Hybrid model to simulate urban traffic

Patrick Taillandier[a], Arnaud Banos[b], Nathalie Corson[c]

[a]*MIAT, INRA, France*
[b]*UMR Géographie-cités, France*
[c]*LMAH, University of Le Havre, France*

Keywords: Traffic Simulation, Multi-level Models, Agent-based Model, LWR Model, Model coupling

1. Introduction

This model (or rather these 3 models) aims at comparing a macro model to a microscopic, agent-based, one and to couple these two models. The goal is to take advantage of these two levels of representation by coupling them: more specifically, the agent-based component will be used to represent finely detailed behaviors at crossroads while the LWR model will be used to handle the flow of vehicles on main roads.

2. Micro-model

In order to present the micro-model used, we use the first section - Overview - of the ODD protocol [1].

2.0.1. Purpose

Our model aims at simulating the behavior of drivers at a fine temporal scale (1 simulation step = 1 second). It was designed in order to let the possibility to easily integrate unexpected events (car accident, natural or technological hazards) and take into account non normative behaviors.

2.0.2. Entities, state variables, and scales

Our model take into account three types of agents:

- *road*: each road is represented by a polyline composed of road sections (segments). A road is considered as a directed link. For bidirectional roads, 2 roads have to be defined corresponding to both directions.

A road can be composed of several lanes and the vehicles are able to change at any time and even use the lane of the reverse road if needed, for example to overpass a vehicle.

- *node*: a junction between two roads. A junction can be an intersection between several roads and contains traffic signals (e.g. traffic lights).
- *driver*: main agent of the model.

The main variables of the road agents are the following:

- *lanes*: number of lanes.
- *max speed*: maximum authorized speed on the road.

The main variable of the node agent is the following:

- *list of stop signals*: for each stop signal, gives the list of roads concerned by the signal.

Concerning the driver agents, their main variables are the following:

- *final target*: final location that the agent wants to reach (its goal).
- *current path*: the path that the driver agent is following. A driver can compute the shortest path (in terms of travel time) between its current location and its final target using a graph structure (each road will be an edge of the graph).
- *vehicle length*: length of the vehicle.
- *max acceleration*: maximal acceleration of the vehicle.
- *max speed*: maximal speed of the vehicle.
- *right side driving?*: do drivers drive on the right side of the road?
- *speed coef*: coefficient that defines if the driver will try to drive above or below the speed limit.
- *security distance coeff*: coefficient for the security distance. The security distance will depend on the driver speed and on this coefficient.

- *proba lane change up*: probability to change lane to a upper lane if necessary (and if possible).
- *proba lane change down*: probability to change lane to a lower lane if necessary (and if possible).
- *proba use linked road*: probability to take the reverse road if necessary (if there is a reverse road).
- *proba respect priorities*: probability to respect left/right (according to the driving side) priority at intersections.
- *proba respect stops*: probabilities to respect each type of stop signals (traffic light, stop sign...).
- *proba block node*: probability to accept to block the intersecting roads to enter a new road.

Note that all these variables are dynamic: the values of these variables can be modified at any time during the simulation. For example, the probability to take a reverse road can be increased if the driver is stuck for several minutes behind a slow vehicle.

The complete list of variables for roads, nodes and drivers can be founded in [2].

2.0.3. Process overview and scheduling

One step of the simulation represents 1 second. The dynamic of the model is based on 2 consecutive steps:

1. Each traffic signal computes its new state.
2. Drivers drive. Note that the driving step is asynchronous. agents move one after the other. The order of activation of the driver agents depend on their distance to the end of their current road: the drivers closer to the road end are activated first.

Traffic signal update

Each traffic light updates its state counter and if necessary changes its color.

Driving step

We defined two alternative behaviors (the choice of a behavior depends on a global parameter) for the driver agents:

- driving toward a target: each driver agent executes the *drive* action as follow (Figure 3): while the agent has time to move (*remaining_time* ¿ 0), it first defines the expected speed. This speed is computed from the *max_speed* of the road, the current *real_speed*, the *max_speed*, the *max_acceleration* and the *speed_coef* of the driver (see Equation 1). Then, the agent moves toward the current target and computes the remaining time. During the movement, the agent can change lane (see below). If the agent reaches its final target, it stops; if it reaches its current target (that is not the final target), it tests if it can cross the intersection to reach the next road of the current path. If it is possible, it updates its next target (target node of the next road) and continues to move.

- wandering: each driver agent executes the *wander* action that works as the *drive* action except that the next road is not defined through a path but chosen randomly among the roads outgoing from the current node.

$$\begin{aligned} speed_{driver} = \; & Min(max\_speed_{driver}, \\ & Min(real\_speed_{driver} + max\_acceleration_{driver}, \\ & max\_speed_{road} * speed\_coef_{driver})) \end{aligned} \tag{1}$$

The function that defines if the agent crosses or not the intersection to continue to move works as follow (Figure 4): first, it tests if the road is blocked by a driver at the intersection (if the road is blocked, the agent does not cross the intersection). Then, if there is at least one stop signal at the intersection (traffic signal, stop sign...), for each of these signals, the agent tests its probability to respect or not the signal (note that the agent has a specific probability to respect each type of signals). If there is no stopping signal or if the agent does not respect it, the agent checks if there is at least one vehicle coming from a right (or left if the agent drives on the left side) road at a distance lower than its security distance. If there is one, it tests its probability to respect this priority. If there is no vehicle from the right roads or if it chooses not to respect the right priority, it tests if it is possible to cross the intersection to its target road without blocking the intersection (i.e. if there is enough space on the target road). If it can cross the intersection, it crosses it; otherwise, it tests its probability to block the node: if the agent

