Development of Road Traffic CA Model of 4-Way Intersection to Study Travel Time

Anna T. Lawniczak[1] and Bruno N. Di Stefano[2]

[1] Department of Mathematics & Statistics, University of Guelph
50 Stone Road East, MacNaughton Building, Guelph ON Canada N1G 2W1
alawnicz@uoguelph.ca
[2] Nuptek Systems Ltd. Toronto, Ontario, Canada
nuptek@sympatico.ca, b.distefano@ieee.org

Abstract. We describe our development of a road traffic CA (Cellular Automata) model of the four most common types of 4-way intersection (Yield-controlled intersections, Stop-controlled intersections, Signal-controlled intersections, and Roundabout-based intersection). We developed this model to study how these four different types of 4-way intersection affect road traffic flow and congestion in general and "travel time" in particular. In this paper we describe the model and 4WayCA.exe, the traffic simulator software package in which the model has been implemented. We focus in particular on the model abstractions and on the simulator architecture.

Keywords: Traffic Modeling, Travel Time, Cellular Automata, Agent Based Simulation.

1 Introduction

In the field of road transportation, an intersection is the point where two or more roads meet. When roads pass above or below one another, we have a grade separation and no conflict exists between the traffic flow of one road and the traffic flow of any other road. When the intersecting roads are at the same level they are said to be "at grade" and a conflict arises between the traffic flow of one road and the traffic flow of the intersecting roads. In this case, a controlling mechanism is required.

We study, by means of a Cellular Automata (CA) model, how the choice of controlling mechanism affects "travel time", that is "the total time required for a vehicle to travel from one point to another over a specified route under prevailing conditions", [1].

We limit our study to four types of 4-way intersection: Yield-controlled, Stop-controlled, Roundabout, and Signal-controlled (traffic light controlled). For our purposes, we consider only single lane traffic for each travel direction and model four lanes (West-East, East-West, North-South, South-North) and the intersection controlling mechanism of which we consider four different configurations, Fig. 1.

Our paper is organized as follows. In Section 2 we briefly introduce Elementary Cellular Automata Rule 184, the mathematical abstraction from which our algorithm

J. Zhou (Ed.): Complex 2009, Part II, LNICST 5, pp. 2040 – 2049, 2009.

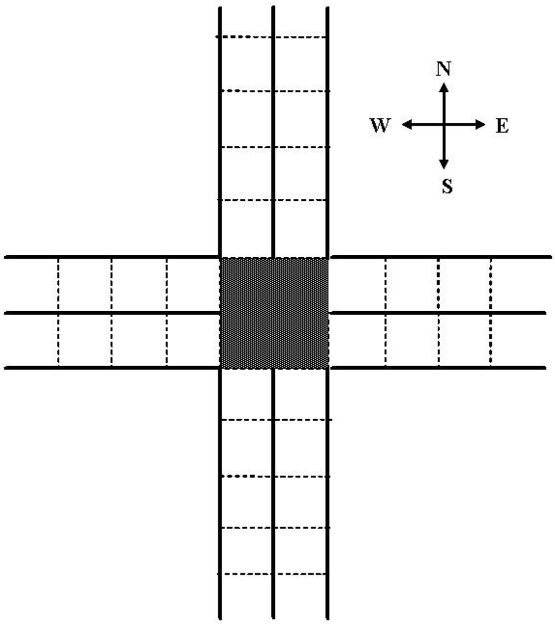

Fig. 1. Simplified drawing showing a generic 4-way intersection

is derived. In Section 3 we outline our digital experiment and in Section 4 we describe the architecture of our model.

2 Rule 184

Unidirectional single-lane traffic, with no intersections & no entry or exit ramps, is the simplest scenario of highway traffic that one can model. We can easily describe this scenario by means of a one-dimensional Cellular Automaton (CA) with periodic boundary conditions, Elementary CA (ECA) Rule 184, sometimes called the “traffic rule", [2].

