Traffic Models of a 2D road network [1]

Bastien Chopard, Pierre-Antoine Queloz and Pascal O. Luthi

Département d'Informatique, University of Geneva
24 rue Général-Dufour, 1211 Genève 4, Switzerland
email: [chopard/queloz/luthi]@cui.unige.ch

Abstract

We consider two different traffic flow models and their implementation on a Connection Machine CM-200. The first approach is based on a cellular automata rule and allows north, east, west and south car motions. Road junctions are represented in a satisfactory way, as rotaries. In a second approach, we consider a fully continuous traffic flow simulator. Congestion and queue formation are studied within both frameworks, in the case of a Manhattan street network.

1 Introduction

Simulation of road traffic is currently receiving a great deal of attention. On one hand, the management of traffic in urban environment is crucial for transportation policy and road planning. On the other hand, the collective behavior of a large number of vehicles mutually interacting is quite interesting from the physical point of view.

Many researchers have proposed simple, fully discrete traffic models, based on a cellular automata (CA) dynamics[1]-[5]. Concepts such as phase transitions and critical densities can be used to describe jamming in a traffic system.

Whereas much work has been done on one-dimensional (single lane) CA traffic, the situation of a road network (typically a Manhattan-like city) has not been investigated in detail from the point of view of its physical behavior. Current two-dimensional CA traffic models consider a rather unrealistic situation in which no street exits; cars can turn at any site of a 2D grid, and interact in a primitive way in regards to what happens in real road junctions[6, 7].

In this paper we propose a new CA rule[8] to deal with road intersections. Our approach is to represent a junction as a rotary, made up of a two by two block of cells and connected to four entering and four exiting lanes. From the basic rule of motion, several extensions can be considered to model different type of priority (for instance, traffic lights, right priority,...).

In addition to the above CA model, we have also developed a parallel micro-traffic simulation which allows any off-lattice motion and associates to each car its own behavior[9]. Our approach is quite similar to that developed by the authors

[1] Presented at the 3rd European Connection Machine Users Meeting, Parma, Oct. 1995

of PARAMICS[10], at the Edinburgh Parallel Computing Center. However, our algorithms and implementations are not the same.

In the current implementation of our model, each driver is very well behaved and strictly obeys the model regulations: speed limits, priority, distance to the preceding car and so on. Each car responds to its external environment by accelerating or braking and can avoid any crash. For the time being, no passing capabilities are given to the cars. Any type of intersection can be modeled in our system. We have developed a protocol at each road junction in order to resolve the conflicting motion of several incoming cars.

Dealing with a general street configuration represents an irregular problem. As such, it is difficult to parallelize microtraffic modeling, especially on a SIMD platform like the CM-200. We will discuss our implementation, the data structures we have used and the algorithm for managing the intersections.

Using both models (CA and continuous), we have simulated large networks of streets organized in a Manhattan-like fashion (other street layout could be considered, though). We have studied the traffic flow regimes as a function of the average car density in the system and the formation of car queues.

Although the performance of our continuous traffic simulator are encouraging (similar to PARAMICS but one or two orders of magnitude slower than the CA), the simulations are too slow to solve a realistic trip planning optimization problem. However, it is appropriate to study the generic physical properties of car traffic in an urban environment. In a Manhattan-like street network, we have performed several simulations which validate our cellular automata results.

2 The cellular automata model

The basic idea of a cellular automata traffic model is to consider a set of adjacent cells representing the street along which a car can move. The rule of motion is that a car jumps to its nearest neighbor cell, unless it is already occupied by another car. When this is the case, the car stays at rest.

Accordingly, the motion can be expressed by the following general rule

$$n_i(t+1) = n_i^{in}(t)(1 - n_i(t)) + n_i(t)n_i^{out}(t) \tag{1}$$

where $n_i(t)$ is the car occupation number ($n_i = 0$ means a free site, $n_i = 1$ means that a vehicle is present at site i). The quantity $n_i^{in}(t)$ represents the state of the cell from which a car could come and $n_i^{out}(t)$ indicates the state of the destination cell. Rule (1) means that the next state of cell i is 1 if a car is currently present and the next cell is occupied, or if no car is currently present and a car is arriving.

