1. Introduction

Simulation tools are an integral part of transportation planning and research. As described by Lieberman (2004), the history of traffic simulators stretches back to the 1950s, and has proceeded alongside developments in the theory of traffic. This trend has continued to the present day: about half of the papers in the Transportation Research Board’s 2018 issue on “Intelligent Transportation Systems” (2018) involve a simulation model. A literature review by the Joint Traffic Simulation Subcommittee of the TRB (SimSub) found 194 papers that used traffic simulation models in the 2019 annual meeting (Hale, 2019).

Transportation models can be classified broadly into macroscopic, mesoscopic, and microscopic models. We adopt here the terminology of Van Wageningen-Kessels et al. (2015). Macroscopic models do not distinguish individual vehicles, but instead view traffic as a continuum. This approach originates with the work of Lighthill and Whitham (1955) and Richards (1956), who coupled the “fundamental diagram” of Greenshields (1935) with the law of conservation of vehicles. In contrast, mesoscopic and microscopic models are vehicle-based. Microscopic models compute vehicle trajectories based on car-following rules. Most models in this category use ordinary differential equations to represent the acceleration of a vehicle as a function of the state of its neighbours. The car-following approach is one of the oldest in transportation modelling, dating back to the work of Chandler et al. (1958). It is used today in most microscopic simulation software, including SUMO (Lopez et al., 2018), Aimsun, VISSIM, and CORSIM. Another sub-category of microscopic models are those based on cellular automata. Here space is considered as fundamentally discrete, and the neighbourhood of a vehicle consists only of its neighbouring cells. These models were introduced to transportation by Nagel and Schreckenberg (1992). Their compatibility with image processing algorithms has led to extremely fast implementations, such as that of Korček et al. (2011). Mesoscopic models also distinguish individual vehicles; however, their movements depend on aggregate quantities such as capacity and jam density, in addition to the states of nearby vehicles. Queueing models fall into this category (Zhou et al., 2014). It should be noted that this common classification of traffic models into macro, meso, and microscopic does not account for models that simulate the fine details of vehicular dynamics, nor vehicle/vehicle and vehicle/infrastructure communications. Such “sub-microscopic” models are widely used in the study of connected and autonomous vehicle technologies. Examples include Prescan by TASS International (2020) and Zhang et al. (2018).

It has long been recognised that none of these model types is superior to all others, but instead, each has a domain of application (Bourrel and Lesort (2003); Burghout et al. (2005)). For example, models that stem from the kinematic wave theory capture the propagation of congestion waves with fewer parameters than microscopic models. They are more parsimonious for the study of congestion-based interventions, such as ramp metering and variable speed limits on freeways. On the other hand, the dynamics of arterial roads is strongly influenced by traffic signals, with congestion wave speed being of
secondary importance. Vehicles form queues at signalised intersections, and hence mesoscopic queueing models are a good choice for the study of signal control algorithms.

Observations such as these have motivated the development of hybrid approaches to traffic simulation, in which different models are applied to different regions of the network. The main challenges for designing hybrid simulators relate to the interface where vehicles move from one model to another. These interfaces should meet the following basic requirements:

(1) They should not create nor destroy vehicles.
(2) They should preserve any properties of the traffic stream that the modeller wishes to track.
(3) They should propagate congestion from the downstream model into the upstream model.

Similar requirements for hybrid simulators have been articulated previously by other authors (e.g., Burghout et al. (1934)). The first requirement is fundamental and relatively simple. The second depends on which properties the modeller is interested in tracking. In this paper, we always preserve the routing characteristics of vehicles. All other properties of interest are gathered into the notion of a vehicle type (Section 2.1), which is conserved by the simulator.

The third requirement – the backward propagation of congestion – is the most difficult to satisfy. Figure 1 is used for illustration. In Figure 1(a), vehicle x is in an upstream section managed by model $m_u$ while vehicle y is in the downstream section managed by model $m_d$. Assuming $m_u$ is microscopic, then the acceleration of x will depend on the headway and its rate of change, which is given by the position and velocity of vehicle y. However, if model $m_d$ is macroscopic, as in Figure 1(b), then there will be no explicit representation of vehicle y, and thus no headway. Most authors have resolved this problem by creating a “transition zone” between the two models, where both representations coexist (Figure 1(c)). The dynamics of microscopic vehicles within the transition zone are influenced by their lead vehicles and also by the macroscopic density state. The problem of congestion propagation is thus resolved; however, a new difficulty is created: how to preserve consistency between the macroscopic and microscopic representations within the transition zone? “Consistency” here is not a well-defined term, but can generally be taken to mean that the two models report similar aggregate metrics within the transition zone. This is of course not possible to guarantee for a wide class of possible models, and so most studies of hybrid traffic simulation have focused on solving the problem for particular pairs of models.

