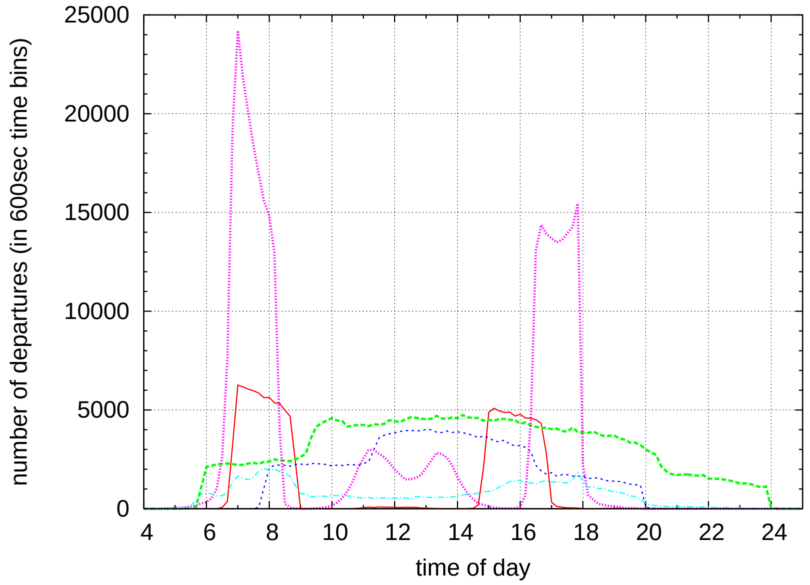


planomat: A comprehensive scheduler for a large-scale multi-agent transportation simulation

Konrad Meister, IVT, ETH Zurich
Michael Balmer, IVT, ETH Zurich
Kay W. Axhausen, IVT, ETH Zurich
Kai Nagel, VSP, TU Berlin

Conference paper
Session 1.4 Multi-agent simulation

The Expanding Sphere of Travel Behaviour Research

11th International Conference on Travel Behaviour Research
Kyoto, 16-20 August 2006

planomat:
A comprehensive scheduler for a large-scale multi-agent transportation simulation

Konrad Meister
Research Associate
IVT, ETH Zurich
ETH Hönggerberg, CH-8093 Zurich
Phone: +41-44-633 30 88
meister@ivt.baug.ethz.ch

Michael Balmer
Research Associate
IVT, ETH Zurich
ETH Hönggerberg, CH-8093 Zurich
Phone: +41-44-633 27 80
balmer@ivt.baug.ethz.ch

Kay W. Axhausen
IVT, ETH Zurich
ETH Hönggerberg, CH-8093 Zurich
Phone: +41-44-633 39 43
axhausen@ivt.baug.ethz.ch

Kai Nagel
VSP, TU Berlin
Salzufer 17-19, D-10587 Berlin
Phone: +49-30-314 23 308
nagel@vsp.tu-berlin.de

Abstract

An external strategy module for an iterative multi-agent microsimulation of traffic systems is presented. This module called *planomat* optimizes the time allocation and route choice of activity plans, which are the agent-based representation of travel demand. The module combines broad search for alternative timing decisions with an optimization procedure for a scoring function that evaluates activity plans. As part of the existing framework MATSIM-T, regional traffic systems of several 100'000 agents can be simulated. The scenario presented here is the Canton of Zurich, the biggest metropolitan area of Switzerland, with 550'000 agents. The comprehensive optimization of activity plans leads to a system relaxation within an acceptable number of 60 iterations. The quality of the time allocation optimization is shown by departure time distributions.

Keywords

activity plan, time allocation, scoring function, planomat, MATSIM-T

Preferred citation style

Meister, Konrad, Michael Balmer, Kay W. Axhausen and Kai Nagel (2006) planomat: A comprehensive scheduler for a large-scale multi-agent transportation simulation, paper presented at the 11^{th} International Conference on Travel Behaviour Research, Kyoto, August 2006.

1. Introduction

MATSIM-T is an iterative multi-agent framework for the micro-simulation traffic systems (MATSIM-T, 2006). It mainly consists on one side of a simulation of traffic flow and on the other side of different modules adapting travel demand to generalized travel costs. They are called alternately until the system reaches its stationary state, which corresponds to user equilibrium in the case of traffic systems. In MATSIM-T, travel demand is represented by individual agents that follow an activity plan. Each activity plan is assigned a score. The higher the score, the better is the plan. Convergence to the stationary state is, among other measurements, judged by the development of the score aggregated over the whole agent population.

This paper is about *planomat*, a flexible module which adapts the activity plans to travel times the agent experiences during the subsequent simulations of traffic flow. Since changing generalized costs of travel affect each aspect of travel demand, it would be desirable that this module was as comprehensive, allowing for choice of activity durations, departure times, activity locations, modes, and other desired attributes. In the implementation presented here, *planomat* optimizes activity durations, departure times and routes according to a time-of-day dependent appproximation of travel times.

The paper is structured as follows. Our concept of an agent-based microsimulation of traffic systems is presented in section 2. Details on the new module *planomat* are given in section 3. Section 4 describes input data, assumptions about activity parameters as well as algorithm details. Results concerning choice of activity timing and system performance are presented in section 5. Finally, an outlook is given in the last section.

2. Micro simulation framework

In this section, the concepts required for understanding the *planomat* functionality are described briefly. For a comprehensive and more detailed framework description, see Raney (2005).

2.1 The activity plan concept

The representation of an agent's travel demand is an activity plan, an alternating sequence of *activities* and *trips*. As shown in the example in Figure 1, the framework uses XML to store and exchange plans (W3C, 2006). The most important XML elements are the following.

person Each person is identified by an `id` by which its socio-economic attributes can be found in the synthetic population. A person can hold several plans.

plan Each plan can be assigned a `score` according to a scoring function (see section 2.2). The attribute `selected="yes"` states that the plan was chosen for execution in the previous iteration of the traffic flow simulation.

activities Each activity `<act>` is characterized by a type, a hectare-based location coordinate, a network link associated to that location, and its temporal extent defined by two of three attributes `start_time`, `end_time`, and `dur` (activity duration). The start of the plan is defined as the end time of the first activity, `07:35:04` in the case of the plan in Figure 1. In the example shown, first and last activity are the same activity (`"h"`, which means home). The location coordinates refer to "Swiss Grid", the Swiss geodetic reference system (Swisstopo, 2006).

legs Movements between activities are called legs. The attributes of a `<leg>` include a mode, a departure time and a duration. A leg can be characterized by a route, which is a sequence of numbers of the network nodes that are passed.