In order to discuss further traffic rule (1), we have to specify how n_i^{in} and n_i^{out} are defined. For a simple one-dimensional periodic road, we simply have

$$n_i^{in} = n_{i-1} \qquad n_i^{out} = n_{i+1} \tag{2}$$

and the microdynamics reduces exactly to rule 184 of Wolfram, used by several authors to model 1D traffic (e.g [1]).

For a two-dimensional road network, these quantities can be generalized as follows. We assume that horizontal roads consist of two lanes, one for eastward motion and the other for westward motion. Similarly, vertical streets are composed of northbound and southbound lanes. Along a road segment, the rule of motion is equations (1), supplemented by (2).

The problem is to define the rule of motion at a road junction. This can be realized in a simple way if one assumes that a rotary is located at each crossing. A vehicle in a rotary has the priority over any entering car.

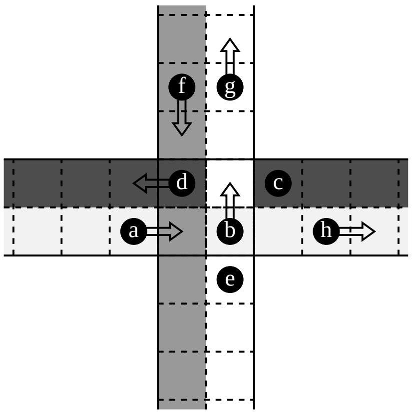


Figure 1: *Example of a traffic configuration near a junction. The four central cells represent a rotary which is traveled counterclockwise. The gray levels indicate the different traffic lanes: white is a northbound lane, light gray an eastbound lane, gray a southbound lane and, finally, dark gray is a westbound lane. The dots labeled a, b ,c, d, e, f, g and h are cars which will move to the destination cell indicated by the arrows. Cars without an arrow are not allowed to move.*

The implementation we propose for this rule is illustrated in figure 1, where a four-corner junction is shown. The four middle cells constitute the rotary. A vehicle on the rotary (like b or d) can either rotate counterclockwise or exit. A local flag F is used to decide of the motion of a car in a rotary. If $F = 0$, the vehicle (like d) exits in the direction allowed by the color of its lane (see figure caption). If $F = 1$, the vehicle moves counterclockwise, like b. The value of the local turn flag F can be updated according to the modeling needs: it can be constant for some amount of time to impose a particular motion at a given junction, completely random, random with some bias to favor a direction of motion, or may change deterministically according to any user specified rule.

Model I In a first model (let us call it model I), a vehicle moves only when its destination cell n^{out} is empty, even in a rotary. In addition, a car wanting to enter the rotary has to check two cells because it has not the priority. For instance, car c cannot enter the rotary because b is going to move to the white cell. Car e cannot either move because it sees b (and cannot know whether or not b will actually move). Car a, on the other hand can enter because it sees that d is leaving the rotary and that the gray cell ahead is free.

Similarly, the incoming vehicle n^{in} to a given cell is computed differently inside and outside the rotary. The light gray cell occupied by car b has two possible inputs: with priority, it is the vehicle from the gray cell at west; but if this cell is empty, the input will be the incoming lane, namely the car labeled e.

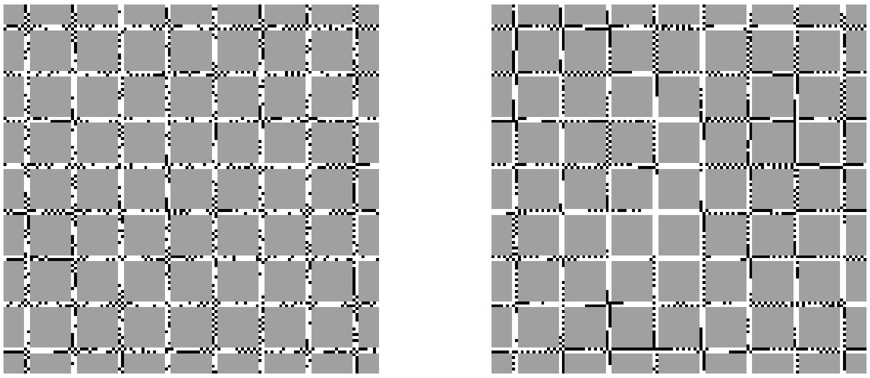
Model II The rule of model I is appropriate when drivers have a random wandering motion, i.e. no given destination, nor preferred route. However, when the drivers decide to take a given rotary exit and do not change their mind, a junction deadlock is likely to occur: if four cars enter simultaneously the same rotary, from north, south, east and west, and all want to keep turning in the rotary, none of them will be able to progress (the destination cells being all occupied). To prevent these problems we have introduced a different rule of motion in a rotary, namely that a car in a rotary will always move (either keep turning or leaving the rotary if the desired exit is free). This is model II, in which a car choose its exit at each junction. In contrast to model I, two cars can now follow each other with a zero headway in a rotary.

