Large-scale traffic simulations for transportation planning*

Kai Nagel, Jörg Esser, and Marcus Rickert
Santa Fe Institute, 1399 Hyde Park Rd, Santa Fe NM 87501, U.S.A.
and
Los Alamos National Laboratory, Los Alamos NM 87545, U.S.A.
email kai@santafe.edu, esser@santafe.edu, rickert@santafe.edu

January 10, 2002

Abstract

An agent-based approach to simulation for transportation planning applications offers a lot of conceptual flexibility. Many millions of agents plus many hundreds of thousands of elements of transportation infrastructure need to be represented. For transportation planning applications, the demand needs to be sensitive to changes in supply, which implies that besides the realistic representation of the transportation system it is also necessary to represent people's decision-making process leading to the demand. Consistency between the dynamics in the transportation system and the assumptions about it during demand generation is achieved by iterating between demand generation and demand execution, often up to hundred times. This provides a considerable computational challenge.

1 Introduction

Nobody likes traffic jams. Yet, they seem to be one of these unavoidable facts of modern life. In fact, they have been a fact of life for long times – there are reports of congestion in ancient Rome. No building of highways, high-speed trains, airports, or other transportation infrastructure has been able to solve this problem in general.

Even without going into the reasons why that is so (which are as much cultural, economical, and sociological as they are technological), it is clear that it would be useful to have tools that allows people to analyze the situation in detail, and to make predictions. Since a lot of complicated reality as well as nonlinear behavior is involved, most such tools will have to use the computer.

*In: Annual Reviews of Computational Physics VII, edited by D. Stauffer, World Scientific, 2000, pp. 151–202

Indeed, computational tools for transportation have been around since at least the 1950s. Today, driving simulators simulate the behavior of cars, simulations are used for the design of individual intersections, or computational algorithms are used to optimize signal phasings along arterials. See Ref. [1] for an overview of traffic micro-simulation models. In this paper, we want to concentrate on one of the largest challenges that can arise: tools that deal with whole metropolitan areas, which can contain 15 million people or more.

In such systems, one is typically interested in long-term planning questions. For example, will highway construction indeed relieve congestion, or will it maybe just lead to a reorganization of where people live, work, and shop? What are the environmental consequences of such a change? What if one would construct a light rail instead of the highway, or do nothing? Such questions typically deal with a 20-year time horizon, that is, for the evaluation of such a construction project one would like to know its consequences in 20 years from now.

This makes immediately clear that one needs more than just the simulation of how people drive, or which subway line they use. We need to deal with possible changes in land use, with possible changes in people's preferences, with possible changes in technology, etc. It is however clear to most people in the field that science is currently not, and possibly will never be, able to answer such questions beyond the extrapolation of current trends. For example, nobody would be able to have been able to reliably predict the effects of the Internet 20 years ago. However, what we would like to have is a tool where we can evaluate *scenarios*. For example, we could assume that, in twenty years from now, people work two thirds of their work time at home, and ask what effect this would have on congestion.

What is implied by such an argument is that we are interested in tools where we have "access" to such changes. We would like tools which allow us to evaluate all kinds of scenarios, especially those that the programmers did not anticipate when they wrote the code. This is a main argument why such tools should be microscopic and modular, which means that they represent the world as closely as possible in a one-to-one way. In principle, in such a tool people should be represented as people, cars should be represented as cars, and traffic lights should be represented as traffic lights. And not as, say, departure rates, traffic streams, and capacities, respectively. A microscopic approach is one that keeps our science open for development.

The downside of this is that we are suddenly faced with a problem of enormous dimensions. Even for a medium-sized problem such as Portland/Oregon, which is the current test-case for the TRANSIMS project (see below), 1.5 million people or more need to be represented in the computer, plus 500 000 cars, about 150 bus lines, 200 000 road links, 80 000 intersections, etc. This shows at least two aspects of the problem: (i) The size of the problem is large, but it is within what Computational Statistical Physics can already deal with. However, (ii) the problem is much more complicated than typical Statistical Physics problems in at least one aspect: The units are not homogeneous. Cities typically have one or more central business districts, shopping districts, different neighborhoods,

suburbs, and open space, not to talk about geographical constraints such as mountains or rivers. People have different quantifiable characteristics (such as income, age, gender) as well as different preferences. Cars are all different from each other, signals all have different timing plans, etc.

This leads to two questions:

- Is all this complicatedness necessary to obtain realistic answers?
- How much of this do we actually know?

And this is where, besides for the computing methods, a basic and systematic science such as Statistical Physics comes in: Because these questions can be approached bottom-up, one can formulate simple models for regional traffic, and then investigate which aspects of the system they represent correctly and which ones they do not. For the aspects they do not represent correctly, one can search for simple model additions that correct these problems, etc. In this way, instead of getting stuck over collecting all the complicated data from reality, one can build item by item an understanding of how to build traffic simulation tools, somewhat similar to how Statistical Physics works: Particles with no interaction lead to the ideal gas equation, which is already very powerful for many problems; adding excluded volume leads to the Van-der-Waals equation, which explains gas-to-liquid phase transitions qualitatively; etc. up to granular flow and weather forecasting.

In this paper we will look at the absolutely necessary modules that make up a transportation planning simulation. Before going into the details, we will give a "quick tour" of these modules (Sec. 2). The first module is demand generation (Sec. 3) which generates the demand for travel. This demand is executed in the transportation micro-simulation, also called supply simulation (Sec. 4). Feedback is necessary to iterate between these two in order to make them consistent (Sec. 5). No simulation is useful without analysis, which includes validation (Sec. 7). That section is preceded by a short section about two TRANSIMS real world scenarios (Sec. 6). Sec. 8 explains some computational issues, followed by the usual summary and conclusion (Sec. 9). In an appendix, we describe a very simple set-up to build a very simplified, but complete transportation simulation package. The text will to a large extent follow TRANSIMS, the TRansportation ANalysis and SIMulation System project at Los Alamos National Laboratory [2], since that is the system the authors are most familiar with and can draw most from their experience. Another review, focusing on the theoretical and analytical results of cellular automata models for traffic flow, can be found in Ref. [3].

2 A quick tour

2.1 Introduction

There are several ways to lay out the different parts of a transportation simulation. They will be discussed in Sec. 3. For this quick tour, we will just

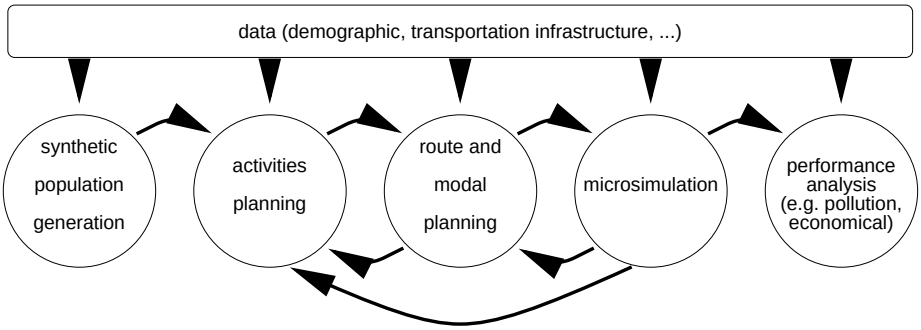

Figure 1: TRANSIMS design

look at the most important modules that are necessary to get a reasonable package. These modules are: demand generation, supply (= transportation micro-simulation), feedback, and analysis (Fig. 1). The following four sections will give short introductions into these four modules.

2.2 Demand simulation

The demand generation module generates the demand for the transportation simulation system. Two important methods here are: (i) origin-destination matrices, and (ii) activities-based demand modeling.

Origin-destination (OD) matrices are the more traditional method. OD matrices contain the number of trips from n starting points to n destinations; it is therefore an $n \times n$ matrix. These matrices can refer to arbitrary time periods. Until a couple of years ago, one typically used 24-hour time periods; these days, people often concentrate on "morning peak" and "afternoon peak" periods since the main direction of travel is obviously different between these periods.

In many situations, it is desirable to have information about demand generation that goes beyond OD matrices. In such situations, the more far-reaching method of activities-based demand modeling is an alternative. Here, the simulation includes models of human behavior with respect to the planning of a day. This includes where and when to eat, sleep, work, shop, etc. For example, a person may start the day at home, be at work at 8am, work for eight hours, go shopping which takes an hour, then be at home for the rest of the day. Assuming that all the transportation pieces take half an hour, this would fix the transportation schedule to: leave home at 7:30am, be at work at 8am, leave work at 4pm, arrive at shopping at 4:30pm, leave shopping at 5:30pm, arrive home at 6pm.

Once the simulation "knows" where and when people do their activities, transportation is generated via connecting activities that take place at different locations. Note that it is not necessary (and probably not possible) to forecast such activities for specific persons; however, there is hope that we will be able to get useful ensemble averages similarly to Statistical Physics.

Once trips (e.g. starting times, starting locations, and destination locations) are known, the exact transportation for these needs to be generated. This includes modal choice (walking, bicycle, train, car, etc.) and the precise routing.

2.3 Supply simulation

The demand generation in some sense describes the "plans making" of individuals. The synthetic individuals in the simulation compute their plans for the following time period, which may for example be a day. Now these plans need to be executed. This is sometimes also called the supply simulation, since it simulates how the transportation system, i.e. the supply of transportation capacity, reacts to the demand. These simulations come at many different levels of resolution and fidelity, reaching from the traditional steady-state flow-based cost function to very detailed micro-simulations.

If one is interested in time-dependent results, as for example the queue built-up during the onset of rush periods, the simulation needs to be sufficiently realistic to contain such dynamics. Traditional flow-based cost functions are *not* able to realistically deal with such dynamical effects, at least not in a straightforward way. Thus, the right supply simulation has to be chosen according to what aspects of the dynamics one wants to have represented for a given question.

2.4 Feedback

The supply simulation needs input from the demand generation, since it executes the plans from the demand generation. However, the demand generation depends on the supply simulation because for example congestion only shows up in the supply simulation, and demand adjusts to such shortages. In order to deal with this situation, one iterates between demand and supply simulation. For example, the demand simulation is run assuming no congestion, the resulting supply simulation is run, then the demand simulation is run again now including the congestion from the last supply simulation run, etc., until a steady state is reached. That is, the system is systematically relaxed towards a consistent state.

It should be noted at this point that there is no a priori reason why a real system should be relaxed. For example, during unique events such as trade shows, the transportation system is probably not relaxed. The research here just follows the usual path in such situations: First understand the steady state solution, and then move on to the transients. Note that the steady state here refers to the comparison from one iteration to the next, *not* to a steady state across time-of-day.

2.5 Analysis

Once a representative run or collection of runs of the supply simulation has been obtained, it can be analyzed. For example, one can see where congestion will show up, and which people get stuck in it. Analysis is the other aspect of the

system that influences the decision about the level of realism in the modules. For example, if one is interested in emissions, one needs a micro-simulation of the driving behavior with enough information on, e.g., acceleration in order to derive the necessary quantities. Or if one is interested in the possible rescheduling of activities as a consequence of transportation infrastructure changes, one needs to model the effect of "trip chaining", i.e. the fact that people can for example go shopping on the way back from work, but they could also put in a stop at home before they go shopping.

3 Demand simulation

3.1 Introduction

Transportation simulations need to work across many scales: from 20-year time horizons for urban planning to split-second driving decisions, from metropolitan areas with scales of several hundred kilometers down to driving which happens on scales of meters. It is, however, virtually impossible to design a monolithic software and modeling package that treats all these scales in one module. The task would be overwhelming, both in terms of software maintenance and in terms of design. The latter stems from the fact that the decisions on these different scales are highly interconnected. For example, the decision to go to work at a certain time of day can be pre-planned several months in advance, but it may be changed in a couple of minutes if a child gets sick.

In order to avoid this problem, most transportation simulation packages consist of several modules, such as activities planning, route planning, and transportation micro-simulation (Fig. 1); and the dependencies between the modules are resolved via feedback. Another way to conceptually modularize transportation simulation packages is into the demand and the supply side (e.g. [4]). The transportation micro-simulation is then sometimes called the supply simulation.

Whichever classification one chooses, one needs to recognize that none of them is entirely sharp. If one attempts to draw clean distinctions between modules, one will probably never arrive at a working system. The alternative is to explicitly live with the tension between the modules, and accept that for example a mode decision can reasonably be implemented both in the activities module (do not want to carry goods from shopping to home by bus) or in the routing module (car is faster than bus). Thus, emphasis needs to be put not on the impossible task to conceptually resolve these tensions, but on reconciling them so that a believable result emerges. This task of reconciliation is done by the "feedback" between the modules, described in a section of its own. This section will describe the demand generation module.

