Parallel traffic micro-simulation by cellular automata and application for large scale transportation modeling

Kai Nagel[a,b], Christopher L. Barrett[a,b], Marcus Rickert[a,c]

[a] Los Alamos National Laboratory, TSA-DO/SA MS M997, Los Alamos NM 87545, U.S.A., kai@lanl.gov, barrett@tsasa.lanl.gov, rickert@tsasa.lanl.gov, llsmith@lanl.gov

[b] Santa Fe Institute, 1399 Hyde Park Road, Santa Fe NM 87501, U.S.A.

[c] Center for Parallel Computing ZPR, University of Cologne, 50923 Köln, Germany

This version: February 21, 1996

A set of very simple driving rules for single lane roadway traffic gives surprisingly realistic results. This observation can be backed up by theory, showing relations both to fluid-dynamical models for traffic, and to control system approaches to driving. Extensions to multi-lane traffic are straightforward and work well. Because of its simplicity, it is easy to implement the model on supercomputers (vectorizing and parallel), where we have achieved real time limits of more than 4 million kilometers (or more than 53 million vehicle sec/sec). The model can be used for applications where both high simulation speed and individual vehicle resolution are needed. We describe results of freeway network simulations of the German land Northrhine-Westfalia, where drivers have individual route plans which they adapt according to the delays they experience. Finally, we discuss the TRANSIMS microsimulation design, where such a simple high-speed microsimulation will be used as one microsimulation option.

I. INTRODUCTION

In many countries, the transportation planning process is currently turning from unlimited expansion to a fine-tuning of the existing system. This demands the development of better tools. Due to the complexity of the problem, many of these tools will be computer simulation tools, a development which is reinforced by the explosion of computational power over the last years.

In the transportation simulation field, some agreement has been reached that *micro*simulation, i.e. a computational resolution down to the level of individual travelers, may be the only answer to a wide variety of problems. An important part of these microsimulations will be the vehicle traffic part. For at least three reasons, this microsimulation should be extremely fast:

(1) RELAXATION PROBLEMS: For example, traffic jams, which can only be produced on the microsimulation level, affect people's modal and route choices. The simulation may have to loop several times back and forth between a route planning and a driving microsimulation part, thus producing a need to execute the microsimulation several times instead of only once until a result is achieved.

(2) REAL TIME APPLICATIONS: Online simulation applications such as state extrapolation for traffic control demand that the model runs much faster than real time.

(3) MONTE CARLO ANALYSIS: The apparent global stochasticity of traffic is captured in most modern simulation approaches. New methods of analysis are necessary in these noisy high dimensional systems. For example, a

simple method involving averaging state values over several simulation runs may be necessary to produce relevant information from the data.

The method to achieve this goal of extremely fast simulations is to construct *minimal* models of the activity under scrutiny. The definition of "minimal" depends on the question, and can usually only be answered by a thorough theoretical understanding of the system.

One possibility for traffic microsimulations are cellular automaton models. Here, the road is discretized into boxes which are either empty or fit exactly one vehicle. Dynamics and movement are executed on this grid (see Sec. II). Despite their simplicity, these models make astonishingly realistic global traffic patterns (Sec. III). The efficacy of the granular methods can be backed up by theory (Secs. V and VI).

Granular models run extremely fast, especially on modern parallel supercomputer architectures. A simple single lane cellular automaton has been implemented on 6 different supercomputers, including a CM-5, using 3 different algorithms. The experiences of these implementations will be described in detail (Sec. IV).

Using extensions of this model (multi lane traffic (Sec. VII), freeway networks containing ramps and junctions and drivers with plans (Sec. VIII)) have been implemented as well; the computing speed drops less than a factor of two. Results will be given.

Finally, the current status of the design of the parallel TRANSIMS microsimulation will be described (Sec. IX).

II. DESCRIPTION OF THE MODEL

The principal idea of our approach is to produce the behavior of vehicles by a set of rules which are as simple as possible. This obviously implies a tradeoff between realism and simulation speed. In the following, we describe a set of rules for a simple single-lane traffic flow model [16]; multi-lane modeling will be sketched out in a later section of this paper.

The model is defined on a one-dimensional array, cutting the street into "boxes" or "cells" of length $\epsilon \approx 7.5\,m$ ($\approx 23\ feet$). This is an average length modeling the approximate distance one vehicle occupies in a congested condition. Each box is either empty, or occupied by one vehicle. Since positions are limited to integer array positions (the integer index of the occupied box), velocities can also only be integers, with possible values between 0 and some v_{max}. For instance, when a vehicle has the velocity 3, it will jump 3 boxes forwards in a time step.

We use $v_{max} = 5$. This speed, in cells per update, corresponds to an actual velocity of 112 km/h (see below). In fact, the phenomenology of the model (i.e. the formation of start-stop-waves) is independent of the choice of v_{max}, but the form of the resulting fundamental diagram (throughput vs. density, see below) becomes unrealistic for values much larger or smaller than 5 [16].

During an update of the simulation, before a car is moved along the lattice, its new velocity is adapted according to its specific situation:

- *slowing down:* If the next car ahead is too close ($gap < v$, where gap is the number of empty boxes between a car and the next one ahead), then the velocity is reduced: $v := gap$. [1]

[1] Note that, for integer variables, $gap < v$ and $gap <= v - 1$ are equivalent.

- *acceleration:* If, however, the gap is large enough ($gap \geq v+1$) *and* if the velocity is smaller than the maximum velocity ($v < v_{max}$), then the velocity is increased by one: $v := v + 1$.

These rules use as input only “old” information which is not changed or generated during the update. Therefore, this deterministic part of the velocity update may be performed “in parallel”, i.e., simultaneously on all vehicles.

In order to take into account the natural fluctuations of driving behavior, we add a

- *randomization:* When velocity is 1 or larger ($v \geq 1$), then with probability $p = 0.5$ the velocity of each vehicle may in addition to the above rules be reduced by one: $v := v - \gamma(p)$, where $\gamma(p)$ is one with probability p and zero else.

