [act-type] (7)*; start of performing an activity,
- *actend [act-type] (8)*; end of performing an activity,
- *departure (6)*; departure from a location,
- *wait2link (4)*; entering the transport network,
- *left link (2)*; leaving a link of the transport network,
- *entered link (5)*; entering a link of the transport network,
- *arrival (0)*; arriving at a location, and
- *stuckAndAbort (3)*; abort an execution of an agent's plan.

The last item is used to accelerate the iterative optimization process of MATSim-EA and has no meaning in reality. After the relaxation process, no agent should abort their plans anymore (see Cetin, 2005, for more information about stucked vehicles).

UNTERFELDSTRASSE 25 · 8050 ZURICH · SWITZERLAND

TELEPHONE: +41 44 633 27 80 · MOBILE: +41 76 365 98 76 · E-MAIL: BALMERMI@GMAIL.COM

MICHAEL BALMER

PERSONAL INFORMATION

◇ Marital status: married
◇ Nationality: Swiss
◇ Date of birth: 1st of January 1974
◇ Place of birth: Scherzingen, Switzerland

DEGREES

DOCTOR OF SCIENCES IN CIVIL ENGINEERING, 2007

Focus:	Large-scale agent-based micro-simulation
Dissertation:	"Travel Demand Generation for Multi-Agent Transport Simulations: Algorithms and Systems"
University:	Swiss Federal Institute of Technology (ETH), Zurich, Switzerland

Advisor:	Prof. Dr. Kay W. Axhausen
	Institute for Transport Planning and Systems
	HIL F 32.3
	Wolfgang-Pauli-Str. 15
	CH-8093 Zurich
	+41 44 633 39 43 Telephone
	+41 44 633 10 57 Fax
	`axhausen@ivt.baug.ethz.ch`
	`http://www.ivt.ethz.ch`

MASTER OF SCIENCE IN COMPUTER SCIENCE, 2003

Focus:	Theoretical computer science
Diploma Thesis:	“Finding the Sink in a Unique Sink Oriented 5-Cube”
GPA:	5.21 out of 6.00; Graduated Magna Cum Laude
University:	Swiss Federal Institute of Technology (ETH), Zurich, Switzerland

Advisor:	Dr. Tibor Szabó, Prof. Emo Welzl
	Institute for Theoretical Computer Science
	CAB G 15.2
	Universitätstr. 6
	CH-8092 Kreuzlingen
	+41 44 632 73 70 Telephone
	+41 44 632 10 63 Fax
	`welzl@inf.ethz.ch`
	`http://www.ti.ethz.ch/`

UNIVERSITY-ENTRANCE DIPLOMA TYPE C, 1994

Focus:	Natural science
GPA:	4.3 out of 6.00; Graduated Cum Laude
High school:	Kantonsschule Kreuzlingen, Switzerland Pestalozzistrasse 7 CH-8280 Zurich +41 71 677 46 46 Telephone +41 71 677 46 47 Fax `http://www.ksk.ch/`

RESEARCH INTERESTS

Computational methods for simulating and solving science and engineering problems, especially those in the field of transport planning. Special foci are agent-based (micro) simulation methods, computational optimization, graph theories and algorithms, performance issues and software management.

COMPUTER SKILLS

Operating systems	Windows
	Unix & Linux
Programming languages, software & data formats	Java
	C, C++ & STL
	Perl
	Shell
	LaTeX
	CVS
	XML & HTML
	various APIs and scientific applications

LANGUAGE SKILLS

Native Language:	German
Other Languages:	English
	very basic French

EDUCATION & EXPERIENCE

GRADUATE RESEARCH, 04/2004 - 03/2007

Institute for Transport Planning and Systems
Department of Civil Engineering, ETH Zurich, Switzerland
◇ Multi-agent micro-simulations
◇ Agent based learning
◇ Demand generation
◇ Lecturing: “Simulation of traffic systems”
◇ Project management
◇ Software development for integrated transport systems
◇ Data analysis
◇ System integration
◇ Automated bibliography environment (CVS & LaTeX)

GRADUATE RESEARCH & TEACHING ASSISTANT, 04/2003 - 03/2004

Institute of Computational Science
Department of Computer Science, ETH, Zurich, Switzerland
◇ Multi-agent micro-simulations
◇ Agent based Learning
◇ Software development for integrated transport systems
◇ Data analysis
◇ System integration
◇ Lecturing in C++ & STL

STUDY IN COMPUTATIONAL SCIENCE, 10/1998 - 03/2003

Department of Computer Science, ETH, Zurich, Switzerland

STUDENT ASSISTANT, 03/2002 - 07/2002

Institute for Theoretical Computer Science
Department of Computer Science, ETH, Zurich, Switzerland
◇ Accompanying lectures in theoretical computer science

TECHNICAL ASSISTANT, 04/1999 - 03/2000

Concepts-ims AG, Horgen, Switzerland
◇ Marketing
◇ Consulting
◇ System administration
◇ Technical support (Citrix-Metaframe)