A cellular automaton is an abstraction consisting of a regular grid of cells (in any finite number of dimensions), each in one of a finite number of states. We study the evolution of the CA over discrete time steps as the collective change of state of all cells of the grid. The state of a cell at time t is a function of the states of a finite number of cells (called its neighbourhood) at time t − 1. To every cell the same rule for updating is applied, based on the values in this neighbourhood. Each time we apply the rule to the whole grid, we create a new generation of cells. A common view is to consider the state of each cell as dependent on the presence or absence of particles in parts of its neighbourhood. In our case, as we are modelling vehicular traffic, this view is convenient and the particles under consideration are vehicles.

Fig. 2 shows a simplified block diagram depicting how the transition from time $t = \tau - 1$ to time $t = \tau$ is computed for any one-dimensional CA. For each cell c_i at time t-1

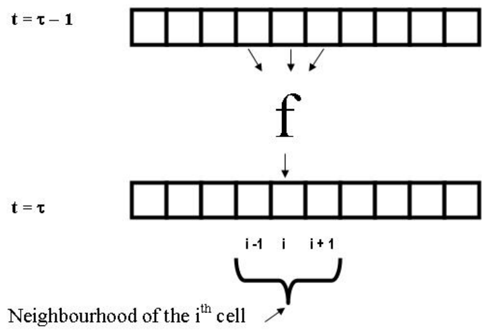

Fig. 2. Simplified drawing showing the transition of an ECA from time t = τ - 1 to time t = τ

we take the values of cells c_{i-1}, c_i and c_{i+1} at time t-1 and apply function f of Fig. 2 to compute the value of cell c_i at time t. We can write this general relationship, valid for all one-dimensional CA, as in equation (1).

$$c[t, i] := f[c[t-1, i-1], c[t-1, i], c[t-1, i+1]] \tag{1}$$

Evolving a CA means computing equation (1) inside two loops, a space loop in which variable i varies from zero to the maximum size of the CA and a time loop in which variable t varies from zero to the maximum number of required discrete time steps. We show below the pseudo-code for these two loops.

```
// initialization
for (t = 0; t < tmax; t++)
{
     // applicable
     for (i=0; i < imax; i++)
     {
        // applicable code
        c[t, i]:= f[c[t-1, i-1], c[t-1, i], c[t-1, i+1]]
        // applicable code
     }
     // applicable code
}
// program termination
```

To identify a suitable "f" to model road traffic is equivalent to choosing one of the existing ECA. We can calculate the possible binary states for the three neighboring cells c_{i-1}, c_i, and c_{i+1} as 2 x 2 x 2 = 2^3 = 8. Thus, cells c_{i-1}, c_i, and c_{i+1} can evolve according to 2^8 = 256 different CA rules. Perusing the literature we notice that many authors decided to use ECA Rule 184 to model vehicle traffic. ECA Rule 184 is one of the rules that exhibit motion patterns from one time step to another.

In ECA Rule 184, every particle (i.e., a Boolean "1"), representing a vehicle, at every time step, moves one cell forward provided that the cell ahead does not contain any other vehicle (i.e., it contains a Boolean "0", representing an empty cell), [3] and [4]. The truth table of Rule 184 is shown in Table 1, where the first row shows all

Table 1. Truth table of ECA Rule 184

Time t	Current Pattern	111	110	101	100	011	010	001	000
Time t+1	Centre Cell	1	0	1	1	1	0	0	0

possible configurations for three adjacent cells at time t and the second row shows the next state for the centre cell, at time t+1.

Each of the 256 different CA can be identified by indexing it according to the second row of the truth table applicable to the given CA, [5]. The general formula is

$$N = \sum_{i=0}^{7} \alpha_i 2^{7-i} \tag{2}$$

where N is an integer such that $0 \leq N \leq 255$ and $\alpha \in \{0,1\}$.

