An event-driven queue-based microsimulation of traffic flow

An Event-Driven Queue-Based Microsimulation of Traffic Flow

2006-08-01

David Charypar
Institute for Transport Planning and Systems (IVT)
ETH Zurich
Switzerland
Phone: +41 44 633 35 62
Fax: +41 44 633 10 57
Email: charypar@ivt.baug.ethz.ch

Kay W. Axhausen
Institute for Transport Planning and Systems (IVT)
ETH Zurich
Switzerland
Phone: +41 44 633 39 43
Fax: +41 44 633 10 57
Email: axhausen@ivt.baug.ethz.ch

Kai Nagel
Transport Systems Planning and Transport Telematics (VSP)
TU-Berlin
Germany
Phone: +49 30 314 23308
Fax: +49 30 314 26269
Email: nagel@vsp.tu-berlin.de

Words:	4497
Tables and Figures:	3 (= 750 words)
Total:	5247

ABSTRACT
Simulating traffic flow is an important problem in transport planning. The most popular simulation approaches for large scale scenarios today are aggregated models. Unfortunately, these models lack temporal and spatial resolution. On the other hand, microsimulations are very interesting for the simulation of traffic flow as they are capable of very accurately simulating features that require both high temporal and spatial resolution including traffic jams and peak hours. So far, most of the microscopic approaches come at high computational costs and therefore require expensive large computers to run them within reasonable time. In this work we present how it is possible to reduce these costs by using a queue-based model and an event-driven approach jointly. Our approach makes it possible to run large scale scenarios with more than one million simulated person-days on networks with 10k links in less than ten minutes on single CPU desktop computers present in most offices today.

INTRODUCTION

The simulation of traffic flow is an important problem in transport planning, as it represents the final joint between the intangible description of travel demand and the emergence of flow densities, volumes and travel speeds. In today's practice traffic flow is most often simulated using aggregated models as these models are easy to use and well established in the community, but compared to aggregated models the microsimulation of traffic flow has certain outstanding advantages:

- Very high spatial and temporal resolution. This makes it possible that features as traffic jams and rush-hours can be captured much more accurately.
- Traffic is described in a very natural way: The cars that travel, the roads and the intersections are simulated directly.
- Microsimulations of traffic flow can be easily coupled with other microscopic approaches such as agent based demand modeling.
- It is possible to make inverse analyses to find out where certain cars are coming from and why they can be found on a certain specific location of the road network.

However, these properties come at the price of high computational burden. This very often makes it necessary to use complex software architectures together with large parallel computers which in turn makes these methods expensive. The other option is to use microsimulations only for small applications and switch to the above-mentioned aggregated methods for large scale problems. Unfortunately, these methods often differ to such an extent that it is nearly impossible to transfer know how from one field to another.

It is the aim of this work to present a new event-driven queue-based approach that makes the microsimulation of large scale traffic flow problems feasible on affordable, desktop computer hardware and within reasonable time.

RELATED WORK

In this section we will give a selection of previous work that is somehow related to our microsimulation. There exists a large range of different traffic flow simulation approaches. The physically based microsimulations (e.g. AIMSUN(*1, 2*), MITSSIM(*3, 4*), VISSIM(*5*)) generally try to capture as many traffic flow phenomena as possible. They simulate car following and lane changing behavior and use a continuous representation of space and constant very small time-steps to simulate the cars on the roads.

A different microscopic but less physical simulation approach is represented by cellular automata (e.g. (*6*)) as they are used for example in TRANSIMS(*7, 8, 9*). Here, cars move through cells that they can occupy like particles. Although in cellular automata we have a coarser level of detail, features like densities and travel speeds still emerge from the simple interactions of the cars and are not computed at an aggregated level. This changes when we move on to mesoscopic modeling. In mesoscopic models (e.g. METROPOLIS(*10, 11*), DynaMIT(*12, 13*), DYNASMART(*14, 15*)), while the demand side of traffic is still represented microscopically the supply side is computed on the aggregated level.

The highest level of abstraction can be found in the macroscopic models that compute all traffic quantities on the aggregated level. One example of a model that simulates traffic as a one dimensional incompressible fluid is NETCELL(16).

