ADJUSTMENT OF ACTIVITY TIMING AND DURATION IN AN AGENT-BASED TRAFFIC FLOW SIMULATION

Michael Balmer, Institute for Transport Planning and Systems (IVT), ETH Zurich, Switzerland

Bryan Raney, Institute of Computational Science (ICOS) ETH Zurich, Switzerland

Kai Nagel, Institute for Land and Sea Transport Systems, TU Berlin, Germany

ABSTRACT

One possibility to bring activity-based demand generation into the transportation planning process is to use it to replace the first three steps of the 4-step-process. In this case, the activity-based demand generation produces a standard origin-destination (OD) matrix, which is then fed into the existing assignment model. Both the OD matrix and the assignment model could be time-dependent.

Since both the activity-based demand generation and modern dynamic traffic assignment models work on the level of individual travelers, using the OD matrix to couple those travelers means that one has synthetic travelers on both sides, but they are not related. We, in contrast, propose to use a truly agent-based representation of the traffic system and the assignment process. In a truly agent-based representation, each person remains individually identifiable throughout the whole simulation process. This approach allows a completely consistent modeling of the behavioral processes related to transportation.

In our work, we have already demonstrated that such an approach is possible for large scale scenarios with several millions of travelers. The newest version of our modeling system is able to call arbitrary "strategy generation" modules, which are able to use individualized (or aggregated) performance to update plans of each agent. Those updated plans can be fed back

into the traffic micro-simulation. In this paper, we demonstrate, besides the use of a routing module, a time allocation module. This module plays the role of departure time choice modules in OD-based models, but it makes travelers adjust *all* their timing, i.e. not just departure time, but also duration of activities etc., throughout the whole day. Our paper shows that already a simple version of a time choice module, together with a utility function to "score" different plans, leads to plausible results. The simulation is run both on test cases and on real world scenarios. The latter will concentrate on the larger Zurich metropolitan area, with about 1 million of travelers. We include validation and sensitivity testing in comparison with real-world traffic counts as far as such data is available.

Keywords: traffic simulation, transportation planning, route planning, learning, multi-agent simulation, activity-based demand generation

TABLE OF CONTENTS

1. INTRODUCTION

One possibility to bring activity-based demand generation into the transportation planning process is to use it to replace the first two (or three) steps of the 4-step-process. In this case, the activity-based demand generation produces a standard origin-destination (OD) matrix, which is then fed into the existing assignment model. Both the OD matrix and the assignment model could be time-dependent.

The advantage of this method is that it ties in with the arguably most sophisticated and best understood part of the 4-step-process, which is the route assignment. Yet, some of these advantages go away when the OD matrices are time-dependent: In that situation, very few of the mathematical results of static assignment carry over. In addition, coupling activity-based demand generation to network assignment via an OD matrix severs the connection between individuals and their performance. Any iterative feedback of the traffic system performance to the activity generation can only be based on aggregate measures, such as link travel times, not on individual performance of the traveler. An obvious case where the coupling via OD matrix goes wrong is that it is possible for a person to leave an activity *even before he/she has arrived at it*.

In this situation, we propose, as a way toward further progress, to use a truly agent-based representation of the traffic system and the assignment process. In a truly agent-based representation, each person remains individually identifiable throughout the whole simulation process. In particular, the traffic micro-simulation assumes the role of a realistic representation of the physical system, including, say, 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 found "on the ground", rather than to aggregated conditions.

This multi-agent concept consists of basically two parts: (i) the simulation of the "physical" properties of the system, and (ii) the generation of the agents' strategies. The simulation of the physical system is the place where the agents interact with each other—car drivers produce congestion, traffic lights change their intervals dependent on the amount of traffic, pedestrians wait for the next train to catch, and so on. The agents make their strategies based on what they experienced in the physical simulation—car drivers try other routes to avoid congestion, pedestrians need to leave earlier to catch the train, traffic lights favor the main streets to maximize the throughput of an intersection, etc.

We are in the process of implementing such a multi-agent simulation for the whole Switzerland. This paper concentrates on the Zurich area, with about 260,000 agents that cross this region. The challenges with such an implementation are many: availability and quality of input data, computational implementation and computational performance, conceptual understanding of agent learning, and validation.

In past work (Raney *et al.*, 2003), we have reported first results based on typical transportation planning data: standard origin-destination matrices; the transportation planning network from the corresponding Swiss federal planning authority; and it performed route assignment (dynamic traffic assignment; DTA) based on these input data. The two main differences to other DTA systems, such as DYNASMART (www.mit.edu/its) or DYNAMIT (www.dynasmart.com), were that our system uses individual route plans for each agent while standard DTA systems store the routing decisions in the network, and our system was run on really large scale scenarios with several millions of travelers. A newer version of DYNASMART, however, now also uses individual routes, and other systems also move towards larger and larger scales. In contrast to TRANSIMS (www.transims.net), which has used individual routes and large scales for many years now, we used a so-called agent database, which keeps track of several plans per agent.

This paper goes further by now also internalizing the time structure of the input data. In other words, it is possible for the simulation system itself to predict when agents begin and end their main activities. The main result is that it is possible to completely ignore the time structure of the time-dependent OD matrices without compromising the quality of the predictive power. This is similar to the approach used and results obtained with METROPOLIS (De Palma and Marchal, 2002). The main difference is that our implementation uses complete daily activity chains, while METROPOLIS only uses trips. We believe that our method, while computationally more demanding, opens the door to more flexible transportation planning models in the future.

This paper will continue with an outline of the general simulation structure (Sec. 2), where we also describe the modules we will use in more detail. Next we introduce the network and the scenario (Sec. 3). The results (Sec. 4) of the different setups of the scenario are compared with traffic count data (Sec. 3.3) of this region. Some computational issues are pointed out in section 5. The paper is concluded by a section on future work (Sec. 6) and a summary (Sec. 7).

2. SIMULATION STRUCTURE

2.1. Overview

As pointed out before, our simulation 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 our simulation. The overall approach consists of three important pieces:

- Each agent independently generates a so-called *plan*, which encodes its intentions during a certain time period, typically a day.
- All agents' plans are simultaneously executed in the simulation of the physical system. In this paper, this is a *traffic flow simulation*. In other papers, we use the term *mobility simulation* in order to stress that the simulation of the physical system can go beyond traffic.
- There is a mechanism that allows agents to *learn*. In our implementation, the system iterates between plan generation and traffic flow simulation. The system remembers several plans per 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. Further details will be given below.