Burghout et al. (1934) studied the coupling of the mesoscopic model Mezzo with the microscopic model MITSIMLab. Bourrel and Lesort (2003) proposed the combination of the LWR with a compatible microscopic model, such as Newell (2002). This approach was extended by Leclercq (2007), who noted that no transition zone is needed if the models are mutually consistent since, in this case, estimates can be made from upstream information alone. That paper developed a technique for coupling a macroscopic and a microscopic model, both of which were consistent with LWR.

The present paper is, to the best of our knowledge, the first attempt at a general solution to the hybrid simulation problem – meaning one that applies to any pair of models in a wide class. The result is achieved by dispensing with the transition one, and thus removing the difficulty of having to guarantee consistency. In order to meet the third requirement, we allow model $m_u$ to query model $m_d$ for the position of its upstream-most vehicle (vehicle y in this case). This quantity is well defined whenever $m_d$ is microscopic or mesoscopic, and there are reasonable definitions for macroscopic models. Section 5.3.4 provides a possible formula for the macroscopic cell-transmission

Figure 1. The boundary between two models: $m_u$ and $m_d$. 
model. In any case, it is the responsibility of the downstream model to respond to the query in a reasonable manner. That is, in a way that will prevent vehicles from crossing the boundary when the downstream link is full.

The concepts described in this paper have been implemented in the Open Traffic Models software (OTM). Figure 2 illustrates the components of OTM. The main module is the hybrid simulator, which contains the representation of the road network described in Section 2. The hybrid simulator also executes the protocol for coordinating the transfer of vehicles between models (Sections 3 and 4). The models are implemented as plugins that are registered with the hybrid simulator. Any number of models can run simultaneously. To participate in the simulation, a model must implement a set of methods that constitute the model interface. The figure shows two models simulating a small city. Model A is used to simulate the main corridor while Model B is applied to the rest. We will see that the methodology captures a wide range of models, including microscopic, mesoscopic, and macroscopic models of first and second order, discretized with a Godunov scheme. Section 5 describes the three models that have been implemented in OTM. These are the cell transmission model (macro), the spatial queueing model (meso), and Newell’s car-following model (micro).

The hybrid simulator also exposes a control interface which allows modellers to implement and include controller plugins (Section 6). The layer of actuators and sensors mediates the interaction of the controllers with the road network.

2. The road network

All traffic simulation models are built upon some representation of the road network. Macroscopic models tend to use simple representations, consisting of a directed graph with edge capacities, maximum densities, and minimum travel times. There are macroscopic models, such as that of Laval and Daganzo (2006), that use lane-by-lane representations and simulate lane-changing behaviour. Microscopic models usually require lane-specific details, including width, curvature, etc. One of the goals of this work is to allow the user to change the model that is applied to a road segment without having to change its representation. This section describes the data structures that are used to achieve this. We begin with the specification of traffic demands.

2.1. Vehicle types, routing, and demands

As was stated in the introduction, a requirement for hybrid simulators is that they should conserve certain characteristics of the traffic stream as it passes from one model to another. The routing behaviour of vehicles is an essential property of traffic and should always be conserved. Other characteristics depend on the application. For example, the modeller may assign different performance or emission characteristics to
different populations, such as cars and trucks. Or they may wish to follow different populations of vehicles for the purpose of computing performance metrics. It is also possible that different vehicles have different accesses to the network infrastructure – e.g., high-occupancy vehicles can use the HOV lane; connected vehicles and drivers with routing-apps have access to an information service. All of these are captured with a user-defined vehicle type. The vehicle type gathers all distinguishing characteristics of the population that the modeller is interested in tracking. The population is then split according to vehicle type, and each type is assigned a unique id.

Vehicles enter the network through a source, attached to a link. A source produces a stream of vehicles of the given type (multiple sources can be attached to a single link). The time-varying intensity of the source is specified by the user as a discrete-time profile. Excess demand, i.e., demand that exceeds the flow capacity or holding capacity of the link, is held in a limitless buffer. The process that produces the vehicles is specified by the user, and also depends on the model that operates on the source link. A vehicle-based model, for example, may create vehicles according to a Poisson process, while a fluid model may use a sequence of independent random variables.

