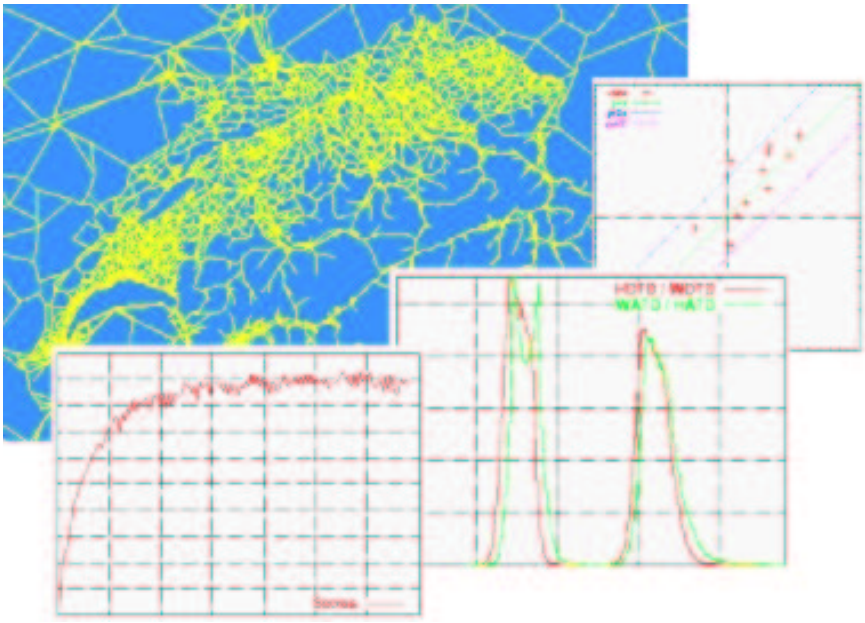


Agent-Based Activities Planning for an Iterative Traffic Simulation of Switzerland – Activity Time Allocation

Michael Balmer, Dept. of Computer Science, ETH Zürich
Bryan Raney, Dept. of Computer Science, ETH Zürich
Kai Nagel, Dept. of Computer Science, ETH Zürich

Conference paper STRC 2004

STRC

4th Swiss Transport Research Conference
Monte Verità / Ascona, March 25-26, 2004

Agent-Based Activities Planning for an Iterative Traffic Simulation of Switzerland – Activity Time Allocation

Michael Balmer
Department of Computer Science
ETH Zürich
CH-8092 Zürich, Switzerland

Phone: 01-632 03 64
Fax: 01-632 13 74
eMail: balmermi@inf.ethz.ch

Bryan Raney
Department of Computer Science
ETH Zürich
CH-8092 Zürich, Switzerland

Phone: 01-632 08 92
Fax: 01-632 13 74
eMail: raney@inf.ethz.ch

Kai Nagel
Department of Computer Science
ETH Zürich
CH-8092 Zürich, Switzerland

Phone: 01-632 54 27
Fax: 01-632 13 74
eMail: nagel@inf.ethz.ch

Abstract

In a multi-agent transportation simulation, travelers are represented as individual "agents" who make independent decisions about their actions. We are implementing a large-scale version of such a simulation in order to model the traveler behavior in all of Switzerland.
Our simulation is composed of separate modules. Each module models a different set of decisions made by an agent and calculates those decisions independently for each agent. The modules we use are listed below.
The "activities generator" module generates a complete 24-hour day-plan for a given agent, including each major activity (sleep, eat, work, shop, drink beer) the agent wishes to accomplish during the day, their times, and their locations. Activity times may be flexible, and can be specified by some combination of starting time, duration, and/or ending time.

The "route planner" module determines the mode of transportation, as well as the actual route plan taken, for each leg of the agent's chosen activity plan. Each leg simply connects one activity to another. At present our router generates routes only for automobile travel.
The "micro-simulation" module executes all the agents' plans simultaneously and in consequence computes the interaction between different travelers leading, e.g. to congestion. The micro-simulation provides data to the other modules about what happened during the simulated day, including the travel-time as well as the time and location of significant events to happen to agents while they are being simulated (such as arriving at their first activity destination, etc.).
There is an interdependence between the above modules. For example, plans depend on congestion (via travel-times) but congestion is a result of the execution of plans. The "feedback and learning" module resolves the interdependence via an iterative method, where an initial plans set is slowly adapted until it is consistent with the resulting travel conditions, or until some other stopping criterion is fulfilled. In this technique, each iteration of the simulation can been seen as a separate "day" in the life of the agents. Each day the agents' view of the traffic network is updated, and they learn about which plans work well and which do not.
We implement the feedback mechanism with a so-called "agent database" that allows agents to have a "memory" of the plans they have used during previous days in the current iteration sequence. The agents also use the events data from the micro-simulation to determine performance "scores" for each plan. Each day of the iteration, every agent chooses a plan from its memory based on their scores, or decides to generate a brand-new plan (either a new set of route plans for a given activity-set or a new activity-set and associated route plans) based on information from the previous iteration.
This paper goes beyond what we have done in the past by investigating for the first time the effects of time choice inside the feedback loop. That is, travel time results are fed back to a time allocation module, which "mutates" existing activity schedules, allowing the learning mechanism of the agents to choose the plans that work best. We present preliminary results from using this module with the rest of the framework applied to the Switzerland morning rush-hour scenario – with special interests into the Zürich area, which contains approximately 260'000 agents.
See also Raney and Nagel (2004).

Keywords

traffic simulation– transportation planning– route planning– learning– multi-agent simulation– 4th Swiss Transport Research Conference – STRC 04 – Monte Verità

1. Introduction

There are different accepted computational models to forecast traffic. The one which is described in this paper is “multi-agent simulation”. The main characteristic of this model leads to the fact that each entity is described as an individual object or agent, which makes independent decisions about its actions. Typical entities are travelers, traffic light, ITS-entities like variable message signs, and so on...
This multi-agent concept consists of basicially two parts: (i) the simulation of the “physical” properties of the system, and (ii) the generation of the agents’ strategies. The simulation 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 strategies of agents are made by the experiences of the 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 troughput of an intersection, etc.
We are implementing such a multi-agent simulation for the whole Switzerland – but specifically concentrating on the Zürich area. About 260’000 agents are simulated who 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. This paper reports first results which are essentially based on typical transportation planning data: it uses standard origin-destination matrices; it uses the transportation planning network from the corresponding Swiss federal planning authority; and it performs route assignment based on these input data. The main difference to typical planning tools, such as EMME/2 (http://www.inro.ca) or VISUM (http://www.ptv.de), is that our implementation is indeed completely agent-based; that is, the individuality of each traveler is fully maintained throughout the process. Furthermore, compared to last years results (Raney *et al.*, 2003), we show a simple but quite effective way to calculate the time window when trips occur during the day. With this it is no longer nessesary to process time dependent O-D-Matrices. In other words, it is possible for the simulation itself to predict when agents executes their trips.