Figure 2 shows a typical traffic configurations in a Manhattan-like city, using model I. In figure (a), a vehicle has a probability 1/2 to exit at each rotary cell. In figure (b), the turn flag F has an initial random distribution on the rotary. This distribution is fixed for the first 20 iterations and then flips to $F = 1 - F$ for the next 20 steps an so on. In this way, a junction acts as a kind of traffic light, which for some amount of time, imposes a given flow pattern (e.g right turns on red). We observed that the global traffic pattern is different in the two cases: it is much more homogeneous in the random case and a higher traffic flow is measured.

2.1 Jamming in a street network

In this section we investigate in more detail the behavior of a the road network shown in figure 2 (a) when varying the number of cars in the system. More precisely, we are interested in the average velocity $< v >$ (i.e the average number of cars able to move) as a function of the average car density ρ (i.e. the total number of cars divided by the number of available road cells).

As more cars are added to the system, many of them cannot move at every time steps because, in order to have a free motion, there must be no more than one car every other sites (at least along the road segments connecting two consecutive



(a) (b)

Figure 2: *Traffic configuration after 600 iterations, for a car density of 30%. Streets are white, buildings gray and the black pixels represent the cars. The Situation (a) corresponds to an equally likely behavior at each rotary junction, whereas image (b) mimics the presence of traffic lights. In the second case, queues are more likely to form and the global mobility is less than in the first case.*

intersections). In addition, road junctions hinder the car flow and reduces the capacity of the road segments. Consequently, when traffic density is larger than some threshold, queues start to form before the rotaries.

An interesting aspect of this dynamics is that the system self-organizes in three regions of different car densities:

1. The queues before the intersections of density ρ_j (subscript j stands for jammed) and velocity $< v_j >= (1-\rho_j)/\rho_j$ (because the velocity in a queue is governed by the density of empty site).

2. The "free" road segments right after the intersections, with density ρ_f and velocity $< v_f >= 1$ (all cars can move).

3. The region inside a rotary of average density ρ_r.

From these observations, some simple analytical relations can be derived to describe the system, in a mean-field approximation[1, 8].

Assuming that the traffic flow $J = \rho < v >$ is constant throughout the system and, in particular, at the interface between the three regions

$$J_{queue} = J^{in}_{rotary} = J^{out}_{rotary} = J_{free} \tag{3}$$

we obtain

$$\rho_f = 1 - \rho_j \tag{4}$$

The expressions for J^{in}_{rotary} and J^{out}_{rotary} (which are the probabilities of entering and exiting the intersection, respectively) depend on the rule of motion in a rotary.

For model II, we have

$$J^{in}_{rotary} = \rho_j \frac{1-\rho_r}{2} \qquad J^{out}_{rotary} = (1-\rho_f)\frac{\rho_r}{2} \tag{5}$$

The term ρ_j (respectively $1-\rho_f$) is the probability of having a car ready to enter the rotary (respectively able to leave it); the term ρ_r is the probability of having a car in the rotary; the factor of 1/2 indicates that a car has probability 1/2 to exit the rotary and 1/2 to keep turning in it. The solution of (3) is then

$$\rho_j = \frac{4}{5} \qquad \rho_r = \frac{1}{2} \qquad \rho_f = \frac{1}{5} \tag{6}$$