The work presented here builds on the approach of MATSIM-T(17) which while using simplified, queue-based dynamics to be computationally efficient still estimates all quantities microscopically and therefore has to be located somewhere between the mesoscopic approaches and cellular automata.

It can be observed that time-step based approaches have been more popular in the past than event-driven approaches. It is not clear why this is the case but one reason might be the more straightforward implementation of time-step based simulations.

CLASSIFICATION OF MICROSIMULATION METHODS OF TRAFFIC FLOW

Cellular Automata

Cellular automata are used in traffic flow microsimulation to accurately model the behavior of cars traveling on a road network (see for instance (6)). The basic idea is to discretize space in cells of equal size each of which can be occupied by at most one vehicle. The cars drive through these cells by looking forward and judging from the space available in front of them they decide how fast they should go. The advantage of cellular automata is their simplicity while retaining a fair amount of spatial resolution and therefore the general ability to simulate many interesting phenomena like car following and lane changing as demonstrated in (6, 7). An additional advantage is that link capacities are generated from the properties of the links and the behavior of the drivers.

Although the mentioned properties are generally desirable the major drawback of this method is its computational cost. Its high temporal and spatial resolution is achieved by granting processor time to every simulated agent in every simulated time-step (usually one second). This becomes unpractical if the number of simulated agents rises above roughly 100k, especially if it is desired to simulate the traffic flow iteratively say as part of an iteration search for a steady-state solution. If we want to stick with this kind of solution the only way to achieve sufficient speed for large problem sizes is to use massively parallel computers like it was done in (*4*). Unfortunately, many will not have sufficient funding to follow this approach.

Queue-Based Simulations

If the computational speed of cellular automata is not sufficient for a certain kind of application one way of achieving higher performance is to simplify the model by using a queue-based approach as it is used in MATSIM-T, see (*9*). Here, the basic assumption is that the main features of (at least) urban traffic can be described at the intersections as either the traffic is flowing more or less freely on the links or the cars are queuing in front of the next intersection and waiting for the car in front of them to move. Based on these assumptions cellular automata can be modified in the following way:

- The cars traveling through the network are no longer simulated directly but instead we change perspective and the links between intersections are being simulated.
- For every link there is a queue that stores the cars that enter the link. Also for each car the entry time is stored in this queue.
- The capacity of a link as well the space available for cars are given as parameters
- When a car wants to leave link A and wants to enter link B, in the next simulation time-step link A checks if the car is able to leave the link according to various constraints (capacity, free speed travel time, intersection precedence,

space available at the next link) and if all preconditions are met the car leaves link A and enters link B.

The advantage of this model is that it reduces the amount of information computed as it no longer has to compute the fine grained stop and go interactions between following cars. As the basic simulation unit is the road segment and no longer the car we gain roughly a factor of 10 to 100 compared to cellular automata.

Event-Driven Queue-Based Simulations

Although queue-based simulations are much faster than cellular automata it is still challenging in terms of computing time to simulate the traffic flow of a whole region microscopically. Consequently, it is desirable to find even faster ways of computing the same information.

The key observation is that queue-based simulations are inefficient in areas where the flow density is very low. Here, the links have to be simulated in every time-step although there is little to compute at all: In most of the time-steps the links only realize that there is no car in the position to leave the link and enter the next. These empty operations cost the simulation most computing time when simulating off-peak and nightly hours. Therefore, it is desirable to speed up the simulation of low density traffic.

In order to achieve higher computational efficiency we chose to substitute the constant time-step for the direct treatment of actions happening on the road network. Each such action gets reflected by an event. Every time a car enters or leaves a link a corresponding event gets processed. This means that the number of events happening in a specific time period directly corresponds to the amount of traffic flow present on the network and consequently the computational effort is proportional to the traffic load. There are two main advantages of the event-driven approach:

- The computational effort distributes over the simulated time of day equal to the traffic flow during any time of the day. As a result, most of the computational time is spent where traffic flow is maximal and almost no time is spent where the traffic network is empty.
- The total computing time needed for the simulation of a certain travel demand (e.g. 1 million cars traveling from one part of a city to another) is almost independent of the total amount of simulated time needed to process the traffic demand of the scenario, i.e. it does not matter if the cars travel during 24 hours or one month. This is because the simulation cares only about the events to be simulated and the total number of events in both scenarios is the same.