As this is an application to traffic forecasting, a plan contains the itinerary of activities the agent wants to perform during the day, plus the intervening 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. A typical plan could look like this:

- stay at home till 07:38 AM
- take the car to drive to work by the following route (encoded in a street network): 321, 2, 34, 65, 21, 34
- arrive at work at 07:58 AM
- work for 8 hours and 24 minutes until 04:22 PM
- take the car to drive home by the following route: 33, 25, 81, 109, 3, 322
- arrive at home at 04:41 PM

This paper concentrates on "home" and "work" as the only activities, and "car" as the only mode. Note that in this case we do not distinguish between a trip (between two activities) and a

leg (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 depend on one another, but the decisions made by one agent are independent of those made by another. We divide the task of generating a plan into sets of 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 plan with some of the decisions updated.

Some possible modules are:

- *Activity pattern generator*: Decides which activities an agent actually wants to perform during the day, and in what order it will perform them. At present this module is not used, and we have a fixed pattern of "home-work-home" for all agents.
- *Activity location generator*: Determines where the agent will perform each activity. At present this module is not used, and we have a fixed location for each agent's "home" and "work" activity.
- *Activity time allocator*: Determines the timing attributes the agent will utilize for each activity in its 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 "activity end time", whichever is earlier. Activities cannot overlap in time.
- *Router*: Determines which route and which mode the agent chooses for each trip leg that connects activities at different locations. The study presented in this paper uses car as the only mode.

A specialty of our approach is that arbitrary numbers and types of such modules can be "plugged in" to the framework, as long as they generate some information that contributes to a plan. For that reason, it is easy to, say, combine activity and mode choice into one module, or to add residential or workplace choice. This paper will employ two modules only: "activity time allocator" and "router". Other modules will be the topic of future work.

Once the agent's plan has been constructed via the modules, it can be fed into the traffic flow simulation module. This module executes all agents' plans simultaneously on the network, allowing agents to interact with one another, and provides output describing what happened to the agents during the execution of their plans.

The above modules produce dependencies about their computed information. The outcome of the traffic flow simulation module (e.g. congestion) depends on the planning decisions made by the decision-making 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 decision-making modules and the traffic flow simulation module.

We need these modules to be consistent with one another, so we introduce feedback into the traffic flow simulation structure (Kaufman *et al.*, 1991; Nagel, 1995; Bottom, 2000). This sets up an iteration cycle which runs the traffic flow simulation with specific plans for the agents, then uses the time allocator and the router to update the plans; these changed plans are again fed into the traffic flow simulation, 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.

In the following sections we describe the modules in more detail.

2.2. Activity Time Allocator

This module is called to change the timing of an agent's plan. At this point, a very simple approach is used which just applies a random "mutation" to the duration and end time attributes of the 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 [-30 min, +30 min] and adds it to the attribute. Any negative duration is reset to zero; any 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 very sophisticated, it is sufficient in order to obtain useful results (Sec. 4.3). 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.

2.3. Router

The router is implemented as a *time dependent Dijkstra algorithm*. It first calculates link travel times from the events output of the previous traffic flow simulation (see Sec. 2.4). The link travel times are aggregated into 15 minute time bins, and then used as the weights 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 this and the knowledge about activity chains, 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 traffic flow simulation.

2.4. Traffic Flow Simulation

The traffic flow simulation simulates the physical world. It is implemented as a queue simulation (Gawron, 1998; Cetin and Nagel, 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 it is filled up, no more agents can enter this link. Third, there is a flow capacity which limits the number vehicles that can leave the link in any given time step.

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

The output that the traffic flow simulation produces is a list of events for each agent, such as entering/leaving link, left/arrived at activity, and so on. Data for an event includes which agent experienced it, what happened, what time it happened, and where (link/node) the event occurred. With this data it is easy to produce different kinds of information and indicators like link travel time (which i.e. will be used by the router), trip travel time, trip length, percentage of congestion, and so on.

2.5. Agent Database—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 traffic flow simulation.

It is very important to note that our framework always uses *actual plan performance* for the score. This is in stark contrast to all other similar approaches that we are aware of—these other approaches always feed back some aggregated quantity such as link travel times and reconstruct performance based on those (e.g. URBANSIM—www.urbansim.org; Ettema *et al.*, 2003). 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:

1. *Initial conditions*: Start with a plans-file that specifies one complete plan per agent. The agent database loads these plans file into the memory of the agents. Each agent marks its initial plan as its "selected" plan.
2. *Simulate*: The agent database sends the set of "selected" plans (one per agent) to the traffic flow simulation. The simulation executes the plans simultaneously and outputs events.
3. *Process events*: The agent database reads the events outputted 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.
4. *Plan pruning*: The number of plans kept in an agent's memory for reuse can be limited to N plans for purposes of memory conservation. 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 (see below), 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).
5. *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:
   (a) *(10 %) New plan, routes only*: The agent sends an existing plan (chosen with equal probability among all plans in memory) to the router. 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".

(b) *(10 %) New plan, times and routes*: The agent sends an existing plan (chosen with equal probability among all plans in memory) to the activity time allocation module. 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 replanning. When it comes back from the route replanner 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.)

(c) *(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".

(d) *(Rest) 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

$$p \propto e^{b \cdot S_j},$$

where S_j is the score of plan j, and b is an empirical constant. This is equal to a logit model from discrete choice theory. The chosen plan is marked as "selected".

6. The cycle returns to step 2, and continues until the system has reached a relaxed state. At this point, there is no quantitative measure of when the system is "relaxed"; we just allow the cycle to continue until the outcome seems stable.

Note that in the above, when an agent reuses an existing plan, its previous score is not forgotten, but averaged with its new score:

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

with the blending factor α. This allows the agent to base plan selection on more of the plans' history than just the last iteration. With $\alpha = 0$ no score will be updated and the agents will not learn. With $\alpha = 1$ 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.

2.6. 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). In this work, a simple utility-based approach is used. The approach is related to the Vickrey bottleneck model (Arnott *et al.*, 1993), but needs to be modified in order to be consistent with our approach based on complete daily plans (Charypar and Nagel, 2003; Raney and Nagel, 2004). The elements of our approach are as follows:

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

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

- where U_{total} is the total utility for a given plan; n is the number of activities, which equals the number of trips; $U_{perf,i}$ is the (positive) utility earned for performing activity i; $U_{late,i}$ is the (negative) utility earned for arriving late to 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:

$$U_{perf,i}\left(t_{perf,i}\right) = b_{perf} \cdot t_i^* \cdot \ln\left(\frac{t_{perf,i}}{t_{0,i}}\right),$$

  where $t_{perf,i}$ is the actual performed duration of the activity, and t_i^* is the "typical" duration of an activity, and b_{perf} is the marginal utility of an activity at its typical duration. b_{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 becomes negative, implying that the agent should rather completely drop that activity. A $t_{0,i}$ only slightly less than t_i^* means that the marginal utility of activity i rapidly increases with decreasing $t_{perf,i}$, implying that the agent should rather cut short other activities. This paper uses

$$t_{0,i} = t_i^* \cdot e^{-V/\left(p \cdot t_i^*\right)},$$

where V is a scaling constant set to 10 hours, and p is a priority indicator, here set uniformly to one. Note that with this specific form, $U_{perf,i}(t_i^*) = b_{perf} \cdot V$, independent of the activity type. This "consequence" is actually the motivation for the specific mathematical form of the activity performance utility contribution, which was used because no better argument was available; future research should clarify this issue.