3.2 Population generation

Since our methods run on synthetic populations of individuals, these synthetic individuals need to be generated. In principle, the method is to use census data

	age 0-30	age 31-60	age > 60	
number of males	?	?	?	200
number of females	?	?	?	200
	200	200	0	400

	age 0-30	age 31-60	age > 60	
number of males	0	10	0	10
number of females	5	0	0	5
	5	10	0	15

	age 0-30	age 31-60	age > 60	
number of males	0	200	0	200
number of females	200	0	0	200
	200	200	0	400

Table 1: Example tables for synthetic population generation. *Top:* Table for a block-group as it could be given by the census. *Middle:* Table as it could be given by the micro-census (PUMS). *Bottom:* Table for a block-group as it would be generated from this information.

and to randomly generate a realization of individuals and households with the correct demographics. For example, a census could give us the number of people and their gender distribution for a certain block group, and the algorithm would have to put households on the street network, and populate them with the right number of male and female people.

The problem is that, given typical census data, this is an under-specified problem. For example, assuming we have the following information about people in a census block (see Table 1): number of males, number of females, number with age 0-30, number with age 30-60, number with age above 60. This results in four independent parameters; however, we would need six (for male/0-30, male/30-60, male/> 60, female/0-30, etc.) to run the above algorithm.

One way to solve this is to use additional census information, sometimes called the micro-census (Public Use Microdata Sample (PUMS) in the United States [5]). The micro-census is taken only from a sample, but it provides the missing correlation information. For example, it might tell us that in a specific area, all females are age 0-30, and all males are age 30-60. This would allow us to fill out the whole table: male/0-30 would be zero, male/30-60 would contain the whole male population of our block group, etc. Naturally, this assumes that the two informations are consistent, for which there is no guarantee that they are. For example, in our example, the census could have the same number

of males and females, but more people aged 0-30 than 30-60. Then it would be impossible to reconcile this census with the micro-census from above. See Ref. [6] for a more sophisticated method that deals with such inconsistencies in a systematic way.

3.3 Activities generation

Once the synthetic population is generated, all other modules act directly on the agents. What is necessary here is a procedure that as a result generates travel demand, i.e. the wish of people to move from one location to another. As already said in Sec. 2.2, two important methods here are: (i) origin-destination matrices, and (ii) activity-based demand modeling.

3.3.1 Origin-destination matrices

As also already said in Sec. 2.2, origin-destination (OD) matrices contain the number of trips from n starting points to n destinations; it is therefore an $n \times n$ matrix. As also said, these matrices can refer to arbitrary time periods; these days, one typically uses "morning peak" and "afternoon peak" periods.

There are many ways to obtain origin-destination matrices. In transportation planning, the typical methods is to anchor them to the land use, and to use behavioral "rates" to determine trip frequencies (e.g. [7]). Residential areas "produce" so and so many trips per capita; commercial areas "attract" so and so many trips per capita. The matching of origins to destinations is done via gravity methods, i.e. the probability of a trip to go to a certain destination is some function of the attraction of this destination and the generalized cost of getting there.

Another method is to derive OD matrices from traffic counts. Here, one collects counts on as many links of the transportation network as possible, and then uses statistical estimators to derive OD matrices from this (e.g. [8]). Statistical estimators are necessary because the problem is under-determined. Sometimes, the two approaches are combined, i.e. the historical OD-matrices are used as starting points, but they are corrected via traffic counts [4].

3.3.2 Activities-based demand modeling

The problem with OD matrices is that they fix the travel demand once they have been derived. Thus, they fail to generate the effect of "induced" travel, which usually happens when one expands capacity. For example, a new freeway may induce people to make more trips, thus increasing overall travel. This means that one needs a demand generation method that is elastic with changing supply.

Activity-based methods attempt to achieve this by generating directly what people do during a day and where; transportation demand is thus derived by connecting activities at different locations (Fig. 2). There are at least two different methods to generate activities: econometric, and heuristic.

Figure 2: Example of a sequence of activities for a person in Portland/Oregon. From R.J. Beckman.

In principle, one can derive OD-matrices from activities, and many groups do this because it connects activity-based demand generation to existing models. This has, however, to be done with care since one loses important information. An important example of lost information are trip *chains*, where a person may go to work, may go shopping, and then home. If the person gets stuck on the way to shopping, the trip from shopping to home will take place later than anticipated; such effects do not get picked up in the OD matrix. Also, a universal reaction to changes in congestion seems to be to add or suppress intermediate stops at home, i.e. to replace home-work-home-shop-home by home-work-shop-home or vice versa. One would have to be careful to not suppress these possibilities when translating the trip chains into OD-matrices.

Econometric Econometric methods [9, 10] are based on utility theory. Utility theory assumes that the utility a person i sees in a certain action a is composed of a measurable and a non-measurable part:

$$U(i,a) = V(i,a) + \eta(i,a) \ .$$

Under a variety of assumptions, e.g. that η is a random variable and follows a certain distribution, this leads to an equation for the probability to choose action a.

An often-used discrete choice model is the so-called logit model. Its main assumptions are:

- Individuals and actions are characterized by certain attributes, that is, two individuals with the same attributes will be modeled by the same equation. This also means that i and a are replaced by a vector of attributes, $\vec{x}_{i,a}$.
- The measurable part of the utilities, V, is a linear function of the attributes, i.e. $V = \vec{\beta} \cdot \vec{x}$.
- The random variables η do not depend on the attributes $\vec{x}_{i,a}$, and they are Gumbel distributed, i.e. the generating function is

  $$F(\eta) = \exp[-e^{-\mu\,(\eta-\gamma)}] \;,$$

  which results in the distribution

  $$f(\eta) = \mu\, e^{-\mu\,(\eta-\gamma)}\, \exp[-e^{-\mu\,(\eta-\gamma)}]$$

  γ is a location parameter, and μ is a positive scale parameter. This distribution is somewhat similar to an asymmetric version of the normal distribution; its main advantage is that it leads to a closed form solution of the choice model.

With a logit model, the probability to choose the bus in a decision between bus and car could look as follows:

$$P(bus) = \frac{\exp[-\beta_b\, t_b]}{\exp[-\beta_b\, t_b] + \exp[-\beta_c\, t_c]} \;. \tag{1}$$

t_b and t_c are the respective travel times the trip would take by bus or by car. β_b and β_c are factors which weigh time in the bus vs. time in the car, i.e. they are "values of time". For example, one could say that time in the bus is more productive than in the car because one can read, resulting in $\beta_b > \beta_c$. However, usually the car is faster, compensating for this effect. – Note that Eq. 1 has the same functional form as a Boltzmann distribution.

The β_b and β_c are estimated from surveys, for example via maximum likelihood methods. A sample of the population with different car and bus travel times is asked about their choices, and the β_x are determined such that the probability according to Eq. 1 to re-generate the survey is maximized.

For applications inside a transportation simulation, this becomes a lot more complicated. An implementation for Portland/Oregon [11] determines activity patterns (for example home-work-home or home-work-shop-home), activity timing, activity locations, mode choice, etc. As long as one wants to treat all alternatives simultaneously, this has the problem that the number of coefficients grows exponentially. For example, if one has five activities patterns, and three

modes of transportation, this means 15 different choices and thus 15 parameters. If however one does not treat the alternatives simultaneously, one can make mistakes: For example, a person could have a strong preference for a pattern home-work-home-shop-home when averaged over *all* possible circumstances, but may prefer home-work-shop-home when really good bus service is available. When choosing first the pattern and then the transportation mode, this information gets misrepresented.

Heuristic methods The econometric method has a solid theoretical foundation, and it is currently the only method that is functional for transportation simulations. However, sometimes it seems like it does not really represent how people behave. The discrete choice method pretends that people calculate utilities for all possible alternatives and then choose the alternative with the highest utility. (Remember that the randomization just comes in because of "unobserved attributes".) However, people do not do this. For example, they may discard an activity pattern home-shop-work-home right away without calculating the utilities of all possible constellations.

Heuristic methods attempt to better represent such human planning processes. For example, research shows that humans make their planning decisions on many time scales simultaneously [12]. The time for work is usually alloted way in advance, shopping may be planned a day in advance, and then the whole schedule may be changed short-term because the child gets sick. Prototypes for such models exist, but they seem currently far away from being operational in any meaningful way.

It should be noted that heuristic and econometric methods can be combined. For example, one could use a heuristic method to determine which decisions are made how far in advance, and use an econometric method to make the actual decision. Or the econometric method could calculate the probability for each activities pattern, the heuristic method could decide to retain the two most important patterns, the econometric method than could calculate the utilities for these two patterns for all mode and time combinations, etc.

Summary of activities-based methods Activities-based demand generation models are a promising method for transportation simulation. Some implementations of these methods have reached the state where they can be used for actual applications [13]. However, so far there are only very few results about coupling these methods together with transportation micro-simulations, as intended with the transportation planning simulation packages described in this article. The only functional system that we are aware of uses a very simple method of demand generation; it is described in the appendix. But we are optimistic that research in the next couple of years will expand the boundaries in these areas enormously.

3.4 Route planners

Activity generators should return locations of activities, and information about times. Route planners are modules that compute sequences of links that lead from a starting point to a destination in a given graph, i.e. they connect activities at different locations. Such routes could be optimal according to some criterion, but do not have to be. The currently most used route planners in transportation simulations are optimal time-dependent fastest path algorithms, for example a time-dependent Dijkstra [14]. Assuming that link travel times are known in, say, 15-minute time intervals, then the algorithm calculates the fastest path with either the arrival time at the destination or the starting time at the start given.

This can be expanded in many directions. For example, it is possible to replace time by another cost, for example the number of left turns, or a function of both. It is also possible to force the algorithm to search under constraints. For example, one can search for a fastest path under the restriction that first a car and then a bus (i.e. park-and-ride) should be used for during the trip. As long as these constraints are simple enough (technically: the constraint can be formulated as a context free language), the time complexity of the algorithm remains polynomial [15].

Another open question is how to generate "reasonable" second-best, third-best, etc. options. It is possible to systematically generate k-shortest paths (although this may be computationally costly when time-dependent) [16], but most of these options would never be chosen by a human. A typical case is that the best option uses a freeway, and the second-best option gets off the freeway somewhere, just to cross the intersection and to get on again. The third-best option may do the same thing at a different location, the fourth-best option may do both things, and only the fifth-best option may provide a true alternative using a completely different freeway. There are methods to avoid some of these problems, but often they generate other artifacts [17, 18].

3.5 Summary of demand generation

The demand generation part of a transportation planning simulation package consists of at least a trip generation module, and a routing module (which includes modal choice). Established for trip generation are origin-destination-matrices, but they have shortcomings especially when one expects demand to be elastic. Promising for the future are methods which are entirely based on individual agents. A prerequisite of any agent-based method is a module which generates realizations of agent populations for our metropolitan area. Once agents exist in the simulation, one can plan their activities, i.e. their patterns and locations of activities, such as work, shop, eat, sleep, etc. Connecting activities at different locations generates travel demand, which gets routed by the router. The result of the demand generation should be a list of plans for each individual, containing not only times and locations, but a precise description of the intended transportation, which could be car, but also walking or taking a

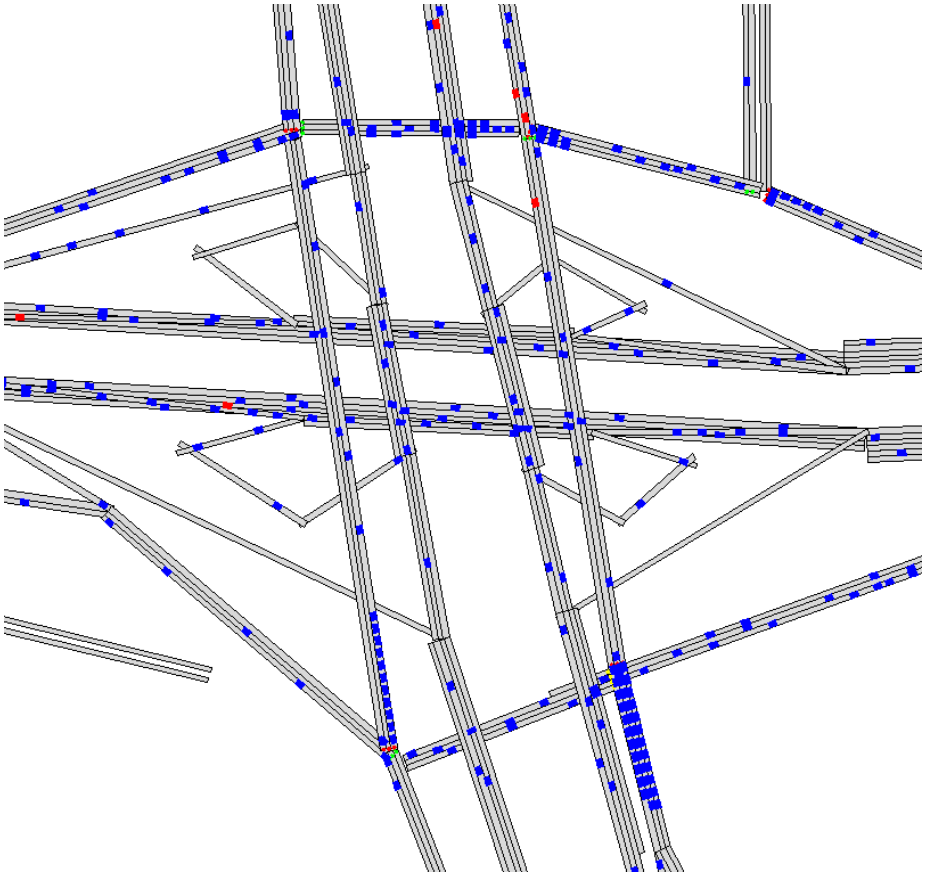

Figure 3: Snapshot from a transportation micro-simulation.

bus.