If we apply formula (2) to the truth table shown in Table 1, we obtain

$$N = 1(2^7) + 0(2^6) + 1(2^5) + 1(2^4) + 1(2^3) + 0(2^2) + 0(2^1) + 0(2^0) \tag{3}$$

$$N = 128 + 0 + 32 + 16 + 8 + 0 + 0 + 0 = 184$$

We can derive the Boolean expression for ECA Rule 184 from the truth table shown in Table 1. This can be done in various ways, depending if one wants to derive a minimal Boolean expression or not. Similarly, one may derive a different expression depending if one wants to use an expression as sum of maxterms or product of minterms, and depending if one wants to restrict or not the type of Boolean operators that one uses (i.e., {AND, OR, NOT} or, alternatively, {AND, OR, NOT, XOR}, or, alternatively, {NAND} or {NOR}). For instance, we use formula (4), i.e.

$$p \cap (\neg q) \cup (p \cup q) \cap r \tag{4}$$

where

$p = c_{i-1}$; $q = c_i$; $r = c_{i+1}$; $\cup$ is the OR, $\cap$ is the AND, and $\neg$ is the NOT operator, while E. Georgiadis [6] provides formula (5), i.e

$$p \oplus (q \cap (p \oplus r)) \tag{5}$$

where

$p = c_{i-1}$; $q = c_i$; $r = c_{i+1}$; $\cap$ is the AND and $\oplus$ is the XOR operator.

Formulas (4) and (5) are just different ways of making the relationship of formula (1) explicit and both can replace the right hand side of formula (1) inside the double loop shown above as pseudo-code.

By implementing a traffic model based on ECA Rule 184 is possible to reproduce most of the properties of road traffic as observed by traffic engineers for some types of traffic. This is adequate only if we assume to model one-way road traffic characterized by constant speed and null acceleration. However, if we want to model a more realistic situation of non-constant speed, we must be able to make each vehicle advance by more than one cell per time step. This means that we must be able to consider a neighbourhood of more than one cell ahead (i.e., to the right) and one cell behind (i.e., to the left). We must be able to "look ahead" and know if the road is empty, that is if the cells ahead are empty as needed. Boccara and Fuks have generalized Rule 184 to higher velocities [7], [8].

We extend and modify Rule 184 for our purposes, i.e. to model, how the choice of controlling mechanism for 4-way intersections affects "travel time", that is "the total time required for a vehicle to travel from one point to another over a specified route under prevailing conditions", [1].

3 Our Experiments

To study how the choice of controlling mechanism affects "travel time", we develop a "digital laboratory" and conduct "digital experiments".

We study how a large number of vehicles travel through a realistic itinerary consisting of left & right turn, and drives straight through the intersection. Fig. 3 shows an example of possible itinerary. We show only the itinerary actually traveled by the vehicle and not other intersecting roads. The arrow at the end of each segment indicates the presence of an intersection and the direction of traffic arriving to the intersection. We assume that all segments are of equal length. We call these segments of equal length "primitive segments". Whenever multiple arrows appear on a line segment, they signify the presence of a number of primitive segments making up a sub-route. For instance, the bottom horizontal line segment of Fig. 3 with arrows indicating traffic direction from

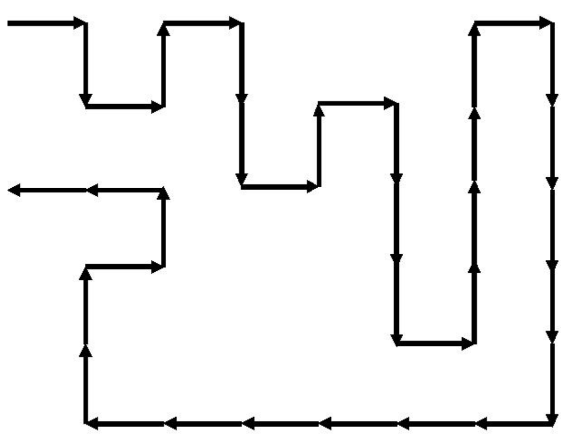

Fig. 3. Example of possible route. This is route {RoadWE, RoadNS, RoadWE, RoadSN, RoadWE, RoadNS, RoadNS, RoadWE, RoadSN, RoadWE, RoadNS, RoadNS, RoadNS, RoadWE, RoadSN, RoadSN, RoadSN, RoadSN, RoadWE, RoadNS, RoadNS, RoadNS, RoadNS, RoadNS, RoadEW, RoadEW, RoadEW, RoadEW, RoadEW, RoadEW, RoadSN, RoadSN, RoadWE, RoadSN, RoadEW, RoadEW}.

right to left consists of six primitive segments. This means that the vehicle travelling through this sub-route passes six intersections and goes through each intersection in the direction East-West. We show this sub-route again in Fig. 4 emphasizing that other intersecting roads exist and that the vehicle could have taken them. Thick lines show itinerary actually traveled by the vehicle while thin lines show intersecting roads that the vehicle could have taken but did not take.