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 (Sec. 2.1). Next we introduce the network and the scenario (Sec. 3). Some computational issues are pointed out in section 5. The results (Sec. 6) of the different setups of the scenario are compared with counts data (Sec. 4) of this region. The paper is concluded by a conclusion (Sec. 7), future work (Sec. 8) and a summary (Sec. 9).

2. Simulation Structure

To constuct a multi-agent simulation we need to split up different aspects into modules which try to fulfill those aspects. The goal is to create a complete plan in a certain time period – usually a typical workday. I.e. this **dayplan** could look like this:

- staying at home till 07:38
- take the car to drive to work by using the following route (encoded in a street network): 321 - 2 - 34 - 65 - 21 - 34
- arriving at work at 07:58
- working for 4 hours till 12:00
- taking the bus nr 34 to go to lunch using the following stations (encoded in a public transport network): station#4 - station#16 - station#14
- arriving at station#14 at 12:14
- walking for 7 minutes to the restaurant using the following route (encoded in a pedestrian network): 101 - 103 - 23 - 54
- arriving ... and so on...

To generate such a plan, we need to cut it into "elemental pieces", so that we can create each with a seperate module. The peaces are defined as the following:

- **Population generation**. Demographic data is disaggregated so that one obtains individual households and individual household members, with certain characteristics, such as a street address, car ownership, or household income (Beckman *et al.*, 1996). – This module is not used for our current investigations but will be used in the future.
- **Activities:** They describe what an agent actually wants to do during the day and where these activities should be processed. It is also useful to define "staying at home" and "being at work" as the **two primary activities**, since typically a citizen first decides where to live and where to work before he plans other activities during his day like "having lunch", "shopping", "leisure", and so on. Please notice, that at the moment the activities and their locations will be treated as the same.
- **Pattern:** It describes in which order the choosen Activities should be executed. Typically, the pattern starts and ends at home.
- **Arrival, Duration, Departure Time:** Activities have two or all three attributes defined. Activities cannot overlap in time.
- **Trips:** Trips connect the activitiy chain. They define which route and which mode an agent chooses to get from one activity to the next.

For each of the items above described, we are now able to implement a seperate module, which produces the necessary inforamtion.

First it is to define where the agent lives and where it works. The **primary location**[1] **choice module** generates these information using i.e. microcensus or other statistical data from the area simulated. There are two ways to aggregate home and work location. One defines first the home location and computes a suitable work location (comparable to blue workers). The other does it vice versa (comparable to white workers).
The **secondary location choice module** includes other activities. This can be done with various statistical approaches.
A **pattern generation module** now uses this information to produce a daily activity pattern for an agent (Vaughn *et al.*, 1997; Bowman, 1998). It typically uses surveys to produce "realistic" patterns.
Now we know what an agent wants to do during the day, but we also need to define when such an activity should start, how long it should take and when it should be finished. **Time allocation module** takes care of this. Some of the times are more variable (i.e. start/end time and duration of leisure), some have fixed times (i.e. shopping activity can't be done when shops are closed) and some of them should have a minimum or maximum duration (i.e. and agent should work about 8 hours).
We have now a day plan of the agent. But there are still two things missing. How does the agent get from one activity to another and what kind of transporation system it will use? A **modal choice module** desides the travel mode. It is dependent on the household of the agent. If an agent doesn't have a car, it needs to take train, bus, tram, and so on. It also can go by foot. A bike or a car taken in the morning should be at home again when the day ends.
Then the **router module** finds out which route the agent should take to go from one activity to the next.
The day plan is now constructed and can be feed into the **mobility simulation module**. It simulates all agents simultaniously on the network. Because of the interaction of the agents, congestion can occur. So, some day plans cannot be fulfilled as they were planned. Those plans have to be recalculated by the modules again and the process starts over. This is the **feedback** part of the multi-agent simulation. This widely accepted method is systematic relaxation (Kaufman *et al.*, 1991; Nagel, 1994/95; Bottom, 2000) – that is, make preliminary plans, run the mobility simulation, adapt the plans, run the simulation again, etc., until consistency between modules is reached.

In this paper we use the the following subset of the modules described above:

- **activity end time allocation module:** It changes the end time of an activity. The duration is fixed to a certain period (i.e. work duration is 8 hours).
- **router module:** It produces the fastest path from one activity to the next with the information about the link travel time output of the previous mobility simulation.
- **mobility simulation:** It simulates the "real world" traffic of the agents over a day.
- **agent database and feedback:** It keeps track of the day plans of each agent and is responsible for the iteration process (feedback).

[1] in this approach, location and activity are treated as being equal. other models seperate these two issues

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

2.1 Modules

2.1.1 Activity Time Allocation Module

If an existing plan of an agent needs to be changed, this module will be called. In this approach, we choose randomly 10% of all agents to replan one of their plans in the databese. At the moment, the module just randomly shifts the end time of the activities of the agents a little bit around. Of course, it is not really sophisticated, but it demonstrates the concept of the time allocation module. But since each module is implemented as a "plugin", this module can be replaced by a more enhanced implemetation.

2.1.2 Router

The router will be used in a similar way as the time allocation module. It is implemented as a *time dependent Dijkstra algorithm*. As an input it uses the information of the link travel times of the previous mobility simulation. The link travel times are encoded in 15 minute time bins, so they can be used as the weights of the links in the network graph. Apart from relatively small and 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 one activity to the next as a function in time. This path will be used by the agents for the next mobility simulation.