When more cars are added to the network, the values of ρ_j, ρ_f and ρ_r do not change, but the length ℓ of the queues increases. Assuming that all car queues are of same length, the relation between ℓ and the separation L between two consecutive junctions is

$$4(L-2-\ell)\rho_f + 4\ell\rho_j + 4\rho_r = 4L\rho \tag{7}$$

where L is measured as the network period. In the case of large L, we can approximate this result as

$$\frac{\ell}{L} = \frac{\rho-\rho_f}{\rho_j-\rho_f} \tag{8}$$

which is similar to the result of [1] for a 1D traffic model with reduced capacity.

Equation (8) provides a generic description of jamming in the system:

1. for $\rho \leq \rho_f$ we have $\ell < 0$, which means that no queues are formed and we have a free traffic regime.

2. For $\rho_f < \rho < \rho_j$, ℓ vary from 0 to L. This is the maximum flow regime (when ρ increases, $< v >$ decreases so that $j = \rho < v >= 1/5$ is constant). The queues act as reservoirs which can absorb the traffic excess.

3. For $\rho > \rho_j$, we have $\ell > L$ which indicates that the queues have grown up to the precedent intersection and, therefore, prevent the rotaries to deliver their maximum flow. This is a strongly jammed regime.

From the numerical simulations, we observe a departure of these mean-field predictions, mostly due to the fact that all queues are not of the same length. The fluctuations in time and across the various rotaries cause a very unbalanced traffic distribution over the network and, at a particular time, some regions get much more loaded than the others. Therefore, the transition from maximum flow to strong jamming begins at lower density than predicted by the mean-field derivation.

From the cellular automata simulations, we can conclude that the distance L separating two consecutive junctions, as well as the drivers behavior at the rotaries (i.e model I or II) are determinant for the collective dynamics of the vehicles.

Figure 3 shows the measured average density before, after and at a rotary and the flow diagram. Figure 4 presents the average density profile along the road segments.

In the limit of small L, we obtain a behavior similar to the 2D grid model of [7]. But, having the possibility to form a queue of length $L \neq 0$ is a crucial aspect of the street network model and makes it more like the 1D traffic model of [1].

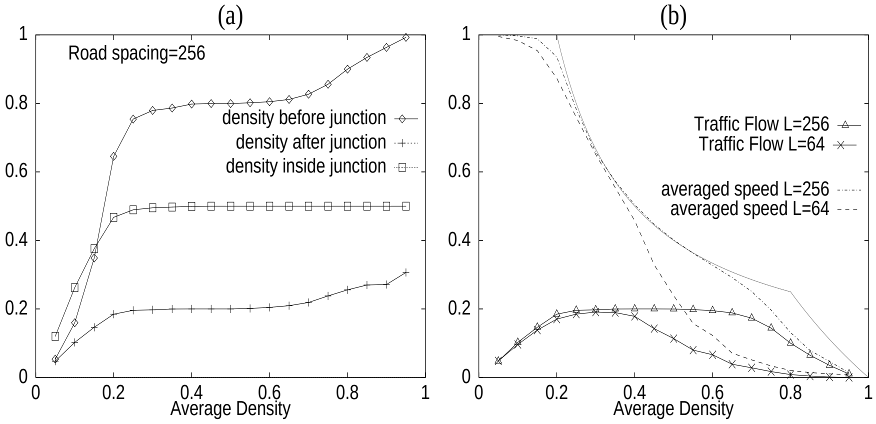


Figure 3: *CA Simulations of Model II, in a Manhattan-like street network. (a) Car density in the three traffic regions (free regions, queues and rotaries), as a function of the average density, for a road spacing $L = 256$. (b) Traffic flow J and average velocity $< v >$ for two different road spacings. The plain solid line represents the mean-field predictions.*

Another difference of 2D road network compared with 1D single lane model of [1] is that the traffic flow is metastable. It can never be excluded that a fluctuation takes place so that a loop of adjacent junctions mutually block each other. This produces a junction deadlock (or gridlock) from which one cannot recover unless a driver deliberately decides to change its route. When one or several junctions get blocked, the problem rapidly propagates to the other intersections and a complete jamming occurs where no car can ever move. Whereas model I is quite sensitive to junction deadlocks, model II is much more robust with this respect. On the typical observation times we have considered, almost no deadlocks took place.