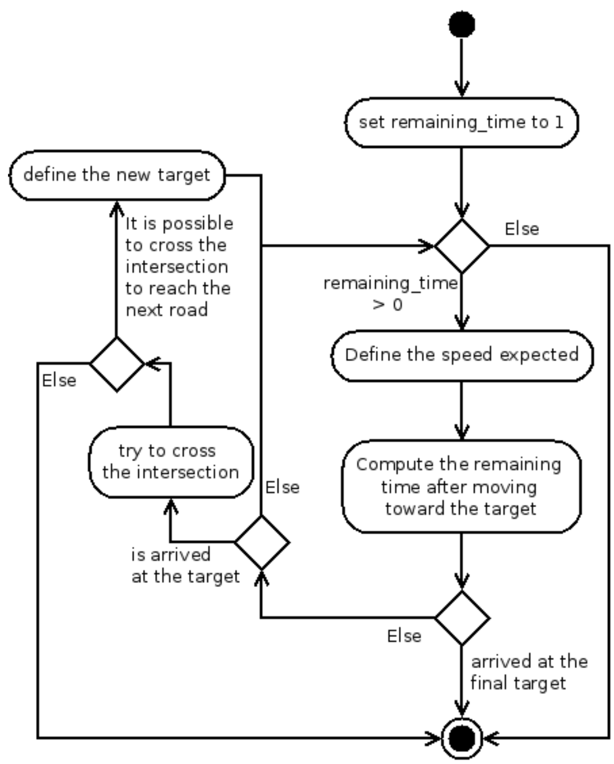

Figure 1: Drive action

decides nevertheless to cross the intersection, then the perpendicular roads will be blocked at the intersection level (these roads will be unblocked when the agent is going to move).

Concerning the movement of the driver agents on the current road (Figure 5), the agent moves from a section of the road (i.e. segment composing the polyline) to another section according to the maximal distance that the agent can move (that will depend on the remaining time). For each road section, the agent first computes the maximal distance it can travel according the remaining time and its speed. Then, the agent computes its security distance according to its speed and its *security_distance_coeff*. While its remaining distance is not null, the agent computes the maximal distance it can travel (and the corresponding lane), then it moves according to this distance (and updates its current lane if necessary). If the agent is not blocked by another vehicle and can reach the end of the road section, it updates its current road section and continues to move.

The computation of the maximal distance an agent can move on a road section consists in computing for each possible lane the maximal distance the agent can move. First, if there is a lower lane, the agent tests the probability to change its lane to a lower one. If it decides to test the lower lane, the agent computes the distance to the next vehicle on this lane and memorizes it. If this distance corresponds to the maximal distance it can travel, it chooses this lane; otherwise it computes the distance to the next vehicle on its current lane and memorizes it if it is higher than the current memorized maximal distance. Then if the memorized distance is lower than the maximal distance the agent can travel and if there is an upper lane, the agents tests the probability to change its lane to a upper one. If it decides to test the upper lane, the agent computes the distance to the next vehicle on this lane and memorizes it if it is higher than the current memorized maximal distance. At last, if the memorized distance is still lower than the maximal distance it can travel, and if the agent is on the highest lane and if there is a reverse road, the agent tests the probability to use the reverse road (linked road). If it decides to use the reverse road, the agent computes the distance to the next vehicle on the lane 0 of this road and memorizes the distance if it is higher than the current memorized maximal distance.

This agent-based microscopic model allows defining in a detailed way both infrastructure (traffic signals, roads with different lanes...) and driver's behavior. However, it requires a lot of computation even for simple cases (when cars are stucked in a traffic jam for example) whereas a macro model

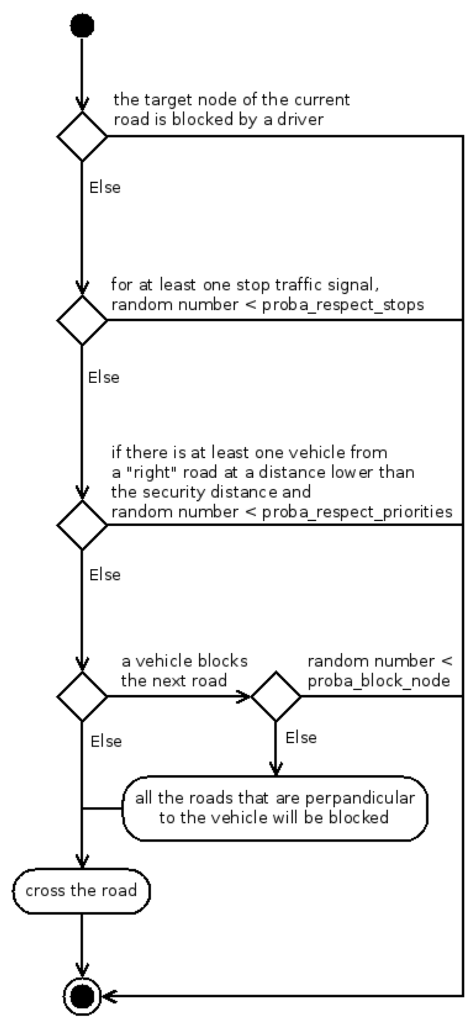

Figure 2: Crossing of an intersection (case where *right_side_driving* is true)