Each vehicle type is assigned a routing behaviour. The routing behaviour can be either deterministic or probabilistic. Deterministically routed vehicle types are assigned a route (a.k.a. a path), which is a sequence of links starting with the given source link. These vehicles travel along the route and are removed from the network upon exiting the last link in the sequence. They can be diverted to another route only by a routing actuator, such as a routing app. Probabilistically routed vehicles choose their next link at each junction according to turning probabilities or split ratios. These vehicles are removed from the simulation when they exit a terminal link in the network. The split ratios are provided as discrete-time sequences for every junction and vehicle type. They can be modified during the run by an event (e.g., an accident) or a split ratio actuator.

2.2. Road segments

The road network consists of an interconnection of road segments or links. The state of each link is managed by its model. Figure 3 shows a generic link. Each link has a positive number of full-length lanes. It may also have partial-length lanes, such as turn pockets. The four possible partial lane structures are in the “inner-upstream”, “inner-downstream”, “outer-upstream”, and “outer-downstream” positions. Each partial lane structure is characterised by its position, number of lanes, length, and gates.

Figure 4(a) shows an approach to a signalised intersection. The turn pocket is captured as an inner-downstream partial lane. Figure 4(b) shows an...
onramp merge. The merge lane is represented as an outer-upstream partial lane in link 2. In both cases, access to the partial lanes is unrestricted. Figure 4(c) depicts a freeway segment with a priority lane with restricted access. The priority lane can be represented as an inner-downstream (or upstream) partial lane structure with length equal to the full length of the link. Access to the priority lane can be restricted to a series of gates or unrestricted.

2.3. Road parameters

The driving characteristics of the road segments are captured by a set of road parameters. These parameters are, in general, specific to the model that manages the link. However, there are basic roadway characteristics that are common to many traffic models. These are the three parameters of the so-called triangular fundamental diagram: the road capacity ($f$), the speed limit ($v$), and the jam-density ($\rho$). See Figure 5.

It is required that the per-lane values of these three quantities be defined for each road segment. These are interpreted by the models and translated into model-specific parameters. Thus, a basic degree of physical consistency between models is achieved. The models may also require additional parameters to be defined, and these can be provided separately. Section 5 describes how these quantities are interpreted by each of the three canonical models included with OTM.

2.4. Road connections

Thus far, we have focused on the geometry and characteristics of isolated road segments. We will now describe how these road segments are connected.

The standard approach for macroscopic traffic models is to use a graph in which links are identified with edges and their interconnections are represented as vertices. This approach ignores the finer-grained interconnections between lanes, which are needed by most microscopic models. Here we use road connections to specify the relations between links at the lane level.

A road connection is a tuple with two elements: a set of upstream lanes and a set of downstream lanes. $r = (l_{up}^r, l_{dn}^r)$. The interpretation of a road connection is that vehicles that depart the upstream link from a lane contained in $l_{up}^r$ may enter the downstream link through any lane in $l_{dn}^r$. Figure 6 shows a representation of a freeway onramp. The three links are the upstream freeway segment (lanes 1 to 4), the downstream freeway segment (lanes 5 to 8), and the onramp (lane 9). The road connections

Figure 5. The fundamental diagram and required road parameters.

Figure 6. A freeway onramp.
are \( r_1 = (\{1, 2, 3, 4\}, \{5, 6, 7, 8\}) \) and \( r_2 = (\{4\}, \{9\}) \). Movements that do not follow road connections are prohibited. For example, vehicles are not allowed to exit the freeway from lane 3. The granularity of the representation can be controlled by adding or removing road connections. For example, vehicles can be made to preserve their lanes when moving from one link to the next by assigning road connections to every individual lane. However, this level of detail is often not needed.

There may be an overlap between the upstream lanes of two road connections \((l_{up}^f \cap l_{up}^g \neq \emptyset)\), and thus a vehicle within the overlap may have turning options (lane 4 in Figure 6). It is required that these options lead to different links, so that split ratios specified at the link level may be translated into split ratios between road connections.

### 2.5. Lane groups

We use the term lane group for a set of adjacent lanes that share the same set of exiting road connections. Lanes in a single lane group can be expected to advance with approximately equal speed. The model that operates on the link can, therefore, assign to each lane group a single speed-synchronised data structure. For a microscopic model this would be a lane; for a mesoscopic model, a FIFO queue; for a macroscopic model it would be an instantiation of the differential equation. The model may choose to create a finer representation (e.g., lane-by-lane), but this is not required.