We model each primitive segment using the basic building block of Fig. 1. For each experiment, we generate a set consisting of a large number of vehicles each with randomly pre-defined: (1) starting location, (2) starting time, and (3) complete route.

In our digital experiments, all vehicles cover a trip of equal length, but of different route, and navigate through an equal number of intersections. What each vehicle does arriving to any given intersection depends on its predefined route. We repeat the experiment using the same set of vehicles with the same characteristics, including the same predefined individual routes, but we apply it, separately, to all types of intersection under investigation.

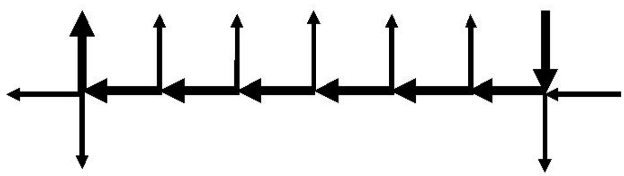

Fig. 4. Sub-route {RoadNS, RoadEW, RoadEW, RoadEW, RoadEW, RoadEW, RoadEW, RoadSN} of the example of possible route shown in Fig. 3. Thick lines show itinerary actually traveled by the vehicle while thin lines show intersecting roads that could have been taken by the vehicle but have not been taken.

Each experiment is characterized by:

- The length of the primitive segment, i.e. each section of road (e.g, West-East, etc)
- The number of intersections that each vehicle must cross
- The total number of vehicles that must be simultaneously in our road
- The maximum velocity allowed for any vehicle
- The maximum velocity allowed for specific vehicles
- The relative duration of the traffic light for the cases when the adopted intersection controlling mechanism is the Signal-controlled intersection
- The duration of the experiment, in time steps

We repeat each experiment four times, once for each type of intersection.

4 Our Model

To implement a road network over which the different routes can take place, we define four one-dimensional CA: RoadWE, RoadEW, RoadNS, and RoadSN, where the coordinates have the meaning of Fig. 1. Thus, RoadWE is the road that a vehicle takes when travelling from West to East, RoadEW is the road that a vehicle takes when traveling from East to West, RoadNS is the road that a vehicle takes traveling from North to South, and RoadSN is the road that a vehicle takes when traveling from South to North. Each of these roads is implemented as a one-dimensional CA.

If we call CELLS the total number of cells for each CA, each CA starts at cell i = 0 and ends at cell i = CELLS-1, which wraps around to cell i = 0. However, this wrap-around happens by means of a discontinuity, because a vehicle may. either restart from the beginning of the current CA or start from the beginning of a different CA, as it may be required by the predefined route. Each vehicle arriving to cell i = CELLS-1 must slow down and eventually stop if warranted by the algorithm and by the state of the specific intersection controlling mechanism (e.g., if the intersection controlling mechanism is a traffic light and if its state is RED). Depending on the predefined route, when the state of the intersection controlling mechanism allows it, the vehicle moves as either continuing on the same CA, just restarting from cell i=0, or switches to a different CA, starting from cell i=0 of a different CA. Thus, we can model a trip like the one depicted in Fig. 3 by concatenating subsequent routes crossing an intersection like the one depicted in Fig. 1. For instance, we can move across an hypothetical route like the one shown in Fig. 3 by transiting through the sequence {RoadWE, RoadNS, RoadWE, RoadSN, RoadWE, RoadNS, RoadNS, RoadWE, RoadSN, RoadWE, RoadNS, RoadNS, RoadNS, RoadWE, RoadSN, RoadSN, RoadSN, RoadSN, RoadWE, RoadNS, RoadNS, RoadNS, RoadNS, RoadNS, RoadEW, RoadEW, RoadEW, RoadEW, RoadEW, RoadEW, RoadSN, RoadSN, RoadWE, RoadSN, RoadEW, RoadEW}. In this sequence, we identify each tract of the route by the name of the CA used to model that tract. Thus, in this example:

- After "reading" the first RoadWE, the vehicle travels eastward from cell i=0 to cell i=CELLS-1 in CA RoadWE
- The vehicle crosses the intersection according to the rules specified by the intersection controlling mechanism
- After "reading" RoadNS, the vehicle travels southward from cell i=0 to cell i=CELLS-1 in CA RoadWE
- Etc.