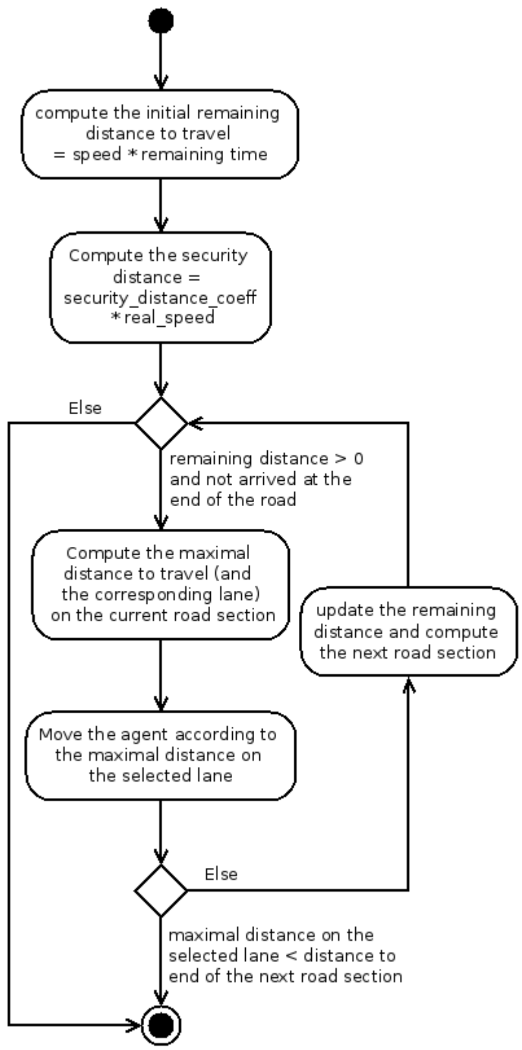

Figure 3: Move on the current road

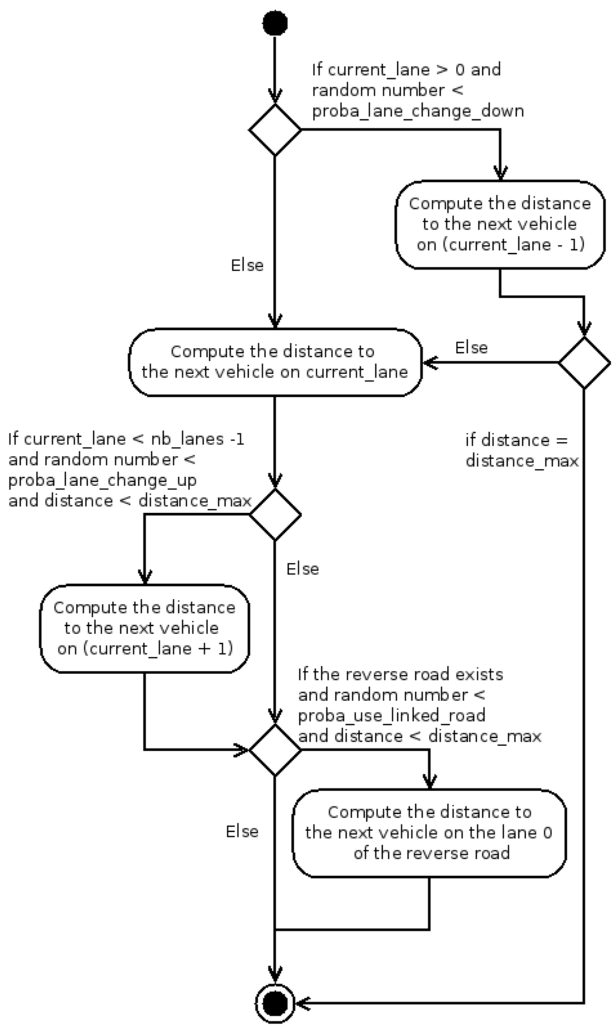

Figure 4: Define the maximal distance possible to travel and the corresponding lane (case where *right_side_driving* is true)

could be used more efficiently in some situations.

3. Macro-model

We used the classic LWR model proposed by Lighthill and Whitham [3] and Richards [4]. This model describes the evolution of the traffic on a single road. It describes the traffic at a global level by considering it as a flow without taking into account the individual behavior of drivers. It is based on the fundamental diagram [5], that define the equilibrium state of the traffic and explicit the linked between the density and the flow.

This model is based on several principles:

- The road is decomposed into sections of constant length x.
- The time is decomposed into simulation steps of duration t.
- The density is assumed to be homogeneous on each section.
- A vehicle cannot cross more than a road section in one simulation step.

Three main components are used to describe the flow of vehicles:

- $Q(x, t)$ = flow : number vehicles passing through section x at time t ;
- $K(x, t)$ = density : number vehicles located in section x at time t ;
- $V(t, x)$ = mean speed of vehicles located in section x at time t ;

Equation (2) shows the evolution of these components.

$$\begin{cases} Q(x,t) = K(x,t)V(x,t) \\ \dfrac{\partial K(x,t)}{\partial x} + \dfrac{\partial K(x,t)}{\partial t} = 0 \\ V(x,t) = V_e(K(x,t)) \end{cases} \tag{2}$$

The first equation corresponds to the definition of a flow: $Q = K.V$. The second equation corresponds to the conservation of traffic. Indeed, variation in time of the concentration depends on the in-flow and the out-flow of vehicles. This corresponds to the variation in space of the flow. The third equation corresponds to the fundamental diagram 3. It gives the speed as a function of concentration. Indeed, while the critical concentration of the road is not reached, vehicles can circulate at the maximal authorized speed.

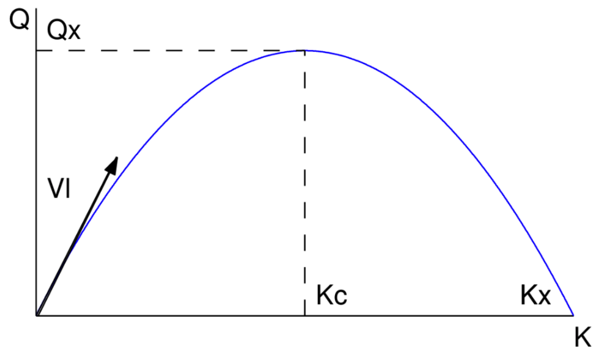

Figure 5: Fundamental diagram

However, once this critical concentration is reached the speed decreases ad we observe a congestion phenomenon. Let us notice that in the case of the LWR model, the critical concentration has to be equal to half of the maximal concentration of the road.

The LWR model is a PDE model and a classic approach to compute the evolution of the number of vehicles (density) in each section consists in discretizing time and space using a Godunov discretization method [6]. Δx is the discretization step of the road in sections (Δx is the size of section i). Δx is a constant and therefore, all sections have the same size. Δt is the discretization step of time.

For each section, we define the offer and demand functions, that respectively defines the number of vehicles that can enter a section and that can exit a road section. These offer and demand functions can be represented according to the density value as shown in figure 6. The values of the offer and demand depend on the critical density of the section (Kc) that can be defined from the properties of the section (perimeter and number of lanes).