2.1.3 Mobility Simulation

The mobility simulation simulates the physical world. It is implemented as a queue simulation, which means that each street (link) is represented as a FIFO (first-in first-out) queue with two 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.
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 mobility simulation produces are events for each agent, such as entering/leaving link, left/arrived at activity, and so on. With this it's easy to produce differnet 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.

2.1.4 Agent Database – Feedback

The Feedback is the most important part of this framework. Without it, agents cannot learn. This leads from the fact that the modules produce dependencies about their computed information. For example, congestion (computed by the mobility simulation) depends on plans (which were created by the allocation modules and the router). On the other hand, plans depend on congestion. The iteration process runs the mobility 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 mobility simulation, etc, until consistency between modules is reached.
But how do we know when a plan of an agent is a good plan?
To measure how good a plan for a certain agent is, we first feed it into the mobility simulation. Because of the interaction between the agents, the defined plan normally cannot be executed as it was planned. Whenever an agent arrives too late at a certain activity the plan gets a punished by a (negative) score. It will also be punished for the duration of travel time. For executing an activity the plan gets a reward (positive score). The calculation of the **scores** (one score point has the value of one Euro) is described below:

- an agent will lose $18 Euro/h$ for being late.
- an agent will lose $6 Euro/h$ for traveling.
- The optimum duration of work is 8 hours (t_{opt_w}), the one for being at home is 16 hours (t_{opt_h}). So, a plan will get full score if an agent spends all of its time executing these activities for the optimum duration trough the whole day, meaning it won't travel nor be late or early.

The caluation of the activity duration is intoduced by Charypar and Nagel (2003). The score is caluatated by the following equation:

$$U_{dur}(t_{dur}) = \beta_{dur} \cdot t_{opt} \cdot \ln(\frac{t_{dur}}{t_0})$$

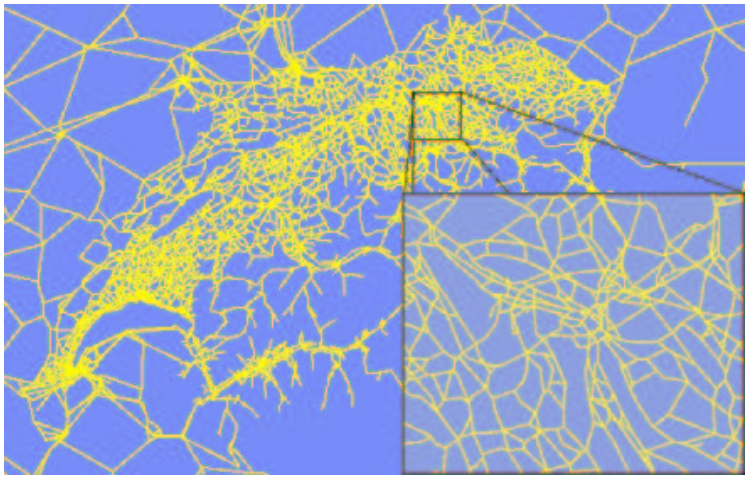
$$t_0 = t_{opt} \cdot e^{-10\ hours/(t_{opt})}$$

where t_{dur} is the actual performed duration of the activity and β_{dur} is the weight which is set to $6 Euro/h$. With this calculation, the maximum score is $120 Euro$ ($60 Euro$ per activity).
The total score of a plan will therefore be calculated as:

$$U_{tot} = \sum_{i=1}^{n} U_{dur_i} - \sum_{i=1}^{m} U_{late_i} - \sum_{i=1}^{k} U_{travel_i}$$

After each mobility simulation all executed plans are scored. Then an agent can decide if it wants to keep this plan for future use or if it removes this plan. An agent can keep a certain number of plans to reuse those again. In this approach, each agent is allowed to keep the 5 best plans (which have the highest scores).
For the next iteration, some of the agents will then pick their best plan of their own database,

Figure 1: Switzerland network



some just randomly pick one out, some try the same plan again and the rest generates a new plan by using the departure time allocation module and the router module (for more details 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 für Raumentwicklung), and covered Switzerland. It was extended with the major European transit corridors for a railway-related study (Vrtic *et al.*, 1999). The network supposedly contains the status for 1999, but contains at least one major error (a high capacity tunnel in Zürich is missing). Our initial simulations resulted in traffic gridlock in Zürich, which was also reflected in the VISUM assignment displaying V/C ratios significantly above 100%. A manual comparison with a higher resolution network of Zürich led to the conclusion that capacity in Zürich was in general significantly underestimated; in consequence, we manually increased the corresponding road capacity for transit corridors through Zürich in our network.
After our modifications, the network has the fairly typical size of 10564 nodes and 28624 links (Fig. 1). Also fairly typical, the major attributes on these links are type, length, speed, and capacity.

3.2 Zürich Area Scenario

Our starting point for demand generation for the full Switzerland scenario are 24-hour origin-destination matrices from the Swiss regional planning authority (Bundesamt für 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 (Bundesamt für Statistik und Dienst für Gesamtverkehrsfragen, 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 counts are available from the counting stations on the national motorway system. 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 6:00 AM and 9:00 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, one million agents would not fit into memory anymore. So we restrict our interests to the Zürich Area only. This is done with the following steps:

- We define the area of interest as a circle of 26 km radius with the center placed on the “Bellevue” in Zürich City
- Each agent which does not cross this area in its daily plan is removed
- The remaining 260275 agents with the daily pattern “home-work” are then extended to the “home-work-home” pattern, where the two homes are – of course – at the same location.
- The duration of the “work” activity are set to 8 hours

- The paths of the trips "home to work" and "work to home" are created by the router module based on the free speed link travel times of the network
- For the work activity a starting time window is defined between 7:08am and 8:52am [2].