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

$$U_{late,i} = b_{late} \cdot t_{late,i},$$

where $b_{late} \leq 0$ is the marginal utility (in Euro/h) for being late, and $t_{late,i}$ is the number of hours late to activity i. To be able to calculate the utility of being late, a starting time window for the activities has to be given (see sec. 3.4 for more details).

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

$$U_{travel,i} = b_{travel} \cdot t_{travel,i},$$

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

Note that the utilities $U_{perf,i}$, $U_{late,i}$ and $U_{travel,i}$ are independent in contrast to (Weiss, 1999), where e.g. activity utilities can depend on previous activities. Those restrictions are not relevant for this paper but have to be considered for future work.

At this point, our 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 if 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 (see sec. 3.4 for the particular value), and arriving before this time causes the agent to wait until then before actually starting the activity. This means that arriving early to an activity does not gain an agent any activity performance utility.

2.7. Verification of Implementation

We have verified that the simulation structure as described above works as we intended by running it on a simple test scenario consisting of a circular network with 2,000 agents going back and forth between home and work. All agents have the same "home" location on one side of the circle and the same "work" location on the other side. Nine routes are available between home and work, and one route is available between work and home. We ran three setups with various combinations of decision-making modules enabled:

- *New plan, routes only*: The agents are only allowed to use the router module. They may do so with a 10 % probability.
- *New plan, times only*: The agents are only allowed to use the activity time allocation module. They may do so with a 10 % probability.
- *New plan, times and routes*: Agents may use the router module with 10 % probability, or both modules, with 10 % probability. This is just as described in the step 5 of Sec. 2.5 above.

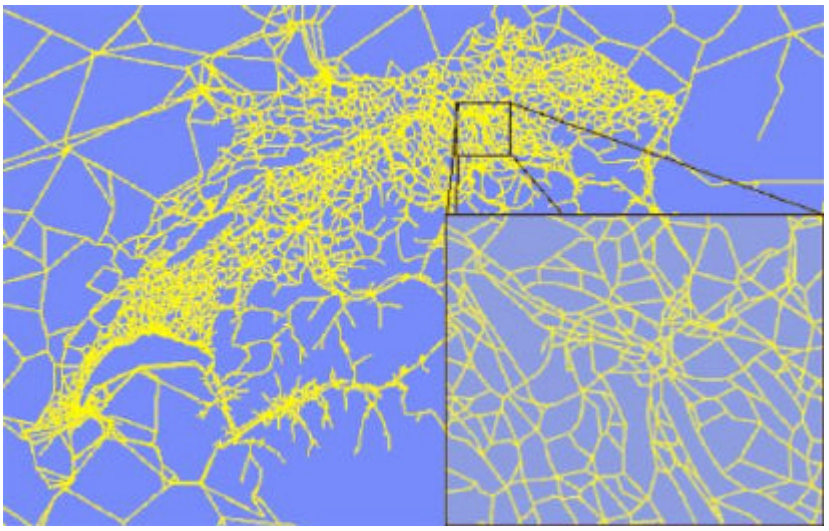
The results from these three scenarios were as expected. For more details on these verification tests, please see Raney and Nagel (2004).

3. INPUT DATA & SCENARIO

3.1. Network

The street network that is used was originally developed for the Swiss regional planning authority (Bundesamt fuer Raumentwicklung), and covered Switzerland. It was extended with the major European transit corridors for a railway-related study (Vrtic *et al.*, 1999). Some further modifications, in particular a capacity increase inside the Zurich city area, are described by Raney *et al.* (2003). The resulting network has the fairly typical size of 10,564 nodes and 28,624 links (Figure 1). Also fairly typical, the major attributes on these links are type, length, speed, and capacity.

Figure 1: Switzerland network



3.2. Zurich Area Scenario

Our starting point for the full Switzerland scenario demand generation is 24-hour origin-destination matrices from the Swiss regional planning authority (Bundesamt fuer Raumentwicklung). The original 24-hour matrix is converted into 24 one-hour matrices using a three step heuristic (Vrtic and Axhausen, 2002). The first step employs departure time probabilities by population size of origin zone, population size of destination zone and network distance. These are calculated using the 1994 Swiss National Travel Survey (BfS, 1996). The resulting 24 initial matrices are then corrected (calibrated) against available hourly counts using the OD-matrix estimation module of VISUM (www.ptv.de). Hourly traffic count data are available from the counting stations on the national motorway system (see Sec. 3.3). Finally, the hourly matrices are rescaled so that the totals over 24 hours match the original 24h matrix. VISUM assignment of the matrices shows that the patterns of congestion over time are realistic and consistent with the known patterns.

For the multi-agent simulation, these hourly matrices are then disaggregated into individual trips. That is, we generate individual trips 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 assume traffic analysis zones (TAZs) while in our simulations trips start on links. We convert traffic analysis zones to links by the following heuristic:

- The geographic location of the zone is found via the geographical coordinate of its centroid given by the data base.
- A circle with radius 3 km is drawn around the centroid.
- 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 leads 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 restrict our interests to the Zurich Area only. This is done with the following steps:

- All trips are routed using free flow travel times.
- We define the area of interest as a circle of 26 km radius around the center ("Bellevue") of Zurich City.
- 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 we get 260,275 agents that have an initial day plan.

3.3. Traffic Count Data

There are about 230 automatic counting stations registered with the Swiss Federal Roads Authority (Bundesamt fuer Strassen). Of those, we had hourly traffic count data for 75 stations. Out of those, we were able to locate 33 of them unequivocally on our network. Since we are just interested in the Zurich area, only a subset can be used. Unfortunately there are only 6

useful counting stations left. Since they are bi-directional, this means that we can compare 12 links to reality.

For the future it is planned to use the counting data of Kanton Zurich (www.laerm.zh.ch). There are about 300 "Taxomex" counting stations available for the whole area of the Kanton, although unfortunately none of these counting stations lie within the cities of Zurich and Winterthur.

3.4. Simulation Parameters

Here we describe some of the specific parameters used in the simulation setup for the results presented in the next section.