### 3. Model interactions

A model is a program that manages the state of one or more links in the simulation. The responsibilities of a model are

1. to maintain the internal state of the lane groups in its links,
2. to emit vehicle packet release requests to the hybrid simulator, in order to send vehicles to a downstream model,
3. to ensure that vehicles only leave the link along road connections that are consistent with their routing,
4. to receive incoming vehicle packets sent from upstream links,
5. to respond to requests for information from the hybrid simulator.

The system requires that models keep track of the type and routing information of all of the vehicles in their links. The routing information is captured by the route id, if the vehicle is of a deterministically routed type, and by the id of the next downstream link, if the vehicle is of a probabilistic type. It is assumed that all vehicles (probabilistically routed vehicles in particular) select, upon entering a link, the link to which they will proceed following that link. This assumption is in contrast to many graph-based models, in which the split ratio is applied as vehicles flow through a multi-output node; that is, when they *exit* the link. This selection of a next link, whether deterministic (provided by the route) or probabilistic (provided by split ratios) is computed by the hybrid simulator and attached to the vehicle packets as they enter the link (see Section 3).

The state index is a tuple that captures the type and routing information of a vehicle. Models must track all state indices separately. Table 1 shows the possible state for a lane group in a macroscopic model. It consists of four state indices (first column) with their respective state values (column two). There are two vehicle types that use this lane group: types 1 and 2. Assuming that type 1 is deterministically routed, the second item in the tuple is the route followed by vehicles with this state index. There are 11.2 vehicles of type 1 on route 7, and 4.7 vehicles of type 1 on route 4. The lane group must, therefore, belong to a link that is on routes 7 and 4. Assuming that type 2 is probabilistically routed, the second item in the tuple is the id of the next link selected by the vehicle. There are 0.3 vehicles of type 2 continuing to link 23, and 9.1 vehicles of type 2 continuing to link 34. The lane group must, therefore, belong to a link that is immediately upstream of links 23 and 34.

Table 1 is an instantiation of a flux packet. Flux packets are used by the program to transfer bits of state from one model to another. While state indices are universal, state values differ from one model to another. As in the table, a first-order macroscopic model will have scalar state values. A second-order macroscopic model may have a two-element array (e.g., number of vehicles and momentum). The state values for a microscopic model will consist of arrays of vehicle objects.

Each of the individual models engaged in the simulation is responsible for managing their internal states. The interactions between these models are coordinated by the hybrid simulator using the protocol described below. This protocol is intended to be compatible with the Godunov scheme for partial differential equations, but also capable of handling discrete vehicles. The Godunov scheme, as illustrated by the cell-transmission model, determines the flow across link boundaries as the minimum between what the

**Table 1. Typical lane group state for a first-order fluid model.**

| State-index | State-value |
|-------------|-------------|
| (1,7)       | 11.2        |
| (1,4)       | 4.7         |
| (2,23)      | 0.3         |
| (2,34)      | 9.1         |
upstream link can send (the demand) and what the downstream link can receive (the supply). The method can also be used to solve second-order PDEs by generalising the definitions of demand and supply Lebacque et al. (2007).

The protocol is triggered when the upstream model \( m_u \) requests to send a packet \( p \) from link \( u \), along a road connection \( r \), to downstream link \( d \) managed by model \( m_d \). Figure 7 shows the elements involved. The road connection \( r \) leads to a set of lane groups \( D_r \) in \( d \). The set of links reached by road connections that leave link \( d \) is denoted with \( \mathcal{E} \).

Upon receiving the request from the upstream model to release \( p \) along \( r \), the system asks the downstream model how much space the packet \( p \) would occupy in link \( d \), and how much space is available. This is done by invoking the following methods in the model interface, as implemented by \( m_d \).

\[
\text{get\_packet\_size} : (p, r) \rightarrow |p| \in \mathbb{R}^+ \tag{1}
\]

\[
\text{get\_max\_packet\_size} : (p, r) \rightarrow \hat{p} \in \mathbb{R}^+ \tag{2}
\]

Here \(|p|\) is the size of packet \( p \) and \( \hat{p} \) is the largest allowable packet size, as determined by the downstream model. These are then used to compute a scaling factor for packet \( p \),

\[
\alpha = \min \left( 1, \frac{\hat{p}}{|p|} \right) \tag{3}
\]