The randomization introduces speed fluctuations for free driving, over-reactions at slowing down, and retardations during acceleration. These random fluctuations can be viewed as elaborating driver control logic in surprising and non-trivial ways ([3] and see below).

For the whole velocity update of one vehicle we use only *position* information of the predecessor—this is in contrast to what some car following theories suggest [6]. Nevertheless, this leads to realistic results (see later).

As already mentioned above, after the velocity update each vehicle is advanced v boxes to the right (“*propagation step*”). Since this step, as well as the randomization, can again be performed in parallel, the whole update consists of parallel rules. This fact enormously simplifies efficient programming on supercomputers.

When taking randomization into account, the average free speed of a vehicle in our model is $4.5 \times 7.5\ m/\Delta t$, where Δt is the model time step. This should be equal to, as we stated above, about 112 km/h (70 mph), which results in $\Delta t \approx 1\ sec$.

III. PHENOMENOLOGY OF THE MODEL

Fig. 1 shows a typical evolution of the system from random initial conditions. The horizontal direction is the space direction; time is pointing downwards. Each black pixel corresponds to the position at a point in space and time of a vehicle moving from left to right; each horizontal line therefore shows the configuration of the simulated road segment at a different time step. In fact, the plot is similar to the well-known space-time-diagrams from aerial photography [33], except that time and space axes are exchanged, and that one has to connect all pixels which represent one car in order to get its trajectory.

One observes in the figure that the details of the random initial conditions quickly become irrelevant and that the system's appearance is dominated by traffic jams of high vehicle density (black), separated by zones of free traffic. Jams form and dissolve at arbitrary times and positions, as can be seen at some of the smaller jams.

As a next step of our investigation, we measured the relation between throughput and density for our model. Averages over $T = 200$ time steps, measured locally at one site, result in the scatter-plot of Fig. 2.

Our simple model reproduces the linear relation between flow and density for low traffic correctly. Furthermore, near the density corresponding to maximum capacity, it shows the strong fluctuations in the capacity values [12,36], and these fluctuations as well as the average throughput decrease approximately linearly with density at higher densities. The density corresponding to maximum flow is somewhat low compared with reality, but this is corrected when using a model for more than one lane and with slower vehicle types (see below).

IV. COMPUTATIONAL ISSUES

A. Coding

Two obvious ways to implement the described dynamics are (i) *site-oriented*, and (ii) *vehicle-oriented* (see [16] for a more detailed description of all aspects of this section). Site-oriented represents a street by an array $(v_i)_i$ of integers with state values between -1 and v_{max}. A value of $v_i = -1$ means that there is no car on site i, whereas a value v_i between 0 and v_{max} denotes a car with speed v_i at site i. The state of each site can then be updated as a function of the states of its neighbors. The principal disadvantage of this technique is that one needs as much computer time for the empty sites as for the occupied ones; but the advantage is that one can employ sophisticated computing techniques (e.g. single-bit coding) known from advanced cellular automata (CA) simulations [30], which yields very high efficiency especially on SIMD architectures. (An early but elaborated version of this approach is [28]. But it uses the technique in a way which is not vectorizable and can therefore not use the power of many supercomputers.)

In contrast, the *vehicle-oriented* approach represents a street as a list of pairs *(position of vehicle, speed of vehicle)*. Obviously, the vehicle-oriented approach will outperform the grid-oriented one for very low vehicle densities, but for densities corresponding to capacity flow the results depend on the computer architecture and on the system size. In addition, this approach is difficult to handle efficiently as soon as vehicle passing is allowed. This is especially true on parallel computers, where for efficiency neighboring vehicles should reside on the same node.

In addition, we used a third "intermediate approach", whose data structure is in principle site-oriented, but where the update only treats the "interesting sites". Single-bit CA-techniques are no longer applicable, but at least for non-vectorizing computer architectures the loss in performance is never more than a factor of four. In addition, treatment of multi-lane traffic is easiest with this approach.

B. Computer architecture overview

For our comparisons, we used a SUN Sparc10 workstation, a net of these workstations coupled by Ethernet (or optical link) under PVM, a NEC-SX3/11, an Intel iPSC/860-Hypercube with 32 nodes and an Intel Paragon XP-10/S with 64 nodes, a Thinking Machines CM–5 with 32 nodes and a Parsytec GCel-3 with 1024 nodes.

The SX3 is a very advanced example of the traditional vector-computers, comparable to the CRAYs. Its power mostly stems from a combination of vectorization and pipelining, the former meaning that data which lies in some regular way in memory can be treated without loosing time for loading and storing, and the latter meaning that the output from one operation can directly be fed into some other unit which performs the next operation, and which works simultaneously with the first unit. For practical purposes, these machines may be seen as SIMD (single instruction multiple data) machines.

Microprocessors such as those found in workstations or in personal computers are increasingly more powerful and sophisticated during the recent years. Therefore, an obvious idea to obtain high performance is to collect many of these microprocessors into so-called parallel computers. All processors act largely independently (MIMD: Multiple Instruction, Multiple Data). A standard method of information exchange between these different processors is "message passing"; a processor can send out a message containing some data to another processor, but the message is only received when the receiving node explicitly issues a receive command. These message passing commands are added to standard Fortran or C. The Parsytec GCel-3 and the Intel iPSC/860 Hypercube are two examples of this type, the first one being a massively parallel machine containing 1024 relatively slow processors, and the second one modestly parallel with 32 workstation-like CPU's. The Paragon is (for the purpose of the following investigation) similar to the iPSC.