When switching to an event-driven simulation one might be concerned about overhead costs added in situations where the system processes very many events at a time. These are the situations where time-step-based approaches usually are very efficient. It is to be expected that such situations will arise in certain scenarios and for these the queue-based approach with constant time-steps is the most efficient choice. However, it is to be noted that such a scenario would need to have full network load nearly everywhere. Our experience shows that in most simulation networks there is a fair amount of links that experience only little load even during rush-hours. It might be due to this fact that we have not found a scenario so far where the event driven approach is slower than the time-step-based approach even during the most heavily loaded minute of the day. In our tests the event-driven simulation obtains an overall speedup of 20 or more over the time-step-based implementation when simulation a 24 hours scenario.

A comparison of the spatial and temporal discretization schema of the simulation approaches described above is shown in Table 1.

TABLE 1 Discretization Schema of Different Microsimulation Approaches

	Spatial discretization	Temporal discretization
Cellular Automata	equidistant	constant time-step
Queue-based simulation	road segments	constant time-step
Event-driven, queue-based simulation	road segments	adaptive, event-driven

IMPROVED DYNAMICS OF THE EVENT-DRIVEN QUEUE-BASED APPROACH

Apart from the changes that were made to our model in order to switch from fixed time-steps to event-driven information processing we have also incorporated a couple of behavioral changes to the road segment dynamics. We want to give an overview of these and briefly discuss the reasons for the modification.

- *Gaps, gap travel times*: In the classical queue-based approach as it was in use in MATSIM-T so far (*9*) there is no concept of gaps between cars. Assume a full road segment. If the first car in the row is able to leave the road in one time-step the space available on the link will become visible upstream at the entry of the link exactly one time-step later. That is, congestion can dissolve much quicker than usually observed as the backward traveling speed of gaps is very high. In order to avoid that problem we have introduced a concept of backwards traveling gap on the road segments. These gaps travel in the opposite direction of the cars at a predefined and parameterized speed. In this way it is possible to reproduce the typical behavior of backward traveling gaps on links during unloading.
- *Capacity constraints at inlet*: In the classical queue-based approach link capacities were only enforced at the downstream end of links. While this is certainly sufficient to reproduce the correct capacities on a linear sequence of road segments more complex behavior can be observed at intersections: Especially on converging Y-type connections with high capacity incoming roads and a relatively low capacity outgoing road one would expect to observe breakdowns at the bottleneck. However, without limiting the incoming capacity of a road segment the breakdown does not occur until much later downstream. To avoid this problem we have extended our model to take care of the inflow capacity as well. Note however that it is often not easy to get the corresponding data, and setting incoming and outgoing capacities identically often is an unsatisfactory choice as the incoming capacity is often close to the theoretical capacity of a link of a certain type while the outgoing capacity is often chosen taking traffic light timings into account.
- *Handling of gridlocks*: One issue with all types of microscopic traffic flow simulations with predefined routes that is not very apparent at first are gridlocks. In heavily loaded road networks it is possible that individual agents

block each other when waiting for the next link on their individual routes to become free for them. At a certain point, this will result in a gridlock happening and this in turn will prevent the involved agents to arrive at their desired locations. It is therefore essential to incorporate some sort of handling of gridlocks into the simulation model. In our simulation the problem is solved using a guaranteed minimum capacity of each link. This can be for instance one percent of the nominal capacity or any constant desired. The simulation then simply guarantees that no matter whether a road segment is already full or not it will always accept incoming cars at least at this specified minimum rate. This can lead to over full links but effectively avoids the situation where car stay in the simulation forever. In our experiments, this method has proven to be practical and quite simple to implement.

TECHNICAL DESCRIPTION OF THE SOFTWARE DESIGN

In the following section we are going to present the different parts of our event-driven queue-based microsimulation of traffic flow and we will explain how these parts work together in order to solve the simulation problem at hand.

Description of Travel Demand

Travel demand is the input for any microsimulation of traffic flow and therefore it is central to find a consistent description for it. In this work we use the same formalism as used in (*10*):

- We use a population of agents each of which has a complete 24 hours activity plan.
- An activity plan consists of an activity pattern describing the type and order of the activities executed, location choice information describing where in the world the activity in question should be executed, timing information defining the time of day when the activities are taking place, and routing information describing the sequence of road segments to take to travel between subsequent activities.