At the end we get 260275 agents which has an initial day plan, looking like this:

```
<person id="6357267">
  <plan>
    <act type="h" x100="626820" y100="171700" link="13151"
         end_time="06:43:48" />
    <leg mode="car" dep_time="06:43:48">
      <route>
        4366 4341 4343 4344 4311 4310 4309 4308 4322 4373 4372
        4371 4370 4378 4377 4368 3830 3829 3828 3827 3826 3814
        3813 4207 4208 4044 4045 4046 4047 710 709 619 620 621
        797 798 799 622 623 800 499 500 501 502 503 1377 1415
        943 942 915 941 940 938 925 926 927 928 929 930 1405
        1404 1403 1402 1401 1416 1388
      </route>
    </leg>
    <act type="w" x100="659350" y100="272740" link="4003"
         dur="08:00" />
    <leg mode="car" dep_time="14:43:48">
      <route>
        1416 1401 1402 1403 1404 1405 930 929 928 927 926 925
        938 940 941 915 942 943 1415 1377 503 502 501 500 499
        800 623 622 799 798 797 621 620 619 709 710 4047 4046
        4045 4044 4208 4207 3813 3814 3826 3827 3828 3829 3830
        4368 4377 4378 4370 4371 4372 4373 4322 4308 4309 4310
        4311 4344 4343 4341
      </route>
    </leg>
    <act type="h" x100="626820" y100="171700" link="13151" />
  </plan>
</person>
```

[2]The time window is calculated by assuming a normal distribution of the desired arrival time at work. The average arrival time (μ) is 8am and the standard derivation (σ) is 0.5 hours. So the starttime/endtime of the time window is calculated by $t_{start,end} = \mu \pm \frac{2 \cdot \sqrt{3} \cdot \sigma}{2}$.

4. Counts Data

Hourly counts data are available from the counting stations on the national motorways. There are about 230 counting stations registered at the Swiss Federal Roads Authority (Bundesamt für strassen). To use those for comparison purposes, the counting stations have to be matched to the right links in the network.
We are using at the moment 33 stations (66 directions/links) since those had matched well on the Switzerland network. Since we are just interested in the Züerich area, only a subset can be used. Unfortunately there are only 6 useful counting stations left, means we can compare only 12 links to reality.

For the future it is planned to use the counting data of Kanton Zürich (http://www.laerm.zh.ch). There are about 300 "Taxomex" counting station registered in the whole area of the kanton ecxept the cities of Winterthur and Zürich unfortunately. It will be nesessary to use them, ecpecially when a higher resolution network will be used.

5. Computational Issues

5.1 Mobility Simulation

Computational issues for the mobility 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 simultaniously with 12 CPU's per setup using ETHERNET communication. The average computation time is ca. 30 minutes, simulating about 18 hours.

5.2 Performance

One iteration takes in the average about 100 minutes. The average duration of substeps of an iteration is listed below:

- about 10 sec for the departure time allocation module
- about 22 min for the router module
- about 30 min for the mobility simulation module
- about 22 min for handling events

- about 100 sec for writing the new plans
- the remaining time (about 24 min) are used for other I/O processes/bottlenecks, communication and data preparations

With this, about 15 to 20 iterations are done per day.

5.3 Disk Usage

A complete data set generated by one iteration produces about 280 MB of data. 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 keeps at most 5 different plans and each of them needs about 700 Bytes of memory plus some overhead, we end up by about 1 GB of Memory. This is one of the main reason we will implement the agent database also in parallel.
The router module also need about 200 MB of Memory which is ok at the moment but could get to a problem for a high resolution network.

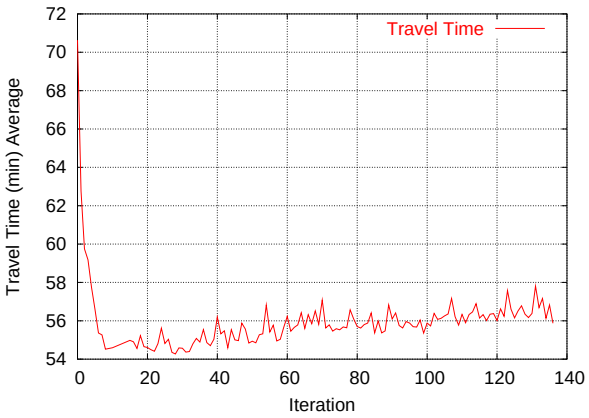
6. Results

We present the results of four different setups of this scenario:

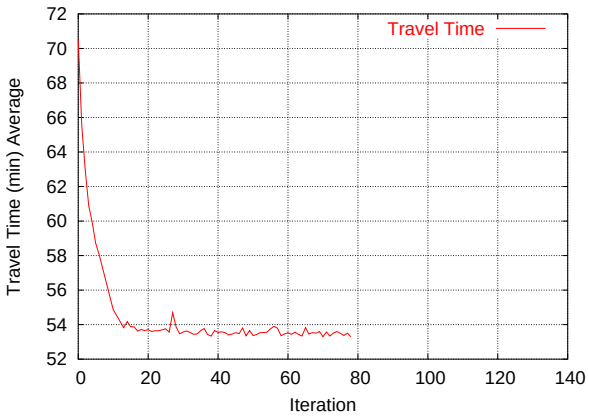
- All agents start their first trip (home-work) according to the OD Matrix described above using route-replanning and activity-time-allocation to improve their plans. We will call this setup *acts-routes-OD-plans*.
- All agents start their first trip (home-work) according to the OD Matrix described above using only route-replanning to improve their plans. We will call this setup *routes-only-OD-plans*.
- All agents start their first trip (home-work) at 6am using route-replanning and activity-time-allocation to improve their plans. We will call this setup *acts-routes-all6am*.
- All agents start their first trip (home-work) at 6am using only route-replanning to improve their plans. We will call this setup *routes-only-all6am*.

We compare the results with the following indicators:

Figure 2: average travel times of acts-routes-OD-plans and routes-only-OD-plans



(a) average travel times per iteration for acts-routes-OD-plans setup



(b) average travel times per iteration for routes-only-OD-plans setup

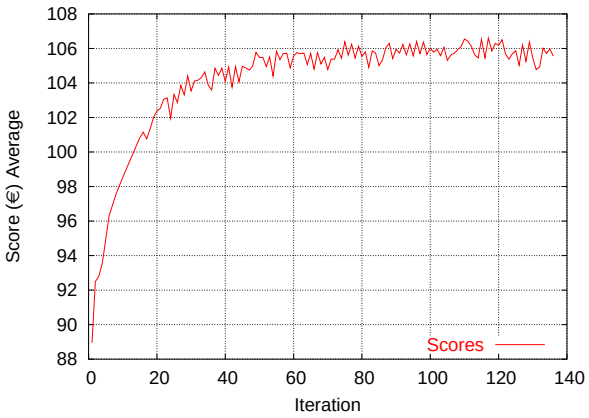
- *departure and arrival time histograms:* The number of agents that arrive/depart from an activity over time during a certain iteration.
- *average travel time:* The average travel time over all agents plans per iteration.
- *avergage score:* The average score over all agents per iteration.
- *Count Data Comparision:* Mean bias and error of of the simulations compared to the counting datas described above.