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 b used to convert plan scores to selection probabilities, is $2.0/Euro$. We use the following values for the marginal utilities of the utility function used for calculating scores:

$$b_{perf} = +6 Euro/h,\ b_{travel} = -6 Euro/h \text{ and } b_{late} = -18 Euro/h$$

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 $b_{perf} = +6 Euro/h$ that it could accumulate instead (opportunity cost). An agent that travels fore-goes the same amount, *plus* a loss of $6 Euro/h$ for traveling. And finally, an agent that arrives late receives a penalty of $18 Euro$ per hour late, but is not losing (or gaining) any time elsewhere by being late.

In addition, this paper will only look at daily activity chains that consist of one home and one work activity. The optimal times will be set to

$$t_h^* = 16\ hours \text{ and } t_w^* = 8\ hours .$$

With these assumptions, the maximum score is $120\ Euro$ ($120\ Euro$ per activity).

For the work activity a starting time window is defined between 7:08 AM and 8:52 AM. These values were set to correspond with those used in a similar study performed by Fabrice Marchal.

The blending factor α (see 2.5) is set to 0.1. This is a useful compromise between zero learning and overreaction. We expect that changes in α will mostly affect the speed of relaxation; this may be the topic of future research.

Note that in the verification tests mentioned in Sec. 2.7, the same simulation parameters were used as are described here, except that the desired time window for agents to arrive at work was set to a 0-width window at exactly 7 AM. As described in Raney and Nagel (2004), that was done to correspond more closely with the Vickrey scenario (Arnott *et al.*, 1993). For the present study, a longer time window seemed more plausible.

4. RESULTS

4.1. Overview

We present the results of four different setups, which result from two different initial conditions and from using time replanning 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 in Sec. 3.2. When the home activity ends, agents immediately depart and drive to work, where they stay for 8 hours, and then return. We will call the two setups where agents initially use externally defined times *times-routes-initial-times-extern* and *routes-only-initial-times-extern* when times replanning 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 in order to have a scenario where the simulation starts with a clearly implausible situation. The question that is tested is if it will recover to a realistic solution by itself. We will call the two setups where all agents depart at 6 AM *times-routes-initial-times-all6am* and *routes-only-initial-times-all6am* when times replanning is enabled and disabled, respectively.

Note that when times replanning is disabled, only 10 % of agents perform route replanning, but when it is enabled, a total of 20 % of agents perform route replanning, with half of those also performing times replanning.

We compare the results with the following indicators:

- *Average travel time*: The average travel time over all agents plans per iteration.

- *Average score*: The average score over all agents per iteration.
- *Departure and arrival time histograms*: The number of agents that arrive/depart from an activity over time during a certain iteration.
- *Traffic count data comparison*: Mean bias and error of the simulations compared to the counting data described above.

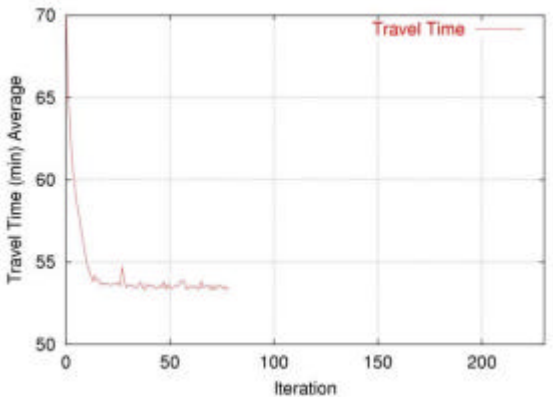
4.2. Initial plans with externally defined departure times

This setup tests whether or not the learning, once time replanning is switched on, drifts away from the time structure given by the external data. Since these initial plans are based on realistic time distributions (see above), one would assume that the time replanning will not affect the result that much. Re-routing alone should decrease the average travel time and congestion.

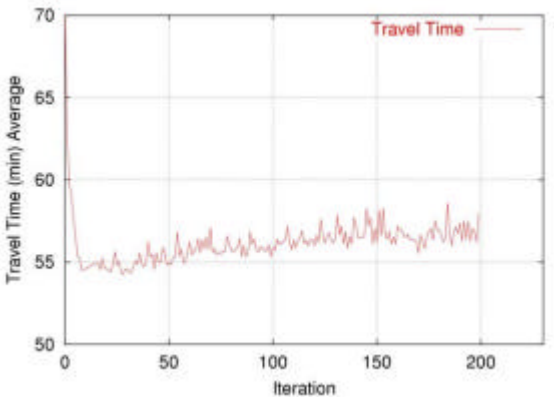
Figure 2 compares the average travel times over the iterations. The routes-only iteration (Figure 2(a)) quickly gets to a stable result because re-routing is the only part which has to be optimized. The small fluctuations are due to the fact that some percentage of the agents always replan, and that the traffic flow simulation is stochastic.

The iterations where time replanning is switched on (Figure 2(b)) behaves in a similar way, but the average travel time is slightly higher than routes-only and also it fluctuates more. However, the scores of the times-routes setup are not worse (see below) 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 parts which have to be optimized.

Figure 2: Average travel times of routes-only-initial-times-extern and times-routes-initial-times-extern



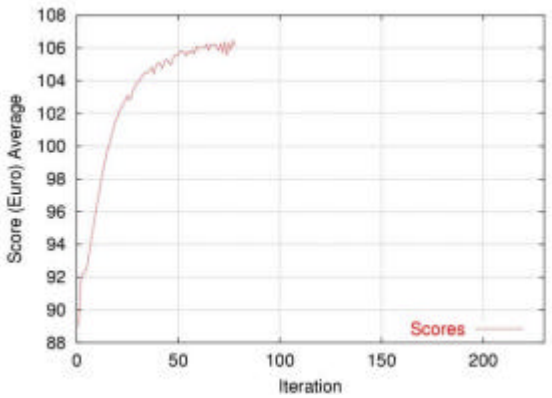
(a) average travel times per iteration for routes-only-initial-times-extern setup



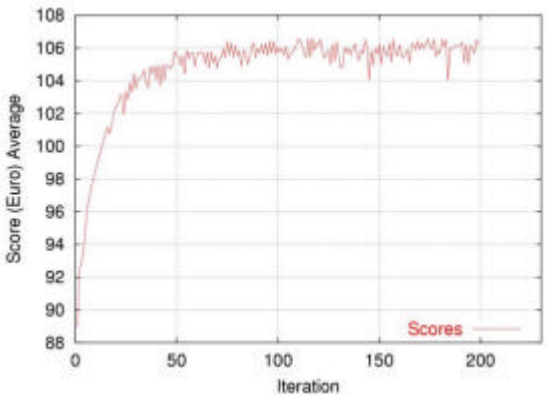
(b) average travel times per iteration for times-routes-initial-times-extern setup

Figure 3 shows the scores per iteration of both setups. They are once more similar to each other, and once more the routes-only setup (Figure 3(a)) has less fluctuation than the setup with time replanning (Figure 3(b)). The reason is the same as described above. Comparing to Figure 2, 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, described in Sec. 2.5.