STUDY IN SHOT FIRING - TYPE C, 07/1999 - 09/1999

Department of Civil Engineering, ETH Zurich, Switzerland

STUDY IN CIVIL ENGINEERING, 10/1994 - 09/1998

Department of Civil Engineering, ETH Zurich, Switzerland

ASSISTANT, 04/1994 - 09/1994

IPG-Keller AG, Kreuzlingen, Switzerland
◇ Streets and building survey

PROJECTS

SINCE 2007

Agenten-basierte Mikrosimulation zur dynamischen Wirkungsanalyse (Canton Zurich, project 5205N-02014, N 20.1.4 / 108.00)

SINCE 2006

Agenten-basierte Simulation für location based services. (KTI project 8443.1 ESPP-ES)

Deriving and Assessing Strategies for Limiting the Spread of Airborne Diseases Using a Social Contact Model: The Case of Influenza. (SNF project 320000-114122)

2004-2005

A unified approach for agent-based learning with application in architecture and in transportation planning. (TH-30/02-3 project 2-74212-02)

2003-2005

Large-scale multi-agent simulation of travel behavior and traffic flow. (ETH project 0-20001-02)

PUBLICATIONS

FORTHCOMING

Beuck, U., K. Nagel, M. Rieser, D. Strippgen and M. Balmer (forthcoming) Preliminary results of a multi-agent traffic simulation for Berlin, *Advances in Complex Systems (ACS).*

Rai, R. K., M. Balmer, M. Rieser, V.S. Vaze, S. Schönfelder and K. W. Axhausen (forthcoming) Capturing human activity spaces: New geometries, *Transportation Research Record.*

2007

Balmer, M. (2007) Travel demand modeling for multi-agent traffic simulations: Algorithms and systems, Ph.D. Thesis, ETH Zurich, Zurich.

Rai, R. K., M. Balmer, M. Rieser, V. S. Vaze, S. Schönfelder and K. W. Axhausen (2007) Capturing human activity spaces: New geometries, paper presented at *the 86th Annual Meeting of the Transportation Research Board, Washington*, D.C., Jan. 2007.

Edelhoff, T., H. Schilling, M. Balmer and R. H. Möhring (2007) Optimal route assignment in large scale micro-simulations, *Working Paper*, **409**, IVT, ETH Zurich, Zurich, URL `https://www.ivt.ethz.ch/vpl/publications/reports/index/edit/ab409.pdf`.

Balmer, M., K. Nagel and B. Raney (2006) Agent-based demand modeling framework for large scale micro-simulations, *Transportation Research Record*, **1985**, 125-134.

2006

Rieser, M., K. Nagel, U. Beuck, M. Balmer and J. Rümenapp (2006) Truly agent-oriented coupling of an activity-based demand generation with a multi-agent traffic simulation, in ISB (ed.) *Stadt Region Land*, vol. 81, 185-192, Institut für Stadtbauwesen RWTH Aachen, Aachen.

Balmer, M. and K. Nagel (2006) Shape morphing of intersection layouts using curb side oriented driver simulation, in J. P. van Leeuwen and H. J. P. Timmermans (eds.) *Innovations in Design & Decision Support Systems in Architecture and Urban Planning*, 167-183, Springer, Eindhoven.

Meister, K., M. Balmer, K. W. Axhausen and K. Nagel (2006) planomat: A comprehensive scheduler for a large-scale multi-agent transportation simulation, paper presented at *the 11th International Conference on Travel Behaviour Research (IATBR)*, Kyoto, Aug. 2006, URL `http://term.kuciv.kyoto-u.ac.jp/iatbr06/`.

2005

Balmer, M., B. Raney and K. Nagel (2005) Adjustments of activity timing and duration in an agentbased traffic flow simulation, in H. J. P. Timmermans (ed.) *Progress in Activity-Based Analysis*, 91-114, Elsevier, Oxford.

Balmer, M., M. Bernard and K. W. Axhausen (2005) Matching geo-coded graphs, paper presented at *the 5th Swiss Transport Research Conference (STRC)*, Ascona, Mar. 2005, URL http://www.strc.ch/2005.html.

Balmer, M., M. Rieser, A. Vogel, K. W. Axhausen and K. Nagel (2005) Generating day plans using hourly origin-destination martices, in T. Bieger, C. Laesser and R. Maggi (eds.) *Jahrbuch 2004/2005 Schweizerische Verkehrswirtschaft*, 5-36, SVWG Schweizerische Verkehrswissenschaftliche Gesellschaft, St. Gallen.

Meister, K., M. Balmer and K. W. Axhausen (2005) An improved replanning module for agent-based micro simulations of travel behavior, *Working Paper*, **303**, IVT, ETH Zurich, Zurich, URL http://www.ivt.ethz.ch/vpl/publications/reports/ab303.pdf.

2004

Balmer, M., B. Raney and K. Nagel (2004) Agent-based activities planning for an iterative traffic simulation of Switzerland: Activity time allocation, paper presented at *the 4th Swiss Transport Research Conference (STRC)*, Ascona, Mar. 2004, URL http://www.strc.ch/2004.html.