Note that while we use a complete description of traffic demand the description of the activity type is not necessary for the simulation of traffic flow.

The Model of Traffic Flow

The basic idea of our model of traffic flow is that we only simulate transitions of agents from one road segment to the next. During the time spent on a specific road segment very little information about the position of the agent is available: Only the position in the queue is stored and the earliest time that the car could leave the link according to free speed limitations. When the agent arrives at the end of a road segment an event occurs and the agent leaves the road segment and enters the subsequent one according to the route stored in his complete activity plan. However, this can only happen if there is sufficient room left on the downstream link to take the oncoming agent. Otherwise the agent has to wait until sufficient space is available.

The travel times derive from the behavior of the agents on the link: If a link is empty the travel time is equal to the time needed to drive down the link at free speed. The travel time can becomes longer if there are other agents traveling in front of the agent. In this case the agent cannot leave the link before all agents in front of it have left the link. The reason for such a delay usually is a high demand coupled with limited outflow due to limited outflow capacities or a blocked road network downstream. That is, as expected high demand and low supply will automatically lead to large travel times.

Actors during the Course of the Simulation

In our simulation software there exist basically three elements that cooperate to compute the traffic flow through the network (see Figure 1). These three elements are:

- The road segment that hold the cars during their travel.
- The agent that represents the object moving through the road network and keeps all information about the travel demand.
- The clock that takes care of registered timers and wakes up the agents at the right moment when there is an action to be taken.

FIGURE 1 Interaction Processes Between Elements of the Traffic Flow Microsimulation.