This paper usually assumes that plans are pre-computed; once they are fed into the micro-simulation, they cannot be changed any more. This is currently true for many implementations, but it does not have to be true in general. For example, one could imagine that the transportation micro-simulation could call the planning (i.e. demand generation) modules on-line, see, e.g., Refs. [19, 20, 21]. Another possibility is to use the feedback mechanism (Sec. 5) to model on-line re-planning [22].

4 Supply simulation: Transportation micro-simulation

As stated in the introduction, supply simulations can reach from fairly simple steady state mappings of link flow into link cost to sophisticated micro-

simulations of all aspects of a transportation system. We will concentrate here on micro-simulations, since they are (a) more open for long-term progress, (b) computationally the most challenging, and (c) maybe closest to Computational Statistical Physics. The first and last arguments refer to the observation that in a microscopic representation, one can always directly add or change elements if the current simulation is no longer judged sufficient to represent reality, as opposed to an aggregated representation. For example, slow vehicles are easy to add in a microscopic formulation, but hard in a fluid-dynamical one. This is similar to Statistical Physics, where one usually starts from a microscopic formulation, and only sometimes obtains a closed-form macroscopic description.

A realistic traffic micro-simulation operates on links, which are connected via nodes or intersections to form the road network (Fig. 3). This means that one has two parts for the dynamics: dynamics on a link, and dynamics on the nodes. The two most important elements of link dynamics are car following (i.e. single lane traffic) and lane changing. The two most important elements of node dynamics are signals, and unprotected turns (e.g. yield signs).

4.1 Car following

4.1.1 Introduction

Any micro-simulation first needs to have a method for simple car following. Such methods can be developed on single-lane loops, similar to a single-lane race track. A good way to start is the rule of thumb of "two seconds time headway", that many of us learn at driving school. We are supposed to have two seconds between the time when the car ahead passes, say, a traffic sign, and the time when we pass it. The reason for this is related to our reaction time. If the car ahead starts braking really hard right at that traffic sign, and we need two seconds to react and start braking at (roughly) the same position, we will barely avoid a crash. Thus, time headway needs to be larger than reaction time, which translates into a space headway proportional to speed. As a consequence, most car following models have as their most important term one that makes the velocity a roughly linear function of the space headway or gap.[1] All car following models based on this principle have a similar dynamical behavior. For example, the transition from laminar to start-stop traffic is similar for all these models [23]. Car following models which are used in micro-simulations are usually designed to be free of accidents.

[1] "Gap" denotes the space from my front bumper to the rear bumper of the car ahead, sometimes minus some safety space I would always like to have in between. Space headway is used less uniformly. For example, it sometimes denotes the front-bumper-to-front-bumper space, thus including the length of the car ahead. This last definition is the necessary one to retrieve the density, while the "gap" definition is the one that is relevant for driving behavior.

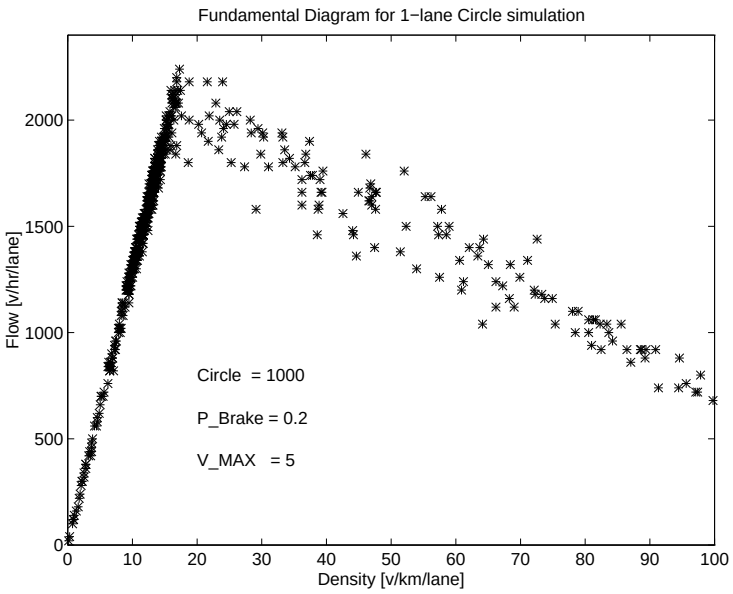

Figure 4: One-lane fundamental diagram as obtained with the standard cellular automata model for traffic using $p_{noise} = 0.2$. From [24].

4.1.2 Discrete space and discrete time: Cellular automata rules

Incarnations of car following can use continuous or discrete time, and continuous or discrete space. While continuous space and continuous time is more realistic, discrete space and time are more natural for a digital computer. And recent research has shown that, in the spirit of Statistical Physics, extremely simple and even unrealistic rules on the microscopic level can still lead to reasonable behavior on the macroscopic level, up to a point [25, 26, 27]. In consequence, cellular automata (CA) techniques, which are discrete in space and time, plus have a parallel local update, can actually simulate traffic quite well. Typical CA for traffic represent the single-lane road as an array of cells of length ℓ, each cell either empty or occupied by a single vehicle. Vehicles have integer velocities between zero and v_{max}. A possible update rule is [28]

(1) $v(t+1) = \min[gap, v(t)+1, v_{max}]$

(2) $x(t+1) = x(t) + v(t+1)$

gap is the number of empty cells between the vehicle under consideration and the vehicle ahead, and v is measured in "cells per time step". This rule is similar to the CA rule 184 according to the Wolfram classification [29]; indeed, for $v_{max} = 1$ it is identical. This model has some important features of traffic, such as start-stop waves, but it is unrealistically "stiff" in its dynamics.

ℓ is the length a vehicle occupies in a jam, it is often taken as $\ell = 7.5\ m$. In order to get realistic results, a time step of one second is a good choice (remember the reaction time), and then $v_{max} = 5$ corresponding to 135 km/h is a good choice. In applications, v_{max} can be set according to a speed limit on the link. Note that in the traffic CA community distances and speeds are often given without units, which means that they refer to "cells" or "cells per time step", respectively.

One can add noise by adding a randomization term:

(1b) With probability p_{noise} do: $v(t+1) = \max[v(t+1) - 1, 0]$.

This makes the dynamics of the model significantly more realistic. $p_{noise} = 0.5$ is a standard choice for theoretical work; $p_{noise} = 0.2$ is more realistic with respect to the resulting value for maximum flow (capacity), see Fig. 4 [24].

Real traffic has a strong hysteresis effect near maximum flow: When coming from low densities, traffic stays laminar and fast up to a certain density ρ_2. Above that, traffic "breaks down" into start-stop traffic. When lowering the density again, however, it does not become laminar again until $\rho < \rho_1$, which is significantly smaller than ρ_2, up to 30% [30, 31]. This effect can be included into the above rules by making acceleration out of stopped traffic weaker than acceleration at all other speeds, for example by:

- if ($v(t) = 0$ and $gap \leq 1$) then $v(t+1) = 0$
- else $v(t+1) = \min[gap, v(t)+1, v_{max}]$.

This means that the vehicle needs a larger gap than before to start moving. Such rules are called "slow-to-start" rules in the physics community [32, 33].

CA rules can also be analyzed analytically, by means of statistical techniques which look at sequences of configurations of the dynamical evolution of the system (see, e.g., Refs. [34, 35, 3]). Note that this is possible because the cellular approach makes the dynamical states countable: There is only a finite number of possible states for a given number of cells.

4.1.3 Continuous space and continuous time (CSCT)

Making both space and time continuous results in coupled differential equations. Such models for car following were established quite some time ago, see, e.g., Ref. [36]. For computer implementations they are a bit inconvenient, since they need to be discretized in time in one way or other. Because of the reaction delay, many of these car-following equations are delay equations, where considerable effort needs to be spent for faithful numerical results. Given this observation, it seems to be simpler to build models that use discretized time to their advantage (see next section). This is not to say that continuous car-following models are useless; indeed, they continue to contribute to our understanding of the matter (e.g. [37, 38]). We would expect, however (see below), that any faithful discretization of these equations will run a lot more slowly on a computer than the model presented in the next section, which explicitly uses discrete time.

Another possible implementation of continuous space and time would be event-driven. This works best when particles move with constant velocity for periods of time, interrupted by events where they change it. Molecular dynamics with hard core interactions is an example. Since human driving behavior can probably indeed be characterized like that [39], this should be a promising approach. However, parallel implementations of event-driven simulations are hard [22] and therefore large scale simulations currently not done with this method.

4.1.4 Continuous space but discrete time (CSDT)

Krauss [25] has done a systematic investigation of models that are derived from the CA approach, i.e. they have a parallel update based on local rules, but they allow continuous space and therefore continuous velocities. Starting from a more detailed version of the "no crash" rule from Sec. 4.1.1, he derived conditions under which CSDT models are crash-free. One of these conditions is that the simulation time step cannot be larger than the reaction time, but making it smaller does not yield any clear improvement, either. Since the number of numerical operations on these models is not much higher than for the CA models, and these days floating point operations are often faster than integer operations, CSDT models do not run more than 50% slower than CA models [40]; however, CSDT models still run about 50 times faster than CSCT models [40]. For this model, an extensive investigation of its properties is available [25], including a phase diagram with three different phases, which depend on acceleration and braking capability.

4.2 Lane changing

All lane changing rules follow a similar scheme (e.g. [41]): In order to change lanes, drivers need an incentive, and the lane change needs to be safe. An incentive can be that the other lane is faster, or that the driver eventually needs to make a turn. Safety implies that one needs enough space on the target lane. Thus, a simple lane changing condition can read as [42]:

(I1) incentive: $gap_{other} > gap$, i.e. the gap on the other lane is larger than the gap on the current lane, allowing a higher speed on the other lane

(S1) safety: $gap_{other,back} > v_{back}$, i.e. the *backwards* gap on the other lane is large enough that a vehicle approaching with v_{back} does not have to slow down immediately.

This is indeed the lane changing criterion currently used in the TRANSIMS micro-simulation, and it seems to work reasonably well for American traffic [24].

The above lane changing criterion is symmetric, since changing to the left happens according to the same criterion as changing to the right. One result of this is that people stay in the left lane until some incentive pushes them out of it, again not totally unrealistic for traffic in the United States. For European

(and other) countries, one better uses asymmetric lane changing rules, with the right lane as the default lane. A useful construction criterion for such rules is that the incentive to go right is the logical negation of the incentive to go left [27]. For example, the incentive to go to the left could again be

(I2a) $gap_{left} > gap_{right}$

and together with the above safety criterion (S1) changing to the left would be ruled by exactly the same criteria as in the symmetric rule above. If one, in contrast to above, makes the negation of (I2a), i.e.

(I2b) $gap_{left} \leq gap_{right}$,

the incentive criterion for going right, then vehicles return to the right lane as soon as there is space available. This assumes a sight distance limitation for drivers, i.e. if there are no cars ahead within a certain distance on either lane, then both gap_{left} and gap_{right} should be equal to this sight distance, and the criterion is fulfilled.

An interesting observation here is that again microscopic lane changing rules that are macroscopically meaningful do not have to be completely realistic on the microscopic level. For example, all lane changes according to the above rules happen in one simulation time step, i.e. usually in one second, whereas in reality this takes much longer (3–5 seconds). Also, the above rules result in much too many lane changes when traffic on both lanes is similar – an effect that is annoying in demonstrations (see, for example, one of the TRANSIMS videos) but macroscopic relations such as fundamental diagrams still come out correct [42, 27].