Read the example plan as follows:

- Agent No. 22018 is at home until 7:35:04. Its home location *"h"* is at the coordinates (703600;236900).
- The agent leaves its home to drive to work (*"w"*). This trip takes 16 minutes and 31 seconds, using the route along the nodes `1900 1899 1897`.
- The agent stays at work more than 8 hours, then leaves for a leisure activity (*"l"*). The trip from the work location on route `1899 1848 1925 1924 1923 1922 1068` to the leisure location takes about 1 hour and 10 minutes.
- After leisure, the agent returns home after a trip of ≈34 minutes.
- Read the plan as a 24-hour wrap-around, so the end of the home activity is also at 7:35:04 the next day.
- The plan has a score of 157.72€.

An activity plan can be interpreted in different ways: It can be either a *strategy* expressing what the agents wants/plans to do, or a *demand description* what an agent actually did in a certain iteration. The character of a plan is even more general: Since many attributes are not required, it is essentially a *working file* in the demand generation process. The formal requirements for an XML file are specified in a DTD (Document Type Definition) file. The various DTDs used in MASTIM can be found at MATSIM-T (2006).

2.2 Scoring

The quality of an activity plan is measured by a score. The corresponding scoring function was introduced first by Charypar and Nagel (2005), and is with slight modifications also used in our

Figure 1: Example activity plan

```
<person id="22018">
  <plan score="157.72" selected="yes">
    <act type="h" x100="703600" y100="236900" link="5757" end_time="07:35:04" />
    <leg num="0" mode="car" dep_time="07:35:04" trav_time="00:16:31">
      <route>1900 1899 1897</route>
    </leg>
    <act type="w" x100="702500" y100="236400" link="5749" dur="08:12:05" />
    <leg num="1" mode="car" dep_time="16:03:40" trav_time="01:10:22">
      <route>1899 1848 1925 1924 1923 1922 1068</route>
    </leg>
    <act type="l" x100="681450" y100="246550" link="2140" dur="01:20:00" />
    <leg num="2" mode="car" dep_time="" trav_time="00:34:35">
      <route>1067 1136 1137 1921 1922 1923 1924 1925 1848 1899</route>
    </leg>
    <act type="h" x100="703600" y100="236900" link="5757" />
  </plan>
</person>
```

current work on traffic micro simulation. This subsection presents the basic parts of the utility function, while subsection 2.3 demonstrates its use in the micro simulation framework. Since here is given a compressed description, the interested reader is referred to the original paper by Charypar and Nagel.

The score of an activity plan U_{plan} is given by the sum of the utilities of all performed activities i, and the travel disutilities for trips necessary to get from one activity location to the other:

$$U_{plan} = \sum_{i=1}^{n} U_{act}(type_i, start_i, dur_i) + \sum_{i=2}^{n} U_{trav}(loc_{i-1}, loc_i)$$

The utility of an activity i is the sum of four terms, each of which is modeling a certain aspect of the utility function.

$$U_{act,i} = U_{dur,i} + U_{wait,i} + U_{late.ar,i} + U_{early.dp,i} + U_{short.dur,i}$$

$U_{dur,i}$ denotes the utility of executing an activity for a certain duration, $U_{wait,i}$ denotes the (dis)utility of waiting for an activity to start (for instance waiting for a shop to open), $U_{late.ar,i}$ and $U_{early.dp,i}$ denote penalties for coming too late or leaving too early that activity respectively, and $U_{short.dur,i}$ is a penalty if an activity is performed for too a short time.

U_{trav} denotes the (dis)utility of traveling from the location of activity $i-1$ to the location of the current activity i.

There is no penalty for *not* performing an activity that might have been planned. Only performed activities contribute to the plan score.

Utility of performing an activity

All terms in the activity utility function except U_{dur} are modeled to be linear in time needed for that activity aspect. The time performing an activity is assumed to have a logarithmic impact on activity utility to reflect diminishing marginal utility:

$$U_{dur} = \begin{cases} \beta_{dur} \cdot t^* \cdot \ln(\frac{t_{dur}}{t_0}) & (t_0 \leq t_{dur}) \\ 0 & (0 \leq t_{dur} < t_0) \\ \beta_{neg.dur} \cdot |t_{dur}| & (t_{dur} < 0) \end{cases} \text{, with}$$

$$t_0 = t^* \cdot \exp^{-10/p \cdot t^*}.$$

t_{dur} denotes the actual activity duration. t^* is the so called *operating point* of the activity, the duration at which the marginal utility equals β_{dur}. So, the value of t^* can be interpreted as the typical duration of an activity, while its effect in the activity plan context is the following: The t_i^* yield the ratios of the durations of different activities in equilibrium.

t_0 is the activity duration at which the logarithmic curve has its null. It is chosen proportional to the operating point, and is influenced by the priority p of the activity. Usual values for p are 1,2,3..., with 1 being the highest priority. The higher the priority, the smaller will be t_0. In busy plans, high-priority activities tend to stay in the plan while low-priority activities will be dropped when for instance traffic conditions worsen. In the current state of our work on activity generation, we use fixed, revealed activity chains, and activity dropping is not allowed. All activities have the same priority $p = 1$. This is why this issue is not described in more detail here.

The utility of performing an activity with a positive duration cannot be negative. Due to the interpretation of an activity plan as 24 hour-wrap round, in the first iterations of the micro simulation framework negative durations can occur. They are penalized linearly with $\beta_{neg.dur}$. This reflects a very undesired plan where it took the agent more than 24 hours to fulfil its plan.

Penalties

The penalty terms of the utility function are penalized linearly according to Vickrey's model of departure time choice (e.g. Arnott *et al.*, 1993):

$$U_{trav}(t_{trav}) = \beta_{trav} \cdot t_{trav},$$

$$U_{wait}(t_{wait}) = \beta_{wait} \cdot t_{wait},$$

$$U_{late.ar}(t_{start}, t_{latest.ar}) = \begin{cases} \beta_{late.ar} \cdot (t_{start} - t_{latest.ar}) & (t_{start} > t_{latest.ar}) \\ 0 & (t_{start} \leq t_{latest.ar}) \end{cases}$$

(where t_{start} is the starting time of the activity and $t_{latest.ar}$ the latest possible starting time of that activity),

$$U_{early.dp}(t_{end}, t_{earliest.dp}) = \begin{cases} \beta_{early.dp} \cdot (t_{earliest.dp} - t_{end}) & (t_{end} < t_{earliest.dp}) \\ 0 & (t_{end} \geq t_{earliest.dp}) \end{cases}$$

(where t_{end} is the ending time of the activity and $t_{early.dp}$ the earliest possible ending time of that activity), and

$$U_{short.dur}(t_{start}, t_{end}) = \begin{cases} \beta_{short.dur} \cdot (t_{shortest.dur} - (t_{end} - t_{start})) & (t_{end} < t_{start}) \\ 0 & (t_{end} \geq t_{start}) \end{cases}$$

(where $t_{shortest.dur}$ is the shortest desired duration for that activity).