Figure 3: Average scores of routes-only-initial-times-extern and times-routes-initial-times-extern



(a) average scores per iteration for routes-only-initial-times-extern setup

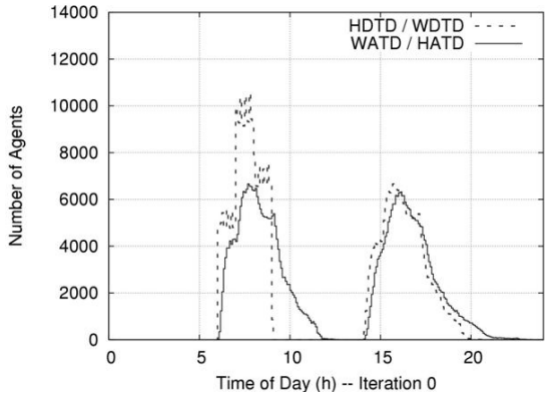


(b) average scores per iteration for times-routes-initial-times-extern setup

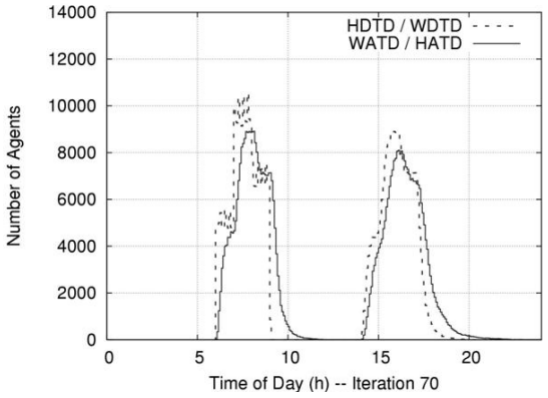
The histograms (Figure 4) show how the re-planning affects the agents. Starting with the same configuration (Figure 4(a)), the routes-only iteration only tries to minimize travel times, so that the periods of arrivals decreases (see bold graph of Figure 4(b)), while departure from home stays the same (see dotted graph of Figure 4(b)).

Switching on time replanning changes also the dotted graph (see Figure 4 (c)). The two peaks of the arrival (bold) graph are at 7:08 AM and 8:52 AM which is the border of the time window we defined for these scenarios (see Sec. 3.4). The reason for that is the fact that agents which are too late or too early at 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. Since an "optimal" plan for an agent is still to have short travel times, more and more agents try to arrive earlier in the defined time window. That is why the left peak is higher than the right one.

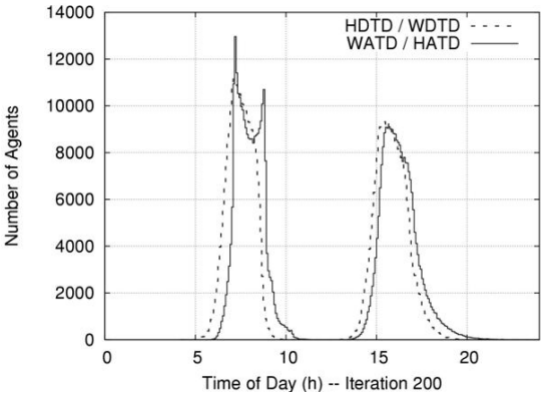
Figure 4: Arrival and departure histograms when the initial plans have "plausible" departure times.



(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 70 with time replanning switched off



(c) arrival and departure histograms (5 min time bins) of iteration 200 with time replanning switched on

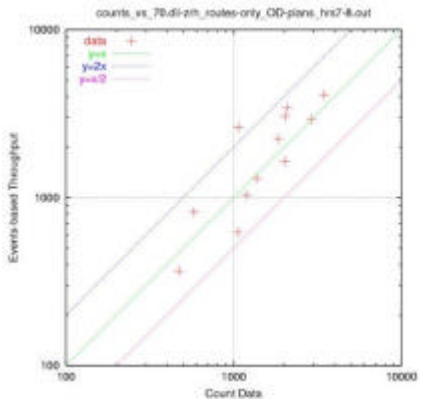
At last we look at the traffic count data. Figure 5 shows the relations of the two setups and the real data given by the already mentioned 12 links (see Sec. 3.3). As expected, the two results do not differ very much, and they are comparable to reality (see also Raney *et al.*, 2003). Also the quantitative measures of bias and errors are similar (see Table 1).

The mean absolute bias is $\langle q_{sim} - q_{field} \rangle$, the mean absolute error is $\langle |q_{sim} - q_{field}| \rangle$, the mean relative bias is $\langle (q_{sim} - q_{field})/q_{field} \rangle$, and the mean relative error is $\langle |q_{sim} - q_{field}|/q_{field} \rangle$, where $\langle . \rangle$ means that the values are averaged over all links where field results are available.

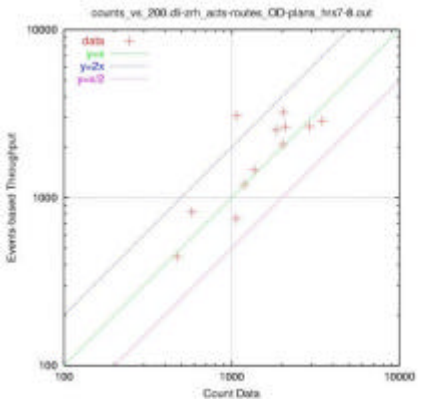
Table 1: Bias and Error of routes-only-initial-times-extern and times-routes-initial-times-extern compared to field data at 7-8 AM

	initial-times-extern	
Mean / Bias	routes-only 7-8 AM	times-routes 7-8 AM
Mean Abs. Bias:	+338.403	+306.320
Mean Rel. Bias:	+20.2 %	+25.3 %
Mean Abs. Error:	529.219	503.768
Mean Rel. Error:	37.0 %	35.4 %

Figure 5: Initial plans with externally defined departure times: comparison to traffic count data



(a) traffic count data vs. 70th iteration of routes-only-initial-times-extern setup at 7-8 AM



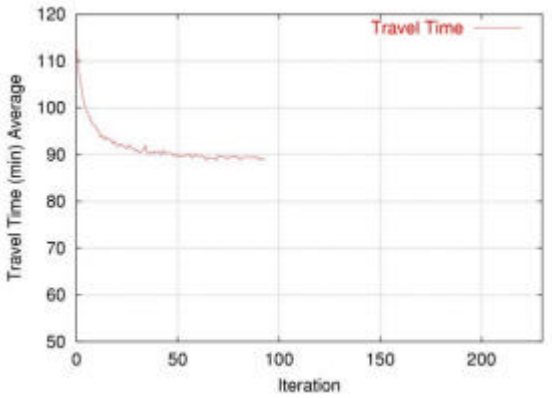
(b) traffic count data vs. 200th iteration of times-routes-initial-times-extern setup at 7-8 AM