It is assumed that by multiplying the number of vehicles in the packet by \( \alpha \) (uniformly over all state indices), while keeping fixed all other states and vehicle characteristics, the scaled packet \( \alpha p \) will fit in the downstream link. This is true for first-order fluid models, and for the class of generic second-order models described in Lebacque et al. (2007) (the GSOM class). If the sending model is vehicle-based, then the packet must be split into two integer-valued parts, while complying with the space constraint of Equation (3). In this case, the calculation is more complicated: each state index is scaled separately by the largest amount that preserves whole vehicles while not exceeding the value of \( \alpha \) from Equation (3).

In the particular case of a downstream mesoscopic model with vertical queueing, \( \alpha \) always evaluates to 1, since all packets can be accepted without restriction. For second-order fluid models, such as those in the GSOM class, the maximum packet size depends on the composition of the packet, in addition to the state of the downstream link. This is the reason for including \( \alpha \) as a parameter in Equation (2).

It should be noted that it is not necessary that all models share the same norm; that is, they need not agree on the size of a packet. They must, however, return a value \(|p|\) such that \( \alpha p \) can be accommodated.

Next, before delivering \( \alpha p \) to the downstream model, the system determines the next link for probabilistically routed vehicles. This is done by sampling the split ratios assigned to links in \( \mathcal{E} \) in Figure 7. If the upstream model is fluid-based, then \( \alpha p \) is divided into as many smaller packets as non-zero split ratios exist for the given vehicle type. If it is vehicle-based, then each vehicle is assigned a next link with probabilities corresponding to the split ratios. The resulting collection of vehicle packets is then passed to the downstream model using the method

\[
\text{send\_packets} : (\mathcal{P}, r) \rightarrow \text{void} \tag{4}
\]

Here, \( \mathcal{P} \) is the collection of packets being sent, and \( r \) is the road connection along which they travel. Upon receiving the packets, the downstream model must distribute them amongst the lane groups in \( D_r \). One possible strategy for doing this is to split each packet in \( \mathcal{P} \) into \(|D_r|\) equal packets. Another is to split them in a way that seeks equilibrium between the lane groups. OTM provides implementations of these two strategies that can be used by model plugins.

After lane group assignment, the final step of the process is to translate all of the packets into the native representation of the receiving model and incorporate them into the state. This is done by the downstream model using vehicle classes and translators provided by the program.

4. Simultaneous requests

The previous section described the interaction between an upstream and a downstream model when a single vehicle packet is sent between the two. In general, however, packets may be sent simultaneously through a multi-input/multi-output connection. This is the case,

![Figure 7](image)

Figure 7. Link \( u \) requests to send packet \( p \) along road connection \( r \). Links are drawn with a dotted rectangle, lane groups with a solid rectangle.
for example, with macroscopic models where all lane groups emit packets at every time step. In this section, we describe the extension of the model coordinator for this case. This is known in the transportation literature as a "node model". The purpose of the node model is to scale the packets that travel through a MIMO node such that they can be accommodated by the downstream lane groups, while satisfying some criterion such as maximising flow (Tampère et al., 2011). We describe here the particular node model that has been implemented in OTM, which is an adaptation of the method of Wright et al. (2017) to a network representation consisting of lane groups and road connections, instead of nodes and links.

Figure 8 provides an illustration. A set of packets is requested to be sent from upstream lane groups in the set \( \mathcal{G} \), through road connections in the set \( \mathcal{R} \), to downstream lane groups in the set \( \mathcal{H} \). The "node" in the figure has three upstream lane groups \{1, 2, 3\}, which connect via three road connections \{1, 2, 3\} to three downstream lane groups \{4, 5, 6\}. The figure also shows a graph representation of the connectivity between these elements. This graph is encoded in "upstream" and "downstream" sets for each element. The downstream set of lane group 1 is road connection 1, the upstream set of lane group 6 is road connections 1 and 3, etc. These sets are denoted with \( D_g \), \( U_u \), \( U_d \), and \( U_h \) for the upstream and downstream sets of elements in \( \mathcal{G}, \mathcal{R}, \) and \( \mathcal{H} \).