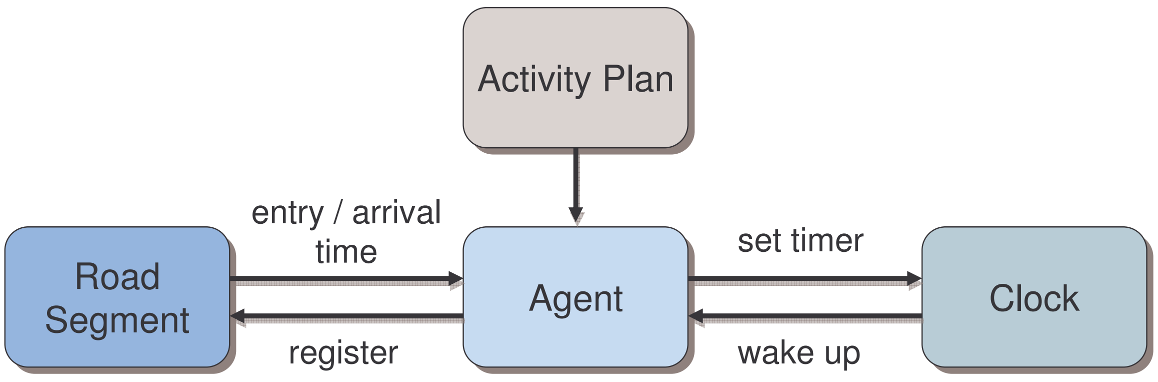
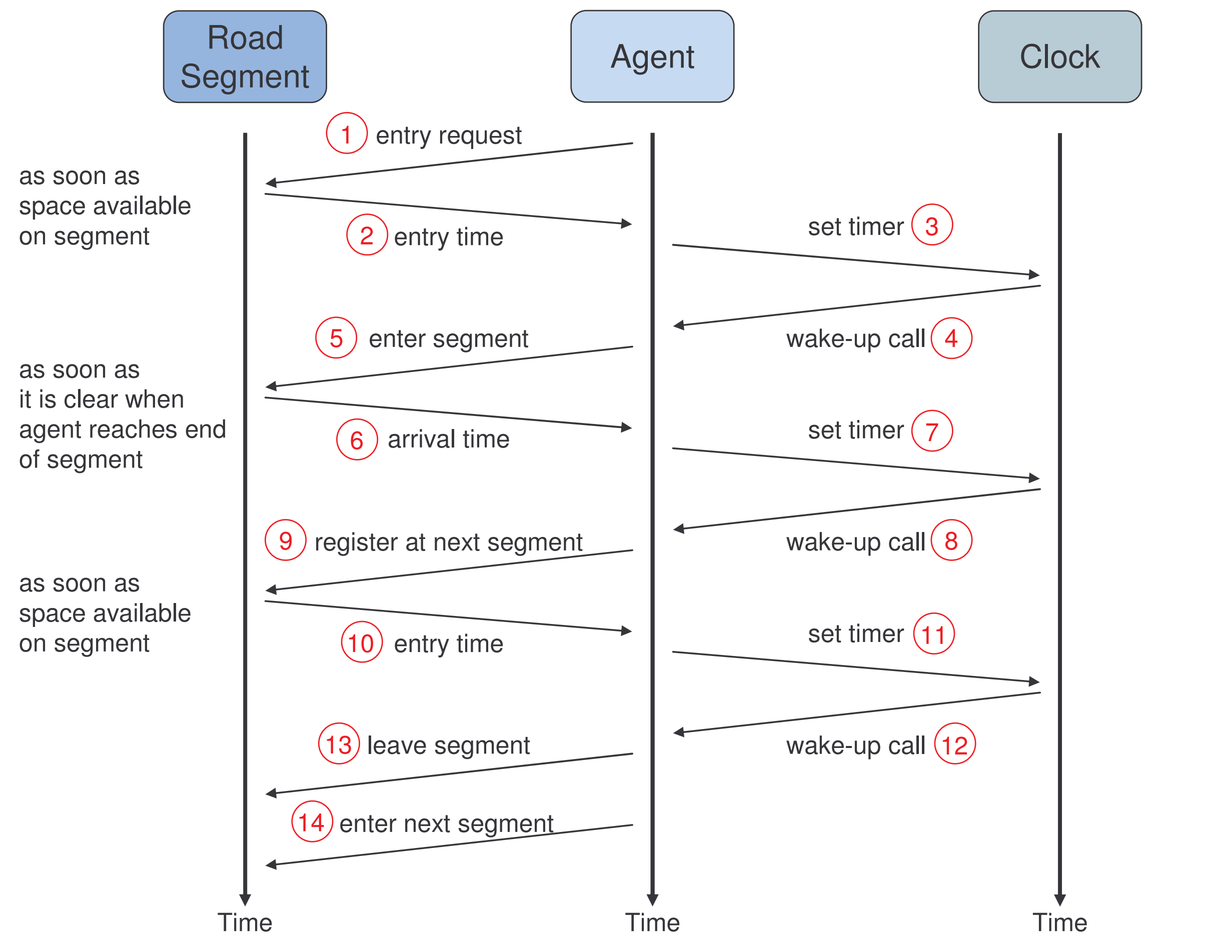


FIGURE 2 The traffic flow protocol.



The Traffic Flow Protocol

The actors mentioned above all interact and communicate according to a certain protocol which is most easily explained using an example that is illustrated in Figure 2 where one agent wants to travel along a certain sequence of road segments. The protocol consists of the following steps:

1. The agent informs the road segment that it wants to enter. The road segment stores this request chronologically together with all other requests of this kind.
2. As soon as it is clear when a gap will be available for the agent, the road segment sends this entry time to the agent.
3. The agent registers a timer for this time with the clock.
4. As soon as the timer expires, the clock sends a wake-up call to the agent and informs it about the time.
5. The agent then enters the road segment. With this step the first part of the protocol finishes. Now the agent continues to travel down the road. Note that this does not need any actions to be taken by the road segment, the agent or the clock.

6. When the agent becomes first in the road queue, the road segment can compute the time when the agent is going to arrive at the end of the road segment and sends this information to the agent.
7. Similar to step 3 the agent registers a timer with the clock.
8. After the timer expires the clock calls the agent.
9. Now the agent is at the front of the road segment. It then checks with its activity plan which is the next road segment to travel and registers there.
10. When it is clear at what time there will be a gap available for the agent to enter the next link it receives a message with the predicted entry time.
11. The agent again registers a timer for this time.
12. When the timer expires the agent is woken up by the clock.
13. The agent informs the last road segment that it is leaving.
14. It also informs the next road segment that it enters. From here on steps 6 to 14 repeat until the last road segment of the trip is reached where step 14 is left out and the protocol stops.