These architectures behave in many respects like coupled workstations; and it is indeed possible to use coupled workstations as a parallel machine. Software packages like PVM ('parallel virtual machine')[2] offer message passing routines to be included into standard Fortran or C programs, and these messages are transmitted e.g. via the standard Ethernet. However, besides the slow speed of the standard networks compared to those of the dedicated parallel machines, they encounter a more serious principal problem: Networks like Ethernet or even FDDI (optical link) support only one message at a time on the whole network. Adding further machines to a network *reduces* therefore the amount of time each machine can use the network for communication. This is different for dedicated parallel architectures, where adding further processors usually does not change the bandwidth between two (neighboring) processors.

Another machine we used was a CM-5, which has not only a workstation processor on each node, but in addition 4 vector units. If one does not use the vector units, the machine behaves essentially similar to the iPSC/860; but using the vector nodes represents a combination of the traditional vector machines and the new parallel machines. Using the vector nodes involved at that time the use of a data parallel programming language (High Performance Fortran or C$*$).

C. Computational speed

Table I gives computational results for the implementations we tested; further details of the implementations, how they relate to the different machines, and more detailed performance data are given in [16].

When comparing performance data, it is necessary to give the size of the simulated system. This becomes imperative for parallel computers, since too small systems perform poorly because of the communication overhead. All values of Table I have been obtained by simulations of systems of size $L = 10,000$ km ($6,250$ $miles$, $1,333,333$ $sites$) with an average traffic density of 13.4 $vehicles/km$ (8.4 $vehicles/mile$, $134,000$ $vehicles$ in the whole system). This is a system size which seems relevant for applications. Moreover, it is a system size small enough to still fit into memory of our single node machines, but which is at the same time large enough to run relatively efficient on our parallel machines. Quantitatively, this means that both the GCel and the CM-5 were operating at 40% efficiency. A larger system size would be even more efficient.

References in the literature sometimes give a "real time limit" as measure of their model's performance, which then is the *extrapolated* system size (or number of vehicles) where simulation is as fast as reality. As explained above, we found these values practically useless in the area of parallel computing, except when given in conjunction with the system size which has actually been simulated.

In consequence and in order to avoid confusion, our primary table entries are the CPU times we needed on the different machines in order to simulate the system as defined above. For convenience, we also calculated the real time limits in km and in $vehicle\ sec/sec$ from these values. But it should be kept in mind that, if one really simulates system

[2] Persons having access to electronic mail can obtain information on PVM by sending email to `netlib@ornl.gov` with the line *send index from pvm* in the subject field.

sizes near 1 million km on the parallel machines, one will find much better real time limits for these system sizes (e.g. 2 million km instead of 900,000 km on the GCel).

Noteworthy features of the table are: (i) The bit-coded CA-algorithm is far superior over the "intermediate" one on the vectorizing machines (NEC SX-3/11 and CM-5), slightly faster on the workstation-based architectures, and slightly slower on the massively parallel Parsytec GCel-3. (ii) Both algorithms can take good advantage of the parallelism. (iii) The multi-lane-version (which is developed from the "intermediate" algorithm; see below) is, on the workstation, only a factor of about two slower than the corresponding single-lane version.

We are therefore confident to reach, for a realistic network setup, real time limits of 1,700,000 single lane kilometers ($23,000,000$ $veh\, sec/sec$) on 512 nodes of a CM-5, even without using the vector nodes.

V. CONNECTIONS TO FLUID DYNAMICS

The CA model for traffic flow seems fairly crude at the first glance. Yet, the surprisingly good correspondence to traffic measurements (Section III) should have made it clear that, for some reasons, it captures many of the essentials of traffic flow dynamics. This point can be reinforced by rigorous arguments, where one can show correspondences to traditional traffic flow theory.

More technically, sub-cases of the CA correspond to the Lighthill-Whitham theory [17,20,18,21]. This correspondence refers to the so-called hydrodynamical limit, where one smoothes out and averages over the "particle-like" vehicles in a traffic flow.

For the complete CA as defined in Sec. II, no fluid-dynamical limit is known as of now. Yet, the way in which this CA goes beyond the sub-cases are exactly the way in which Payne and others [24,12,10,11,8] go beyond the Lighthill-Whitham theory in fluid-dynamics — that is, by introducing the effect of inertia. This means that vehicles no longer adapt *instantaneously* to the surrounding traffic, but have some delay in doing so, caused by, e.g., reaction times, mass of the car, etc. Surprisingly or maybe not so, both in the CA and the fluid-dynamical description of traffic flow, the dynamics turns out to change in comparable ways after the inclusion of the effect of inertia: Now, the *spontaneous* initiation of jams is possible; and one has "stable" and "instable" laminar regimes, where in the stable regime, all initial disturbances eventually dissolve, whereas in the instable regime, initial disturbances can survive forever. In between is a phase transition, see [19,17,20,18,10,11].

Yet, there is a significant difference between the CA and the fluid-dynamical description: The CA *directly* includes noise in the behavior modeling of individual elements, simply meaning that the same initial condition can develop into different outcomes. Most notably, in the unstable regime, an initial disturbance *can* dissolve, whereas in the fluid-dynamical model, it will always develop into a fully-grown jam. In a more technical sense, *averaging* over all realizations of the CA starting from the same initial condition will lead to the fluid-dynamical picture.

Thus, one can safely argue that the CA reproduces all dynamical phenomena which are reproduced by the best recent fluid-dynamical models for traffic flow. Yet, the fluid-dynamical models give only an *average* behavior, whereas the CA gives individual *realizations* of this behavior, which lead *only after averaging* to the fluid-dynamical behavior. This allows the representation of *individuals* using no more information than is required for the representation of aggregate populations (the rules for all "particles" are the same). Iterated local interactions with random noise terms have the power to evolve detailed properties of the individuals' time series.

In addition, the CA includes *microscopic* elements in a straightforward way, which will be important for, e.g., multi-lane traffic with different vehicle types, or when using trip plans.

VI. CONNECTIONS TO MICROSCOPIC DRIVING

Also the details of the relationship of the local lattice site transition rules to some kind of driver control model can be made clearer [3].