Summary of parameters

The parameters of the utility function have the following values:

$\beta_{dur} = 6$€/h,

$\beta_{trav} = -6$€/h,

$\beta_{wait} = 0$€/h,

$\beta_{late.ar} = -18$€/h,

$\beta_{early.dp} = 0$€/h,

$\beta_{short.dur} = 0$€/h,

$\beta_{neg.dur} = -18$€/h.

The parameters for the penalty terms are chosen to reflect the relations in Vickrey's model of departure time choice:

$\beta_{wait} : \beta_{trav} : \beta_{late.ar} = 1 : 2 : 3$

This relation is not obvious on first sight when looking at the parameter values:

$\beta_{wait} : \beta_{trav} : \beta_{late.ar} = 0 : -6 : -18$

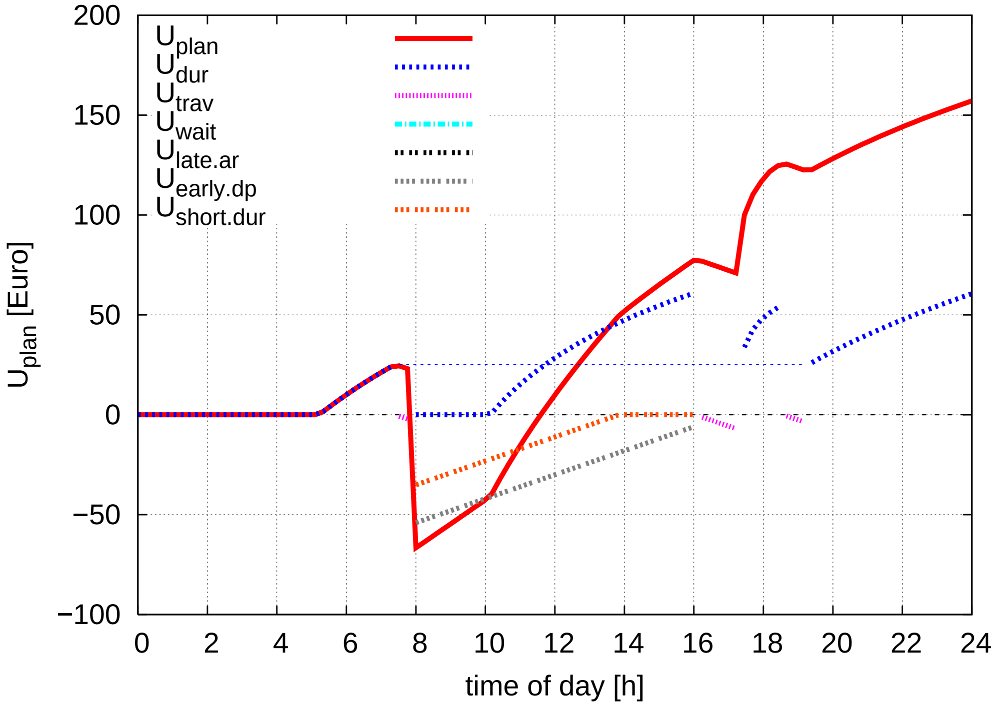
Considering the opportunity costs of *not* performing an activity while waiting or traveling, one has to subtract β_{dur} from β_{wait} and β_{trav}. So, the effective parameter values are the following:

$\beta_{wait,eff} : \beta_{trav,eff} : \beta_{late.ar,eff} = -6 : -12 : -18$,

which means the Vickrey type model is yielded. These values are different from the ones used in Charypar and Nagel (2005), who already discussed the issue of opportunity costs.

Figure 2 demonstrates the utility calculation using the example activity plan shown in Figure 1.

Figure 2: Utility plot of example activity plan



The graph U_{plan} represents the plan score depending on time of day as this plan was canceled at that certain time of day. One clearly sees positive utility of activity performance (log-shape graphs), the various penalties (linear graphs starting on the x-zero axis) as well as the overall plan score yielded at 24:00.
The very low score value between 8:00 and 10:00 can be explained as follows: On one hand, only the home activity and a small part of the work activity including the (penalized) home-work trip were performed. On the other hand, the penalties for early departure $U_{early.dp}$ and short activity performance $U_{short.dur}$ are very high.
The activity parameters used here are listed in Table 2, which is part of the scenario description in section 4.
For explanatory reasons, in this figure $\beta_{early.dp} = \beta_{short.dur} = -6$€/h, instead of 0€/h.
Based on Balmer (2005, p.15 ff.)

2.3 Simulation

The task of a simulation is to find the stationary state of the system modeled. In the case of our transport system model, the stationary state is the state where an agent cannot improve its score by altering the plan. This is analogous to what is called Nash equilibrium in game theoretic models or user equilibrium, the term for Nash equlibrium used in aggregate traffic assignment models (Ortúzar and Willumsen, 2001).

As pointed out, an iterative approach is used to solve this maximization problem. where travel times as a representative for generalized travel costs are the central feedback element. The overall simulation system consists of the following steps (compare Raney, 2005, p.77 et seqq.):

1. *Initialize:* A first set of plans has to be generated, assuming initial states of the network as well as the plan attributes. For example, the agent might assume free speed travel time for its preliminary set of legs and a random start time of the plan. For each agent, one plan is generated which will be marked as "selected", indicating it has chosen that plan for execution in the traffic flow simulation.

2. *Simulate:* The simulation of traffic flow executes the plans, that is it "moves" agent objects through a model of the traffic network when trips are planned. Currently, a queue-based, time-sliced model of traffic flow is used (Cetin, 2005). The output of the simulation is the so called events file which keeps detailed information about which agent "did" what during the simulated day.

3. *Scoring:* The agent database reads the events file and sends each event to the agent identified within it. Each agent uses its events to calculate the new score of its selected plan – the one it most recently sent to the traffic flow simulation. New plan scores are calculated as described in section 2.2, and are averaged with old plan scores. Score averaging is a simple mechanism to permit agents to learn about their plans performance over time. The agent averages scores according to:

$$S_p = (1 - \alpha) \cdot S_p + \alpha \cdot S_p^{'}, \tag{1}$$

   where S_p is the stored score for plan p, $S_p^{'}$ is the newly calculated score, and $\alpha \in [0, 1]$ is a blending factor. In the setup described here, a blending factor of $\alpha = 0.1$ is used.