The algorithm of the node model is shown in Figure 9. With each iteration, the upstream models send a portion of their packets. The portion is limited either by upstream demand or downstream supply, and hence, as with Wright et al. (2017), the algorithm is guaranteed to complete within a number of iterations not exceeding the number of lane groups. We confine the description to the simpler case, where the total supply in downstream lane groups is independent of the composition of the packets being sent to those lane groups. This case covers first-order fluid models and most vehicle-based models. The more general case has been developed by Wright and Horowitz (2017) for graph representations.

### 4.1. Node model steps

(step 0) Initialisation. Use the model interface to obtain the demands (flux packet sizes) and supplies (maximum flux packet sizes) for all lane groups that are incident on the node. We use \( d_g^r \) to denote the size of the packet emitted by lane group \( g \in \mathcal{G} \) along road connection \( r \in \mathcal{R} \), and \( s^h \) for the available supply in lane group \( h \in \mathcal{H} \). The demands are obtained with equation (1) and the supplies with equation (2). This function is expected to return zero whenever lane group \( h \) is full, and thus prevent vehicles from entering.

(step 1) \( \forall g \in \mathcal{G} \), find which upstream lane groups are blocked or empty. For each lane group \( g \), we first construct the set of road connections with positive demand:

\[
D_g^+ = \{ r \in D_g : d_g^r > 0 \} \quad \forall g \in \mathcal{G} \tag{5}
\]

Boolean variables \( \Box^F \), \( \Box^R \), and \( \Box^H \) are used to indicate that a lane group or road connection is "blocked". A downstream lane group \( h \) is blocked if it has zero supply \( s^h \).

\[
\Box^H = [s^h = 0] \quad \forall h \in \mathcal{H} \tag{6}
\]

A road connection is blocked if all of its downstream lane groups are blocked. This reflects the assumption that packets travelling along road connection \( r \) may be placed in any of the lane groups in \( D_r \).

\[
\Box^R = [\Box^H]_{h \in D_r} \quad \forall r \in \mathcal{R} \tag{7}
\]

The variable \( \Box^F \) is true whenever lane group \( g \) is either blocked or empty. An upstream lane group is blocked whenever any of its exiting road connections are blocked, provided there is a demand for that road connection (Equation (8)). This reflects the FIFO assumption within the lane group: a single vehicle moving along a blocked road connection will block the entire upstream lane group. A lane group is empty if its demand has been delivered, and therefore the set \( D_g^+ \) is empty.

\[
\Box^F = [\Box^R]_{r \in D_g^+} \lor [D_g^+ = \emptyset] \quad \forall g \in \mathcal{G} \tag{8}
\]

![Figure 8. Node model.](image)

![Figure 9. Node model algorithm.](image)
The computation terminates if all upstream lane groups are blocked or empty.

\[
\text{stop if } \left[ \exists g : \gamma_{g} \in G \right] \quad (9)
\]

(step 2) \( \forall r \in \mathcal{R} \), calculate the total demand on each road connection by aggregating over incoming lane groups.

\[
d^{r} = \sum_{g \in U_{r}} d_{r}^{g} \quad (10)
\]

Compute the downstream supply available to each road connection, disregarding the demands from other road connections. This calculation involves \( \lambda_{h}^{r} \in (0, 1] \), the portion of lane group \( h \) that is accessible to road connection \( r \). In Figure 8, \( \lambda_{h}^{r} = 0.5 \) since road connection 1 has access to only one of the two lanes in lane group 6.

\[
s^{r} = \sum_{h \in D_{r}} \lambda_{h}^{r} s_{h}^{r} \quad (11)
\]

Compute the apportionment of the demand on connector \( r \) to each of its downstream lane groups \( h \). This is done in proportion to the available supply.

\[
\alpha_{h}^{r} = \begin{cases} 
1 & \text{if } s^{r} = 0 \\
\lambda_{h}^{r} s^{r} / s^{r} & \text{otherwise} 
\end{cases} \quad \forall h \in D_{r} \quad (12)
\]

(step 3) \( \forall h \in \mathcal{H} \), compute the total demand on downstream lane group \( h \) as the sum of demands on its incoming road connections scaled by the apportionment factor (Equation (13)). \( y_{h}^{r} \) is the per cent excess demand on lane group \( h \) (Equation (14)).

\[
d_{h}^{r} = \sum_{r \in U_{h}} \alpha_{h}^{r} d^{r} \quad (13)
\]

\[
y_{h}^{r} = \max\left(0, 1 - \frac{s_{h}^{r}}{d_{h}^{r}}\right) \quad (14)
\]