Initially, site transition algorithms such as the N-S rules were considered "low fidelity driving models". Issues and problems in applications [4] and research [25,2] have led to a reconsideration of this view. We now view the N-S rules as generator functions that, in interaction with the lattice, the state update algorithm, and in local interaction among themselves, give rise to an intermediate emergent phenomenon that is justifiably called a local vehicle control system (see Fig. 3). That is, the "driver" emerges between the levels of the generators and the global traffic phenomenology. The driver logic in this sense is a much higher fidelity representation than was previously suspected. This kind of an emergent dynamical hierarchy in simulation is of considerable theoretical and practical interest in its own right and apart from any issue in traffic science [1,25].

The identification of the controller is, however, a non-unique determination. By parsimony arguments and with some semantic understanding of what constitutes a driver, we believe that the noise term in N-S rules are probably best understood as serving to elaborate the controller in the diagram in Fig. 3, rather than, for example, elaborating a model of the sensory and state estimation errors of the driver. Clearly, we could code such logic explicitly in an encapsulated software actor-object in the object architecture implied by Fig. 3. Such an object could be considered a driving controller object and, as such, a simulation employing these objects would produce the same global traffic behavior with transparently verifiable driver assumptions. This accessibility would be offset by a decrease in computational efficiency and, depending on the detail of the derived driving logic model, an intractable data collection problem for the validation process of the human-like controller.

VII. MULTI-LANE TRAFFIC

As argued above, the CA corresponds to advanced fluid-dynamical models of traffic flow. Since these fluid-dynamical models are also used for multi-lane traffic, one can safely argue that already a collection of parallel one-lane CA models would be as good as the fluid-dynamical models.

But the CA models can actually do better. One can simply include heuristic lane changing rules and thus obtain a coupling between the different lanes [15,26,27]. One can even, with a little more understanding of the system, chose these lane-changing rules in a way that several macroscopic quantities such as lane usage or lane change rates as functions of the density are reproduced realistically [34,35,22]. By doing this, these models are, at least on the macroscopic level, as good or better than most existing more complicated models [29,14]. One has to keep in mind that this is not due to a more advanced modeling technique, but simply because CA models are easier to handle than more complex models, primarily for two reasons: (i) The space of possible parameter settings is much smaller due to their discrete nature. (ii) Since they run fast on computers, trying out different parameter settings and obtaining statistically valid output is much faster.

Thus, we have two more arguments why CA models for traffic are useful:

- With respect to fluid-dynamical models for traffic flow, a CA multi lane model is better because it *explicitly* represents multi-lane traffic. This is for example important when certain types of vehicles (trucks) are slower than others and thus platoons of these types move differently in the traffic system.

- With respect to higher fidelity microscopic models, CA multi lane models are, as of now, at least comparably good as other microscopic methods simply because the research with respect to reproducing macroscopic quantities is more advanced.

VIII. NETWORK IMPLEMENTATIONS

The CA approach to traffic has been used for several road network implementations so far. Rickert has implemented fairly realistic simulations both of the freeway network of the German land Northrhine-Westfalia (NRW) and of whole (western) Germany, using realistic modules for ramps and intersections [26], thus showing the general feasibility and the computational speed advantage of this approach. Nagel has implemented a simplified version of the same NRW road network, but this time using trip plans, that is, each car follows its own route [17]. This simulation used heuristic decision rules, where individual drivers made their route choice based on previous experiences. Both the TRANSIMS project and the "Verkehrsverbund NRW" are currently working on more realistic and more operational implementations of this.

Rickert made thorough investigations of the computational speed which can be reached on parallel machines with these implementations. He could, for example, run a version with two directional lanes[3] of the freeway network of western Germany (12 000 km $\times$ 2 directions $\times$ 2 lanes = 48 000 km; 660 000 vehicles simultaneously in the system) in 10 times faster than real time on an Intel Paragon XP-10/S with 64 computational nodes. Extrapolated this means real time limits of 480 000 km and 6 600 000 veh sec / sec.

Some comparable simulation projects—that is, projects which simulate large scale traffic systems *microscopically*, i.e. resolved to individual vehicles—are the traditional TRAF/NETSIM [31,13], PARAMICS [23], TRANSIMS (see below), plus a thesis by Schütt [28]. The fastest implementation of TRAF/NETSIM reaches a real time limit of 1000 km on a Cray XMP; PARAMICS reaches 250 000 veh sec / sec on a 16k Connection Machine 200 with 512 Floating point units and 360 000 veh sec / sec on 32 nodes of a Cray T3D; Schütt reaches 222 km on a 86286 PC with 10 MHz. In summary, we can say that our own implementations are at least a factor of 10 faster than any other implementation so far that we are aware of.

Nagel's work was more directed towards showing the general feasibility of a *trip-plan based* approach, where drivers use individual decision rules based on past "simulated" experience. Roughly, for a plot like Fig. 4, the following was done in the simulation:

- At the beginning of each "period" ($\approx$ rush "hour"), there were 20 ordered queues of vehicles with drivers waiting to enter into the network. Each queue consisted of 2000 vehicles.
- Each simulated driver had an individual destination, and the set of the 10 shortest paths to that destination to select from.
- Each driver randomly selected a yet untried path; or in case all paths had been tried, he/she selected the path which had performed best in the past (with a small random chance to try something else again).
- The simulation is executed, with each driver following his/her path. If too many vehicles attempt to use the same road section, this creates congestion such as in Fig. 4.

---

[3] Better data was not available to us at that time.

This scheme was executed for several consecutive periods, until the congestion pattern "relaxed" to a pattern which did not change much from one period to the next. This was usually reached after simulating 15 periods; note that 10 periods were necessary until each driver had tried each of his/her options.

This work also showed the robustness of the results against changing of the trip-plan distributions, which gives raise to the hope that traffic patterns such as jams to a large extent are robust against the microscopic details supposedly causing them. Or more intuitively: If a certain route is simply advantageous for a lot of connections, it will always be congested.