4. *Plan pruning:* The agent database may limit the number of plans agents can store in memory. New plans are accumulated until the maximum number N_{plans} is reached. Any agent having a number of plans $P > N_{plans}$ in its memory deletes the $(P - N_{plans})$ plans with the lowest score in this step. Note that in the step following this one, an agent may obtain a new plan. When this happens to an agent that has already N_{plans}, it temporarily keeps $N_{plans} + 1$ plans in memory until the new plan has been scored. Then, in this step, it deletes the first plan (even if it is the newest one). Thus, the agent will have only N_{plans} to choose from when selecting from old plans.

5. *Replanning:* A subset of the agents is chosen for plan modification/new plan generation by so-called *external strategy modules*. These modules, of which *planomat* is one, can capture one or more travel behavior attributes. In the current setup, planomat is the only strategy module because it captures all the travel behavior aspects varied during the iterations. A random 10% of all agents are chosen to obtain new plans by planomat.

6. *Return* to step 2 until the system has reached a relaxed state which will be interpreted as the result of the simulation. The state of the system is called relaxed (or stationary) if there is no significant improvement in the average score of the plans selected by the agents for simulation in the last iteration.

3. Methods of planomat

Our idea for the external strategy module called *planomat* is to have a module that generates plans which are optimal in the sense of the scoring function described before. This is completely different to previous implementations of rescheduling modules where

- activity plan attributes were altered randomly (e.g. shifting activity durations / departure times $\pm 30min$), or
- optimization was performed for only a fraction of the travel behavior attributes that are varied in the iteration process (e.g. route optimization without the opportunity to alter the departure time).

Here, we propose a comprehensive rescheduler that suggests optimal plans considering the traffic conditions the agent experienced in the last iteration of the traffic flow simulation. In this section, first a method for travel time approximation is presented. It is followed by a description of the implementation of the genetic algorithm we currently use to solve the optimization problem.

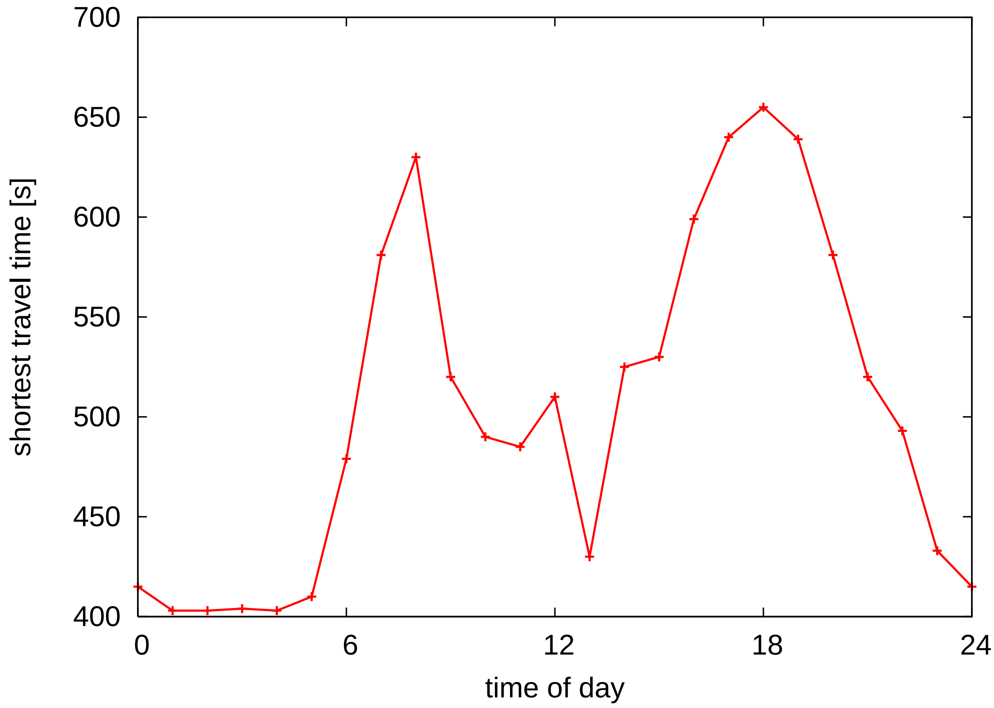
3.1 Travel time information

As pointed out, travel time is the only aspect of generalized travel costs in the proposed scoring function. The agent needs a time-of-day dependent approximation of travel times in order to react on traffic conditions varying throughout the day.

Our current approach to this is a very basic one: For each trip the agent has planned the location coordinates resp. the associated network links are given. For the agent it was wishful to exactly know what travel times are yielded at every point in time on every feasible route to decide which is the best activity timing/routing decision. The availability of such detailed information is not only unrealistic, but also infeasible to compute in useful time. Furthermore, such a level of exactness would only make sense if a particular agent was the only one performing a replanning. In this case the state of the network would be the same in the previous and the next iteration. But since 10% of all agents will obtain new plans, this assumption will most likely not hold.

In order to approximate the travel time for a given OD-pair, we obtain the shortest path and the associated travel time of each trip in certian time intervals. If an agent requests a travel time information for scheduled departure time, a linear interpolation between the two nodes in front of resp. after the departure time is returned. Currently we use 1h as node interval. So, if an agent plans a trip from A to B at 11:36 AM, it will be returned the linear interpolation of the shortest travel time information between 11:00 AM and 12:00 AM. Since we currently simulate daily activity plans, information at 12:00 PM will be set to the value at 0:00 (see Figure 3).

Figure 3: Approximation of OD travel time



The 1h-wise routing is done using a time-of-day dependent Dijkstra shortest path algorithm (Raney, 2005, p. 38 et seq.). So, for an agent which had three trips planned, $3 \cdot 24 = 72$ routings would have to be performed. This number is constant because every following travel time lookup is no more than a linear interpolation. Concerning the interval size, a fraction of 1h would possibly increase the quality of the plan, but also remarkably increase the computational effort. An even better method was one that samples more travel time information at times of day where many changes in trends are expectable (e.g. at the beginning of a peak period), and less where the trend is constant (e.g. close to free speed travel time in night hours).