6.1 OD Matrix Based Initial Plans

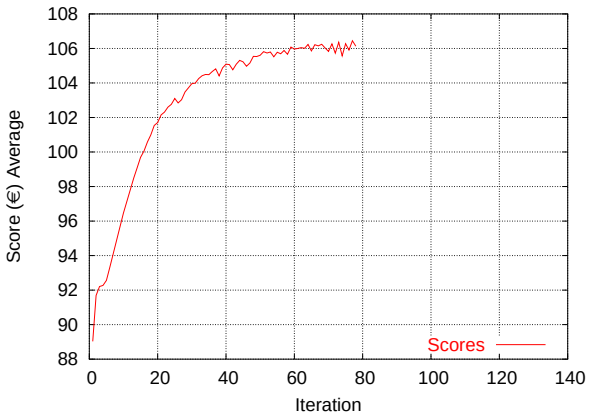
Since these plans are based on realistic time distributions (see above), we would assume that the activity-time-allocation module will not affect the result that much. The re-routing module should decrease the average travel time and congestion.
Figure 2 compares the average travel times over the iterations. The routes-only iteration (figure 2(b)) quickly gets to a stable result because re-routing is the only part which has to be optimized. The small fluctuation leads from the fact that some percentage of agents still replan.
The acts-routes iteration (figure 2(a)) behaves in a similar way, but the average travel time is a little bit higher than routes-only and also it fluctuates more. This is not surprising, since there are two dependent re-planning parts which have to be optimized.

The figure 3 shows the scores per iteration of both setups. They are also similar to each other while the acts-routes setup (figure 3(a)) has again more fluctuation than routes-only (figure 3(b)). The reason is the same as described above.

Figure 3: average scores of acts-routes-OD-plans and routes-only-OD-plans



(a) average scores per iteration for acts-routes-OD-plans setup



(b) average scores per iteration for routes-only-OD-plans setup

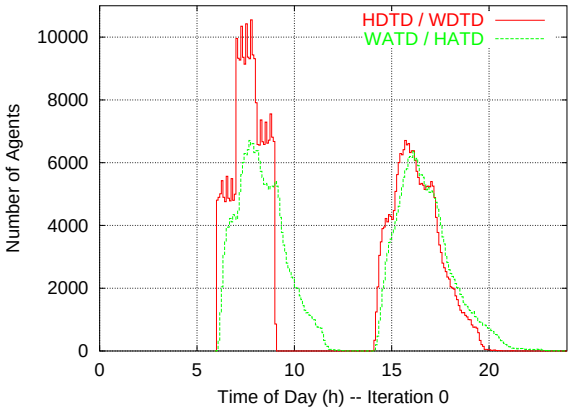
The histograms (figure 4) show, how the re-planning affects the agents. Starting with the same configuration (figure 4(a) and 4(b)) the routes-only iteration only tries to minimize travel times, so that the periods of arrivals decreases (see green graph of figure 4(d)), while departure stays the same (see red graph of figure 4(d)).
On the other hand, re-planning departure time and routes changes also the red graph (see figure 4(c)). The two peaks of the arrival (green) graph are at 7:08am and 8:52am which is the border of the time window we defined for these scenarios (see section 3.2). The reason for that leads from the fact that agents which are too late or too early at work try to "squeeze" into this time window and will more or less stay at this plan if they succeeded. Because the "departure time allocation module" still shuffles the departure times, these two peaks will flatten as more iterations are performed.

At last we look at the counts data. Figure 5 shows the relations of the two setups and the real datas (see section 4). As expected, the two results do not differ very much, and they are comparable to reality (see also Raney *et al.* (2003)). Also bias and errors[3] are similar (see table 1).

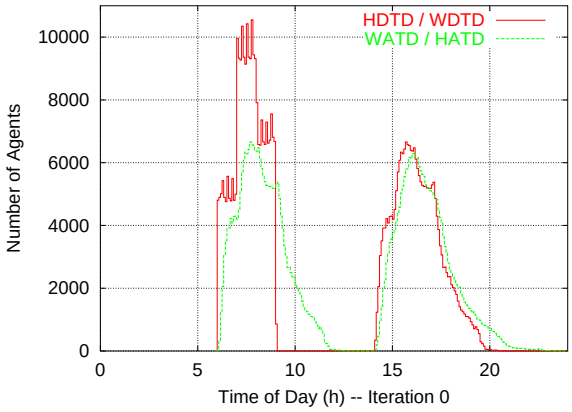
These results could lead us to the question, why do we need to use "departure time allocation" modules? Since we used the OD-Matrix it is not very surprising that the routes-only setup ends up with the simulation being similar to the real world data. The next section (Sec. 6.2) will show us what happens, if we do not know anything about the initial departure times of the agents.

[3]Mean absolute bias is $\langle q_{sim} - q_{field} \rangle$, mean absolute error is $\langle |q_{sim} - q_{field}| \rangle$, mean relative bias is $\langle (q_{sim} - q_{field})/q_{field} \rangle$, 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.