We model the evolution of the offer and demand at every simulation steps by respectively Equation 6 and Equation 4.

$$O(K_i) = \begin{cases} Q_x & if K \leq K_c \\ \frac{-Q_x}{K_c} K + 2Q_x & if K > K_c \end{cases} \tag{3}$$

$$D(K_i) = \begin{cases} \frac{Q_x}{K_c} K & if K \leq K_c \\ Q_x & if K > K_c \end{cases} \tag{4}$$

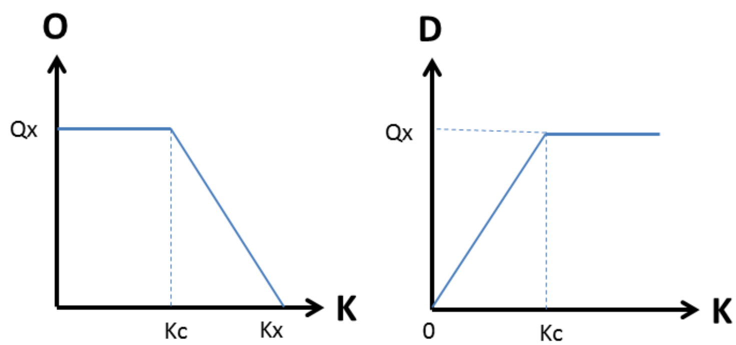

Figure 6: Value of the offer and demand according to the density, with Kc the critical density of the road, Kx the maximal density, and Qx the maximal flow

We can then define $K_i^{t+\Delta t}$ as the concentration of section i at time t+Δt.

$$\begin{aligned} Q_i^{t \to t+\Delta t} &= \min\left(D(K_i), O(K_{i+1})\right) \\ K_i^{t+\Delta t} &= K_i^t + \frac{\Delta t}{\Delta x}\left(Q_{i-1}^{t \to t+\Delta t} - Q_i^{t \to t+\Delta t}\right) \end{aligned} \tag{5}$$

The first equation of (5) represents the fact that, for section i, the flow at time $t + \Delta t$ corresponds to the minimum between the number of vehicles wanting to enter section i and the number of vehicles able to enter the following section.

The second equation of (5) represents the fact that the concentration of section i at time $t + \Delta t$ corresponds to the concentration at time t to which we add the number of vehicles going out of the previous section $(i - 1)$ and subtract the number of vehicles outgoing Section i.

Note that this can be done if in one time step, a vehicle cannot go further than the next section. This condition can be written as: $\dfrac{\Delta x}{\Delta t} \geq V_l$ where V_l is the maximal speed.

To sum up, the scheduling for a unique road for one step is the following:

1. ask all the sections to compute O (Equation 3) and D (Equation 4)
2. ask all the sections to compute Q (Equation 5)
3. ask all sections to compute K (Equation 5)

However, we want to take into account the connection between several roads. Once arrived at the end of the road, a driver will randomly choose the next road among all the outgoing roads. In our macro model, this is formulated as follow: the demand of the final section of a road is distributed among the first sections of the outgoing roads. Note that we have to take into account the fact that several incoming roads can be connected to several outgoing roads and that we have to take into account at the same time all the demand of these incoming roads as well as all the offer of the outgoing roads to have a fair distribution of demand to the outgoing roads. We chose then to integrate this distribution computation into the node. The global scheduling for the model for one step is then the following:

1. each section computes O (Equation 3) and D (Equation 4)
2. each node stores the flow (Q) of its incoming and outgoing sections
3. each section - except the ones connected to a node - computes Q (Equation 5)
4. each section - except the ones connected to a node - computes K (Equation 5)
5. each node compute K for its incoming nodes (Equation 5)

The management of the flow by nodes works as follow:

1. ask every incoming sections to normally compute Q (Equation 5)
2. ask every outgoing sections to set $Q_{incoming}$ (flow of incoming vehicles) to 0
3. compute $sumD$ the sum of demand of incoming sections.
4. compute $offering\_sections$, the list of outgoing sections that can still receive vehicles (O ¿ $Q_{incoming}$)
5. While $sumD > 0$ and $offering\_sections$ is not empty, the nodes ask $offering\_sections$ to compute the additional incoming flow: $Q^{tmp}_{incoming} = Minimum(O - O_{incoming}, sumD/length(offering\_sections))$; update $Q_{incoming}$ with: $Q_{incoming} = Q_{incoming} + Q^{tmp}_{incoming}$; subtract $Q^{tmp}_{incoming}$ from $sumD$; if $Q_{incoming} = O$, remove the section from $offering\_sections$
6. ask every outgoing sections to compute K with: $K_i^{t+\Delta t} = K_i^t + \frac{\Delta t}{\Delta x}(Q_{incoming} - Q_i^{t \to t+\Delta t})$
7. compute the sum of incoming flows of incoming sections $sumQ_{incoming}$
8. ask every incoming sections to set Q to 0
9. compute $demading\_sections$, the list of incoming sections that can still give vehicles (D ¿ Q)

10. While $sumQ_{incoming} > 0$ and $demading\_sections$ are not empty, the nodes ask $demading\_sections$ to compute the additionally flow: $Q^{tmp} = Minimum(D - Q, sumQ_{incoming}/length(demading\_sections))$; update Q with: $Q = Q + Q^{tmp}$; subtract Q^{tmp} from $umQ_{incoming}$; if $Q = D$, remove the section from $demanding\_sections$

The LWR model has been implemented with the GAMA platform. It represents a homogeneous traffic evolving from one equilibrium to another, but does not take into account transitory stages. This is why we are interested in a hybrid model, coupling two previously described models in a dynamic way.

4. Coupling macro and micro models

The hybrid model is based on the micro-model that is enriched in order to let the possibility to roads to directly manage the flow of vehicles on them by using the LWR model. In this model, each road is divided in road sections

When a vehicle enters a road that is managed by the macro model, the vehicle is captured by the road and is added to the concentration of the first road section of the road. Then the macro model is used to compute the evolution of the concentration of vehicles on each road sections of the road. In the same way, the computation of the demand of the last section is used to release driver agents.

In that perspective, we define a new type of agent: the road section agents. Each road agent is composed of a list of road section agents and can manage the flow of vehicles on each of its sections.

We therefore add 2 new variables to the road agent:

- *macro behavior?*: does the road agent manage by itself the behavior of the driver flow?
- *sections*: list of road sections composing the road.