3.2 Optimization

For several reasons, the decision was made to use a Genetic Algorithm (GA) to find good solutions in the sense of the utility function:

Flexibility In the current setup of the module, a better time allocation could be much easier calculated. GAs are not the best choice to solve continuous problems like this, they were designed to rather solve combinatorial problems. A gradient-based optimization procedure or an Evolutionary Strategy would probably be much faster and/or produce better results. Experiments are undertaken with the Covariance Matrix Adaptation-Evolution Strategy (CMA-ES), a stochastic population based optimization algorithm for continuous space problems (Hansen and Ostermeier, 2001). However, the goal is to extend *planomat* to a comprehensive replanning module incorporating further, combinatorial dimensions of travel behavior such as activity location choice, mode choice and the choice of the activity pattern. This is why we stick to the GA here.

Experience The GA method proved to be successful in various experiments for activity plan generation for individual agents or households (Charypar and Nagel, 2005; Meister *et al.*, 2005; Schneider, 2003). This paper is about the attempt to integrate this approach into a multi-agent simulation system.

The implementation details of the GA operators in the planomat are as follows, while Table 1 gives an overview of the values chosen for the various GA parameters. All these parameters have to be chosen according to the nature of the problem to be solved. This is often done on a gut level, so is in this case.

Generation of initial population For each agent, the selected plan is read in and the travel time information trajectory is generated as described in section 3.1. The start time of the plan, that is the end time of the first (home) activity, is uniformly selected between 00:00 and 12:00 PM. The same is done for the duration of each activity. All other attributes are kept constant as they came from the input plan (as described, in the current state of the work planomat only optimizes time allocation). For each agent, $popsize$ plan alternatives are generated.

Recombination and mutation The crossover operator recombines two existing plans to a new one by randomly choosing start time and activity durations from one of the parents. The mutation operator alters each time information in a certain range parameterized with the *mutation probability* p_{mut}:

- A new start time is chosen by adding an amount s uniformly selected from range $s \in [p_{mut} \cdot -12h,\ p_{mut} \cdot 12h]$. Values that come before 00:00 (midnight) are reset to that time.
- An activity duration is multiplied with a factor $d = e^X$ with X being uniformly selected from the range $X \in [-p_{mut}/2,\ p_{mut}/2]$.

Preparation for scoring After both the creation and the recombination/mutation operations, the new plan is stretched/compressed to a duration of 24 hours to be comparable to its competitors in the GA population. Furthermore, the anticipated travel times are calculated using the piecewise linear interpolation described before.

Scoring, selection and output Every time a new activity plan was created by the GA, it is evaluated with the scoring function. Since the number of plans held in the GA population at one time is constant, good plans are kept while bad ones are dropped. After a certain

Table 1: GA parameters

Variable	Description	Value
$popsize$	Constant population size.	`50`
n_{gen}	When a fixed stop criterion is used: The optimzation is canceled after n_{gen} individuals were generated by the crossover/-mutation operations.	`1'000`
ϵ_{stop}	When the adaptive stop criterion is used: If the average fitness doesn't increase more than $\epsilon_{stop}\%$ after n_{stop} newly inserted plans, the optimization is canceled.	`1.0`
n_{stop}	see ϵ_{stop}	`50`
p_{mut}	Probability that one element of an activity will mutate according to its respective mutation operator.	Initial: `0.30`, exponentially decreasing to `0.07`
τ_{mut}	Each time a new indivdual was inserted into the population, p_{mut} is adapted. The higher τ_{mut}, the quicker p_{mut} decreases.	
$mindiff$	Minimum fitness difference between two individuals. If a new plan with almost the same score is generated, it will be dropped in favor of the one that is already present.	`0.10`

number of recombination/mutation operations, the optimization is canceled. This may either happen after a fixed number of iterations n_{gen}, or if the average fitness of the population doesn't increase more than a threshold ϵ_{stop} within a number of newly plans that had a high enough score to be inserted in the GA population. The setup presented here uses the latter, adaptive stop criterion.

The best plan currently in the population is chosen as the agent's new strategy to be evaluated in the next iteration of the traffic flow simulation. Before returning the plan to the agent database, it is routed a last time using the router directly (instead of the approximation with the linear interpolation). This is done in order to provide the agent the actual route of whose travel time we assume that it is not too different from what the approximation suggested.

4. Canton Zurich Scenario

The scenario setup includes a regional definition of the study area, the demand generation process, the specification of the traffic network and a list of assumptions about activity-related behavior as well as temporal constraints.

4.1 Study area: Canton Zurich

The case study used for testing the *planomat* is a simulation of the Canton Zurich, the biggest metropolitan area in Switzerland. The demand generation process, as well as the framework used for it, is described in detail in Balmer *et al.* (2006).

First, a synthetic population of the Canton Zurich is generated, using data from the Swiss National Population Census. It is a list of ≈1'200'000 agents with individual attributes like age or sex, and a hectare-based home location (Frick and Axhausen, 2004). Each agent is assigned an activity chain based on the Swiss Microcensus on travel behavior (Rieser, 2004). These activities are distributed in space by several location choice modules (Marchal and Nagel, 2006). The network model used for the assignment with a microscopic traffic flow simulation is the Swiss National Traffic Network model (Vrtic *et al.*, 2003).

4.2 Activity parameters and constraints

The scoring function requires several parameters, either activity or location specific.

Each activity is characterized by a typical duration t^*, a mimimum duration $t_{shortest.dur}$ and desired start/end times $t_{latest.ar}$, $t_{earliest.dp}$. While the typical duration is a mandatory parameter to the utility function, the minimum duration and desired time windows are optional. Table 2 is a list of parameter values used in this scenario.

Furthermore, there exist temporal constraints for the execution of activities, represented here by opening hours. An agent will fail to perform an activity outside these opening hours, and will have to wait instead. In this case, it doesn't gain any score or even loses some in case of $\beta_{wait} < 0$. The temporal constraints are an attribute of a specific facility. In this setup, they are the same all over the modelled region because more detailed data about opening hours was not available yet. This is why they appear activity-specific in Table 3.