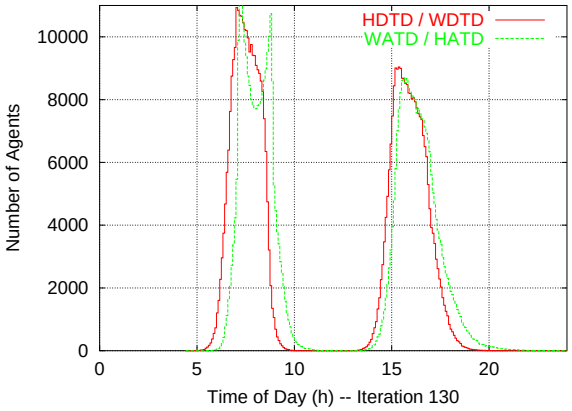
Figure 4: arrival and departure histograms of acts-routes-OD-plans and routes-only-OD-plans



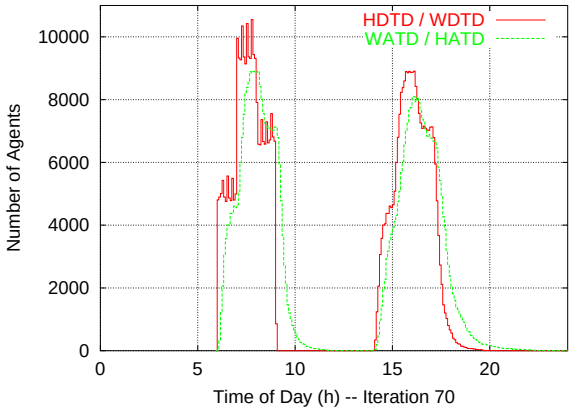
(a) arrival and departure histograms of iteration 0 for acts-routes-OD-plans setup



(b) arrival and departure histograms of iteration 0 for routes-only-OD-plans setup



(c) arrival and departure histograms of iteration 130 for acts-routes-OD-plans setup

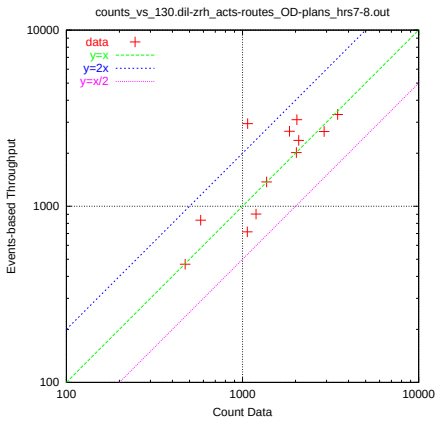


(d) arrival and departure histograms of iteration 70 for routes-only-OD-plans setup

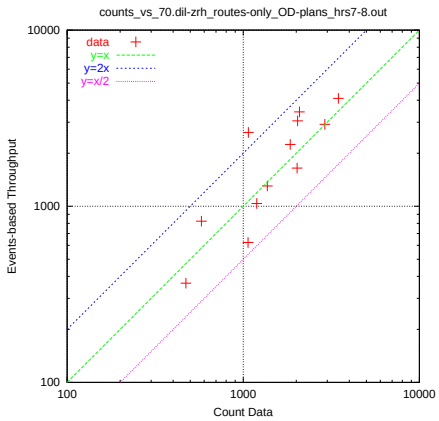
Table 1: Bias and Error of acts-routes-OD-plans and routes-only-OD-plans compared to field data at 7-8am

Mean / Bias	acts-routes-OD-plans	routes-only-OD-plans
Mean Abs. Bias:	271.153	338.403
Mean Rel. Bias:	21.6%	20.2%
Mean Abs. Error:	445.438	529.219
Mean Rel. Error:	33.4%	37.0%

Figure 5: OD-Matrix based plans: comparison to counts data

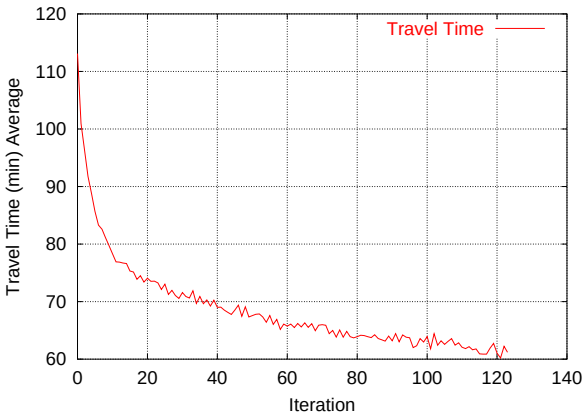


(a) counts data vs. 130th iteration of acts-routes-OD-plans setup at 7-8am

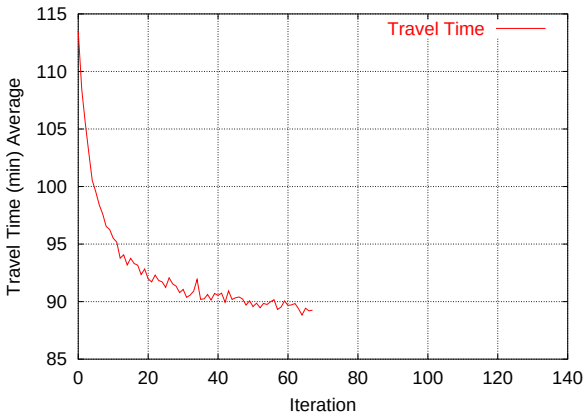


(b) counts data vs. 70th iteration of routes-only-OD-plans setup at 7-8am

Figure 6: average travel times of acts-routes-all6am and routes-only-all6am



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



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

6.2 Initial Plans with departure time at 6am for all agents

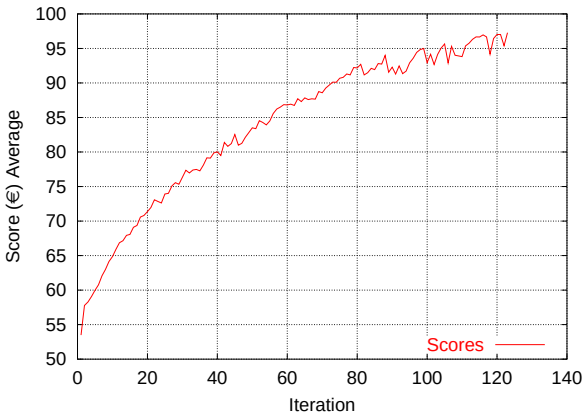
These setups take the same initial plans as the setups of section 6.1 except that all agents leave their homes at 6am instead of the distribution from the OD-Matrix.
Figure 6 shows again the average of travel times for both setups. We see that this time, the routes-only setup decreases travel time slower than before because it is harder to avoid congestions when all agents starts travelling at the same time. Of course at the end, the average travel time will be higher than the one described in section 6.1.
On the other hand the acts-routes setup decreases average travel time faster, because agents are allowed to change their departure time, too.