4.3. Initial Plans with departure time at 6 AM for all agents

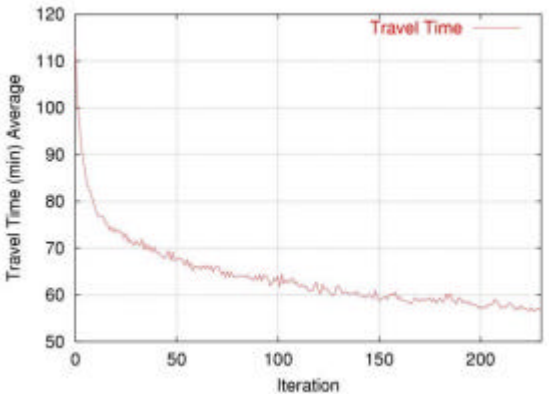
The previous section demonstrated that the results both with respect to the time structure and with respect to validation do not (at least) become worse when time replanning is switched on. However, the initial condition was still based on the externally given time structure. The experiments in this section will test in how far a realistic time structure can 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 6 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 than the one described in section 4.2. With time replanning switched on, average travel times decrease rather quickly, because agents are now allowed to change their departure time, too.

Figure 6: Average travel times of routes-only-initial-times-all6am and times-routes-initial-times-all6am



(a) average travel times per iteration for routes-only-initial-times-all6am setup

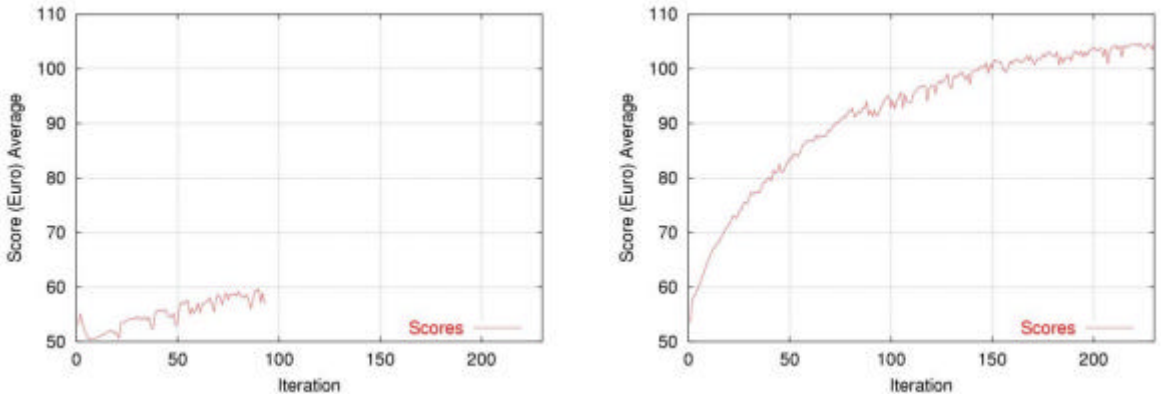


(b) average travel times per iteration for times-routes-initial-times-all6am setup

Also average scores without time replanning (Figure 7(a)) shows no improvement. Only optimizing travel times does not make sense anymore, because a major part of the agents will then arrive at work too early which does not increase scores (which can also be seen in Figure 8(b)).

When the time replanning module is also switched on, agents are now able to have short travel times and still arrive at work in the given time window. Figure 7(b) shows that the average score slowly increases to the same level compared to section 4.2.

Figure 7: Average scores of routes-only-initial-times-all6am and times-routes-initial-times-all6am

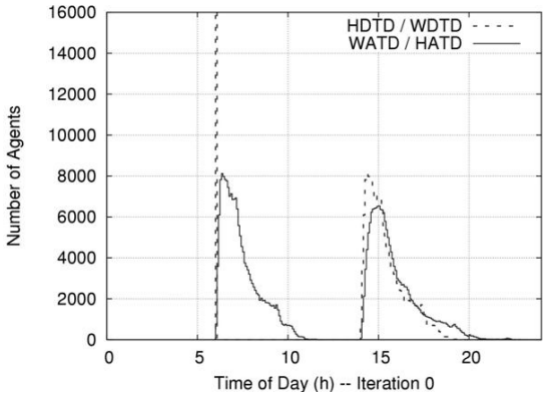


(a) average scores per iteration for routes-only-initial-times-all6am setup

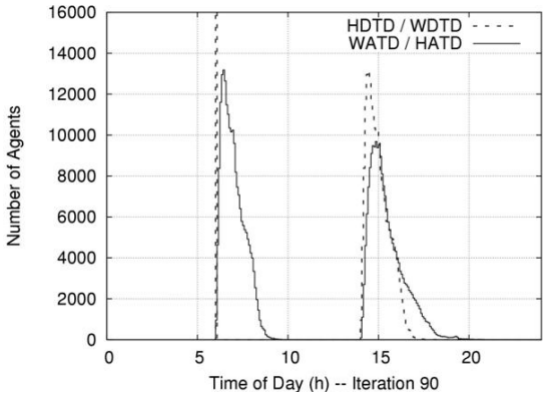
(b) average scores per iteration for times-routes-initial-times-all6am setup

The histograms (Figure 8) also show those facts. There are many more people who arrive between 6 and 7 AM in the routes-only setup (Figure 8(b)) than in the times-routes setup (Figure 8(c)). The peak of the departure time (dotted) graph of Figure 8(c) moved toward the same time as shown in Figure 4(c) of the previous section.

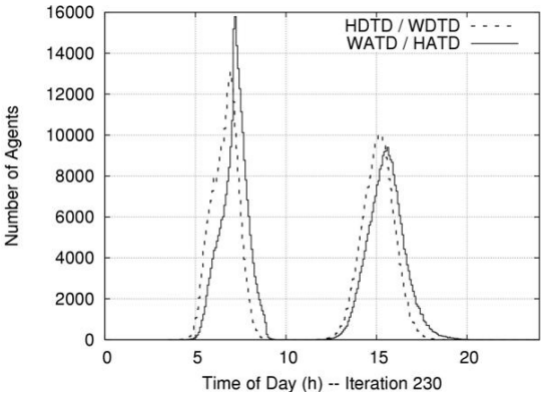
Figure 8: Arrival and departure histograms when in the initial plans everybody departs at 6 AM.