As noted above, the incentive to change lanes could also come from an intended turn movement at the end of the link, and one can partially over-ride the safety criterion with increasing urgency of the incentive criterion.

4.3 Traffic signals

We now turn to intersections, where links with car following and lane changing dynamics are connected. The easiest case are fully signalized intersections since the signal (assuming it is working correctly) is taking care of avoiding crashes. The dynamics resulting from a red light can be generated by placing a virtual car with speed zero into the last spot on the link, and removing this car once lights turn green. Although in principle car-following rules that generate the right capacity on a single lane road also generate the right flow through traffic lights, one should nevertheless check this [24]. Also, the lane changing rules need to allow vehicles to make it to their intended turning lane even under congested conditions at a traffic light [24].

4.4 Unprotected turns

A little more difficult are unprotected turns, i.e. turns that are not regulated by traffic signals and where vehicles need to merge on their own without accidents.

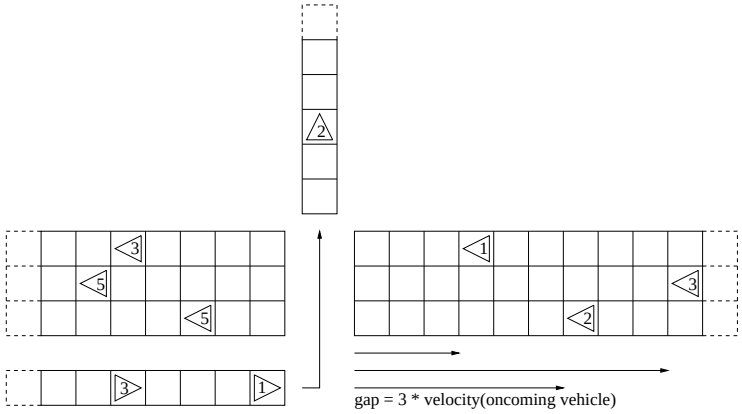

Figure 5: Illustration of gap acceptance for a left turn against oncoming traffic. From [24].

Typical examples of this are yield, stop, "right on red", left turns against oncoming traffic, and on-ramps to freeways. The mechanism here is again a "gap acceptance" similar to the safety criterion (S1) for lane changes (Fig. 5). That is, the vehicle on the incoming road moves into the major road if the gap there is big enough. This gap mostly stretches upstream, since the incoming car does not want the next car upstream on the major road to crash into itself. The standard reference for highway engineers, the Highway Capacity Manual [43] claims that drivers accept gaps that correspond to time headways of approximately 5 seconds or more, i.e. the accepted gap needs to be proportional to the speed of the oncoming car. In our standard CA implementation, this would mean that the accepted gap would have to be at least five times the oncoming vehicle's velocity. However, when implementing this rule, it turns out that a factor of three instead of five gives much more realistic flow rates [24]. It is not totally clear why this is the case.

4.5 Queue model

For some purposes, a model with lane changing, signals, unprotected turns, etc., is too complicated, for example because the necessary data is not available, or because a sufficiently fast computer is not available. In such cases, it may be possible to use a simple "queue" model [44, 45], where:

- cars move to the end of a link with free speed;
- once there, they join the end of a queue;
- vehicles from the queue are released with a rate corresponding to the capacity of the link;

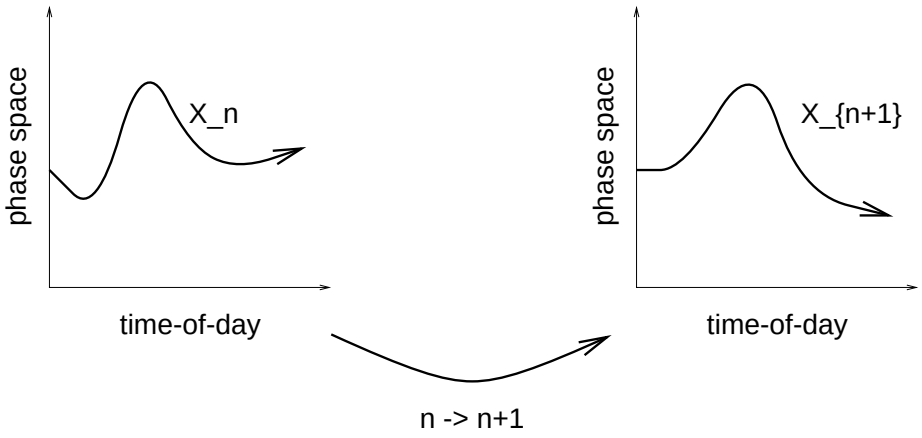

Figure 6: Schematic representation of the mapping generated by the feedback iterations. Traffic evolution as a function of time-of-day can be represented as a trajectory in a high dimensional phase space. Iterations can be seen as mappings of this trajectory into a new one. From [47].

- and they get only released if there is space on the destination link.

The last item is the important distinction to many other models, because it means that links have a storage limit. Once that limit is reached, no more vehicles can be stored on this particular link, and the jam starts spilling back on other links. For more information, see the appendix.

4.6 Other modes

Realistic micro-simulations also need to simulate other modes besides the car, such as busses, light rail, walking, bicycle. As of now, only basic versions of this are being worked on. For example, buses are treated similarly to cars (i.e. they follow a route), with the distinction that every time they approach a bus stop, they move into the right lane and stop there [46]. Light rail is treated similarly as long as it interacts with other traffic; if it does not interact with other traffic, the dynamics is not represented at all and one assumes that it just follows its schedule. Walking and bicycle are similarly assumed to just follow their schedules, which is correct as long as there are no interactions – for example an overloaded subway walkway, or interactions between bicycles and car traffic.

5 Feedback

5.1 Introduction

Results need to be consistent between modules. For example, assume a person expects there will be no congestion and as a result plans quite a lot of trips. If this person encounters severe congestion, he/she will probably plan fewer trips for the following day. We and other groups (e.g. [4, 48, 49, 50]) approach this problem via a learning algorithm (Sec. 5.2). Each individual in the simulation plans his/her day; the plans are executed and performance is recorded; some or all people make new plans; all plans are again executed; etc.

This implies that such iterated simulations can be treated as discrete dynamical systems (Sec. 5.3). A state is the trajectory of the simulation through one day; an iteration is the update from one day (period) to the next (Fig. 6). As such, one can search for things like fix points, steady state densities, multiple basins of attraction, strange attractors, etc. Typically, one would first analyze the steady state behavior, and then the transients.

It is interesting to compare the feedback method to traditional economics and to traditional assignment (Sec. 5.4). It turns out that implementations of traditional assignment do something similar to microsimulation feedback, but for a different reason – and that reason is exactly the economics justification of the method. That is, the traditional assignment assumption is that the system has to be in a Nash equilibrium. In consequence, iterations are just a computational trick to get there.

This leads to a discussion of the behavioral foundation of iterated transportation simulations (Sec. 5.5). For example, if one believes that the steady state describes real systems reasonably well, one could search for computational methods to reach this steady state as fast as possible. This situation would be somewhat similar to traditional assignment. If one, however, believes that the real system relaxes too slowly to ever reach the steady state, one would have to look closely into the transients, plus their behavioral justification.

5.2 Feedback mechanisms

The above feedback mechanism sounds straightforward enough. There are however subtle differences between implementations, with unclear relevance to the results. In this section, we want to discuss some of these differences.

For that discussion, let us make some assumptions about our set-up. Let us assume we have our plans-based micro-simulation as explained above, with no re-planning allowed *during* the micro-simulation. This implies for example that a driver stuck in a jam will *not* take a different route than originally planned to escape the jam; in the worst case, the driver stays in the jam until the computer is switched off. This in itself is already a strong restriction, which needs to be lifted in the near future. Let us further assume that we have run the micro-simulation. Since a micro-simulation computes complete system states on a second-by-second basis, practically all measurable information could in principle

be extracted.

Given this basic set-up, we want to look into aspects of feedback approaches. There are at least three important distinctions between current feedback approaches. One concerns the set of the possible plans, the second the storage of possible plans, the third one the selection of individuals for re-planning. These distinctions will be treated in the next subsections.

5.2.1 The set of possible plans

As we have explained above, each individual operates on a plan. Thus, the easiest implementation is that each individual knows exactly one plan, which is the one that is executed. This is what is currently done in TRANSIMS. During the micro-simulation, for each link, link travel times are averaged into 15-min bins. During the re-planning phase, each re-planning individual simply computes the fastest path based on that information. This can for example be done with a time-dependent Dijkstra algorithm [14], which can compute the fastest path when either the starting time is given or when the arrival time is given. This means that the set of possible routes that could be encountered during the iterations contains every possible route between the starting point and the destination, but the set of possible routes that a traveler has at any given point in time is of size one.

Another approach is that drivers have multiple options, each with a weight, and make a random selection between the options according to their weights. In many implementations [4, 51], the view of the plans here is still global, that is, all individuals have the same information to base their decision on. – This approach is also used for stochastic assignment models of the traditional type [52].

The above methods give a lot of information away that is in principle available from a micro-simulation. For example, the algorithm never compares the performance of the new route to the performance of the previous one. The performance of the new route is the fastest based on the 15-min link travel times, but this does not guarantee that it is faster than the option before for the particular individual. One of the typical examples is a route generator which averages link travel times over the link, i.e. it does not respect the fact that for example turning left against oncoming traffic may take much longer than going straight. An alternative route with fewer left turns, although being considered "longer" by the router, may actually be faster for the individual.

Such aspects could be included into the router. However, they make the router more complicated, and since such problems have a tendency to grow combinatorially in the number of possible options, it may make the problem uncomputable. In addition, this is probably not the way humans really think. It makes therefore sense to base the decision on actual performance for the individual. This would make sure that *any* information being picked up by the micro-simulation would be used for the route selection – and not just the information that survives the 15-min link-long averaging.

Some simulations, e.g. [53, 54], indeed use this agent-based approach. Typically, all of the plans have a weight, but weights belong to individuals. A plan

agent1 time1 link1.1 link1.2 link1.3 ...
agent2 time2 link2.1 link2.2 link2.3 ...

node1 time1.1 link(dest1) link(dest2) link(dest3) ...
node1 time1.2 link(dest1) link(dest2) link(dest3) ...
...
node2 time2.1 link(dest1) link(dest2) link(dest3) ...

Table 2: Plans storage. *Top:* Agent-based plans storage. For each agent, the starting time and the sequence of links is stored. *Bottom:* Network-based plans storage. For each node and each time slice, the next link for each possible destination is stored.

for the next day is picked according to the weights; and weights are updated by performance. A well-performing plan will accumulate more and more weight at the expense of the others. This is sometimes called the classifier approach (e.g. [55]).

According to the above example, one of the challenges here is to generate "reasonable" alternatives. Such plans do not necessarily look optimal from the perspective of the route generator, but as explained above they may turn out to be useful from the perspective of an individual. Some of this was discussed in Sec. 3.4.

All of the above examples concern route plans. However, the same arguments are true for activities plans.

5.2.2 How plans are stored

It may seem like a purely computational issue how plans are stored, and to a certain extent it is. However, any choice of implementation makes certain approaches easy and other ones hard, and so does this one.

The way we have explained it so far, one will probably assume that each individual has computational memory to store his/her plan or plans. The memory requirements for this are of the order of $O(N_{people} \times N_{trips} \times N_{links} \times N_{plans})$, where N_{people} is the number of people in the simulation, N_{trips} is the number of trips a person takes per day, N_{links} is the average number of links between starting point and destination, and N_{plans} is the number of plans that are kept in memory. For example, for our Portland simulations with the 200 000 link network, $N_{people} \sim 1.5$ mio, $N_{trips} \sim 3$, and $N_{links} \sim 300$, which results in

$$1.5 \cdot 10^6 \text{ persons } \times \text{ 3 trips per person } \times \text{ 300 links per trip}$$

$$\times \text{ 4 bytes per link } = \text{ 5.4 GByte}$$

of storage if each individual keeps exactly one plan in memory, and we use 4-byte words for storage of integer numbers.

Since this is a large storage requirement, many approaches do not store plans in this way. They store instead the shortest path for each origin-destination combination. This becomes affordable since one can organize this information in trees anchored at each possible destination. Each node in the network has "signs" for which way to go for any possible destination; a plan is thus given by knowing the destination and following the "signs" at each intersection. The memory requirements for this are of the order of $O(N_{nodes} \times N_{destinations})$, where N_{nodes} is the number of nodes of our network, and $N_{destinations}$ is the number of possible destinations. Traditionally, transportation simulations use of the order of 1000 destination zones, and networks with of the order of 10 000 nodes, which results in a memory requirement of

$$1\,000 \times 10\,000 \times 4 = 40 \text{ MByte} ,$$

considerable smaller than above.