This implementation requires only four CAs, actually implementing an intersection like the one of Fig. 1 to model any route. The number of primitive segments of a route is equal to the number of times a vehicle crosses an intersection. Any stochasticity is implemented during the initialization by selecting randomly an individual route for each vehicle. After the selection of the routes, during the execution of all experiments, the behaviour of the model is fully deterministic.

4.1 Cell Size and Time Step Duration

To use this model for practical traffic engineering applications, we must assign a realistic length value, in metres, to each cell and a realistic time value, in seconds, to each time step of our simulation. Perusing the literature it is possible to find that the most common length chosen for a cell is 7.5 m, see for instance [9], [10], and [11]. This has been chosen because it corresponds to the space occupied by the typical car plus the distance to the preceding car in a situation of dense traffic jam. The traffic jam density is 1000/7.5m approximately 133 vehicles/km, a realistic figure according to traffic engineers. We agree with the cell size of 7.5 m. After careful consideration, we have decided to assign the value of 5 seconds to each time step. Thus, the minimum speed of a vehicle advancing by one cell at each time step is equivalent to 5.4 km/h (that is, 7.5 x 3600/5 = 7.5 x 720 = 5400 m/h = 5.4 km/h). This allows representing most realistic and legal speeds observed in an urban setting with a maximum speed limit of 50 km/h (a vehicle advancing by 9 cells per time step travels at 48.6 km/h).

4.2 Vehicle

We represent each vehicle by means of a data structure containing:

- The ID, a unique identifier
- The Source, i.e. the cell where the vehicle enters into the traffic stream
- The Route, i.e. the way the vehicle travels through the model during the simulation
- The StartTime, i.e. the time step when the vehicle enters the traffic stream
- The Velocity, defined as the number of cells by which the vehicle will advance from time step t to time step t+1 in absence of any obstacle
- The CurrentCell, i.e. is the cell where the vehicle is at time t before the execution of the algorithm that will make the model evolve to time t+1
- The NextCell, i.e. the cell where the vehicle will move after the execution of the algorithm that will make the model evolve to time t+1
- The RouteIndex, i.e. is a variable keeping track of which leg of its route the vehicle is at any given time
- The CurrentTime, such that at any time step CurrentTime = t

For each vehicle, ID, Source, Route, & StartTime remain constant through the simulation, while all other parameters are updated at each time step until the vehicle reaches its destination, where we compute TravelTime = CurrentTime - StartTime.

4.3 Neighbourhood

We assume that a generic cell i, representing a segment of road of 7.5 m, at any given time step may either contain a vehicle or may be empty. If we assume that cell number i is occupied by a vehicle, a cell number smaller than i indicates a segment of road that has already been covered by the vehicle currently at cell i, while a cell number greater than i indicates a segment of road that the vehicle currently at cell i will cover in the future.

As we assume that vehicles can only move forward, thus, cells with cell number lower than i are immaterial from the point of view of the motion of the vehicle located at cell i, as with Rule 184, where the motion can only be in the direction of growing cell numbers. Hence, we omit these cells from the neighbourhood of cell i. The size of the neighbourhood of cell i in the direction of cell numbers greater than i is equal to the maximum speed of the vehicle occupying cell i. We depart from the spirit of Rule 184 and derived rules, [4], [7], and [8], and adopt a speed dependent variable size neighbourhood. We feel that in a realistic model speed increases much more slowly then it decreases. At the end of each time step the Velocity is updated in preparation for next time step. If a vehicle has not yet reached the end of the CA and has not reached the maximum speed, Velocity is incremented, otherwise it is left unchanged. The choice of DeltaVelocity depends on the type of drivers, "locale" (i.e., local rules, regulations, and culture-dependent driver's behaviour), and vehicle being modeled. At each time step, NextCell=CurrentCell+Velocity is computed, and a neighbourhood going from the cell adjacent to the cell where a vehicle is currently located, cell i=CurrentCell+1, to i=NextCell is examined. If any cell i=j for any j < NextCell is occupied by an obstacle, we set NextCell=j, and reduce speed to Velocity=1. One of the difficulties of modeling and simulating road traffic by means of CA is that, at every time step, only the neighbourhood is known. The only information about past history is what can be derived from the current state of the neighbourhood. Thus, from the point of view of a vehicle moving at full speed, a fixed obstacle ahead (e.g., a stalled vehicle) is not distinguishable from a slow moving vehicle ahead. If the obstacle is just a slower vehicle, the trailing vehicle can just slow down and, as soon as conditions allow, it can slowly increase again the speed. If the obstacle is a permanent obstacle (e.g., a stalled vehicle), the trailing vehicle will set its own speed to Velocity=0 at the next time step.