RESULTS

Test Scenario

The test scenario we use to demonstrate the properties of our simulation software consists of the following parts:

- The road network of the federal states of Germany Berlin and Brandenburg. It consists of 11.6k nodes and 27.7k links.
- The synthetic population of the area consists of 7.05M people. Each person has a complete daily activity plan with multiple activities and trips. That is, we are simulating 7.05M person-days.

The average number of trips per agent in our demand is 2.02 and the average length of a trip is 17.5 links. This leaves us with an overall daily demand of 249M road segments to be traveled.

Test Setup

The test scenario was run on a server-system equipped with Dual Core AMD Opteron Processor 275 with 2.2GHz. Only one core was used during our test runs. The system was equipped with 4GiB of RAM of which roughly 3.2GiB were used. The input file for the synthetic population and the daily demand uses 5.4GiB of disk space and the output file generated which contains each event processed during the simulation consists of roughly 17GiB. Note that one reason for the considerable sizes of our input and output files is the fact that we are using ASCII representations (XML in the case of the input files and line-based text files in the case of the output).

Test Results

We ran the test scenario on the machine mentioned above. An average run took an overall execution time of 53 minutes. This time divides into 8 minutes for the reading in and parsing of the input data (network, synthetic population and complete daily plans), 37 minutes for the execution of the plans (i.e. the actual simulation of traffic flow), and roughly 8 minutes for the generation of the output files. Summing everything up this results in an overall real-time ratio of 27 for our test scenario on the described hardware. This means that our simulation ran 27 times faster than real-time. Compared to our original queue-based microsimulation using fixed time-steps we gained a speedup of more than a factor of 20. Please note that the time needed for I/O could be easily reduced to about two minutes by switching to a binary representation of the input and output files.

DISCUSSION AND OUTLOOK

An interesting question is how to derive an estimate of the performance of the microsimulation at hand given the problem to be computed. Fortunately this is very easily possible as the main factor affecting the computational effort needed to simulate the traffic flow is the overall traffic volume. Judging from our test results, it can be estimated that the average simulation speed on the computer hardware described above is about 110k road segments per second (not including I/O). This can be now transferred into an estimate of runtime for a given problem: For instance let us assume a large scale traffic simulation problem of a population of one million agents each executing 3 trips a day. Let the average trip length be 15 road segments. The overall traffic volume is then given as $V = a \cdot t \cdot l = 1\text{M} \cdot 3 \cdot 15 = 45\text{M}$, where V is the total traffic volume, a is the number of agents in the population, t is the average number of trips per agent, and l is the average length of a trip in number of road segments. From $r = V / v$, where r is the expected running time and v is the simulation speed in number of traveled road segments per second it follows that $r = 45\text{M}/110\text{ks}^{-1} \approx 409\text{s}$. We therefore expect that such a simulation will take less than ten minutes to complete on today's computing hardware.

It is interesting to note that the factor limiting scenario sizes the most when using our microsimulation model is computer memory and not processor speed. Even though the large scenario used for our testing can be simulated in only 53 minutes on affordable hardware it needs roughly 3.2GiB of memory meaning that it is not possible to double the size of the scenario on a machine with 4GiB. This is not directly a problem of our implementation but comes from the sheer size of the data describing the demand. It would be interesting to investigate how much the memory requirements can be alleviated by using some sort of streaming of the travel demand: An agent could postpone loading its trips into memory until just before the moment when they are executed. Another possibility might be to use some sort of compression algorithm to reduce the memory size needed.

A completely different way around the problem of memory sizes is parallelization. We are in the course of developing a parallel version of our microsimulation model that will make it possible to take advantage of a group of cheap computers to simulate large scenarios that would not fit on any of these computers alone. The other obvious advantage of such a parallel implementation is the expected higher execution speed.

SUMMARY

We have presented a microsimulation of traffic flow that uses a queue-based and an event-driven approach jointly. Compared to cellular automata this means a certain reduction in spatial resolution that is rated to be acceptable compared to the large speedups gained by this approach. Compared to earlier queue-based approaches the fact that our simulation is event-driven mainly means that we save time in areas of the road network where the traffic load is moderate to small. Over all a gain of 20 can be expected when compared to earlier queue-based approaches.