For analysis, the activity chain types are summarized into five groups:

education-dominated chain types `heeh`, `heh`

leisure-dominated chain types `hlh,hllh,hlslh`

shop-dominated chain types `hsh,hssh`

Table 2: Activity parameter values

Activity type	abbreviation	t^* [h]	$t_{shortest.dur}$ [h]	$t_{latest.ar}$	$t_{earliest.dp}$
home	`h`	12	8	—	—
work	`w`	8	6	9:00	—
work1	`w1`	4	2	9:00	—
work2	`w2`	4	2	—	—
work3	`w3`	8	6	—	—
education	`e`	6	4	9:00	—
education1	`e1`	3	1	9:00	—
education2	`e2`	3	1	—	—
education3	`e3`	6	4	—	—
shop	`s`	2	1	—	—
leisure	`l`	2	1	—	—

All activities have the same priority $p = 1$.
The different work and education activity types can be explained as follows. If an activity chain includes two *work* or *education* activities, it is assumed that their typical activity duration is half the complete-activity duration and will be renamed *work1* and *work2* resp. *education1* and *education2*. An example would be `h-w1-l-w2-h`. If a work or education activity is not the first an the activity chain, it is renamed *work3* or *education3* without the desired start time at 9:00, but all other attributes equal. An example of that would be `h-s-w3-h`
.

Table 3: Opening hours as temporal constraints

Activity type	opening time	closing time
home (`h`)	—	—
work (`w`, `w1`, `w2`, `w3`)	7:00	18:00
education (`e`, `e1`, `e2`, `e3`)	7:00	18:00
shop (`s`)	8:00	20:00
leisure (`l`)	6:00	24:00

work-dominated chain types `hwh`,`hwlwh`,`hwswh`,`hwwh`

other chain types `helh`,`hesh`,`hleh`,`hlsh`,`hlwh`,`hslh`,`hswh`,`hweh`,`hwlh`,`hwsh`

5. Results

5.1 A world without congestion

In order to test the optimization capability of the GA, the plans of all 550'000 agents were generated assuming free speed travel time in the network. The result might be interpreted as "a world without congestion", as the plans will be completely independent of traffic conditions changing throughout the day. They are only determined by the agents' preferences which are formulated in the utility function as well as environmental constraints (e.g. opening times). The result is shown in Figure 4, and to be read like the following:

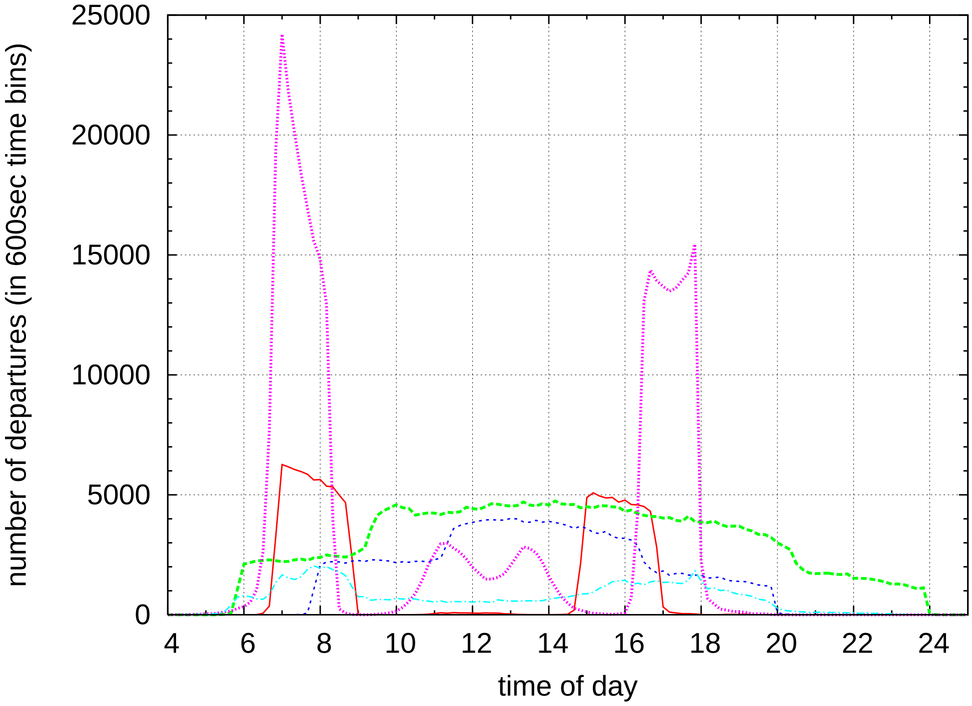
- Peak periods can be seen for the work- and education-activity chain types. They are the result of the trade-off between the latest start times of the main activity (9:00 in this case), and the extension of the time "spent" at home according to the specification of the utility function. The variance of the departure times is only determined by the distribution of trip distances between the home and the work resp. education activity. There are additional, smaller peaks in the time around noon (12:00 AM). These are departures to additonal activities besides the main work activity, e.g. of agents with activity chain type `h-w1-l-w2-h`.
- The departure time distributions of activity chain types which are dominated by shop or leisure activities have quite a uniform shape. They are only constrained by the respective opening/closing times, about which assumptions were made in Table 3. For example, all shop activities in the shop-dominated activity chain type graph are located between 8:00 and 20:00. Since travel times are the same all the day, the utility landscape within these opening time windows is "flat". Each of the graphs has two flat levels. While the lower one represents agents with only one out-of-home activity (e.g. `h-l-h`), the higher one are the departures of the agents with additional activities (e.g. `h-s-s-h`).

There are some time allocations which are likely suboptimal. Considering the work-dominated activity chain types, some agents leave work after 18:00, which is the closing time of the work facilities. After that time, no agent should attempt to perform the work activity because no utility can be derived from time spent waiting. At the moment it is unclear if this shows a limit of the optimization or, unexpectedly, is really an optimal time allocation.

5.2 Complete scenario simulation

The iterative simulation of traffic flow and strategy optimization by planomat were tested with four different setups of the agent database. Agent memory sizes of $N_{plans} = 1$ and $N_{plans} = 3$ were combined with score averaging switched on and off (compare section 2.3). The agent

Figure 4: Departure time distribution by activity chain type: Free speed travel time, ≈550’000 agents



database used to serve as the learning framework selecting the best strategies, when external strategy modules were only optimizing one particular travel behavior attribute resp. randomly altering them. Setups with $N_{plans} = 1$ are simulated to test whether the strategy generation/learning can be performed in a (computer memory-efficient) external strategy module rather than in the (heavily computer memory-demanding) agent database. Setups without score averaging are intended to explore the need of successively averaging provisional solutions of a stochastic optimization procedure like the MATSIM-T microsimulation framework.

For test reasons, the traffic of only a 1% sample of the whole agent population is simulated. In order be able to still produce some congestion and sensitivity of timing decisions to experienced travel times, the network capacity was reduced to a similar fraction as the agent population.