The problem with this second approach is that it explodes with more realistic representations. For example, in our Portland simulations we replace the traditional destinations zones by the links, i.e. each of the 200 000 links is a possible destination. Similarly, that network has of the order of 100 000 nodes. Last, we need the information time-dependent. If we assume that we have 15-min time slices, this results in a little less than 100 time slices for a full day although some of the information can be recycled [56]. The memory requirements for the second method now become

$$200\,000 \cdot 100\,000 \cdot 100 \cdot 4 = 8\,000 \text{ GByte} ,$$

which is now clearly more than the first method and also more than most systems can currently handle. Note that the first method is unaffected by changes in the temporal resolution.

In consequence, when moving to an agent-based view of transportation simulations, it seems both computationally advantageous and conceptually straightforward to store plans in an agent-based way.

In both cases it is possible to compress the relevant information by a factor of at least 30 [57]. This is achieved by instead of encoding the sequence of links, one just encodes the turning directions, like "straight", "left", or "right". Assuming that there are not more than 8 possible turning directions, one could encode this in 3 bits instead of 4 bytes, resulting in a factor of ten of compression. Further compression can be achieved by noting that in most cases people go straight; and twenty straights in a row can be encoded in a more compressed format than twenty 3-bit sequences.

5.2.3 Selection of re-planning individuals

Another aspect of re-planning is how to select the individuals for re-planning. One possibility is to simply select all of them [59, 51]. This however needs to be done with care: It is easy to introduce systematic oscillations. In one iteration, during re-planning one road looks empty, thus everybody picks it, thus it will be

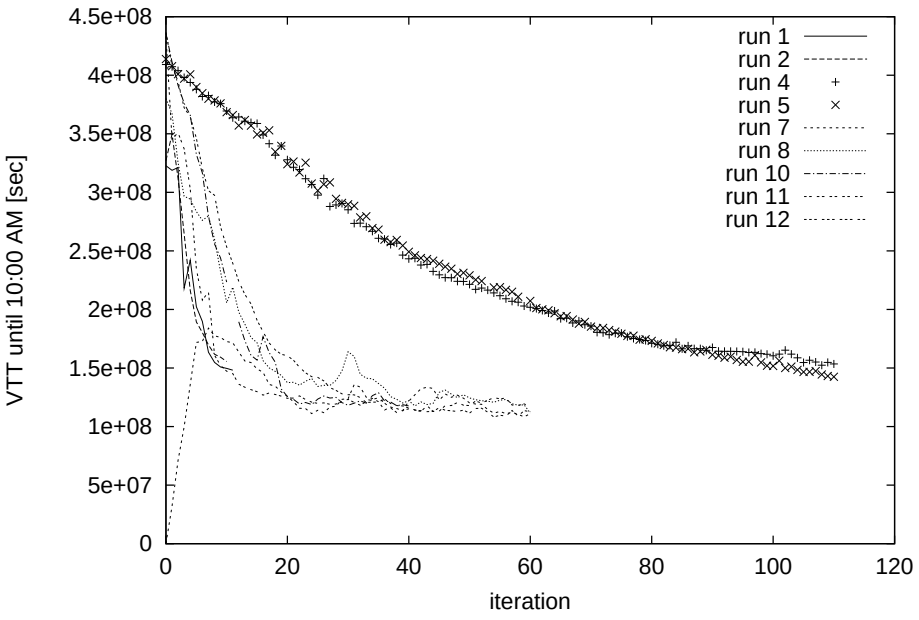

Figure 7: Relaxation of the iteration process for different selection schemes. VTT means "Vehicle Time Travelled", i.e. the sum of all travel times of all trips. From [19, 58].

full and the other one empty, etc. In consequence, such approaches usually select routes probabilistic according to the weights. Options with equal performance should obtain equal weights, and in consequence each one should be selected by 50% of the population, thus avoiding fluctuations.

Weights can be obtained from the classifier system (see above), which means that one needs a behaviorally justified update rule for the weight since it decides about the distribution between options. Another option is to use the same discrete choice theory that was described for the activities (Sec. 3.3.2). One well-known draw-back of such an approach is the failure to group options together (as has already been described in Sec. 3.3.2): If one has three identically performing routes, two of them nearly identical and the third one radically different, then discrete choice theory will give equal weights to all three of them, instead of the behaviorally reasonable solution of giving 25% to each of the first two and 50% to the third. Attempts to overcome this are under way [18].

Another option is to only re-plan a fraction of the population. This option is currently being used in TRANSIMS. It has, however, the same problem with the systematic oscillations since it does not matter if they are caused by all of the population or just a fraction of it. To overcome this problem, TRANSIMS currently "forces" its re-routing to relax by decreasing the fraction of re-planned

individuals with increasing iteration number. A fraction of 5% or less seems to not cause significant oscillations in most cases with a sufficiently stochastic micro-simulation.

If only a fraction of the population is re-planned, one thing that is important is to go through it in some organized way. Otherwise, it easily happens that even after many iterations a portion of the population has never been re-planned, which means that they are still operating on their initial plan. Since these plans are probably inconsistent with the actual situation, they will be changed once these individuals come up for re-planning, and as long as that has not happened, the simulation is not "relaxed" (although this may be realistic; see Sec. 5.5).

One possibility to go through the population in an organized but not absolutely deterministic way is to do "age-dependent" re-planning. Here, the probability of a plan being picked for re-planning increases with the number of iterations since its last re-planning. Technically, we use $p(a) = C \cdot a$ as the probability for a person to be picked for re-planning, with a being the number of iterations since the last re-planning for this person. Independent of the initial age distribution, this converges towards a steady state distribution $f(a)$, and the relation between C and the re-planning fraction f_r in the steady state is [19, 60] $C = (\pi/2)\, f_r^2 \exp[(\pi/4)\, f_r^2]$. All this means that by setting C, one can determine the steady state re-planning fraction; and by setting a at the beginning to a high value, one can force higher re-planning fractions for the first iterations. This method achieves relaxation within 20 iterations or less, see Fig. 7. For more information, see Refs. [19, 58].

5.2.4 Discussion of feedback mechanisms

It is unclear in how far the different methods lead to different results. Most importantly, it is unclear if they lead to different or the same steady state. Clearly, the steady state is only defined for a given re-planning operator. But how different can these operators be until the resulting traffic states are significantly different?

Also, re-planning of all travelers is not computationally efficient. As long as the routes are stored in the network-based way as explained above this is not much of a problem, since routes are calculated for all possible origin-destination pairs anyway, and selecting between these routes is computationally fast. If one however moves to the agent-based view, this is no longer true, and re-planning 10% of the population is ten times faster than re-planning all of it. And remember that the network-based view is not feasible for large enough problems.

Last for this discussion, note that the discrete choice approach is nothing but a noise parameterization. The assumption was that people always choose the option with the highest utility; deviations from this are explained by "unobservable/unobserved" attributes, and in order to make theoretical progress, these are assumed to be randomly distributed. Thus, one may ask the question if one could not replace the noise parameterization by noise itself. For example, our micro-simulation generates different scenarios for each different random seed,

each one potentially resulting in a different fastest route between two points. Thus, a fixed re-planning fraction operating on such a stochastic simulation will generate a *distribution* of paths between any two points even for a deterministic routing algorithm. One should probably compare this distribution when generated "directly" by the simulation noise with one generated by discrete choice theory.

5.3 Dynamical system approach

It is instructive to see such iterated transportation simulations as dynamical systems rather than as assignment problems. These dynamical systems operate on two time-scales, which should not be confused:

- The simulation usually generates a traffic dynamics that depends on the time-of-day. For example, a possible output of such a micro-simulation may be link delays as a function of when vehicles enter the link.
- The iterations generate "periods", often called "days" because, in some sense, one watches a day-to-day evolution of the co-evolution problem where everybody adapts their decision based on what everybody else does (or did in the past). This day-to-day evolution provides a mapping into itself. For example, route plans get executed by the micro-simulation, and based on this, a new set of route plans is generated.

It is the second process (the process of iterations) that we are interested in here.

Let us introduce some terminology. If one looks at the evolution of the system as a function of time-of-day, it can for example be described as a trajectory in $\sum_{i=1}^{N} f_i$ dimensions, where N is the number of particles (vehicles, travelers, "agents", ...) and f_i is the number of their degrees of freedom (position, speed, ...) of the ith particle. The trajectory of this system is thus an object with $1 + \sum_{i=1}^{N} f_i$ dimensions, where the additional dimension is the time-of-day dimension. If the iterations would reach a fix-point, then the trajectory of the system at iteration $n + 1$ would exactly lie on top of the trajectory at iteration n. Thus, if one wants to include the dynamics of the transportation system in the description, one needs to look at mappings in this $(1+\sum_{i=1}^{N} f_i)$-dimensional space (Fig. 6).

It can be proven [61] that such an approach under fairly general conditions will eventually run into a steady state, in the sense that for each individual there will be a stationary probability for each possible plan.

For the moment, most research groups look at the steady state. However, even here research is necessary. For example, the mathematical proof does not say anything about the necessary number of iterations to reach the steady state. It also does not say anything about the possible occurrence of "broken ergodicity" [62], i.e. the property of a system to be mathematically ergodic but to remain in parts of the phase space for long periods of time. An Ising magnet of finite size below the critical temperature would be an example: It would

erratically move back and forth between positive and negative macroscopic magnetizations, with the times between transitions becoming longer with lower temperature and increasing system size. In practice, however, it seems that iterative routing is reasonably well-behaved – we have never found that using a different selection algorithm or a different initial condition leads to different traffic patterns [19, 58].

What we have found, however, is that the microsimulations themselves can fluctuate a lot from one realization to the next. That is, by just changing the random seed of the micro-simulation, one can get very different traffic patterns. Note that this is *not* referring to a different re-planning series, but to a single micro-simulation run with everything, including plans and signal phasing, being the same. The immediate consequence of this is not surprising for people with a Statistical Physics training, but it is still worth mentioning because it is not universally accepted everywhere: Results of stochastic simulations are distributions, not single numbers. In addition, our research indicates that these distributions can be non-Gaussian. This implies that average values may actually be misleading since they can never happen; in our view it is important to keep these things in mind for policy applications of our simulations.

5.4 Traditional assignment and economics

Some implementations of traditional traffic assignment [52] look in praxis very similar to an iterated simulation: Origin-destination (OD) streams are assigned to routes, the resulting link costs (link travel times) are computed, some or all streams are re-assigned, etc. The main technical difference is that the computation of link travel times is much cheaper in traditional assignment than it is in iterated transportation simulations: They are a direct function of flow demand (volume demand) for that link. Since flow demand on a link is just the sum of all streams going via that link, this is a straightforward calculation.

One needs to keep in mind, though, that the trade-off for this simplicity is unrealistic traffic dynamics. The whole approach is only consistent when everything is static, i.e. does not depend on the time-of-day. Then the approach indeed functions very much like a voltage and current calculation in a network with nonlinear resistances, except that the particles in traffic have destinations. Such an approach cannot represent time-of-day dynamics, for example congestion built-up during the morning peak, or the inversion of flow direction from morning to afternoon peak. In consequence, what people have done is to run different assignments for, say, morning and afternoon peak. One cannot, however, make the time slices arbitrarily short.

The assumption behind static assignment (and we will use the word "static" also for methods that do several such assignments for different times-of-day) is the assumption of a Nash or User Equilibrium. In such an equilibrium, no user (or traffic stream) can be better off by unilaterally switching routes.

For the static assignment, under fairly general assumptions one can show that there is only one solution to this problem. Note that this uniqueness is true in terms of the link flows and thus in terms of the link delay, *not* in terms

of the route assignment. Usually, many different route assignments can lead to the same link flow/link delay result. Since we are typically most interested in congestion, which is represented by link delays, this is still very helpful.

Since the result is unique, *any* computational technique that generates this result is valid. Thus, for static assignment the choice of computational techniques is a question of computational resources and desired level of sophistication, but it is irrelevant for the result. It turns out, though, that the most-used techniques to obtain this result are relaxation techniques which resemble very much our iterated micro-simulations: assign routes; calculate link delays; reassign routes; etc. Note, though, that in many similar problems in operations research, the solution can be computed non-iteratively. Thus, it becomes important to remember that the iterations here have no behavioral meaning – they are just a computational method to reach the equilibrium state. Only the equilibrium state has a behavioral justification in the sense that it is assumed that each traveler in a traffic system selects an option for her/himself that cannot be unilaterally be improved upon.