Concerning the road section agents, we define for each of them 6 static variables that define their properties:

- *max capacity*: max number of vehicles on the road section
- *Kc*: critical concentration
- *Vl*: maximal speed

- *Qx*: maximal flow
- *next road section*
- *previous road section*

In addition, we define 4 dynamic variables:

- *K*: current concentration, i.e. number of vehicles on the road section
- *Q*: current flow, current flow on the road section
- *offer*: number of vehicles that can enter the road section
- *demand*: number of vehicles that can exit the road section

The UML diagram of the hybrid model is given in Figure 4.
In terms of scheduling, a simulation step is composed of 4 steps:

1. Each traffic signal computes its new state.
2. Each macro road identifies entering driver agents : these agents are removed from the general scheduler.
3. Each macro road having identified such driver agents activates the LWR model:
   (a) ask its sections to compute O and D
   (b) ask its sections to compute Q: for the final road section, after computing Q, the section computes from Q the concentration of drivers to release ($nb\_drivers = Q * (step\_duration/section\_size)$), with $step\_duration = 1s$ for us, and $section\_size = roadmaxspeed$. Finally, the section releases the corresponding number of driver agents (these agents are added to the general scheduler).
   (c) ask every section to compute K
4. Drivers (that are not "captured" by roads) drive.

An important point concerning the coupling between macro and micro models is the management of float values: indeed the LWR model works with continuous stocks of vehicles (section concentration) while the agent model handles discrete entities (agents). The solution proposed here is to add a new variable to the road agents called *remaining_car* representing the concentration remaining (floating number) once driver agents are released. For example, imagine that a the concentration of driver agents that should

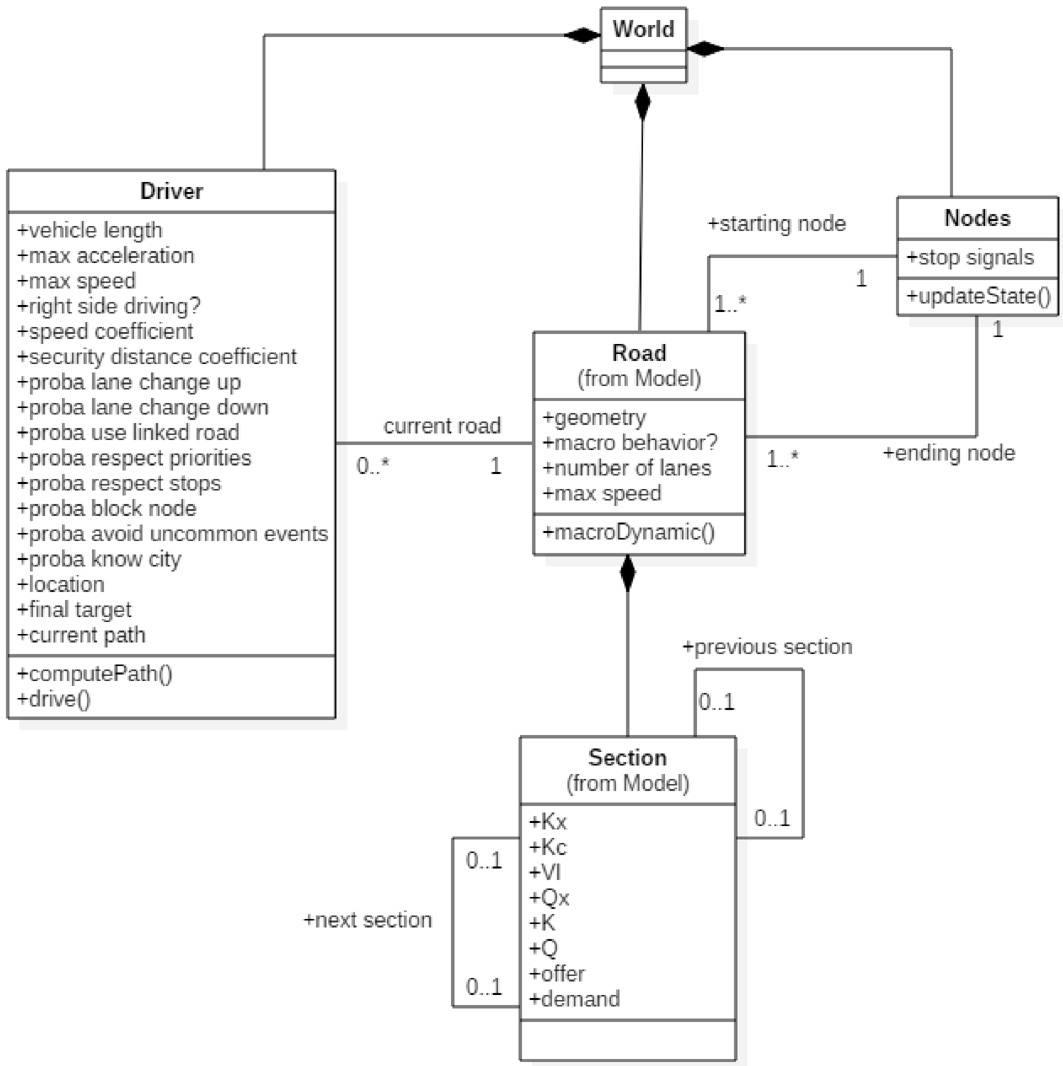

Figure 7: UML class diagram of the hybrid model

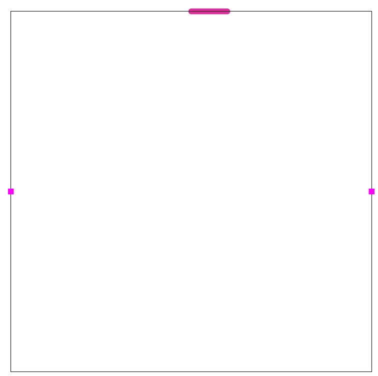

Figure 8: Simple circuit composed of two roads: the two squares represent the connection between the two roads and the magenta line the section on which the measures were taken

be released (*nb_drivers*) from a macro road is 3.6. In our hybrid model, 3 driver agents will be released (at the end of the road) and 0.6 will be added to the *remaining_car* variable. If *remaining_car* is higher than 1, then a new car is released and *remaining_car* is decremented by 1.

5. Experiments

5.1. Simple unidirectional circuit experiments

The first set of experiments carried out compares the two models on a simple road infrastructure composed of only two roads connected as a circuit This set of experiments has two main goals: firstly to compare the results of the two models in terms of flow, mean speed and concentration and secondly to compare the efficiency of the two models in terms of computation time.