The results of these experiments are presented in Figure 5. It shows the development of the average score of the most recently simulated plans across the whole agent population. Its steady-state density is used to determine when the system converges to a user equilibrium, where no agent can unilaterally improve its score. The four upper graphs, each representing a different setup of the agent database, show a tendency towards a limiting value which is reached after ≈60 iterations.

Variation of N_{plans} In general, setups with $N_{plans} = 1$ converge to the same average score level as setups with $N_{plans} = 3$, while convergence speed is slightly higher. This can be explained like the following: The planomat always generates plans optimized for travel times yielded in the previous iteration, assuming this time and space-dependent landscape unchanged in the next iteration. Of course, this is not the case since not only one agent but 10% of the entire population are provided the generation of a new strategy. But as closer this assumption is to what will happen in the next iteration, the better applies the predicted best strategy, and the better the system will perform. However, with $N_{plans} > 1$, for some agents a random plan is chosen for the next simulation of traffic flow. This leads to an additional change in the time-space travel time landscape assumed by the planomat in the previous cycle of strategy generation, and therefore a worse prediction. With $N_{plans} = 1$, each agent whose plan is not optimized by planomat will be simulated with the same plan as before, as assumed by planomat.

Variation of score averaging As Figure 5 shows, setups with score averaging converge slower, but yield a higher steady state as the ones without score averaging. In the first iterations, the plans' scores rapidly increase because there is still a great potential for improvement by finding better routes and/or peak spreading. This effect is dampened by the score averaging technique which explains the slower convergence. The reason why a higher steady state is reached is not yet understood and has to be investigated. A possible explanation is that the result is the same as in the setup without score averaging, but the displayed averaged scores are misleading because they do not represent the true scores yielded in the traffic flow simualtion.

Figure 5.2 presents the departure time distribution of iteration 100, with the agent database

Figure 5: Convergence of average scores

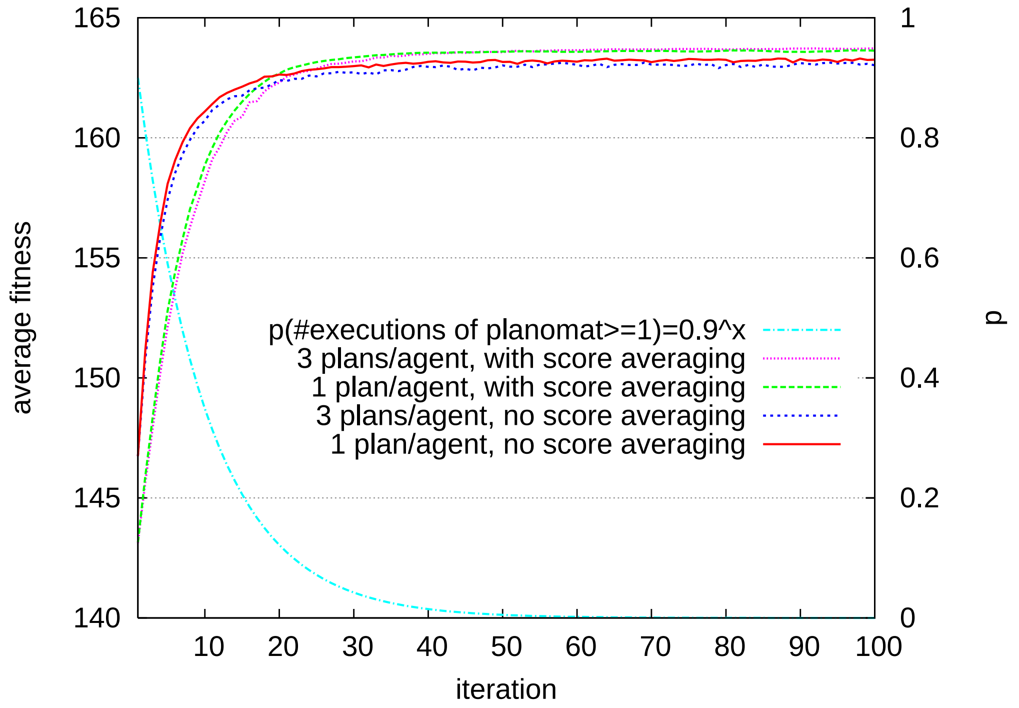
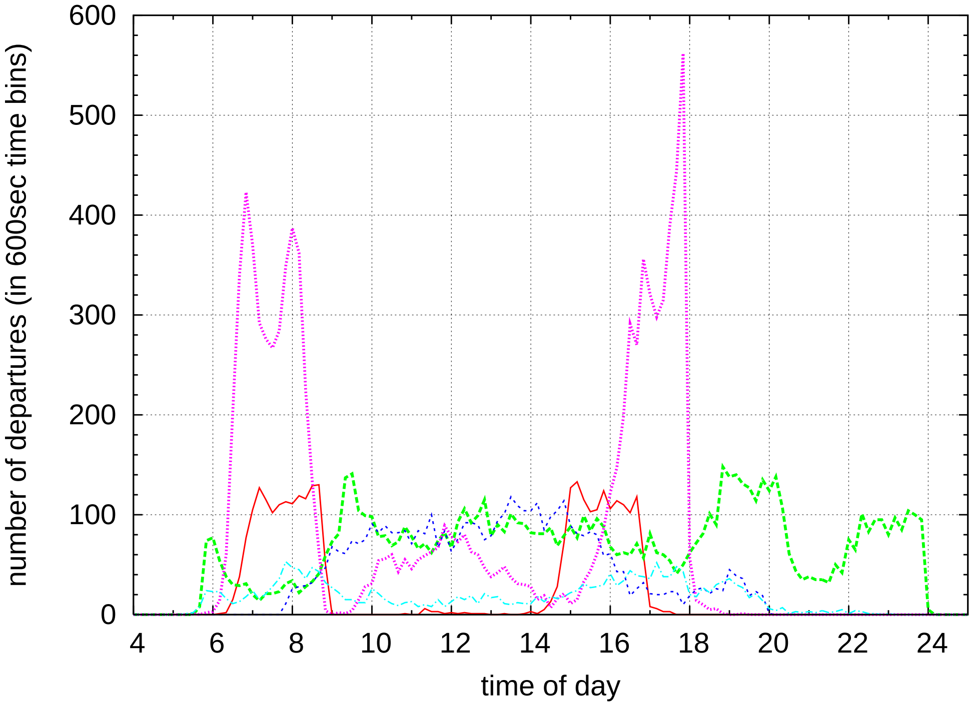


Figure 6: Departure time distribution by activity chain type - iteration 100, ≈12'000 agents



setup N_{plans}=1, no score averaging used. The main differences compared to a free speed travel time world are:

Peak spreading of work trips The peak periods of the work-activity dominated chains have widened, which is a result of an increased level of congestion on the network links around work facilities in the region of desired arrival/departure times. Also, the two local maxima at 11:00 AM and 1:30 PM from Figure 4 have merged into one, wider peak with maximum at 12:00 AM.