3 Continuous Microtraffic Simulations

Our cellular automata model of urban traffic is numerically simple and quite effective. It mostly requires boolean variables. Thus, the car motion is much

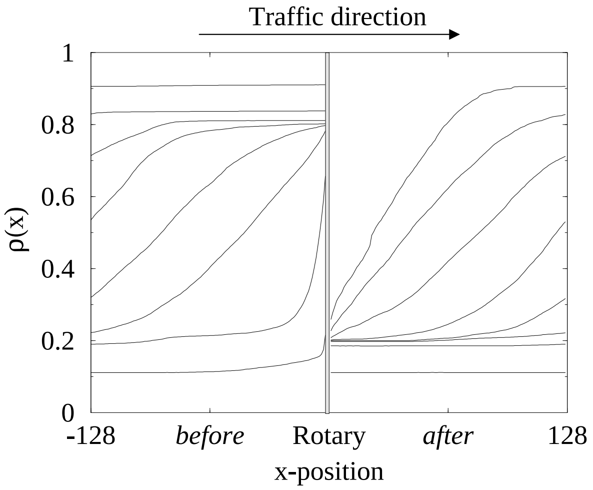


Figure 4: *Time and space average of the car density profile along the segments of the street network. The road spacing is $L = 256$ and the system is periodic with one rotary located at $x = 0$. By space average, we mean that we have summed up the contributions of the four traffic lanes (north, south, east and west). Each of the eight curves corresponds to a different average density, namely* $\rho = 0.1, 0.2, \ldots, 0.8$.

simplified compared with a real situation in which position, velocity and acceleration are real-valued quantities. A important question is how well our idealized discrete CA traffic captures the real case.

In this section, we discuss a continuous microtraffic flow simulator which uses a much more realistic car dynamics. For this reason, it is much more CPU intensive and its parallel implementation less effective due to irregular data structures. However, it can be seen as a reliable model, from which the global properties of street network traffic can be studied.

As mentioned previously, any type of road junction can be implemented in the model, with or without traffic lights. In addition, the divers strictly obey the traffic regulations and cannot pass another car.

3.1 Vehicle representation and dynamics

The main attributes of each vehicle in the system are their length l, maximal acceleration and maximal braking capabilities, α_+ and α_-, respectively. Their position $x(t)$ and speed $v(t)$ are computed repeatedly after some time interval Δt during which the acceleration $a(t)$ (positive or negative) is supposed to be constant. All these quantities are real-valued for more accuracy in comparison with the CA model. Thus, the motion obeys the relation

$$x(t + \Delta t) = x(t) + v(t)\Delta t + a(t)\frac{\Delta t^2}{2}$$

The main point in this simple description is clearly the acceleration, representing the driver's reaction to the perceived situation. We have considered that all drivers are trying to achieve highest speed but with some restrictions: they have to conform to speed limits, traffic lights and junction priorities. They are also very cautious and keep an adequate headway distance so that collisions can be avoided in the case of an emergency braking. When a driver is faced to several constraints, the strongest one prevails to determine the appropriate acceleration.

3.1.1 Acceleration

In our model, we have considered that acceleration results from the driver's intention to adjust his vehicle velocity. We supposed that the diver's response to an environment change is a continuous function of ε, where ε measures the difference between the actual and optimal situation. The next section will show how the optimal situation is computed, according to the various "obstacles" a car may encounter.