5.5 Steady state vs. transients, behavioral foundation, and time scales

In iterated simulations, the situation with respect to the behavioral justification is different from static assignment. Although most of us currently look at the steady state behavior, the justification of what we do is in the claim that there is some realism in how our agents change plans from one iteration to the next. For example, for the classifier system approach, the agents choose the plan that has performed best in the past, and we assume that this is some reasonable representation of human behavior.

As long as we do not say anything about the time scale of switching between plans, this is still a statement that would just prescribe the dynamics for the steady state situation. If, however, we start being interested in transients, it is no longer enough to assume in what kind of behavioral situation people eventually end up; we need to have rules of how they switch from one behavior to the next. The interesting point is that our simulations are now capable of addressing this problem in a real world context; however, much work will be needed until this problem is solved.

Part of the problem concerns the different time scales in the system. For example, human drivers make decisions about lane changes on the time scale of several seconds. Decisions about routes are made on time scales from minutes to hours. Decisions about modes are made maybe on a daily time scale. And decisions about activities are, as already stated earlier, made on time scales between minutes and months.

This implies that any iterated system that claims to represent human behavior needs to represent these time scales. One –simple– way to represent different time scales is to assume that certain choice behaviors are associated with certain typical time scales. For example, one could assume that route choice is done on time scales of the order of a few days, whereas activities choice

is done on roughly weekly time scales. One way to implement this would be that parts (or all) of the population re-plan their routes every night, whereas re-planning of activities is only allowed for a fraction (or all) of the population every five iterations. This is indeed the approach that we have selected in our first implementation of such an activities feedback ([63], see in the appendix for more information). However, different time scales can also be implemented by other means, for example by always allowing for re-planning of both activities and routes, but with different probabilities or rates. For example, if the 5% of the population re-plan their routes but only 1% of the population re-plans their activities over night, then the route re-planning operates on a time scale five times faster.

The challenge here is to both find methods that lead to fast relaxation and to bring real human behavior into this.

6 TRANSIMS scenarios

Secs. 7 and 8 refer to specific TRANSIMS scenarios. In order to avoid repetition, this section here is meant to give a short description of the scenarios, and which version of the modules were used.

6.1 The "Dallas scenario"

TRANSIMS, as part of its development, works on realistic case studies. The first such case study was done using data from the Dallas/Fort Worth area in Texas [64]. Since at that time most of the demand generation modules were not yet functional, the study used the same origin-destination matrices that the regional planning authority uses for their studies. From these matrices, individual trips with starting times were extracted. From here on, TRANSIMS modules were used: in the routing, in the micro-simulation, and in the feedback. The study used a so-called focused road network, which means that for a 5 miles times 5 miles study area *all* streets were represented, whereas with further distance from the study area more and more of the less-important streets were left out. This network contained 9864 nodes and 24 622 uni-directional links, of which 2276 nodes and 6124 links were in the study area. The micro-simulation (supply simulation) ran on the study area only, while the the other modules and information referred to the whole Metropolitan area. To make this work, routes from the router were "clipped" to the study area before they were fed into the micro-simulation. For more information, see, e.g., Refs. [64, 65].

For certain studies, a micro-simulation different from the TRANSIMS micro-simulation was used. It is called PAMINA, for PArallel MIcroscopic Network Algorithm [19, 66]. Both micro-simulations are based on the cellular automaton technique. However, the TRANSIMS micro-simulation includes things such as turn pockets, correct signal timings, lane-changing for plans following, all of which were not implemented for PAMINA.

6.2 The "Portland scenario"

Portland (Oregon) is the site of the next TRANSIMS case study. Since this study is scheduled for the year 2000, no results are available yet. The project is however already in the stages of acquiring the necessary data. We used that data, together with research versions of TRANSIMS, for a research study. This study is extremely preliminary – it uses a simple home-to-work demand generator, time-dependent fastest path routing, and a simple queue micro-simulation. In fact, the set-up used for this study is described in the appendix, as an example of a simple, but complete "toy" version of a transportation simulation package.

The network for this study contains about 20 000 uni-directional links. It is the same network that the Portland transportation planning authority (Portland METRO) uses for their traditional assignment. Some predictions refer to a 200 000 link network for the same city. It is planned to use this larger network for the Portland TRANSIMS case study. This network is derived from the census (TIGER) network and contains *all* public roads in Portland.

7 Analysis

7.1 Validation

Models need to be validated. This may be less important for models that are meant to give insight into the dynamics or "physics" of a certain system, but it is certainly necessary for models on which political decisions will be based.

Unfortunately, at least in our view a validation of complex transportation simulation systems is far from easy. This starts with the fact that the definition of "validation" itself is unclear for complex enough systems. In general, one would probably declare a model "valid", if it reproduced reality within certain error thresholds. Simulation systems as discussed in this paper are however too complex for this approach. The problem is that one needs to decide beforehand which aspects should be validated. For example, when using a cellular automata approach for driving, then the driving behavior itself can most probably not be validated, because the driving dynamics is too unrealistic. However, a fundamental diagram can potentially be "valid".

Another problem is that many modules of our simulation system cannot be validated independently way because measurements of the necessary quantities are problematic. For example, not enough is known about routing decisions of real people.

We prefer to use quantities that are actually available from field measurements. These quantities are currently usually something like hourly flow rates. Figs. 8 and 9 show comparisons between simulation and reality for the "Dallas scenario" (cf. Sec. 6.1) and the "Portland scenario" (cf. Sec. 6.2).

The results from the Dallas comparison can be summarized as follows (see Refs. [67, 47]):

Figure 8: Comparison of approach volumes between simulation result and reality. The lengths of the black bars denote reality; lengths of the gray bars come from the simulation. The data going into this comparison is not entirely consistent, see [67]. From [67].

- Our results are not obviously worse than the results obtained by the Dallas regional planning authority using their assignment model.
- Switching from the realistic TRANSIMS micro-simulation to the much more unrealistic queue simulation described in Secs. 4.5 and in the appendix does not make the results significantly worse. This indicates that deviations from reality are probably caused to a larger extent by the demand generation, including the routing, than by the transportation micro-simulation.

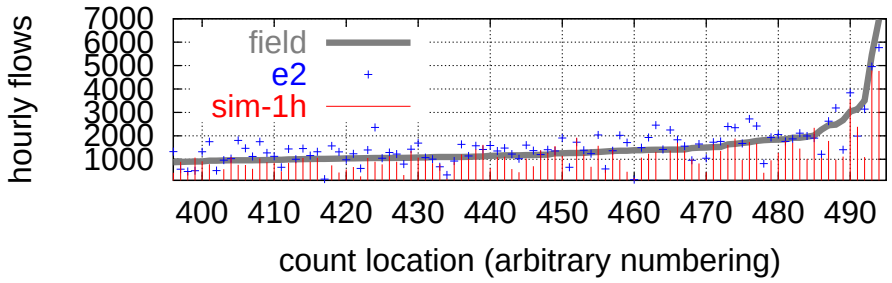

Figure 9: Comparison of approach volumes between: gray bar: reality, spikes: our simulation, +-signs: Portland traditional assignment. From [68].

In Portland, we also generated our own demand. That module is extremely simplified; it just consists of an assignment between workers and workplaces, thus generating home-to-work trips. We assume that such trips contribute about 80% of all traffic during the morning peak. The results from this study can be summarized as follows:

- In contrast to the results obtained by the Portland regional planning authority, our results have a bias towards too low flows.
- However, apart from this systematic bias, in many other statistical aspects our results are not worse than those obtained by Portland.

We conclude, from these preliminary results, the following:

- Our methods are capable of generating real world scenarios, which can be compared both to field measurements and to results from the traditional methods.
- So far, the traditional methods perform better than our methods, but not overwhelmingly so. However, only a tiny portion of the capability of the simulation methods has been explored.
- Current research should be focused in the areas of demand generation, not the transportation micro-simulation.
- Systematic comparison studies are feasible given current technology, but they are hard. For example, in the TRANSIMS project, resources are already strained by getting the real-world data into the micro-simulation at all; few resources are left for systematic comparisons with field measurements.

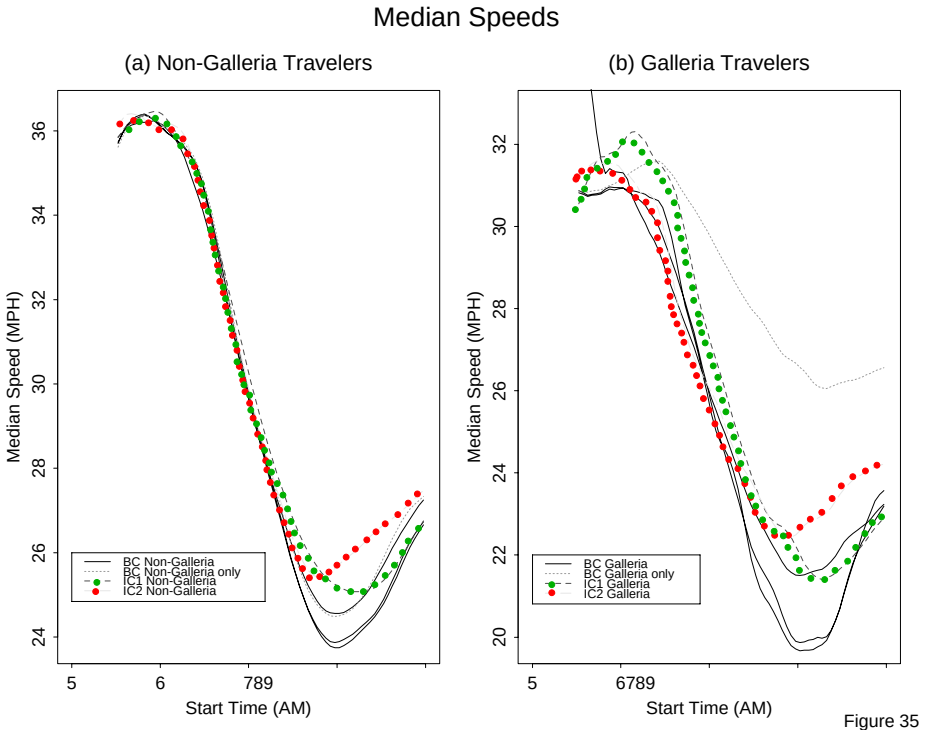

Figure 10: Example of stake-holder analysis for Dallas. Shown in both plots is the median speed as a function of the time-of-day, on the right for a certain subpopulation with destinations inside the simulation area, on the left for people traveling through the simulation area. Different lines in the plots refer to different runs. One sees that the different population groups benefit in different ways from different scenarios. From [64].

7.2 Stake-holder analysis

Some important reasons to base transportation planning on a microscopic approach are: (i) The representation of the dynamics is more correct. (ii) It is straightforward to do analysis by arbitrary sub-populations, since the travelers retain their demographic characteristics throughout the simulation. (iii) Microscopic simulation is more convincing especially to non-scientists.

These arguments are interwoven. For example, it would be perfectly possible to retain characteristics of individual travelers with the traditional four-step process. However, this would not be very useful. For example, say we are looking at a morning peak traditional assignment. The method assigns steady-state traffic streams between origins and destinations. Our traveler would thus be randomly spit out at his/her origin according to the necessary departure rate, and it would be impossible to base, say, departure time choice on behavioral

characteristics. Also, a representation of trip chaining (e.g. going shopping on the way home) is impossible since for the traffic stream representation one has to assume that all trips are executed simultaneously. Thus, if one wants to be able to differentiate subgroups according to demographic criteria, one first needs to make sure that the method really enables these subgroups to behave in different ways. A more realistic, microscopic, representation of the dynamics currently seems to be the most straightforward conceptual way to achieve this. Thus, even if the results currently obtained by the new microscopic methods are not more realistic than the results obtained by the traditional methods, the microscopic methods are nevertheless a necessary step forward because only they will enable us to do the analysis that we are really interested in.

Analysis by sub-populations is sometimes called stake-holder analysis, since it assigns costs and benefits of infrastructure changes to these subgroups instead of to the population in general. As a result, it is possible to identify winners and losers of certain changes, instead of just looking at overall benefits. This will enable planning organizations and politicians to target infrastructure changes much more directly, for example according to social equity, or according to the voting structure of an area. An example of a stake-holder analysis, for a Dallas-Fort Worth case study with TRANSIMS [64], is shown in Fig. 10.

7.3 Emissions

A certainly very important element of transportation planning are emissions from the transportation system. However, for emissions it is very important to have a realistic representation of the vehicle dynamics. For example, it is very different for emissions if the same flow rate is achieved with everybody driving at homogeneous and fairly low speed, or with many people accelerating violently from a near stand-still to freeway speeds. Thus, region-wide microscopic traffic simulations make it possible to calculate realistic region-wide emissions. Such work is in progress at several places (e.g. Refs. [69, 70, 71]).