Nagel's work also allows some evaluation of the computational speed. He used a single-lane version of the whole freeway network of NRW, i.e. 2 000 km x 2 directions = 4 000 km, and ran that, filled with 40 000 vehicles, on a network of two SUN Sparc 10 workstations which were coupled via optical link using PVM. The whole simulation ran approximately 5 times faster than real time. Simulation of the 15 periods until relaxation took somewhat less than 24 hours on this system. Thus, though laborious, it is possible to run this kind of *microscopic* experiments on fairly modest computational equipment — which, as we think, opens new ways for computational experiments such as done by Emmerink and coworkers [5].

IX. TRANSIMS

A. TRANSIMS overview

The TRansportation ANalysis and SIMulation System (TRANSIMS) is part of the multi-track Travel Model Improvement Program sponsored by the U.S. Department of Transportation and the Environmental Protection Agency. Los Alamos National Laboratory is leading its development. TRANSIMS will address issues resulting from the Intermodal Surface Transportation and Efficiency Act of 1991, such as considerations of land use policies, intermodal connectivity, and enhanced transit service. It will support analysis of potential responses to the stringent air-quality requirements of the Clear Air Act Amendments of 1990.

The TRANSIMS project objective is to develop a set of mutually supporting realistic simulations, models, and data bases that employ advanced computational and analytical techniques to create an integrated regional transportation systems analysis environment. By applying forefront technologies and methods, it will simulate the dynamic details that contribute to the complexity inherent in today's and tomorrow's transportation issues. The integrated results from the detailed simulations will support transportation planners, engineers, and others who must address environmental pollution, energy consumption, traffic congestion, land use planning, traffic safety, intelligent vehicle efficacies, and the transportation infrastructure effect on the quality of life, productivity, and economy.

Fig. 5 illustrates the TRANSIMS architecture [32]. The TRANSIMS methods deal with individual behavioral units and proceed through several steps to estimate travel.

TRANSIMS predicts trips for individual households, residents, freight loads, and vehicles rather than for zonal aggregations of households. The Household and Commercial Activity Disaggregation Module creates regional synthetic populations from census and other data (Synthetic Populations submodule). Using activity-based methods and other techniques, it then produces a travel representation of each household and traveler (Activity Demand and Travel Behavior submodule).

The Intermodal Route Planner involves using a demographically defined travel cost decision model particular to each traveler. Vehicle and mode availability are represented and mode choice decisions are made during route plan generation. The method estimates desired trips not made, induced travel, and peak load spreading. This allows evaluation of different transportation control measures and travel demand measures on trip planning behaviors.

The Travel Microsimulation executes the generated trips on the transportation network to predict the performance of individual vehicles and the transportation system. It attempts to execute every individual's travel itinerary in the region. For example, every passenger vehicle has a driver whose driving logic attempts to execute the plan, accelerates or decelerates the car, or passes as appropriate in traffic on the roadway network.

The Travel Microsimulation produces traffic information for the Environmental Models and Simulations to estimate motor vehicle fuel use, emissions, dispersion, transport, air chemistry, meteorology, visibility, and resultant air quality. The emissions model accounts for both moving and stationary vehicles. The regional meteorological model for atmospheric circulation is supplemented by a model for local effects. The dispersion model is used for directly emitted contaminants and handles both local and urban scale problems. The air chemistry model includes dispersion, but is designed to deal with secondary pollutant production on larger scales.

The TRANSIMS team will develop an interim operational capability (IOC) for each major TRANSIMS module during the five-year program. The Travel Microsimulation will be the first IOC. To the extent possible the first Travel Microsimulation IOC will rely on data currently existing at Municipal Planning Organizations (MPOs). Techniques to incorporate these data readily into the TRANSIMS methodology will be developed.

B. TRANSIMS architecture

Before the TRANSIMS team began development specific to the Traffic Microsimulation IOC, it established the overall TRANSIMS software framework and architecture. This comprehensive planning and design effort should result in a flexible, robust structure for research and development of future TRANSIMS capabilities. The methods for the later IOCs should be implemented more quickly within this architecture without significant design changes. To provide computational speed and an operational capability available to end users in the near term, the TRANSIMS team is developing the first IOC to be distributed for parallel computation on a network of SUN workstations.

The resulting framework, shown in Fig. 6, is a layered architecture with the following layers [32]:

- **Application:** The Analyst Toolbox, which provides a centralized interface between the user and TRANSIMS.
- **System:** Interim Household and Commercial Activity Disaggregation, Interim Intermodal Route Planner, High Speed Cellular Automata Microsimulation, other microsimulations, Input Editor, and Output Visualizer centralize access to the major functional components of TRANSIMS. Additional systems will be added for future IOCs.
- **High-Level Subsystem:** Ten subsystems have been identified that provide services to one or more of the TRANSIMS systems. The subsystems enhance the re-usability and flexibility of the software.
- **Low-Level Subsystem:** Six representational subsystems provide basic services (data and operations on data) to the high-level subsystems. They provide a common representation of objects such as vehicles, travelers, the transportation network, etc.
- **Utility Subsystem:** The utility subsystems provide basic domain-independent services to the higher level components of TRANSIMS. They isolate the domain subsystems from dependence on operating systems, file systems, etc.

C. TRANSIMS microsimulation

The TRANSIMS Travel Microsimulation module mimics the movement and interactions of travelers throughout a metropolitan region's transportation system. For this discussion, traveler refers both to human travelers as well as freight loads, etc. The Intermodal Route Planner provides a trip plan to each traveler that he then attempts to execute on the transportation network. In the process he interacts with other travelers and the transportation system. The combined traveler interactions produce emergent behaviors such as traffic congestion.