4.4 Intersection Navigation Algorithms

Upon approaching cell i=(CELLS-1) the behaviour of each vehicle depends on the type of intersection. In the case of traffic light the vehicle will either stop (if RED or AMBER/YELLOW) or proceed (if GREEN), slowing down if the next route involves a turn or maintaining its speed unchanged if it is straight ahead. In the case of a Stop-controlled intersection the vehicle will stop, remaining on cell i=(CELLS-1) for at least one time step. It will then evaluate a selected neighbourhood in the two intersecting lanes to see if any vehicle is approaching, waiting to advance until the road is clear. We can think about this as an extended von Neumann neighbourhood. A similar behaviour will apply to Yield-controlled intersection, where the vehicle does not need to remain in cell i=(CELLS-1) for at least one time step if the extended von Neumann neighbourhood is empty. In the case of Roundabout-controlled, the situation is similar to the Yield-controlled intersection but only a route-dependent subset of the extended von Neumann neighbourhood needs to be considered.

5 Conclusions

We have described our development of a road traffic CA model of the four most common types of 4-way intersections used to study how these types of intersection affect traffic flow & congestion and "travel time". We plan to improve our model and simulator by adding multi-lane scenario, so that we can allow for vehicle passing. Our simulation results will be presented in a future publication.

Acknowledgments. A. T. Lawniczak acknowledges partial financial support from the Natural Science and Engineering Research Council (NSERC) of Canada. B. N. Di Stefano acknowledges full financial support from Nuptek Systems Ltd. The authors thank D. Makowiec for inspiring conversation.

References

1. Transportation Engineering – Online Lab Manual, © 2000, 2001, 2002, 2003, Oregon State University, Portland State University, University of Idaho, Glossary, http://www.webs1.uidaho.edu/niatt_labmanual/Chapters/TrafficFlowTheory/Glossary/index.htm
2. Fukś, H.: Solution of the density classification problem with two similar cellular automata rules. Physical Review E 55(3), R2081–R2084 (1997)
3. Chopard, B., Droz, M.: Cellular Automata Modelling of Physical Systems. Cambridge University Press, Cambridge (1998)
4. Boccara, N.: Modeling Complex Systems. Springer, New York (2004)
5. Wolfram, S.: Statistical Mechanics of Cellular Automata. Rev. Mod. Phys. 55, 601–644 (1983)
6. Georgiadis, E.: A Note on Minimal Boolean Formula Size of One-Dimensional Cellular Automata. Journal of Cellular Automata 2(4), 351–353 (2007)
7. Boccara, N., Fuks, H.: Cellular Automaton Rules Conserving the Number of Active Sites. J. Phys. A: Math. Gen. 31, 6007–6018 (1998)
8. Fukś, H., Boccara, N.: Generalized deterministic traffic rules. Journal of Modern Physics C 9(1), 1–12 (1998)
9. Maerivoet, S., De Moor, B.: Cellular Automata Models of Road Traffic. Physics Reports 419(1), 1–64 (2005)
10. Nagel, K., Schreckenberg, M.: A cellular automaton model for freeway traffic. J. Physique I 2, 2221–2229 (1992)
11. Larraga, M.E., del Rio, J.A., Schadschneider, A.: New kind of phase separation in a CA traffic model with anticipation. J. Phys. A: Math. Gen. 37, 3769–3781 (2004)