In that perspective, we simulated the traffic on a simple circuit composed of two roads of 500 meters, with a speed limit of 90km/h and 2 lanes (Figure 5.1) in which we add progressively more and more vehicles. Indeed, after 500 simulation steps used for initialization (to avoid artifacts due to the addition of vehicles), we compute for a given section (see Figure 5.1) for minutes and for 10 minutes (600 simulation steps) the mean flow, speed and concentration, then we add $0.05 * totalmaxcapacityoftheroads$ vehicles to the road. These vehicles are uniformly located on the road with a given initial speed that corresponds to the speed limit. As the model is stochastic (in particular the characteristic of the drivers), we ran 10 time the simulation.

Concerning the parameter values of the agent model, we chose the following value for the parameter that has a meaning for this model:

- *vehicle length*: 4 meters
- *max acceleration*: random value between 10 and 15 km/h (max speed that can be gained during 1 second)
- *max speed*: 150 km/h
- *speed coef*: random value between 0.8 and 1.1
- *security distance coeff*: random value between 1.5 and 3.0
- *proba lane change up*: random value between 0.5 and 1.0.
- *proba lane change down*: random value between 0.5 and 1.0.

For the macro model, the only parameter to define is the maximal density (Kx) of sections. This one was defined according to the perimeter of the section, the number of lanes and the vehicle length (see Equation 6). Note that we chose for the section perimeter the maximal distance (defined by the speed limit on the road) that can travel a driver during one time step (1 second for us).

$$Kx = sectionperimeter * numberoflanes/vehiclelength \tag{6}$$

Figure 5.1 shows the phase diagrams obtained with the LWR model, Figure 5.1 with the agent model.

As can be observed on the diagram, the two models give results that are consistent with each other. The general tendency between the two models are similar even if we can observe that the curves of the agent model are more fuzzy, due to the stochastic nature of the model.

The second type of results concerns the computation time of each model. Indeed, we expect that using a macro model such as LWR is less time consuming than using a micro agent-model when dealing with a high number of vehicles. So, we compared the computing time for different number of vehicles. Figure 5.1 gives the results for the LWR model and Figure 5.1 for the agent one. We can observe that, as expected, the macro model is a lot more efficient when dealing with a high number of vehicles than the micro model.

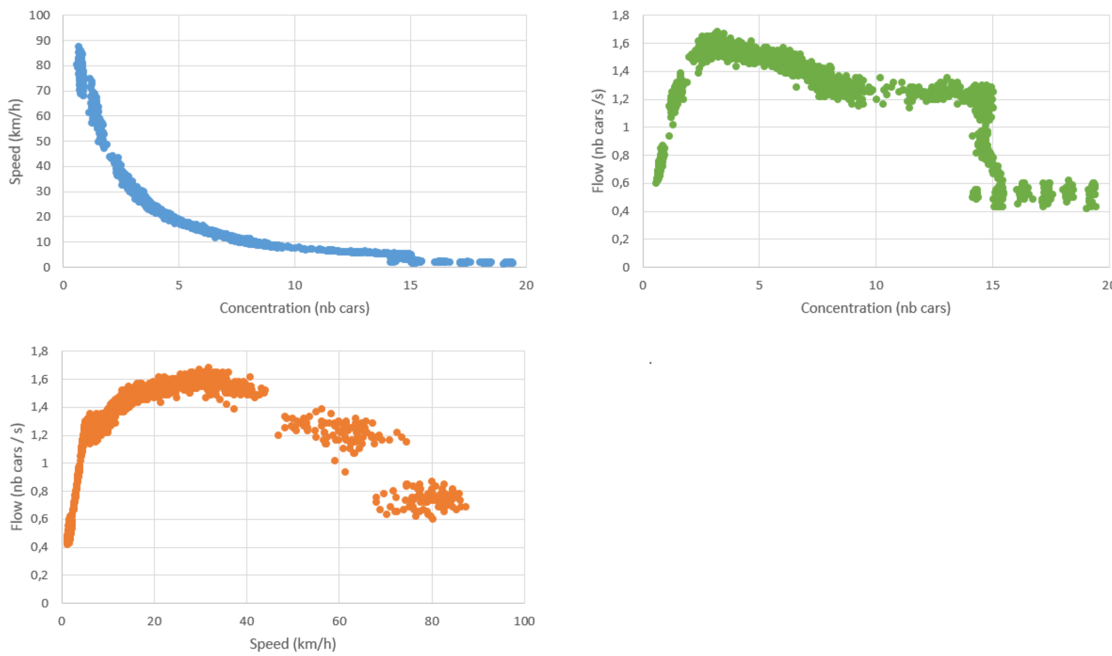

Figure 9: Phase diagrams of the Flow, Speed and Concentration obtained with the LWR model on the simple circuit