The TRANSIMS Travel Microsimulation models many transportation modes including automobiles, trucks, buses, light rail, commuter rail, bicycles, and pedestrians. Thus, the microsimulation includes roadway, transit, rail, bikeway, and pedestrian networks. The roadway network includes freeways, highways, streets, ramps, turn lanes, grades, and intersections (signalized or unsignalized). In executing their trip plans, vehicle drivers accelerate, decelerate, turn, change lanes, pass, and respond to other vehicles and signs and signals. Drivers exhibit behavior between aggressive and passive. Vehicles have weight and acceleration and deceleration characteristics. Analysis requirements determine the necessary microsimulation detail.

Increasing the microsimulation's detail increases its behavioral representation of real transportation systems, but it also increases its computational burden. The representation quality is called the model's fidelity. One goal is to find the minimum computational detail necessary to produce the fidelity needed for specific analysis. This minimum computational detail is called critical complexity.

The TRANSIMS research team is studying two approaches to the microsimulation.

In the first approach, the links (roadway segments) of the network representation of the transportation system are a continuous domain. A vehicle can be positioned along any point on the segment. The vehicle driver evaluates the current situation and decides his next action that advances the vehicle to a new position. The vehicle and driver objects retain their characteristics as they move through the network.

The second approach is to use the cellular automata (CA) microsimulation. A CA microsimulation is lower fidelity, but provides a means to simulate large numbers of vehicles and maintain a fast execution speed. Increasing the fidelity by decreasing the cell size, adding vehicle attributes, and expanding the rule set results in slower computational speed.

An early demonstration version of the TRANSIMS approach, designed for the town of Albuquerque, only used a demonstration version of the first approach. This demonstration project showed both that the methods designed for TRANSIMS do principally work and that the approach was computationally much too slow, even given expected hardware improvements. For that reason, the current TRANSIMS Microsimulation development concentrates on the CA algorithm in order to be able to compute (an "relax") large scale transportation problems *at all.* A high fidelity but computationally much slower microsimulation will probably be added at a later time. Yet it needs to be repeated that the CA is not as low fidelity as it seems: We know quite a bit about the connection to its macroscopic, fluid-dynamical properties (cf. Sec. V); and we can derive the microscopic CA driving model as an adaptively compensated derivative feedback control system (cf. Sec. VI).

And another point needs to be stressed here. TRANSIMS is not a CA-based, parallel high performance computing project *per se*, but it is driven to using these methods to solve the computational problems which arise with its approach.

X. SUMMARY

We have presented a class of very granular microscopic traffic simulation models based on very simple rules describing driving behavior. Nevertheless, this model proves to yield astonishingly realistic behavior. This observation can be backed up by theory: Relations to fluid-dynamical models show that the *macroscopic* quantities of the model are similar to those of fluid-dynamical models for traffic; and by deriving the CA driving logic as an adaptively compensated derivative feedback control system, we can establish the connection of the CA model to higher fidelity microscopic approaches. Extensions to multi-lane traffic are straightforward and seem to work well. We have implemented two different codings of the model on six different computer architectures, showing that, due to its simplicity, the model runs efficient on all of these machines. We have achieved real time limits of more than 4 million kilometers (or more than 53 million vehicle sec/sec). We further described two implementations of the freeway network of the German land Northrhine-Westfalia: One fairly realistic, especially with respect to junctions etc., which established the usefulness of the method for road network simulations; another one less realistic but including individual route plans and decision dynamics, thus showing that it is very well possible to do such investigations on the *microscopic* level (i.e. on the level of the traveler). Finally, we discussed the TRANSIMS microsimulation design, where such a simple high-speed microsimulation will be used as a traffic representation in many or most analysis settings envisioned for that system.

XI. ACKNOWLEDGMENTS

Much of the TRANSIMS description is an abridged version of [32].

---

[1] N. Baas, Emergence, Hierarchies, and Hyperstructures, *Artificial Life III*, edited by C.G. Langton, Santa Fe Institute Studies in the Sciences of Complexity XVII (Addison-Wesley, 1994).

[2] N. Baas (personal communication).

[3] C.L. Barrett, M. Wolinsky, and M.W. Olesen, Emergent local control properties in particle hopping traffic simulations (extended abstract), in *Traffic and Granular Flow, Proceedings of a workshop in Jülich Oct 9-11, 1995*, edited by A. Bachem, M. Schreckenberg, and D.E. Wolf (World Scientific, in press). [4]

[4] C.L. Barrett, S. Eubank, K. Nagel, J. Riordan, and M. Wolinsky, Issues in the representation of traffic using multi-resolution cellular automata, Los Alamos Unclassified Report LA-UR 95-2658 (1995).

[5] R.H.M. Emmerink, K.W. Axhausen, P. Nijkamp, and P. Rietveld, The potential of information provision in road transport networks with non-recurrent congestion, Paper TI 94-30 (Tinbergen Institute, Roetersstraat 11, 1018 WB Amsterdam, The Netherlands, 1994); R.H.M. Emmerink, K.W. Axhausen, P. Nijkamp, and P. Rietveld, Effects of information in road transport networks with recurrent congestion, Transportation **22**, 21 (1995).

[6] D.L. Gerlough and M.J. Huber, Traffic Flow Theory, Special Report 165 (Transportation Research Board, National Research Council, Washington, D.C., 1975).

---

[4] Most papers from the TRANSIMS team can be retrieved via the World Wide Web at `http://www-transims.tsasa.lanl.gov/misc/` or `http://www-transims.tsasa.lanl.gov/research_team/papers/`.

[7] D. Helbing, A fluid-dynamic model for the movement of pedestrians, Complex Systems **6** 391 (1992).

[8] D. Helbing, Improved fluid-dynamic model for vehicular traffic, Phys. Rev. E **51**(4), 3164 (1995)

[9] D. Helbing, Theoretical foundation of macroscopic traffic models, Physica A (in print, 1995) ; D. Helbing, High-fidelity macroscopic traffic equations, Physica A (in print, 1995) ; D. Helbing, Gas-kinetic derivation of Navier-Stokes-like traffic equations (preprint, 1995) ; D. Helbing, Modeling multi-lane traffic flow with queueing effects (preprint, 1995).