Since the acceleration is bounded, we assume that it is can be given by

$$a(t) = \begin{cases} \alpha_+ \tanh(\varepsilon(t)/\alpha_-) & \text{if } \varepsilon \geq 0 \\ \alpha_- \tanh(\varepsilon(t)/\alpha_-) & \text{if } \varepsilon < 0 \end{cases} \tag{9}$$

Provided that $\alpha_+ \leq \alpha_-$ and $\Delta t < 1$, a fast convergence to the desired velocity (characterized by $\varepsilon = 0$) is obtained, without oscillations. Note that the same kind of approach is used in [11] but the authors consider that erratic oscillations around the optimal situation is an interesting ingredient to simulate the unpredictable change of behavior of each driver (i.e velocity fluctuations). Practically, several features imply a modification of the motion:

Speed limits In our model, the road network contains signals like speed limits and traffic lights that the drivers see when passing by and remember afterwards. At each time step, the actual speed $v(t)$ is compared with the recorded speed limit V. The difference $\varepsilon(t) = V - v(t)$ is used to compute the necessary acceleration, according to (9). This behavior leads to a perfect observance of the regulations (this is of course an idealized approximation of the real driver's behavior).

Obstacles In addition to simple speed limits, several obstacles of different kind can limit the vehicle motions:

- Each vehicle can have a predecessor in its own lane; between them, enough distance must be kept, to avoid collisions when the first one is braking.
- When the vehicles are near a junction, they have to stop if the traffic light is red, or if a vehicle with higher priority is coming from another direction;

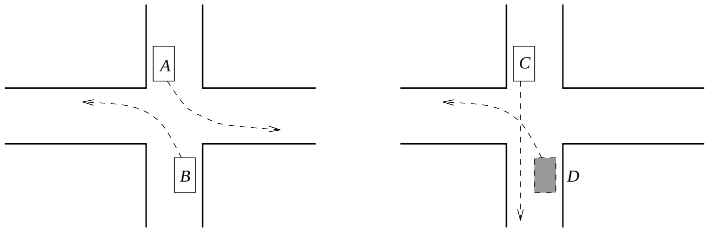


Figure 5: *On the left-hand side, cars A and B can cross but this is not the case of cars C and D, on the right-hand side.*

in both cases, our model assumes there is a motionless obstacle at the end of the lane, just before the junction.

- When they are close to a junction, the vehicles have to choose a destination to continue their trip; the last vehicle in the destination lane can be another obstacle to keep distant from.

Each of these obstacles is represented in the system by its position x_o and its speed v_o. The distance d_o to the obstacle is given by $d_o = x_o - x - l$, where x is the position of the examined vehicle and l its length.

3.1.2 Optimal distance

We have determined that the vehicles are able to stop safely if they start slowing down when

$$d_o < \begin{cases} 2v + (v - v_o)^2/2\alpha_- & \text{if } (v - v_o) > 0 \\ 2v & \text{otherwise} \end{cases} \tag{10}$$

The first term ($2v$) is the distance traveled during two seconds, generally accepted as a sufficient gap between cars with comparable speeds. When the obstacle is slower than the vehicle, more space is necessary and we add the stopping distance in the predecessor's system of coordinates.

Traffic lights suddenly turning red and unexpected prior vehicles at junctions are ignored if the distance to the obstacle d_o is smaller than the vehicle stopping distance $v^2/2\alpha_-$. This reasonable attitude has two requirements : (i) a small period of time is necessary at the crossing during which all lights are red; (ii) oncoming prior vehicles must be seen early enough.

3.2 Intersections

We have developed a protocol to compute the interactions between vehicles simultaneously coming to an intersection. The first point to mention is that not only the origin of the vehicles is important, but that the whole trajectory (origin and target) must be known to compute the priorities (see figure 5). For each

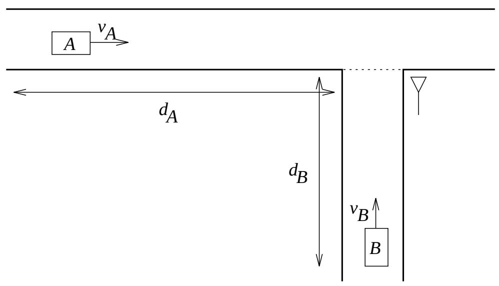


Figure 6: *A prior vehicle A should announce its presence to vehicle B when it gets to a distance d_A from the intersection.*

crossing in the system, priority can be described by a blocking trajectory table, containing every possible origin/destination pairs and their respective level of precedence.

As long as there are cars on a blocking trajectory, all other trajectories with lower priority get blocked. It is worth noticing that the vehicles interacting at a crossing know each other intentions and can exploit the information. This is an ideal situation which does not always hold in everyday's traffic.