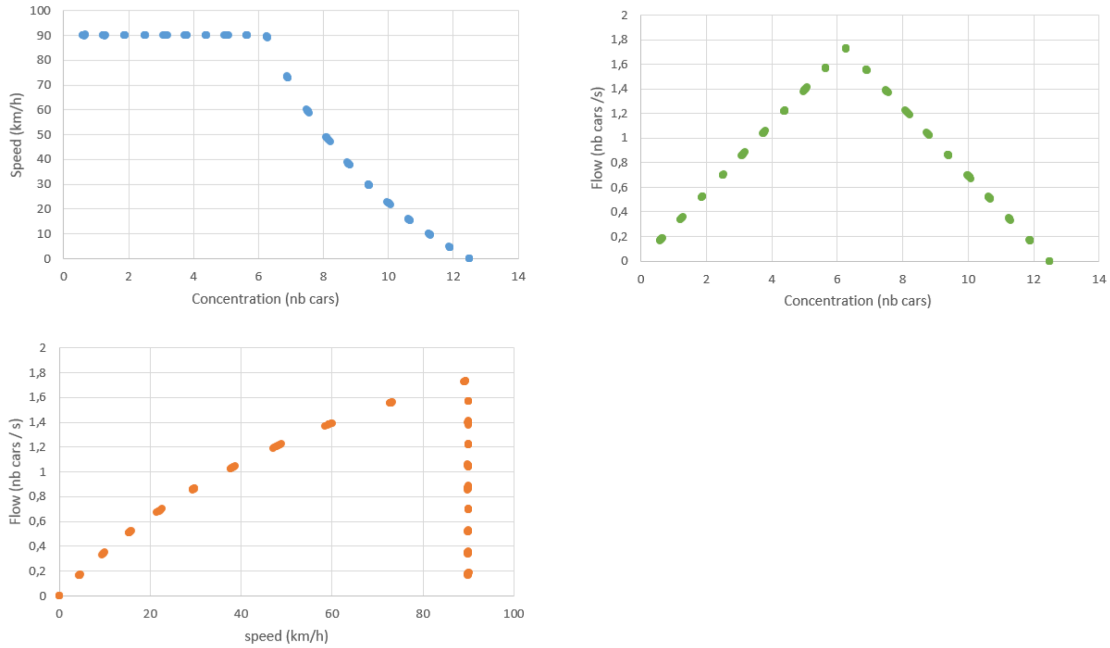

Figure 10: Phase diagrams of the Flow, Speed and Concentration obtained with the agent model on the simple circuit

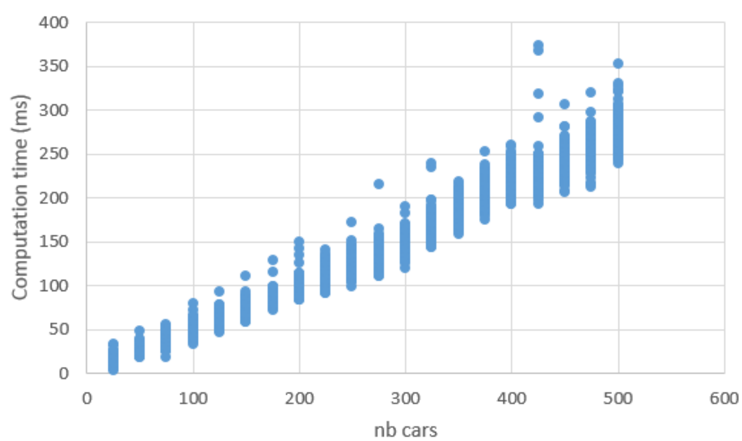

Figure 11: Computation time with the LWR model on the simple circuit

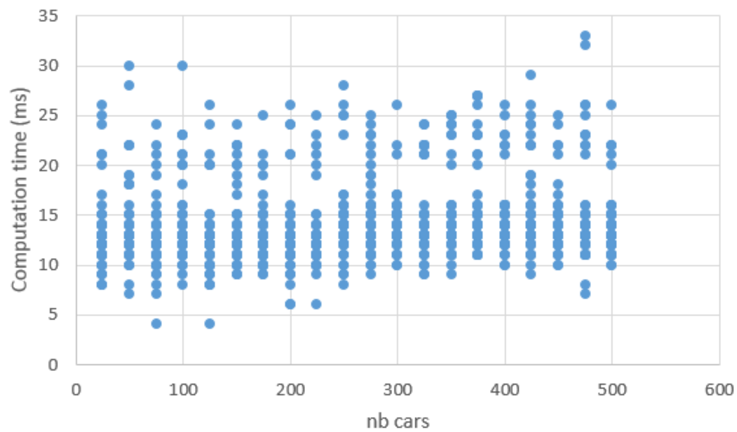

Figure 12: Computation time with the LWR model on the simple circuit

5.2. Artificial network and diffusion

The second series of experiments concerns the comparison of the two models for more complex networks. In particular, we will compare how the vehicles spread in the networks (percolation) and the computation time.

The experiment is the following: each 5 minutes (300 simulation steps), we add 5 new vehicles on the starting road (a predefined road). Thus, at the end, 155 vehicles are moving in the road network. The goal is to observe, after during 1h (3600 simulation steps), the sub-network visited by the vehicles.

We tested 3 networks with a comparable length of roads, and the same type of roads (limit speed at 50 km/h and 2 lanes):

- Regular (Manhattan): Figure 5.2
- Random spatial network: Figure 5.2
- Real network (city of Le Havre): Figure 5.2 . This network is composed of 4548 roads and 1744 nodes. We used the OSM data (converted as shapefiles) of Le Havre.

We use the same parameter value for the two models as the ones used for the simple circuit experiment. We just defined some new parameter value concerning the agent model:

- *proba use linked road*: 0.0; it means that the drivers will never used the reverse road.

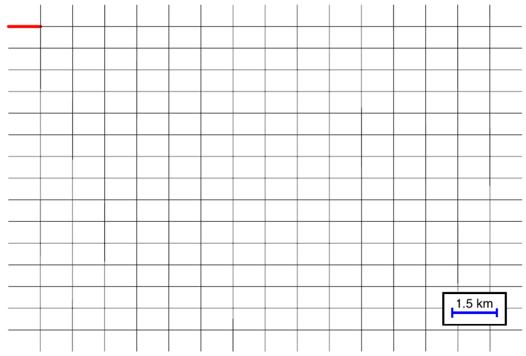

Figure 13: Regular network: the red line is the starting road

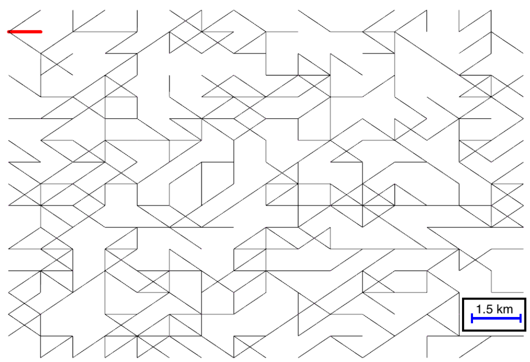

Figure 14: Random spatial network: the red line is the starting road

Figure 15: Real Network: the red line is the starting road

- *proba respect priorities*: 1.0; it means that the drivers will always respect the priority.
- *proba block node*: 0.0; it means that the drivers will never block a crossroad.