On top of the accelerations of the method a couple of other modifications where undertaken. The simulation now accurately handles backward-traveling gaps, it limits the inflow capacities of all road segments in a meaningful way and it has a new way of handling gridlocks in the simulation by using a notion of guaranteed minimal capacities.

The final simulation was tested with a large scale 24 hours scenario with seven million simulated person-days on a network with 28k links. The tests were undertaken on a single CPU workstation computer with a current processor and 4GiB of RAM of which 3.2GiB were used. The simulation took about 53 minutes to finish corresponding to an overall real-time ratio of 27.

REFERENCES

1. Barceló, J., J.L. Ferrer, D. Garcia, M. Florian, and E. Le Saux. Parallelization of microscopic traffic simulation for ATT systems. In P. Marcotte and S. Nguyen, editors*, Equilibrium and advanced transportation modelling*. Kluwer Academic Publishers, 1998, pp. 1-26.
2. AIMSUN. http://www.aimsun.com. Accessed August 2006.
3. Yang, Q. A Simulation Laboratory for Evaluation of Dynamic Traffic Management Systems. PhD thesis., Massachusetts Institute of Technology, 1997.
4. MITSIMLab. MIT ITS Program, Cambridge, MA. http://web.mit.edu/its/mitsimlab.html. Accessed August 2006.
5. VISSIM. http://www.ptv.de/cgi-bin/traffic/traf_vissim.pl. Accessed August 2006.
6. Brilon, W. and N. Wu. Evaluation of cellular automata for traffic flow simulation on freeway and urban streets. In W. Brilon, F. Huber, M. Schreckenberg, and H. Wallentowitz, editors, *Traffic and Mobility: Simulation – Economics – Environment*, Springer, Berlin, 1998, pp. 163–180.
7. Nagel, K, D.E. Wolf, P. Wagner, and P. M. Simon. Two-lane traffic rules for cellular automata: A systematic approach. Phys. Rev. E, 58(2), 1998, pp. 1611–1639.
8. Nagel, K. and M. Rickert. Parallel implementation of the TRANSIMS micro-simulation. Parallel Computing, 27(12), 2001, pp. 1611–1639.
9. TRANSIMS, TRansportation ANalysis and SIMulation System. Los Alamos National Laboratory, Los Alamos, NM. http://transims.tsasa.lanl.gov. Accessed August 2006.
10. Marchal, F. Contribution to dynamic transportation models. Ph.D. thesis. University of Cergy-Pontoise, Cergy-Pontoise, France, 2001.
11. de Palma A. and F. Marchal. Real case applications of the fully dynamic METROPOLIS tool-box: An advocacy for large-scale mesoscopic transportation systems. *Networks and Spatial Economics*, 2(4), 2002, pp. 347–369.
12. Ben-Akiva, M., M. Bierlaire, H. Koutsopoulos, and R. Mishalani. Dynamit: A simulation-based system for traffic prediction. DACCORS Short Term Forecasting Workshop, The Netherlands, 1998.
13. DynaMIT. http://mit.edu/its/dynamit.html. Accessed August 2006.
14. Chang, G.L., T. Junchaya, and A.J. Santiago. A real-time network traffic simulation model for ATMS applications: Part I— Simulation methodologies. *IVHS Journal*, 1(3), 1994, pp. 227–241.
15. DYNASMART. http://www.dynasmart.umd.edu. Accessed October 2005.
16. Carlos Danganzo personal hompage. http://www.ce.berkeley.edu/trans/Daganzo/index.htm. Accessed August 2006.
17. MATSIM, Multi Agent Traffic SIMulation. http://www.matsim.org. Accessed August 2006.
18. Chowdhury, D., L. Santen, and A. Schadschneider. Statistical physics of vehicular traffic and some related systems. *Physics Reports*, 329(4–6), 2000, pp. 199–329.
19. Balmer, M., K. W. Axhausen and K. Nagel. A demand generation framework for large scale micro-simulations. In *TRB 85th Annual Meeting Compendium of Papers.* CD-ROM. Transportation Research Board, Washington, D.C., 2006.