(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 90 with time replanning switched off



(c) arrival and departure histograms (5 min time bins) of iteration 230 with time replanning switched on

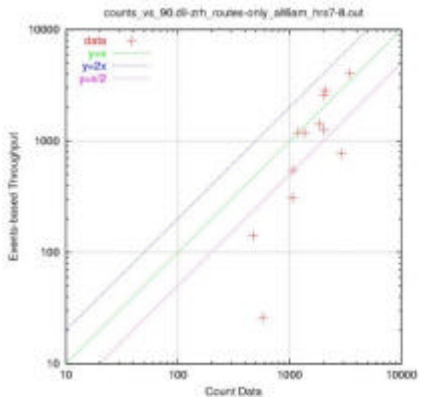
Comparing the results with real word data now shows a high discrepancy between the two setups. The routes-only setup cannot handle the fact that almost everybody starts too early. So it underestimates the throughput between 7 and 8 AM (Figure 9(a)). In the times-routes setup (Figure 9(b)), agents slowly move to more appropriate departure times which—at the end—will converge to similar result as shown in section 4.2.

Of course the calculation of the bias and the error (Table 2) produces now completely different result for the routes-only setup.

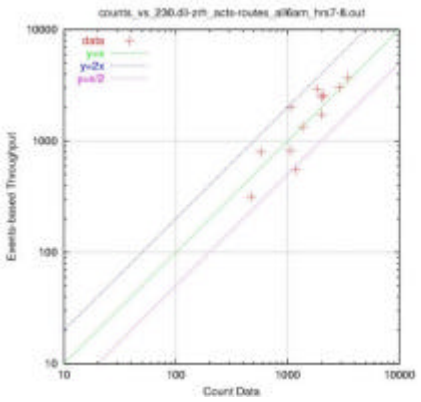
Table 2: Bias and Error of routes-only-initial-times-all6am and times-routes-initial-times-all6am compared to field data at 7-8 AM

	initial-times-all6am	
Mean / Bias	routes-only 7-8 AM	times-routes 7-8 AM
Mean Abs. Bias:	-313.430	+178.236
Mean Rel. Bias:	-29.3 %	+9.4 %
Mean Abs. Error:	631.395	404.886
Mean Rel. Error:	42.8 %	30.4 %

Figure 9: Departure time 6 AM plans: comparison to traffic count data



(a) traffic count data vs. 90th iteration of routes-only-initial-times-all6am setup at 7-8 AM



(b) traffic count data vs. 230th iteration of times-routes-initial-times-all6am setup at 7-8 AM

4.4. Conclusion

The important conclusions of the above study are:

- Departure time information from time-dependent OD matrices can be substituted by a static OD matrix and a time allocation module. It should be noted though, that the approach involves large amount of additional computations.
- This statement, which has already been demonstrated with trip-based models by De Palma and Marchal (2002), remains correct when moving to a fully agent-based approach which uses activity chains instead of individual trips.
- The concept of "plug-in" modules (see also Raney and Nagel, 2004), here applied to the two modules "route (re)planning" and "activity time (re)planning", worked without problems in a real world scenario. This is a small but important step for further development, because it makes it possible to add other modules, such as activity location choice or activity pattern choice, in a simple way.

5. COMPUTATIONAL ISSUES

5.1. Traffic Flow Simulation

Computational issues for the traffic flow simulation are discussed in detail elsewhere (Cetin and Nagel, 2003). The main result from those investigations is that, using a Pentium cluster with 64 CPUs and Myrinet communication, the queue simulation can run more than 500 times faster than real time (excluding input and output), meaning that 24 hours of traffic of all of Switzerland can be simulated in less than 3 minutes. For the results presented in this paper, two setups were run simultaneously with 12 CPU's per setup using Ethernet communication. The average computation time is ca. 30 minutes in order to simulate about 18 hours of traffic.

5.2. Performance

One iteration takes in the average about 102 minutes. The average duration of sub-steps of an iteration is listed below:

- about 18 sec for the departure time allocation module
- about 23 min for the router module, of which about 19 min is spent reading the events
- about 39 min for the traffic flow simulation module (including file input and output)
- about 11 min for sorting events
- about 16 min for reading events and processing them into scores
- about 105 sec for writing the new plans
- the remaining time (about 12 min) is used for other I/O processes/bottlenecks, communication and data preparations

These times allow the calculation of 15 to 20 iterations per day.

5.3. Disk Usage

A complete data set generated by one iteration produces about 280 MB of data (when compressed). These will be kept for the first and the last 5 iterations and also for every 10th iteration. For each of the other iterations only about 40 MB are kept.

5.4. Memory Usage

Since we are simulating about 260,000 Agents which keep at most 5 different plans and each of them needs about 700 Bytes of memory plus some overhead, we end up with a requirement of about 1 GB of memory. The router module also needs about 200 MB of memory, which is feasible. Higher resolution networks will need more memory which might become a problem in the future.

6. FUTURE WORK

At present we only model the "primary" activities of "home" and "work". We are working on adding "secondary" activities, such as shopping and leisure to the system. This requires the addition of two more modules: the activity pattern generator and the activity location generator, which were mentioned in Sec. 2. Another module we are interested in adding is a *Population generation* module, which would disaggregate demographic data to obtain individual households and individual household members, with certain characteristics, such as a street address, car ownership, or household income (Beckman *et al.*, 1996; Frick, 2004). The population would not match reality, but would result in the same statistics. These modules should also be implemented as "plug-ins". We are also investigating other travel modes such as public transport or pedestrian mode.

Another issue of interest is the possibility that agents could also learn during the day. They could re-route while they are stuck in congestion, drop an activity because they are already too late, and so on. This "within day replanning" (e.g. Axhausen, 1990; Cascetta and Cantarella, 1991) should help to improve their strategies faster than only "day-by-day replanning", and the interaction with other entities (like traffic lights, changing traffic signs and other ITS entities) can be added to the traffic flow simulation. Within-day learning is more realistic since some types of decisions are made on time scales much shorter than a day (Doherty and Axhausen, 1998). However, within-day replanning is at adds with the parallel computing approach to the traffic flow simulation (Nagel and Marchal, 2003), which is the reason why it is currently not used in our project.

Simulation speedup can also be improved by elimination of performance bottlenecks. At the moment the agent database keeps track of *all* agents of the simulation. Since recalculating an agents' strategies is completely independent to other agents, it is useful to introduce parallelism into this. These "multiple agent databases" should then be controlled by a separate module which keeps track of the feedback. This leads us to a clear separation of "agent databases" and "feedback".

The activity time allocation module itself can be improved, too. It should recognize when agents are too early or too late, so the adaptation to a more realistic departure time should be done with fewer iterations.

High resolution networks are another issue. Especially if there is more precise information available about locations. The main goal will be that each agent has its home location at a street with a house number, probably a ramp to its garage, a private pedestrian path to the next tram station, and so on. In high resolution networks, intersection bottlenecks can be shown also at

specific points in a big city. Last but not least, high resolution scenarios are indeed a computational challenge.