Also the improvement of the average scores of acts-routes setup (figure 7(a)) shows up, while there isn't any improvement for routes-only (figure 7(b)). The explanation of the flutuation of routes-only is, that during the iterations, some of the agents which are late at work have found other routes which allow them to reach the arrival time window. But on the other hand, some of the agents will find a faster path so that they need to wait longer until they are allowed to start working.

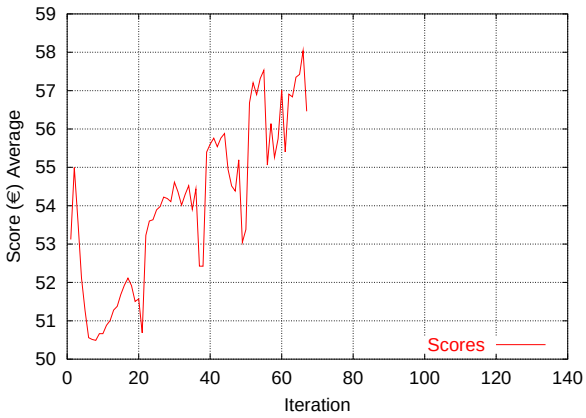
The histograms (figure 8) also shows those facts. There are many more people who arrive between 6 and 7am in the routes-only setup (figure 8(d)) than in the acts-routes setup (figure 8(c)).

Comparing the results with real word data now shows a high descrepandency between the two setups.
The routes-only setup cannot handle the fact, that almost everybody starts too early. So it overes-timates the troughput between 6 and 7am and it will underestimates at the hours later.
In the acts-routes setup, agents slowly move to more appropriate departure times which – at the

Figure 7: average scores of acts-routes-all6am and routes-only-all6am



(a) average scores per iteration for acts-routes-all6am setup



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

end – will convert to similar result as shown in section 6.1.
Of course the calculation of the bias and the error (table 2) produces now completely different result for the routes-only setup.

7. Conclusion

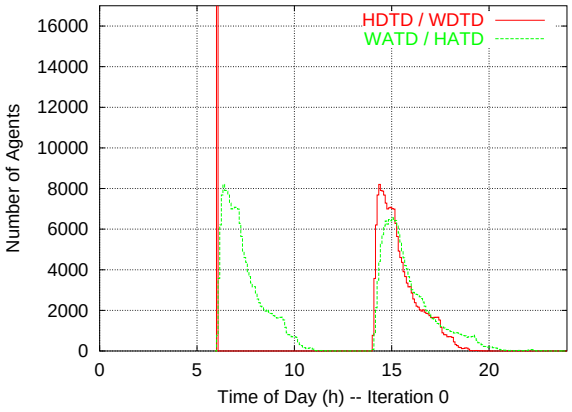
The results show us one important conclusion:

Departure time information of OD-Matrices can be substituted by a time allocation module.

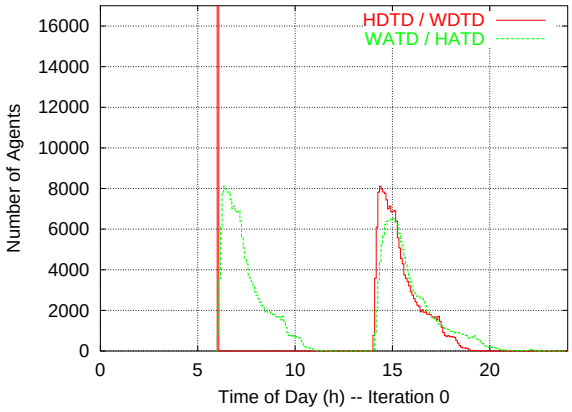
Table 2: Bias and Error of acts-routes-all6am and routes-only-all6am compared to field data at 6-7am and 7-8am

Mean / Bias	acts-routes-all6am 6-7am	routes-only-all6am 6-7am	acts-routes-all6am 7-8am	routes-only-all6am 7-8am
Mean Abs. Bias:	807.292	562.792	-411.514	-312.347
Mean Rel. Bias:	100.8%	69.7%	-27.7%	-29.8%
Mean Abs. Error:	920.252	793.129	510.944	680.645
Mean Rel. Error:	108.1%	81.7%	32.8%	45.7%

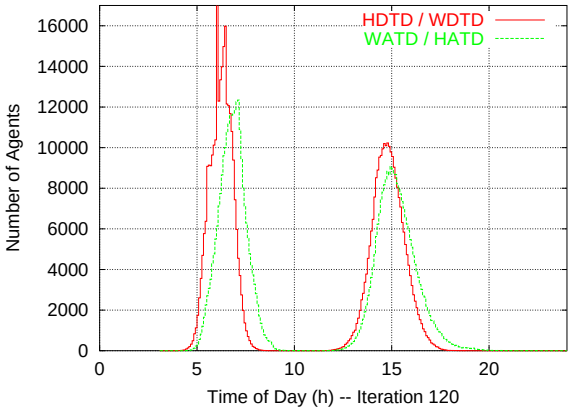
Figure 8: arrival and departure histograms of acts-routes-all6am and routes-only-all6am



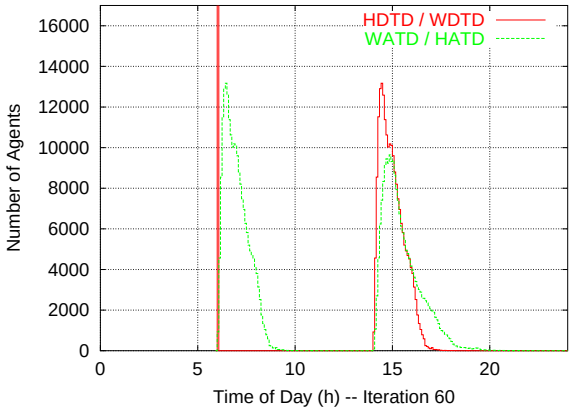
(a) arrival and departure histograms of iteration 0 for acts-routes-all6am setup



(b) arrival and departure histograms of iteration 0 for routes-only-all6am setup

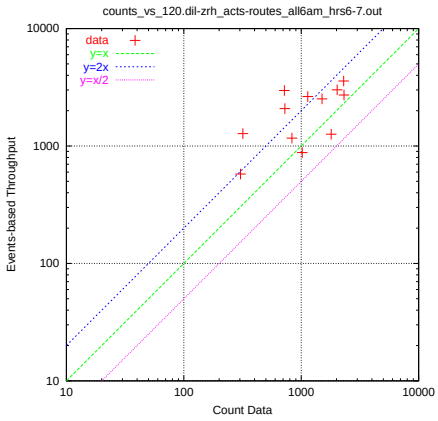


(c) arrival and departure histograms of iteration 120 for acts-routes-all6am setup