The blocking table is a static information. Dynamically, the problem is to estimate whether or not a vehicle has time to cross an intersection before another prior vehicle arrives. Three types of signals are used to announce an intersection: traffic lights, stop signs and priority information. Halting is compulsory at red lights and stop signs. In these cases, vehicles do not need to interact with those arriving from other lanes because they cannot reach the crossing area. When the light turns green or when the vehicle has reached the intersection limits, general priority rules are considered. A typical situation is illustrated in figure 6. Vehicle A, which has priority, should announce early enough its presence to vehicle B. At a suitable distance d_A from the intersection, B is informed of the arrival of A and can stop to prevent a car crash. It can be shown[9] that no collision takes place if

$$d_A \geq V_A(2\Delta t + v_B/\alpha_-)$$

where V_A is the maximum speed allowed for A and v_B the actual speed of B. Similarly, a distance d_B can be determined so that a vehicle B has time to go if its less than d_B from the intersection.

Although this strategy has not been confronted with real traffic data, we have observed its correctness regarding to collisions and we believe it minimizes the waiting time of each car.

Practically, priority management is achieved as follows, in order to determine

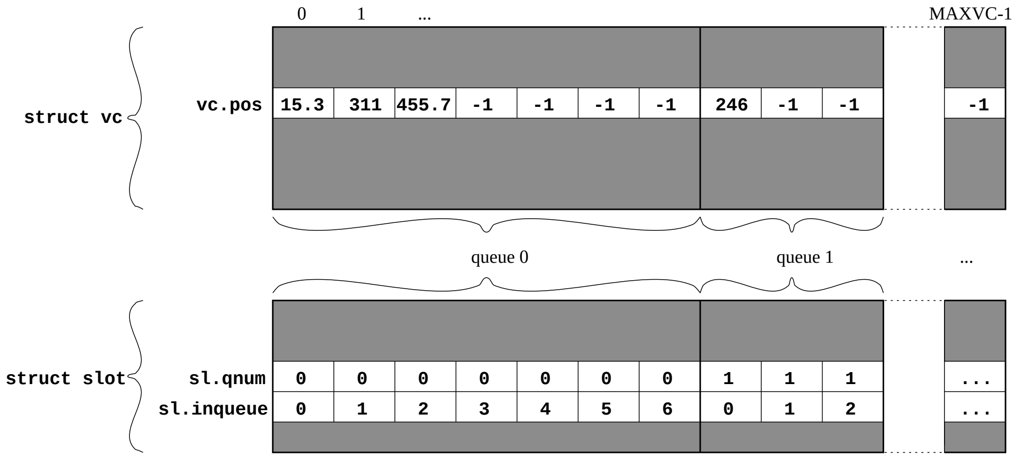


Figure 7: *Data structure used to described the vehicles. A value of -1 in the position slot indicates the absence of a vehicle.*

which vehicle should stop at a crossing: any car, close enough to an intersection, indicates its desired trajectory as triplet of values (i_0, i_1, i_2), i_0 representing an intersection label, i_1 its origin lane and i_2 the destination lane. When several cars register in such a way to the same road junction, the blocking table is looked up, and the non-prior vehicles are informed of the presence of an obstacle which makes them stop.

3.3 Implementation

Our traffic flow simulator has been implemented on a 8k-processor Connection Machine CM-200. This is a SIMD approach, with a synchronous motion update. The program is written in C^*, as the first version of PARAMICS[10], but with a different parallel data structure. The attributes of the vehicles (length, speed, last speed limit recorded,...) are stored in a parallel array of `structures`, as shown in figure 7 and are updated synchronously.

This array has one parallel dimension and each traffic lane is represented by a segment of adjacent array elements. The vehicles within each segment are organized in a first-in first-out order, starting from the leftmost element. They must be shifted each time a new vehicle enters the lane and dynamic reallocation is necessary if a queue develops beyond the number of available slots.

The network description (lanes, signal and intersection) is also stored in the Connection Machine distributed memory to avoid communication with the front-end. In addition, some data are replicated across the memory, in a parallel array similar to that used for the vehicles description, in accordance with the dynamic lane segmentation shown in figure 7. This strategy avoids some expensive general communications[9].