8 Computational issues

8.1 Introduction

Current transportation micro-simulations of large metropolitan areas, even when implemented on powerful parallel supercomputers with 256 CPUs or more, run no faster than 20 times faster than real time. This means that, on such machines, a 24-hour period of the system can be simulated in a little more than 1 hour of computer time. Assuming that we need 100 iterations for one useful result, and even if we assume that the demand *generation* modules use negligeable computer resources, this still means more than 100 hours of computer time on a large supercomputer to obtain one such result. Clearly, iterated transportation simulations are still a computational challenge, and a diligent analysis of the resource utilization is necessary.

8.2 Transportation Micro-simulation

As of now, the transportation micro-simulation can easily be the most compute intensive part of a transportation planning simulation system. It was indeed the Cellular Automata (CA) approach to traffic micro-simulation which helped to convince many people that it was possible with current computers and algorithms to perform realistic micro-simulations of complete metropolitan regions. Even today, there are only two better-known micro-simulations which can treat systems of this size and with this level of realism: TRANSIMS [2] and PARAMICS [72].

In general, we know the following and more:

- Any transportation micro-simulation needs to simulate links (roads) and nodes (intersections). For pure CA-based links, the typical computational speed is 100 000 vehicles in real time on a workstation or Pentium CPU. This number has not changed much over the last couple of years – improvements in hardware get used up by making the algorithm more realistic. Note that the CPU time spent for link dynamics depends, besides on the number of vehicles, also on the length of the link. The value given above assumes a density of 0.1 vehicles/cell or approx. 13 veh/(lane km).

- When composing road networks, the above number decreases since intersections use up computer time. For a network with 20 000 links and 8 500 nodes of Portland/Oregon and 50 000 vehicles in the simulation, the TRANSIMS micro-simulation runs a little more slowly than real time.

- When moving to a network with 200 000 links, as planned, we expect to lose about a factor of five in computational speed. Note that the number of vehicles in the simulation is not changed when moving to this realistic network, and so most of the additional computation is due to the higher number of nodes (intersections).

- We have ported some of our simulations to parallel computers. The parallelization method of choice was domain decomposition. In order to keep the complexity of the parallelization as low as possible, we introduced our boundaries between the domains in the middle of links. This means that we actually distribute the nodes of the the road network, with their attached "half-links" across the CPUs of the parallel computer. Obviously, with this approach we cannot use more CPUs than our network has nodes; in practice, the approach becomes inefficient much earlier due to load imbalances. An investigation of just this aspect leads to the prediction that the 200 000 links network can run efficiently on up to 1024 CPUs [73, 74].

- We use dynamic load adaptation between feedback iterations. That is, the computational loads on the road network are recorded during one iteration, and the domain decomposition is redone at the start of the next iteration. This results in significant speed-ups, especially in situations with high load imbalance, for example when most of the traffic is in some core region of the area [19, 74].

- Taking all this knowledge together, we expect that one would be able to simulate the 200 000 link Portland situation about 20 times faster than real time on a 500 CPU supercomputer. Unfortunately, we are not able to test this prediction.

For more details on computational results, see [75, 73, 19, 74].

The above computational speeds may sound fast, and compared to what was possible a couple of years ago they are. However, as stated in the introduction to this section, typically it is necessary to run through the micro-simulation about one hundred times in order to obtain a relaxed result via feedback iterations. Assuming we wanted to look at a 24-hour time period, this would imply more than 100 hours of continuous computing on a 500 CPU supercomputer for the micro-simulation part alone. Re-planning (i.e. changing of activities and/or routes) is, although also demanding, usually less compute-intensive than the micro-simulation.

9 Conclusion

Agent-based transportation planning simulations are possible with current technology. Such simulations typically keep track of several million people and several hundreds of thousands of street links. For transportation planning, it is necessary to not only model the traffic dynamics, but also the process of demand generation. A promising approach is "activities-based demand generation". Each individual in the simulation plans the activities, such as sleeping, working, eating, shopping, for his/her day. Activities at different locations need to be connected by transportation, which generates travel demand. A routing module decides by which mode and via which route to do these trips. A realistic micro-simulation executes all plans and computes congestion and bottlenecks.

Iterations between these modules are necessary to make them consistent. This means that the micro-simulation needs to run up to one hundred times for a useful result, which makes the problem a considerable computational challenge. Simplified versions of the approach produce results in about two days of computing on a workstation; more realistic versions need a supercomputer and more time.

Such simulations should prove useful not just for transportation planning, but for all issues where a detailed knowledge of the urban system is useful. This includes general urban planning questions beyond the transportation planning; it includes emergency planning; and it includes marketing and location choice problems for companies. We believe that in the future centers that have access to both the necessary data and the simulation capability to run such realistic scenarios will play an important role in addressing many of these problems.

10 Acknowledgments

We thank the TRANSIMS team, especially Chris Barrett and Dick Beckman as technical directors, for providing the technological infrastructure to make this work possible. Peter Wagner and his group in Cologne have provided uncountable insights into many subjects. Michael Schreckenberg and Dirk Helbing helped us to stay abreast of theoretical developments, among many other things. The ITS group at MIT, especially Moshe Ben-Akiva, Michel Bierlaire, Jon Bottom, and Didier Burton, provided much insight from personal discussions. Last, but maybe most importantly, Bill Stein, Dick Walker, and Keith Lawton, at different positions at the Portland Metropolitan Planning organization, and Ken Cervenka from the Dallas/Fort Worth Metropolitan Planning organization, provided all the data and the practical insights, without which much of this work would never have been possible.

A A simple but complete example of a transportation planning simulation system

In this section, we want to give an example of how a simple but in principle complete simulation package for transportation planning applications could look like. Such an application will have the following modules: data base, activities generator, mode and route planner, and a micro-simulation.

A.1 Data base

Transportation simulations need to deal with real world scenarios to be useful. In order to achieve this, it makes sense to write them so that they can read in arbitrary real world configurations, even when the initial intention of the project is to use artificial data.

For this example case, the minimum content of the data base is some information about the road network, and some information about where people live and where people work. The information about the road network is typically given by two files: a file of nodes, and a file of links. The node file typically contains:

- a unique ID number for each node, and
- geographical coordinates.

Additional information can be added for each node, but is not needed for this example. – The link file for this example needs the following information:

- a unique ID number for each link,
- the ID number of the node where the link starts,
- the ID number of the node where the link ends,

- length of the link,
- number of lanes,
- capacity (number of cars that can leave the link per hour, e.g. restricted by a traffic light), and the
- number of workplaces on the link.

In addition, we will have a file that contains the synthetic population for the area. For this example, this has the following entries:

- a unique ID number for each synthetic person,
- the link ID of the link where the person lives,
- an entry that describes if the person is employed or not, and
- the starting time for going to work.

Such a population can for example be generated from census data [6]. – In addition, we need the acceptance function for work trip times, see below.

A.2 Activities planner: Assign work locations to workers

As the first module in our example, we need the activities planner. In general, this module plans, for each synthetic individual, a sequence of activities with locations for the day.

In our example here, we will only look at two activities [63]: (i) Be at home before work. (ii) Work. If the workplace is at a different location than the home, this generates a trip. We will assume that the times when people leave their homes is included in the population information. Then the only thing that is left is to assign workplaces to workers. This is done by observing that workplace selection is influenced by the fact that, all other things being equal, people usually do not want to have really long trip times to get to their jobs. This can be expressed by a function which decreases with increasing travel time, for example $p_{accept}(\tau) \propto \exp(-\alpha\tau)$ with $\alpha > 0$. Thus, one can design a simple algorithm:

- For all working members of the population do:
  - Weigh all workplaces with $p_{accept}(\tau)$. This depends on the travel time τ which would be necessary for this particular individual to get to the workplace.
  - Make a random selection between all the workplaces according to their weights.

  End for.

This assigns a workplace to each worker. Note that the selection depends on the travel time τ, so if congestion increases the travel time, this algorithm is sensitive to this fact.

It should be noted that, for really realistic applications, this method is *much* to simplistic. For example, people certainly do not select their workplaces depending on the travel time only; the algorithm does not keep track of over- or underutilization of the working locations; and in many cases, people first know their working location and then select their homes accordingly.

However, in order to have a somewhat useful result with a simple model, the above algorithm is probably the best chance one has. It should describe between 80 and 90 percent of the early morning rush hour traffic, until until other traffic such as delivery truck traffic sets in.

A.3 Mode and route planner

Next, one needs to connect home and work locations by transportation. For the sake of simplicity, we only look at the car mode, which describes 80 percent or more of all such travel in most western cities. In addition, we assume that travelers are optimizing but non-predicting, and somewhat lazy. This means that in our model they revise their route only every couple of days, and they select the route that would have been fastest on the previous day. Technically, this is done by remembering link travel times from the last day/iteration, and running a time-dependent fastest path algorithm (Dijkstra, [14]) on this information. We usually collect link travel times in 15-min time bins.

The result of this module is a list of the following information for each traveler:

- Unique traveler ID according to the population file.
- Starting location link ID.
- Time when leaving this location.
- Sequence of links to follow in order to reach the destination. Instead of links, one can use nodes.
- Ending location link ID.

This is called a route plan.

A.4 Micro-simulation

The micro-simulation simply executes the route plans. The minimum functionality of this module is to read in the road network information and the route plans, and to calculate time-dependent link delays. In this example, we use a simple queue model (see Sec. 4.5) – a model which is a simple implementation of queues as in queuing theory, with one important difference: there is a size

limit on the number of vehicles that can sit on a link, and this translates into a queue length limitation.

In this model, links are characterized by: flow capacity, storage capacity, and free speed travel time. Storage capacity is computed as $L \times N_{lanes} \times \rho_{jam}$, where L is the length of the link, N_{lanes} is the number of lanes, and ρ_{jam} is the jam density. For consistency with the cellular automaton approach, we use 1/7.5 m for the jam density. Free speed travel time is computed as L/V_{free}, where V_{free} is the free speed on the link.

The algorithm works as follows:

- FOR all links DO:
  - WHILE (vehicles can still leave in this time step according to the capacity) DO
    - Look at the first vehicle in the queue.
    - If the free flow speed arrival time (see below) is larger than the current time, then break and go to the next link.
    - Check if the destination link has space. If not, break and go to the next link.
    - Calculate the free flow speed arrival time on the end of the next link:
      Arrival time = Current time + length/freeFlowSpeed
      (length and freeFlowSpeed of the destination link)
    - Insert vehicle at the end of the destination queue

    END WHILE

  END FOR

The precise condition for "can still leave in this time step according to the capacity" is

$$N_{link} < int(C_{link}) \text{ OR } [\ N_{link} = int(C_{link}) \text{ AND } rnd < C_{link} - int(C_{link})\ ] \ ,$$

where C_{link} is the capacity of the link in vehicles/second, N_{link} is the number of vehicles which already left the link during the same time step, rnd is a random number between 0 and 1, and $int(x)$ is the integer part of x.

Note that the simulation runs on pre-computed route plans, as explained above. Such a simulation can become "stuck" or grid-locked, for example when a loop of full links forms, and the first car on each of these links wants to move into another of these full links [66, 65]. In order to prevent this, we remove vehicles that are first in a queue and have not moved for T_{wait} time steps of the simulation, where T_{wait} is for example 300 seconds. In the iterative procedure, many such vehicles get removed in the first iterations, but their number usually becomes less than 0.5% in later iterations.

A.5 Feedback

The three modules need to be coupled via feedback to obtain a consistent solution. For illustration purposes, one could do 10 iterations where one re-plans the routes for a randomly selected 10% of the population, followed by one iteration where one re-plans the activities for a randomly selected 20% of the population. More sophisticated schemes can lead to faster relaxation, and one should keep in mind that the relaxed solution is not necessarily the behaviorally correct one (see Sec. 5).

References

[1] SMARTEST (Simulation Modelling Applied to Road Transportation European Scheme Test. See http://www.its.leeds.ac.uk/smartest/.

[2] TRANSIMS, TRansportation ANalysis and SIMulation System, Los Alamos National Laboratory, Los Alamos, U.S.A. See http://transims.tsasa.lanl.gov.

[3] A. Schadschneider and D. Chowdhury. Statistical physics of vehicular traffic, in preparation 1999.

[4] DYNAMIT, see http://its.mit.edu.

[5] See http://www.census.gov.

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

[7] D. Lohse. *Verkehrsplanung*, volume 2 of *Grundlagen der Straßenverkehrstechnik und der Verkehrsplanung*. Verlag für Bauwesen, Berlin, 1997.