[10] B.S. Kerner and P. Konhäuser, Cluster effect in initially homogenous traffic flow, Phys. Rev. E **48**(4), R2335 (1993).

[11] B.S. Kerner and P. Konhäuser, Structure and parameters of clusters in traffic flow, Phys. Rev. E **50**(1) 54 (1994).

[12] R. Kühne, Traffic patterns in unstable traffic flow on freeways, in *Highway Capacity and Level of Service*, edited by U. Brannolte, (Balkema, Rotterdam, 1991); R. Kühne, Freeway Speed Distribution and Acceleration Noise – Calculations from a stochastic continuum theory and comparison with measurements, in *Proc. Int. Symp. on Transpn. and Traffic Theory* (MIT press, Cambridge, 1987); R. Kühne, Macroscopic freeway model for dense traffic stop-start waves and incident detection, *9th Int. Symp. Transpn. Traffic Theory* (VNU Science Press, 1984) ; R. Kuehne and R. Beckschulte, Non-linearity stochastics of unstable traffic flow, *Proceedings of 12th Int. Symposium on the Theory of Traffic Flow and Transportation* (Elsevier, Amsterdam, The Netherlands, 1993), p. 367.

[13] H.S. Mahmassani, R. Jayakrishnan, and R. Herman, Network traffic flow theory: Microscopic simulation experiments on supercomputers, Transpn. Res. A **24A**(2), 149 (1990).

[14] M. McDonald and M.A. Brackstone, Simulation of lane usage characteristics on 3 lane motorways, Paper No. 94AT051, in *Proc. 27th International Symposium on Automotive Technology and Automation (ISATA)* (Automotive Automation Ltd, Croydon, England, 1994), p. 365.

[15] T. Nagatani, Self-organization and phase transition in traffic-flow model of a two-lane roadway, J. Phys. A **26**, L781-L787 (1993); T. Nagatani, Traffic jam and shock formation in stochastic traffic-flow model of a two-lane roadway, J. Phys. Soc. Japan **63**(1), 51 (1994); T. Nagatani, Dynamical jamming transition induced by a car accident in traffic-flow model of a two-lane roadway, Physica A **202**, 449 (1994).

[16] K. Nagel and M. Schreckenberg, A cellular automaton model for freeway traffic, J. Phys. I France **2**, 2221 (1992).

[17] K. Nagel, High-speed microsimulations of traffic flow, Ph.D. thesis (University of Cologne, 1995).

[18] K. Nagel, Fluid-dynamical vs. particle hopping models for traffic flow, in *Traffic and Granular Flow, Proceedings of a workshop in Jülich Oct 9-11, 1995*, edited by A. Bachem, M. Schreckenberg, and D.E. Wolf (World Scientific, in press).

[19] K. Nagel and M. Paczuski, Emergent traffic jams, Phys. Rev. E **51**, 2909 (1995).

[20] K. Nagel, Particle hopping models and traffic flow theory, Los Alamos Unclassified Report 95-2908 (1995), Phys. Rev. E (submitted).

[21] K. Nagel, Particle hopping models, traffic flow theory, and traffic jam dynamics, Los Alamos Unclassified Report 95-2658, Transportation Research Board meeting 1996 (submitted).

[22] K. Nagel, D.E. Wolf, P. Wagner (in preparation).

[23] G.D.B. Cameron and C.I.D. Duncan, PARAMICS–Parallel microscopic simulation of road traffic, J. Supercomputing (in press); C.I.D. Duncan, PARAMICS wide area microscopic simulation of ATT and traffic management, Paper No. 95ATS044, in *Proceedings of the 28th International Symposium on Automotive Technology and Automation (ISATA)*, edited by J.I. Soliman and D. Roller (Automotive Automation Ltd, Croydon, England, 1995), p. 475; D. McArthur, Parallel microscopic simulation of traffic on the Scottish trunk road network (preprint 1994); B.J.N. Wylie, D. McArthur, and M.D. Brown, PARAMICS parallelisation schemes, Report EPCC-PARAMICS-CT.10 (Edingburgh Parallel Computing Centre, University of Edingburgh, Edingburgh EH9 3JZ, Scotland, 1992); see also `http://www.epcc.ed.ac.uk/paramics/`.

[24] H.J. Payne, Models of freeway traffic and control, Mathematical Models of Public Systems, Vol.1, (Simulation Council, La Jolla, CA), p. 51; H.J. Payne, FREEFLO: A macroscopic simulation model of freeway traffic, Transportation Research Record 722 (Urban System Operations, National Academy of Sciences, Washington D.C., 1979); H.J. Payne, A critical review of a macroscopic freeway model, in *Proc. Conf. on Research directions in computer control of urban traffic systems*,

1979, edited by W.S. Levine, E. Lieberman, and J.J. Fearnsides (American Society of Civil Engineers, New York, 1979), p. 251.

[25] S. Rasmussen and C.L. Barrett, Towards a Theory of Simulation, in *Advances in artificial life*, edited by F. Moran et al., Lecture Notes in Computer Science No. 929 (Springer, 1995), p. 515.

[26] M. Rickert, Simulationen zweispurigen Autobahnverkehrs mit Zellularautomaten, Master Thesis (University of Cologne, 1994).

[27] M. Rickert, K. Nagel, M. Schreckenberg, A. Latour, Two lane traffic simulations using cellular automata, Physica A (in press).

[28] H. Schütt, Entwicklung und Erprobung eines sehr schnellen, bitorientierten Verkehrssimulationssystems für Straßennetze (Schriftenreihe der AG Automatisierungstechnik TU Hamburg-Harburg No. 6, 1991).

[29] U. Sparmann, Spurwechselvorgänge auf zweispurigen BAB-Richtungsfahrbahnen, Straßenbau und Straßenverkehrstechnik Vol. 263 (Bundesministerium für Verkehr, Bonn, 1978).