Off-peak concentration of shop/leisure trips Activity chain types that are dominated by activities without a desired time window tend to be allocated in off-peak regions. For example, consider the maxima of departures in leisure-dominated chains before the morning peak period around 6:00 AM, after that period around 9:30 AM, and after the evening peak period from 7:00 to 12:00 PM. Also, the major share of the trips in the shop-dominated chains is shifted to the region between the peak periods. This shift is not as obvious as for the leisure activities because shop activites are constrained to opening time windows close to the peak periods anyway.

6. Discussion and outlook

6.1 Computing issues

All figures presented here apply to a *Sun Fire X4100 Extra Large* machine, AMD Opteron 2 Model 275 (Dual Core), 1 MB L2 Cache, 8 GB RAM, Debian Etch with gcc 4.0.3. The entire simulation system was run using a single Dual Core processor.

The overall runtime for one iteration of the 550'000 agents scenario is $\approx$2000 seconds. Sufficient convergence could be shown after 60 iterations, which results in an overall runtime of one and a half days. This is a massive improvement compared to former versions of MATSIM-T, mainly due to the reduction of required iterations from several hundreds to around 60. The following description presents the share of runtime of each element of the simulation system, and discusses approaches to runtime improvements.

Traffic flow simulation The synchronous, queue-based simulation of traffic flow takes 700 s to simulate 24h plans of 550'000 agents, which is a Real Time Ratio (RTR) of 100. Recent experiments with an event-based version of the queue model let expect an RTR of about 300.

Planomat The planomat module yields a replanning performance of 75 agents/s. Of the runtime of ca. 730 s, a fifth is required to read the events produced by the traffic flow simulation. By far the biggest share of runtime takes the routing of the planned trips for travel time approximation by linear interpolation described in Section 3.1. So the replanning performance depends highly on the choice of the travel time information interval (currently 1h). Furthermore, the use of smarter optimization algorithms such as Evolution Strategies might help to reduce the required number of generations during one optimization.

Event file I/O The agent database requires 400 seconds, or 20% of overall runtime to read events in order to score the simulated plans. The main reason is the property of low performance of I/O based on text files.

Plans I/O About 9% or 120 s are required for exchanging plan information between the agent database and the planomat. Our current efforts on system integration include the abolishment of file-based plans exchange during the iterations (Balmer *et al.*, 2006).

Computer memory requirements are no limiting factor to performance, since optimization is done agent by agent. The temporary caching of the events information of 10% of all agents takes several dozens of megabytes which nowadays doesn't create a problem.

The technical improvements described get a high priority considering our vision to include more aspects of travel behavior into MATSIM-T.

6.2 Improvement of the location choice concept

One upcoming modeling goal is the improvement of the location choice concept. The basic difference will be that location choice for secondary activities will be part of the replanning process, instead of its currently limited role as a preprocess to initial demand generation (Marchal and Nagel, 2006).

At first, we will improve the data basis. Up to now, the number of overall workplaces in a spatial aggregate was assumed as predictor for the utility gained there, regardless of the activity type. This is insufficient because the functional organization typical for urban areas is not considered at all. We create an activity-fine set of facilities based on landuse information available on hectare-level for all Switzerland, called the Swiss National Enterprise Census provided by the Swiss Federal Statistical Office (BfS, 2001). Opening time windows will be no more activity-specific, but location-specific. Data about opening times still have to be imputed/revealed. Furthermore, the synthetic facilities will have an activity-specific capacity which in the first run will be proportional to the number of workplaces. An open question is how to include location capacity constraints into the agents' decision making.

For each agent, a choice set of locations is generated. Here, an approach based on revealed activity spaces is chosen. Refer to *activity space* as a continuous spatial representation of the locations visited by a person in a certain time range. We will use activity space generation algorithms developed in Vaze *et al.* (2005). It is then task of the *planomat* to find the best location for each activity in the sense of the scoring function. The complexity of the search space is thus extended with a non-scalar dimension *activity location*. Earlier GA experiments show that this task is feasible, although it will take more time than the comparably simple time allocation problem (Charypar and Nagel, 2005; Meister *et al.*, 2005).

References

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.

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

Balmer, M., K. W. Axhausen and K. Nagel (2006) A demand generation framework for large scale micro-simulations, in *TRB 85th Annual Meeting Compendium of Papers CD-ROM*, Transportation Research Board, Washington, D.C.

BfS (2001) *Eidgenössische Betriebszählung 2001 - Sektoren 2 und 3*, Swiss Federal Statistical Office, Neuchâtel, `http://www.bfs.admin.ch/bfs/portal/de/index/dienstleistungen/servicestelle_geostat/datenbeschreibung/eidgenoessische_betriebszaehlung2.html`.

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

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

Frick, M. and K. W. Axhausen (2004) Generating synthetic populations using IPF and monte carlo techniques: Some new results, in *The 4th Swiss Transport Research Conference (STRC)*, Institut für öffentliche Dienstleistungen und Tourismus (IDT), University of St.Gallen (HSG), Monte Verità, Ascona, `http://www.strc.ch/2004.html`.

Hansen, N. and A. Ostermeier (2001) Completely derandomized self-adaptation in evolution strategies, *Evolutionary Computation*, **9** (2) 159–195.

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. Frick and K. W. Axhausen (2005) A GA-based household scheduler, *Transportation*, **32** (5) 473–494.

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

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

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

Schneider, A. (2003) Genetische Algorithmen zur Optimierung von Tagesplänen für Verkehrsteilnehmer, *Semester Project*, ICoS, ETH Zurich, Zurich, http://www.vsp.tu-berlin.de/archive/sim-archive/publications/schneider.adrian-sa/optimizer.pdf.

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

Vaze, V. S., S. Schönfelder and K. W. Axhausen (2005) Continuous space representations of human activity spaces, *Working Paper*, **295**, IVT, ETH Zurich, Zurich, http://www.ivt.ethz.ch/vpl/publications/reports/ab295.pdf.

Vrtic, M., P. Fröhlich and K. W. Axhausen (2003) Schweizerische Netzmodelle für Strassen- und Schienenverkehr, in T. Bieger, C. Laesser 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.