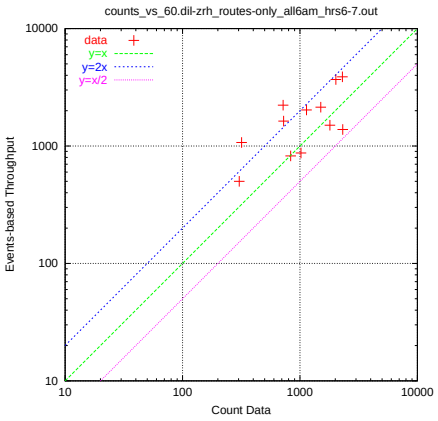


(d) arrival and departure histograms of iteration 60 for routes-only-all6am setup

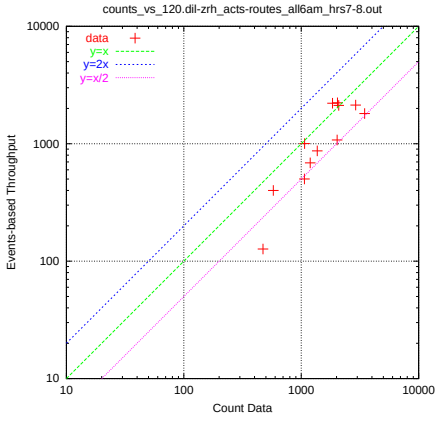
Figure 9: departure time 6am plans: comparison to counts data



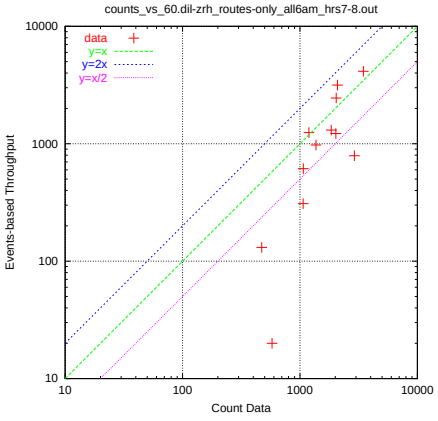
(a) counts data vs. 120th iteration of acts-routes-all6am setup at 6-7am



(b) counts data vs. 60th iteration of routes-only-all6am setup at 6-7am



(c) counts data vs. 120th iteration of acts-routes-all6am setup at 7-8am



(d) counts data vs. 60th iteration of routes-only-OD-plans setup at 7-8am

In other words, it is possible to schedule daily plans with predefined activity chains without any knowledge of departure times of activities. Furthermore, since the time allocation module is not very sophisticated yet, there are possibilities to improve convergance faster than it is at the moment.
The concept of "plugin" modules is robust in the way that it will converge to a similar "quasi-stable" result starting with different initial conditions (plans). It is a small but important step to improve this framework because other modules (see section 8) like pattern replanning modules, etc. require time allocation.

8. Future Work

As already mentioned above, other replanning modules are planned, like other time allocation or pattern replanning modules, which should also be implemented as "plugins". We are also investigating into other travel modes like public transport or pedestrian mode.

Another issue of interest is the possibility that agents could also learn during the day. They re-route while they are stuck in congestion, drop an activity because they are already too late, and so on. This "within day replanning" (Gloor, 2001) 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 mobility 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).

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 controled by a separate module which keeps track of the feedback. This leads us to a clear separation of "agent databases" and "feedback".

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

High resoltion networks are another issue. Especially if there is more preciser information available about locations. The main goal will be that each agent has its home location at a street with a house number, probabely 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 compuational challange.

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

9. Summary

The goal of this work is a multi-agent traffic simulation of a large scenario, like high resolution scenario of the region of Zürich. This work should demonstrate the feasibility of this technology on such a large scale, and it should be useful for research questions. A challenge is to keep the computational performance fast enough so that such problem sizes can be computed over night, rather than over many months as is currently the case with similar packages, e.g. TRANSIMS (www.transims.net).

This paper presents intermediate results of this work. The intent of these intermediate results was to test if the technology is able to calculate the occurence of trips rather than extract this information through surveys. This will go together with an extension of the technology towards activity-based demand generation. Further details of this will be reported in the years to come.

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 mobility simulation. The work also benefited from discussions with Fabrice Marchal.

References

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

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

Bowman, J. L. (1998) The day activity schedule approach to travel demand analysis, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA.

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

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

Charypar, D. and K. Nagel (2003) Generating complete all-day activity plans with genetic algorithms, August 2003. See http://www.ivt.baum.ethz.ch/allgemein/iatbr2003.html.

Doherty, S. T. and K. W. Axhausen (1998) The developement of a unified modelling framework for the household activity-travel scheduling process, in *Verkehr und Mobilität*, no. 66 in Stadt Region Land, Institut für Stadtbauwesen, Technical University, Aachen, Germany.

Gloor, C. (2001) Modelling of autonomous agents in a realistic road network (in German), Diplomarbeit, Swiss Federal Institute of Technology ETH, Zürich, Switzerland.

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, *Tech. Rep.*, **IVHS Technical Report 91-02**, University of Michigan Department of Industrial and Operations Engineering, Ann Arbor MI 48109, May 1991.

Nagel, K. (1994/95) High-speed microsimulations of traffic flow, Ph.D. thesis, University of Cologne. See www.inf.ethz.ch/~nagel/papers.

Raney, B., N. Cetin, A. Völlmy, 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. Similar version Transportation Research Board Annual Meeting 2003 paper number 03-4267.

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

Vaughn, K., P. Speckman and E. Pas (1997) Generating household activity-travel patterns (HATPs) for synthetic populations.

Vrtic, M. and K. Axhausen (2002) Experiment mit einem dynamischen umlegungsverfahren, *Strassenverkehrstechnik*. Also Arbeitsberichte Verkehrs- und Raumplanung No. 138, see www.ivt.baug.ethz.ch.

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