[8] E. Cascetta, D. Inaudi, and G. Marquis. Dynamic estimators of origin-destination matrices using traffic counts. *Transportation Science*, 27(4):363–373, 1993.

[9] M. Ben-Akiva and S. R. Lerman. *Discrete choice analysis*. The MIT Press, Cambridge, MA, 1985.

[10] Th. A. Domencich and D. McFadden. Urban travel demand. In D.W. Jorgenson and J. Waelbroeck, editors, *Urban travel demand*, number 93 in Contributions to Economic Analysis. North-Holland and American Elsevier, 1975.

[11] J. L. Bowman. *The day activity schedule approach to travel demand analysis*. PhD thesis, Massachusetts Institute of Technology, Boston, MA, 1998.

[12] S. T. Doherty and K. W. Axhausen. The developement of a unified modelling framework for the household activity-travel scheduling process. In *Verkehr und Mobilität*, number 66 in Stadt Region Land. Institut für Stadtbauwesen, Technical University, Aachen, Germany, 1998.

[13] A system of activity-based models for Portland, Oregon, Draft final report, 1997.

[14] R. R. Jacob, M. V. Marathe, and K. Nagel. A computational study of routing algorithms for realistic transportation networks. *ACM Journal of Experimental Algorithms*, in press. Also Los Alamos Unclassified Report LA-UR 98-2249, see http://www.santafe.edu/~kai/papers.

[15] C. L. Barrett, R. Jacob, and M. V. Marathe. Formal language constrained path problems. Los Alamos Unclassified Report (LA-UR) 98-1739, Los Alamos National Laboratory, see http://transims.tsasa.lanl.gov, 1997.

[16] T. Kelly and K. Nagel. Relaxation criteria for iterated traffic simulations. *International Journal of Modern Physics C*, 9(1):113–132, 1998.

[17] D. Park and L. R. Rilett. Identifying multiple and reasonable paths in transportation networks: A heuristic approach. *Transportation Research Records*, 1607:31–37, 1997.

[18] E. Cascetta and A. Papola. An implicit availability/perception random utility model for path choice. In *Proceedings of TRISTAN III*, volume 2, San Juan, Puerto Rico, June 1998.

[19] M. Rickert. *Traffic simulation on distributed memory computers.* PhD thesis, University of Cologne, Cologne, Germany, 1998. See http://www.zpr.uni-koeln.de/~mr/dissertation.

[20] J. Esser. *Simulation von Stadtverkehr auf der Basis zellularer Automaten.* PhD thesis, University of Duisburg, Duisburg, Germany, 1998.

[21] J. Esser. City congestion: Can online simulations solve the problem? In D.E. Wolf and M. Schreckenberg, editors, *Traffic and granular flow'97.* Springer, Heidelberg, 1998.

[22] C. L. Barrett. Personal communication.

[23] S. Krauß, K. Nagel, and P. Wagner. The mechanism of flow breakdown in traffic flow models. Los Alamos Unclassified Report (LA-UR) 98-3161, Los Alamos National Laboratory, see http://www.santafe.edu/~kai/papers, 1998.

[24] K. Nagel, P. Stretz, M. Pieck, S. Leckey, R. Donnelly, and C. L. Barrett. TRANSIMS traffic flow characteristics. Los Alamos Unclassified Report (LA-UR) 97-3530, Los Alamos National Laboratory, see http://www.santafe.edu/~kai/papers, 1997. Earlier version: Transportation Research Board (TRB) preprint 981332.

[25] S. Krauß. *Microscopic modeling of traffic flow: Investigation of collision free vehicle dynamics.* PhD thesis, University of Cologne, Cologne, Germany, 1997.

[26] K. Nagel. Particle hopping models and traffic flow theory. *Physical Review E*, 53(5):4655, 1996.

[27] K. Nagel, D.E. Wolf, P. Wagner, and P. M. Simon. Two-lane traffic rules for cellular automata: A systematic approach. *Physical Review E*, 58(2):1425–1437, 1998.

[28] K. Nagel and H. J. Herrmann. Deterministic models for traffic jams. *Physica A*, 199:254, 1993.

[29] S. Wolfram. *Theory and Applications of Cellular Automata.* World Scientific, Singapore, 1986.

[30] B. S. Kerner and H. Rehborn. Experimental features and characteristics of traffic jams. *Physical Review E*, 53(2):R1297–R1300, 1996.

[31] B. S. Kerner and H. Rehborn. Experimental properties of complexity in traffic flow. *Physical Review E*, 53(5):R4275–R4278, 1996.

[32] R. Barlovic, L. Santen, A. Schadschneider, and M. Schreckenberg. Metastable states in cellular automata. *European Physical Journal B*, 5(3):793–800, 10 1998.

[33] D. Chowdhury, L. Santen, A. Schadschneider, S. Sinha, and A. Pasupathy. Spatio-temporal organization of vehicles in a cellular automata model of traffic with 'slow-to-start' rule. *Journal of Physics A – Mathematical and General*, 32:3229, 1999.

[34] A. Schadschneider and M. Schreckenberg. Cellular automaton models and traffic flow. *Journal of Physics A – Mathematical and General*, 26:L679, 1993.

[35] A. Schadschneider. Analytical approaches to cellular automata for traffic flow: Approximations and exact solutions. In D.E. Wolf and M. Schreckenberg, editors, *Traffic and granular flow'97*, pages 417–432. Springer, Heidelberg, 1998.

[36] D. L. Gerlough and M. J. Huber. *Traffic Flow Theory.* Special Report No. 165. Transportation Research Board, National Research Council, Washington, D.C., 1975.

[37] M. Bando, K. Hasebe, A. Nakayama, A. Shibata, and Y. Sugiyama. Structure stability of congestion in traffic dynamics. *Japan Journal of Industrial and Applied Mathematics*, 11(2):203–223, 1994.

[38] M. Bando, K. Hasebe, A. Nakayama, A. Shibata, and Y. Sugiyama. Dynamical model of traffic congestion and numerical simulation. *Physical Review E*, 51(2):1035, 1995.

[39] R. Wiedemann. Simulation des Straßenverkehrsflusses. Schriftenreihe Heft 8, Institute for Transportation Science, University of Karlsruhe, Karlsruhe, Germany, 1994.

[40] P. Wagner. Personal Communication.

[41] U. Sparmann. *Spurwechselvorgänge auf zweispurigen BAB–Richtungsfahrbahnen.* Number 263 in Forschung Straßenbau und Straßenverkehrstechnik. Bundesminister für Verkehr, Bonn–Bad Godesberg, Germany, 1978.

[42] M. Rickert, K. Nagel, M. Schreckenberg, and A. Latour. Two lane traffic simulations using cellular automata. *Physica A*, 231(4):534–550, 1996.

[43] Transportation Research Board. *Highway Capacity Manual.* Special Report No. 209. National Research Council, Washington, D.C., 3rd edition, 1994.

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

[45] P. M. Simon and K. Nagel. Simple queueing model applied to the city of Portland. *International Journal of Modern Physics C*, in press.

[46] S. G. Eubank. Personal communication.

[47] K. Nagel, M. Rickert, and P. M. Simon. The dynamics of iterated transportation simulations. Los Alamos Unclassified Report (LA-UR) 98-2168, Los Alamos National Laboratory, see http://www.santafe.edu/~kai/papers, 1998. Presented at the 3rd TRIannual Symposium on Transportation ANalysis (TRISTAN-III) in San Juan, Puerto Rico.

[48] J. Bottom, M. Ben-Akiva, M. Bierlaire, and I. Chabini. Generation of consistent anticipatory route guidance. In *Proceedings of TRISTAN III*, volume 2, San Juan, Puerto Rico, June 1998.

[49] H.S. Mahmassani, T. Hu, and R. Jayakrishnan. Dynamic traffic assignment and simulation for advanced network informatics (DYNASMART). In N.H. Gartner and G. Improta, editors, *Urban traffic networks: Dynamic flow modeling and control.* Springer, Berlin/New York, 1995.

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

[51] J. A. Bottom. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, In preparation.

[52] Y. Sheffi. *Urban transportation networks: Equilibrium analysis with mathematical programming methods.* Prentice-Hall, Englewood Cliffs, NJ, USA, 1985.

[53] K. Nagel. *High-speed microsimulations of traffic flow.* PhD thesis, University of Cologne, 1994.

[54] C. Gawron. *Simulation-based traffic assignment.* PhD thesis, University of Cologne, Cologne, Germany, 1998. See http://www.zpr.uni-koeln.de/~gawron/diss.pdf.

[55] J. H. Holland. Using classifier systems to study adaptive nonlinear networks. In D Stein, editor, *Lectures in the sciences of complexity.* Addison-Wesley Longman, 1989.

[56] I. Chabini. Discrete dynamic shortest path problems in transportation applications: Complexity and algorithms with optimal run time. *Transportation Research Records*, 1645:170–175, 1998.

[57] B. W. Bush. Personal communication.

[58] M. Rickert and K. Nagel. Issues of simulation-based route assignment. Los Alamos Unclassified Report (LA-UR) 98-4601, Los Alamos National Laboratory, see http://www.santafe.edu/~kai/papers, 1998.

[59] A. Garcia and R. L. Smith. On the convergence of iterative routing-assignment procedures in dynamic traffic networks. Technical Report 95-26, University of Michigan Department of Industrial and Operations Engineering, Ann Arbor MI 48109, November 1995.

[60] M. Rickert. Personal communication.

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

[62] R. Palmer. Broken ergodicity. In D. L. Stein, editor, *Lectures in the Sciences of Complexity*, volume I of *Santa Fe Institute Studies in the Sciences of Complexity*, pages 275–300. Addison-Wesley, 1989.

[63] J. Esser and K. Nagel. Census-based travel demand generation for transportation simulations. In M. Schreckenberg al, editor, *Proceedings of the workshop "Traffic and Mobility*, Aachen, Germany, Sep/Oct 1998. Also Los Alamos Unclassified Report LA-UR 99-10, see http://www.santafe.edu/~kai/papers.

[64] R.J. Beckman et al. TRANSIMS–Release 1.0 – The Dallas-Fort Worth case study. Los Alamos Unclassified Report (LA-UR) 97-4502, Los Alamos National Laboratory, see http://transims.tsasa.lanl.gov, 1997.

[65] K. Nagel and C.L. Barrett. Using microsimulation feedback for trip adaptation for realistic traffic in Dallas. *International Journal of Modern Physics C*, 8(3):505–526, 1997.

[66] M. Rickert and K. Nagel. Experiences with a simplified microsimulation for the Dallas/Fort Worth area. *International Journal of Modern Physics C*, 8(3):483–504, 1997.

[67] K. Nagel, M. Pieck, P. M. Simon, and M. Rickert. Comparison between three different micro-simulations and reality in Dallas. Los Alamos Unclassified Report (LA-UR) 98-2944, Los Alamos National Laboratory, see http://www.santafe.edu/~kai/papers, 1998.

[68] J. Esser and K. Nagel. In preparation.

[69] M. D. Williams, G. Thayer, and L. L. Smith. A comparison of emissions estimated in the transims approach with those estimated from continuous speeds and accelerations. Los Alamos Unclassified Report (LA-UR) 98-3162, Los Alamos National Laboratory, see http://transims.tsasa.lanl.gov, 1998.

[70] M. Kerschgens. Regionale und lokale Immissionsmodellierung auf der Basis von dynamischen Verkehrssimulationen. In *Verkehr und Mobilität*, number 66 in Stadt Region Land. Institut für Stadtbauwesen, Technical University, Aachen, Germany, 1998.

[71] H. Wallentowitz. Auswirkungen neuer Fahrzeug- und Verkehrstechnologien - Analyse von Verkehrsfluß, Kraftstoffverbrauch und Emissionen mit PELOPS. In *Verkehr und Mobilität*, number 66 in Stadt Region Land. Institut für Stadtbauwesen, Technical University, Aachen, Germany, 1998.

[72] G. D. B. Cameron and C. I. D. Duncan. Paramics–parallel microscopic simulation of road traffic. *J. Supercomputing*, 10(1):25, 1996.

[73] K. Nagel, M. Rickert, R. Frye, P. Stretz, P. M. Simon, R. Jacob, and C. L. Barrett. Regional transportation simulations. In *Proceedings of the Advanced Simulation Technologies Conference*, Boston, MA, U.S.A., 1998. The Society for Computer Simulation International.

[74] M. Rickert and K. Nagel. Dynamic traffic assignment on parallel computers. To be submitted to: Proceedings of a Traffic Workshop at HPCN'99 in Amsterdam.

[75] K. Nagel and A. Schleicher. Microscopic traffic modeling on parallel high performance computers. *Parallel Computing*, 20:125–146, 1994.