Finally, the implementation of the priority management is first done with a three-dimensional parallel array W of size $n \times k \times l$, where n is the number of junctions in the network and k, l are the maximum number of input and output lanes of the most complex intersection.

Upon arrival, each car registers in this data structure. The blocking table B is represented as a five-dimensional parallel array of size $n \times k \times l \times k \times l$, indicating whether or not trajectory (i_1, i_2) blocks trajectory (i_3, i_4), at intersection i_0. In this way, all trajectories can be analyzed simultaneously, and using the general communication primitive, the cars of lesser priority get notified.

3.3.1 Performance

The advantages of the data distribution we presented in the previous section (i.e. one car per processing element) are (i) a good load balancing when the number of vehicle is close to the number of virtual processors and (ii) the use of the efficient NEWS communication primitives between vehicles traveling along the same lane.

In contrast, the authors of PARAMICS have chosen to store the vehicles in the same queue on the same processing element. This approach needs less general communications and makes multi-lane with passing easier to implement. On the other hand, it may lead to poor load balancing. However, it is interesting to note that no significant difference in the performance of the two implementation are observed, despite the very different data structures and traffic model used.

Due to the irregularity of the data structures, the execution time is dominated by general communications operations. The update rate depends of the problem size. The best performance we have obtained is $4.7\ 10^5$ car updates per second, for a system comprising slightly less than 2^{19} vehicles. Since Δt is typically of the order of 1 second, our simulation can follow about half a million of cars in real time.

3.3.2 Simulation results

We have tested our traffic flow simulator on a 2D Manhattan-like street network. The same qualitative behavior as observed with the automata model is obtained, but with much more flexibility to adjust various parameters (and the possibility to consider much more complex street layout). These results[9] provide a confirmation that a simplified, fully discrete dynamics can well capture the relevant macroscopic behavior of urban traffic.

Acknowledgments

Part of this work was realized at the Institue for Scientific Interchange, under the project "Complexity and Chaos," grant OPES 95.0046.

References

[1] Satoshi Yukawa, Macoto Kikuchi, and Shin ichi Tadaki. Dynamical phase transition in one-dimensional traffic flow model with blockage. *J. Phys. Soc. Jpn*, 63(10):3609–3618, 1994.

[2] M. Schreckenberg, A. Schadschneider, K. Nagel, and N. Ito. Discrete stochastic models for traffic flow. *Phys. Rev. E*, 51:2939, 1995.

[3] A. Schadschneider and M. Schreckenberg. Cellular automaton models and traffic flow. *J. Phys.*, A(26):L679, 1993.

[4] K. Nagel and M. Schreckenberg. Cellular automaton model for freeway traffic. *J. Physique I (Paris)*, 2:2221, 1992.

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

[6] O. Biham, A. A. Middleton, and D. Levine. Self-organization and dynamical transition in traffic-flow models. *Phys. Rev.*, A(46):R6124, 1992.

[7] J. A. Cuesta, F. C. Martínez, J. M. Molera, and A. Sánchez. Phase transition in two-dimensional traffic-flow models. *Phys. Rev.*, E:R4175–78, 1993.

[8] B. Chopard, P. O. Luthi, and P.-A. Queloz. Cellular automata model of car traffic in two-dimensional street networks. *J.Phys. A*, 1996. submitted.

[9] Pierre-Antoine Queloz. Modèle de trafic routier et simulateur massivement parallèle. Master's thesis, University of Geneva, CUI, 1211 Geneva 4, Switzerland, 1995.

[10] G. Duncan, G. Cameron, and S. Druitt. Paramics-mp: a step nearer to vehicle reality. Technical report, Edinburgh Parallel Computing Center, 1995. ftp://ftp.epcc.ed.ak.uk/pub/paramics/papers/vehicular-reality.ps.z.

[11] Satoshi Yukawa and Macoto Kikuchi. Coupled-map modeling of one-dimensional traffic flow. *J. Phys. Soc. Jpn*, 64(1):35–38, 1995.