With high resolution networks, more precise traffic count data is required to compare with. There is some effort to extract information of the raw data of the Kanton Zurich which gives more precise information about local traffic situations.

7. SUMMARY

The investigation described in here is part of a series of investigations to demonstrate the feasibility of multi-agent traffic simulations for transportation planning. This work should demonstrate the convenience of this technology on large scale research question for transportation planning.

This paper presents intermediate results of this work. The intent was to test if the technology is able to generate plausible morning departure times by itself, rather than having to rely on external information for this. The results show that this is possible, and that the results are no less realistic than those obtained with an externally given departure time distribution. It is important to note that these results are achieved with an implementation which is fully based on 24-hour daily activity plans and completely agent-based. This is in contrast to previous implementations of similar scenarios.

The computational structure of the computational framework is such that fairly arbitrary modules can be plugged in and used. The overall structure of this framework shows that it is able to handle a certain amount of plans for each agent. Simple rules can be defined how a plan can be chosen (or completely new generated) for the next simulation step.

The simple but very flexible approach together with the promising results created by using the router and the activity time allocator makes it worthwhile to develop further on this project.

ACKNOWLEDGMENTS

The ETH-sponsored 192-CPU Beowulf cluster “Xibalba” performed most of the computations. Marc Schmitt is doing a great job maintaining the computational system. Nurhan Cetin provided the traffic flow simulation. The work also benefited from discussions with Fabrice

Marchal. Michael Balmer and Bryan Raney are funded in part by the ETH project "Large scale multi-agent simulation of travel behavior and traffic flow"; Bryan Raney is also funded in part by the ETH project "A unified approach for agent-based learning with application in architecture and in transportation planning".

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.

Avineri, E. and J. N. Prashker (2003). Sensitivity to uncertainty: Need for paradigm shift. *Transportation Research Board Annual Meeting*, Washington, D.C., Paper 03-3744.

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

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

BfS—Bundesamt fuer Statistik und Dienst fuer Gesamtverkehrsfragen (1996). Verkehrsverhalten in der Schweiz 1994. *Mikrozensus Verkehr 1994*, Bern. See also http://www.statistik.admin.ch/news/archiv96/dp96036.htm.

Bottom, J. A. (2000). Consistent anticipatory route guidance, PhD thesis. *Massachusetts Institute of Technology*, Cambridge, MA.

Cascetta, E. and C. Cantarella (1991). A day-to-day and within day dynamic stochastic assignment model. *Transportation Research A*, **25A**(5), 277–291.

Cetin, N. and K. Nagel (2003). A large-scale agent-based traffic microsimulation based on queue model. *Proceedings of Swiss Transport Research Conference (STRC)*. Monte Verita, Switzerland. See also http://www.strc.ch.

Charypar, D. and K. Nagel (2003). Generating complete all-day activity plans with genetic algorithms. *Proceedings of the meeting of the International Association for Travel Behavior Research (IATBR)*, Lucerne, Switzerland, 2003. See also http://www.ivt.baum.ethz.ch/-allgemein/iatbr2003.html.

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

Doherty, S. T. and K. W. Axhausen (1998). The development of a unified modeling framework for the household activity-travel scheduling process. In: *Verkehr und Mobilitaet*, number 66 in Stadt Region Land. Institut fuer Stadtbauwesen, Technical University, Aachen, Germany.

Ettema, D., G. Tamminga, H. Timmermans and T. Arentze (2003). A micro-simulation model system of departure time and route choice under travel time uncertainty. *Proceedings of the meeting of the International Association for Travel Behavior Research (IATBR)*, Lucerne, Switzerland. See also http://www.ivt.baum.ethz.ch/allgemein/iatbr2003.html.

Frick, M. A. (2004). Generating Synthetic Populations using IPF and Monte Carlo Techniques: Some New Results. *Proceedings of Swiss Transport Research Conference (STRC)*, Monte Verita, Switzerland. See also http://www.strc.ch.

Gawron, C. (1998). An iterative algorithm to determine the dynamic user equilibrium in a traffic simulation model. *International Journal of Modern Physics C*, **9**(3), 393-407.

Jacob, R. R., M. V. Marathe and K. Nagel (1999). A computational study of routing algorithms for realistic transportation networks. *ACM Journal of Experimental Algorithms*, **4**(1999es, Article No. 6).

Kaufman, D. E., K. E. Wunderlich and R. L. Smith (1991). An iterative routing/assignment method for anticipatory real-time route guidance. *IVHS, Technical Report 91-02*, University of Michigan Department of Industrial and Operations Engineering, Ann Arbor, MI 48109.

Nagel, K. (1995). High-speed microsimulations of traffic flow, PhD thesis, *University of Cologne*. See also http://www.inf.ethz.ch/~nagel/papers or www.zaik.uni-koeln.de/~paper.

Nagel, K. and F. Marchal (2003). Computational methods for multi-agent simulations of travel behavior. *Proceedings of the meeting of the International Association for Travel Behavior Research (IATBR)*, Lucerne, Switzerland. See http://www.ivt.baum.ethz.ch/allgemein/-iatbr2003.html.

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

Raney, B., N. Cetin, A. Voellmy, M. Vrtic, K. Axhausen and K. Nagel (2003). An agent-based microsimulation model of Swiss travel: First results. *Networks and Spatial Economics*, **3**(1), 23-41.

Raney, B. and K. Nagel (2003). Truly agent-based strategy selection for transportation simulations. *Transportation Research Board Annual Meeting*, Paper 03-4258, Washington, D.C.

Raney, B. and K. Nagel (2004). An improved framework for large-scale multi-agent simulations of travel behavior. *Proceedings of Swiss Transport Research Conference (STRC)*, Monte Verita, Switzerland. See also http://www.strc.ch.

Vrtic, M., R. Koblo, and M. Voedisch (1999). Entwicklung bimodales Personenverkehrsmodell als Grundlage fuer Bahn2000, 2. Etappe, Auftrag 1. Report to the *Swiss National Railway* and to the *Dienst fuer Gesamtverkehrsfragen*, Prognos AG, Basel, Switzerland. See also http://www.ivt.baug.ethz.ch/vrp/ab115.pdf for a related report.

Vrtic, M. and K. W. Axhausen (2002). Experiment mit einem dynamischen Umlegungs-verfahren. *Strassenverkehrstechnik*. Arbeitsberichte Verkehrs- und Raumplanung, No. 138. See also http://www.ivt.baug.ethz.ch.

Weiss, G. (1999). Multiagent Systems. A modern approach to distributed artificial intelligence. *The MIT Press*, Cambridge, MA.