Figure 1 shows the results for the percentage of visited roads for the three network. As shown by the figure, the diffusion is a lot faster with the LWR model. Indeed, in contrary to the agent model where the model deals with discrete entity (the agents), the LWR model deals with continuous stocks (the concentration). As a consequence, the flow is distributed much faster among the roads. Another result is that the exploration of the random network by both models is slower than the exploration of the regular network as the random network is composed of many nodes with high degree (hub) and longer roads. At last, the difference is particularly important for the real network, that is composed of many nodes with a low degree and short roads.

Table 1: Diffusion (percentage of visited roads) for a simulation of 1 hour with the 3 networks

Network type	Agent Model	LWR Model
Regular	9.3%	68.3%
Random	6.4%	43.7%
Real	5.2%	98.4%

Table 2 shows the computation time for both models with the three networks. In contrary to the first series of experiment where the LWR model was a lot more efficient, in this one, the agent model is more than 100 times faster. The lack of efficiency of the LWR model in comparison to the agent one is due to the fact the number of vehicles is very low compared to the number of sections. Indeed, we have here just 155 vehicles but more than 50000 sections. Even with an optimization of the LWR model that prevents some unnecessary computation for roads with a null concentration (and not linked with a road with a not null concentration), the model needs to compute the evolution of the concentration for many sections. In comparison, the agent based model just needs to compute the movement of the 155 vehicles without having to take into account at each time step the global network.

To conclude on this experiment, the use a of continuous concentration for the LWR model can have a deep impact on the traffic result, in particular,

Table 2: Computation time (in seconds) for a simulation of 1 hour with the 3 networks

Network type	Agent Model	LWR Model
Regular	0.15 s	626 s
Random	0.14 s	409 s
Real	0.16 s	1337 s

when dealing with small number of vehicles. In the same way, LWR is not efficient at all when working with city-scale networks with low concentration. It is thus important to limit the use of the LWR model to the main roads with a lot of vehicles.

5.3. City-scale simulation

In order to test the model in "real" conditions, we carried out an experiment concerning the use of the model to simulate the traffic of city of Le Havre (France, Normandie).

As our goal is to test the use of our hybrid model in realistic conditions, we work here with 30000 driver agents, a number that corresponds to the number of cars in Le Havre between 17h and 18h, on a typical week-day. Each driver agent has a random initial location (one of the nodes) and a random final target.

When a driver agent reaches its destination, it simply disappears (we do not integrate car parking at the moment).

We use the same network of Le Havre than the one presented in the previous experiment.

Using the hybrid model requires to choose the road that will use the macro model and the ones that will use the micro model. As stated in the previous experiment, it is interesting to use the macro model for roads that are used by a lot of drivers. A solution to determine which are the most used roads is to run the micro models and then to count for each road the number of drivers that have used this road. However, it requires a lot of computation and depends on the initial and target locations of the drivers. We then chose another solution that just consist in computing the edge betweenness of each road (number of shortest paths passing through the road). This computation only depends on the structure of the road network and not on the drivers, thus this computation can be done only once. After defining the edge betweenness of each road, we used the macro models for the k-roads with the highest value for this indicator. k is a parameter of the model that we tested in this experiment. Note that the macro model can be used only if a vehicle cannot go further than the next section, so the model is not adapted for short roads. As a result, we chose to not use the macro model for the roads of which the perimeter is lower than twice the section perimeter.

In terms of results, we compared the number of simulation steps and the computation time for the complete disappearing of the driver agents for different value of k. The experiment was carried out on a Ubuntu desktop computer with an i7 processor and 16Go of RAM. The computation time takes only into account the dynamic of the agents and the macro models and not the initialization and visualization computation time. As the model is stochastic (in particular the choice of the initial and target locations of drivers), we ran 10 time the simulation per k value.

Concerning the parameter value for the two models, we used the same as in the previous experiment (artificial network and diffusion).

Table 3 shows the results obtained (mean for the 10 simulations): first of all, the results in terms of simulation steps required for the complete disappearing of drivers are very close: the gap increase with the k value but stays small, which shows that the general dynamic was not impacted compared to the use of a purely micro model. Concerning the computation time, the use of the hybrid model with a k of 10% has allowed to improve the computation time by 50% which is rather promising. We can note that with a higher of k the computation time increases, in particular when k is 100% (we use the macro for all roads for which the perimeter is higher than 2 * section perimeter), which proves that using the macro model for roads that are less used by drivers is counterproductive.

Table 3: Mean number of simulation steps and computation time for the complete simulation of Le Havre city for the different k value

value of k (percentage of macro roads)	$Simulationsteps$	$Computationtime$
0% (purely micro model)	5346.2	1329s
5%	5389.5	701s
10%	5450.4	660s
20%	5665.4	686s
50%	5815.5	1019s
100%	5932.6	1430s

Note that another advantages of our hybrid model over a purely macro model resides its flexibility and modularity, as new elements can be integrated quite easily, for example:

- traffic signals: traffic light, stop signals, priorities that can already be managed by the micro agent model.
- adaptive change of plans of the agents: arriving at a crossroad, an agent can decide to dynamically change its path in presence of a traffic jam.
- non-normative behaviors: all the drivers do not respect all the driving rules. The agent-based model allows driver agents not to respect traffic signals, or to block a crossroad, or skip right/left priorities...

6. Acknowledgment

This work was carried out in the context of the RNSC MAPS network ("Agent-based Modeling of Spatial Phenomena").

[1] V. Grimm, U. Berger, D. L. DeAngelis, J. G. Polhill, J. Giske, S. F. Railsback, The odd protocol: a review and first update, Ecological modelling 221 (23) (2010) 2760–2768.

[2] P. Taillandier, Traffic simulation with the gama platform, in: International Workshop on Agents in Traffic and Transportation, 2014, pp. 8–p.

[3] M. J. Lighthill, G. B. Whitham, On kinematic waves. ii. a theory of traffic flow on long crowded roads, Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences 229 (1178) (1955) 317–345.

[4] P. I. Richards, Shock waves on the highway, Operations research 4 (1) (1956) 42–51.

[5] B. Greenshields, W. Channing, H. Miller, et al., A study of traffic capacity, in: Highway research board proceedings, Vol. 1935, National Research Council (USA), Highway Research Board, 1935.

[6] J.-P. Lebacque, The godunov scheme and what it means for first order traffic flow models, in: Internaional symposium on transportation and traffic theory, 1996, pp. 647–677.