[30] D. Stauffer, Computer simulations of cellular automata, J. Phys. A **24**, 909 (1991).

[31] TRAF User reference guide, Publication No. FHWA-RD-92-060 (U.S.Department of Transportation, Federal Highway Administration, 1992).

[32] L. Smith, R. Beckman, K. Baggerly, D. Anson, and M. Williams, Overview of TRANSIMS, the TRansportation ANalysis and SIMulation System (combined one-pagers), Los Alamos Unclassified Report LA-UR 95-1641 (Los Alamos National Laboratory, Los Alamos NM 87545, USA, 1995); L. Smith, R. Beckman, D. Anson, K. Nagel, and M. Williams, TRANSIMS: TRansportation ANalysis and SIMulation System, *Proc. 5th Nat. Transportation Planning Methods Applications Conference*, Seattle, Washington, April 17-21, 1995, LA-UR 95-1664; J. Morrison and V. Loose, TRANSIMS model design criteria as derived from federal legislation, LA-UR 95-1909 (1995). ; C. Barrett, K. Berkbigler, L. Smith, V. Loose, R. Beckman, J. Davis, D. Roberts, and M. Williams, *An operational description of TRANSIMS*, LA-UR 95-2393 (1995); TRANSIMS Travelogue, Travel model improvement program newsletter Issue No. 2 (Texas Transportation Institute, 1600 E Lamar Blvd # 112, Arlington TX 76011, January 1995), p. 9; TRANSIMS Travelogue, Travel model improvement program newsletter Issue No. 3 (July 1995), p. 10; TRANSIMS Travelogue, Travel model improvement program newsletter Issue No. 4 (November 1995), p. 9.

[33] J. Treiterer, Investigation of Traffic Dynamics by Aerial Ohoi, 1975).

[34] P. Wagner, Traffic simulations using cellular automata: Comparison with reality, in *Traffic and Granular Flow, Proceedings of a workshop in Jülich Oct 9-11, 1995*, edited by A. Bachem, M. Schreckenberg, and D.E. Wolf (World Scientific, in press).

[35] P. Wagner, K. Nagel, and D.E. Wolf (in preparation).

[36] H. Zackor, R. Kühne, W. Balz, *Untersuchungen des Verkehrsablaufs im Bereich der Leistungsfähigkeit und bei instabilem Fluß*, Forschung Straßenbau und Straßenverkehrstechnik 524 (Bundesminister für Verkehr, Bonn–Bad Godesberg, 1988).

TABLE I. Computing speed of different algorithms on different computer architectures. "s(ingle) bit", "particle", "intermed(iate)", and "netw(ork)" mean the corresponding algorithms described in the text. For each machine and algorithm, the first table entry gives the time each computer needed to simulate a system of size $10,000\ km$. From this number, we derive the other three entries. The second entry is the computational speed in Mega Updates Per Second, and the third and fourth are real time limits in km and in vehicle sec / sec. For further details see text. The entries for network traffic are due to M. Rickert (personal communication).

FIG. 1. Evolution of the model from random initial conditions. Each black pixel represents a vehicle. Space direction is horizontal, time is pointing downwards, vehicles move to the right. The simulation was of a system of size $L = 10000$ with density $\rho \approx \rho(q_{max}) \approx 0.08$; the figure shows the first 1000 iterations in a window of $l = 1000$.

FIG. 2. *Top:* Fundamental diagram of the model (throughput versus density). Triangles: Averages over short times (200 iterations) in a sufficiently large system ($L = 10,000$). Solid line: Long time averages (10^6 iterations) in a large system ($L = 10,000$). Dashed line: Long time averages (10^6 iterations) for a small system $L = 100$. *Bottom:* Fluctuations of local velocity (see text) vs. density.

FIG. 3. Adaptively compensated, derivative feedback control architecture for the N-S rules.

FIG. 4. Simulated traffic jams in a single-lane implementation of the freeway network of the German land Northrhine-Westfalia (NRW). Situation at "day 16" after 6000 iterations (100 minutes).

FIG. 5. TRANSIMS architecture

FIG. 6. TRANSIMS software framework

	s.bit (F77)	particle (F77)	intermed. (C)	netw. (C)
Sparc10	0.33 sec		0.71 sec	
	4.0 MUPS	3.4 MUPS	1.9 MUPS	1.2 MUPS
	30 000 km		14 000 km	8 800 km
	0.4 e 6 veh		0.19 e 6 veh	
PVM	0.07 sec		0.15 sec	
(5× Sp10)	19.0 MUPS		8.9 MUPS	
	140 000 km		65 000 km	
	1.9 e 6 veh		0.87 e 6 veh	
SX-3/11(1)	0.0025 sec		0.48 sec	
1 CPN	**533 MUPS**		**2.8 MUPS**	
	4 000 000 km		21 000 km	
	53 e 6 veh		0.28 e 6 veh	
GCel-3	0.013 sec		0.011 sec	
1024 CPNs	102 MUPS	211 MUPS	121 MUPS	
	750 000 km	1 550 000 km	900 000 km	
	10 e 6 veh		12 e 6 veh	
iPSC	0.016 sec		0.038 sec	
32 CPNs	83 MUPS		35 MUPS	
	630 000 km		260 000 km	
	8 e 6 veh		3.5 e 6 veh	
Paragon				
64 CPNs				6.6 e 6 veh
				480 000 km
CM-5(1)	0.0077 sec(2)		0.045 sec(3)	
32 CPNs	**173 MUPS**		**30 MUPS**	
	1 300 300 km		220 000 km	
	17 e 6 veh		2.9 e 6 veh	
CM-5(1)				
1024 CPNs				
				[> 1.7 e 6 km]

(1) CPN(s) has/have vector units (SIMD instruction set)

(2) using data parallel Fortran (CMF)

(3) using message passing (CMMD)