Balmer, M., K. Nagel and B. Raney (2004) Large-scale multi-agent simulations for transportation applications, *Journal of Intelligent Transportation Systems*, **8** (4) 205-223.

Balmer, M., N. Cetin, K. Nagel and B. Raney (2004) Towards truly agent-based traffic and mobility simulations, paper presented at *the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, New York, Jul. 2004, URL `http://www.aamas2004.org/`.

2003

Raney, B., M. Balmer, K. W. Axhausen and K. Nagel (2003) Agent-based activities planning for an iterative traffic simulation of Switzerland, paper presented at *the 10th International Conference on Travel Behaviour Research (IATBR)*, Lucerne, Aug. 2003.

Balmer, M. (2003) Finding the sink in a unique sink oriented 5-cube, Master Thesis, Institute of Theoretical Computer Science, ETH Zurich, Zurich.

2002

Balmer, M. (2002) Morphing topologies using roadside oriented driver simulation: Microsimulation of “Central” traffic junction of zurich, Semester Project, ICoS, ETH Zurich, Zurich.

Balmer, M. (2002) Einführung in die Komplexität bei probabilistischen k-SAT Algorithmen, Semester Project, Institute for Behavioral Sciences and Institute of Theoretical Computer Science, ETH Zurich, Zurich, URL `http://www.educeth.ch/lehrpersonen/informatik/unterrichtsmaterialien\_inf/algorithmen\_datenstrukturen/k\_sat/index`.

PRESENTED PAPERS

2007

Balmer, M. (2007) Travel demand modeling for multi-agent traffic simulations: Algorithms and systems, Ph.D. defence, ETH Zurich, Zurich, May 2007, URL `http://www.ivt.ethz.ch/vpl/publications/presentations/v196.pdf`.

Rai, R. K., M. Balmer, M. Rieser, V. S. Vaze, S. Schönfelder and K. W. Axhausen (2007) Capturing human activity spaces: New geometries, poster & presentation, TRB 86th Annual Meeting, Washington, D.C., Jan. 2007.

2006

Balmer, M., K. Nagel and B. Raney (2006) An agent-based demandmodeling framework for large scale micro-simulations, poster presentation, TRB 85th Annual Meeting, Washington, D.C., Jan. 2006.

Balmer, M., K. W. Axhausen and K. Nagel (2006) A demand generation framework for large scale micro simulations, presentation, The 6th Swiss Transport Research Conference (STRC), Ascona, Mar. 2006, URL `http://www.strc.ch/2006.html`.

Balmer, M. and K. Nagel (2006) Shape morphing of intersection layouts using curb side oriented driver simulation, presentation, The 8th International Conference on Design & Decision Support Systems in Architecture and Urban Planning (DDSS), Heeze, Jul. 2006, URL `http://2006.ddss.nl/`.

2005

Balmer, M., M. Bernard and K. W. Axhausen (2005) Matching geo-coded graphs, presentation, The 5th Swiss Transport Research Conference (STRC), Ascona, Mar. 2005, URL http://www.strc.ch/2005.html.

Balmer, M., M. Rieser, A. Vogel, K. W. Axhausen and K. Nagel (2005) Generating day plans based on origin-destination matrices: A comparison study between VISUM and MATSIM based on Kanton Zurich data, presentation, The 5th Swiss Transport Research Conference (STRC), Ascona, Mar. 2005, URL http://www.strc.ch/2005.html.

Balmer, M., A. Vogel and K. Nagel (2005) Shape morphing of intersections using curbside oriented driver simulation, presentation, The 5th Swiss Transport Research Conference (STRC), Ascona, Mar. 2005, URL http://www.strc.ch/2005.html.

Balmer, M. (2005) Nachfrage Generierung (für MATSIM): Aufbau, Generierung und erste Test-Resultate, internal presentation, IVT, ETH Zurich, Zurich, Jan. 2005.

Balmer, M. and K. Meister (2005) Nachfrageerzeugung für agentenbasierte Simulation von Verkehrssystemen, presentation, Deutsche Hochschultagung Verkehrswesen, Wildbad Kreuth, Sep. 2005.

Balmer, M. (2005) MATSIM utility function, internal presentation, IVT, ETH Zurich, Zurich, Nov. 2005.

2004

Balmer, M., B. Raney and K. Nagel (2004) Agent-based activities planning for an iterative traffic simulation of Switzerland: Activity time allocation, presentation, The 4th Swiss Transport Research Conference (STRC), Ascona, Mar. 2004, URL `http://www.strc.ch/2004.html`.

Balmer, M. (2004) Agent-based activities planning for an iterative traffic simulation of Switzerland: Activity time allocation, presentation, Transport Systems Planning and Transport Telematics (VSP), Technical University Berlin, Berlin, Jul. 2004.

Balmer, M. (2004) Multi-Agenten Simulation: Resulate anhand einer Fallstudie der Region Zürich, presentation, Studentenbörse, Deutsche Verkehrswissenschaftliche Gesellschaft (DVWG), Berlin, Dec. 2004.