(step 4) \( \forall r \in \mathcal{R} \), propagate the excess demand factors to the road connectors. The demand on each road connector must be reduced by the worst-case factor. Note that this formula will yield \( y_{h}^{r} = 1 \) for a blocked road connection.

\[
y^{r} = \sum_{h \in D_{r}} \alpha_{h}^{r} y_{h}^{r} \quad (15)
\]

(step 5) \( \forall g \in \mathcal{G} \), compute the reduction factor for each upstream lane group as the largest reduction factor of its outgoing road connections. Only road connections with positive demand \( r \in D_{g}^{+} \) are considered in this calculation. If the lane group is empty, then set the reduction factor to 1.

\[
y_{g}^{r} = \begin{cases} 
1 & \max_{r \in D_{g}^{+}} (y^{r}) = 0 \\
\max_{r \in D_{g}^{+}} (y^{r}) & \text{otherwise} 
\end{cases} \quad (16)
\]

Compute the demand that advances along each road connection exiting lane group \( g \) (Equation (17)), and update the demand that remains (Equation (18)). \( \forall r \in D_{g}^{+} \):

\[
\delta_{g}^{r} = d_{g}^{r} (1 - y_{g}^{r}) \quad (17)
\]

\[
d_{g}^{r} \leftarrow y_{g}^{r} d_{g}^{r} \quad (18)
\]

(step 6) \( \forall r \in \mathcal{R} \), calculate the portion of the advancing flow on each road connection.

\[
\delta^{r} = \sum_{g \in U_{r}} \delta_{g}^{r} \quad (19)
\]

(step 7) \( \forall h \in \mathcal{H} \), collect the flow entering each downstream lane group and reduce the downstream supplies.

\[
\delta_{h}^{r} = \sum_{r \in U_{h}} \frac{1 - y_{h}^{r}}{1 - y^{r}} \alpha_{h}^{r} \delta^{r} \quad (20)
\]

\[
y^{r} \leftarrow y^{r} - \delta_{h}^{r} \quad (21)
\]

With demands and supplies reduced by Equations (18) and (21), we return to step 1 to determine whether there is remaining demand that can advance. If there is none, then the iteration terminates. Once the iteration is done, the scaling factors that are the output of the node model are computed as the proportion of each packet that has been sent.

5. Three models

Models are implemented in OTM as plugins, that is, as relatively small pieces of code that define the functionality specified by the modelling interface. This interface includes methods such as methods (1), (2) and (4), as well as other methods that are needed to track performance measures. Additionally, the model must provide methods for advancing its internal state in time. That is, it must provide the longitudinal and lateral dynamics. All other functionality – routing, traffic control, performance metrics, execution, and process parallelisation – are managed by OTM. The main requirement placed on the model is that it must only release vehicle packets along road connections that are consistent with the routing behaviour of the vehicles. This implies that vehicles must move laterally within a link (from one lane group to another) in order to reach a valid exiting road connection. In short, the model must implement a lane-changing strategy.

This section describes implementations of the three canonical examples for microscopic, mesoscopic, and macroscopic modelling; respectively, Newell’s simplified car-following model, the spatial queueing model, and the cell-transmission model.
5.1. Newell’s car-following model

The model described here is a discrete-time version of Newell (2002), with an additional term to account for segment capacities. Each lane group contains a single first-in-first-out queue of vehicles, numbered starting with the downstream-most vehicle. The position of the \( i \)th vehicle at time \( t \), \( x_i(t) \) is related to the position of the \((i-1)\)th vehicle at a previous time.

\[
x_i(t + \Delta t) = x_i(t) + \max\left(0, \min\left(\delta v^i, h_i(t) - \delta w^i, h_i(t)\delta f^i\right)\right)
\]

(22)

Here \( \Delta t \) is the model’s simulation time step; \( \delta v^i, \delta w^i, \) and \( \delta f^i \) are sampled from distributions whose means correspond to the left and right slopes of the fundamental diagram, and the capacity (see Figure 5).

\[
\delta v^i \sim N(\bar{v}\Delta t, \sigma^v)
\]

(23)

\[
\delta w^i \sim N(w\Delta t, \sigma^w)
\]

(24)

\[
\delta f^i \sim N(\bar{f}\Delta t, \sigma^f)
\]

(25)

\( \sigma^v, \sigma^w, \) and \( \sigma^f \) are user-defined standard deviations. \( w \) is the magnitude of the slope of the right side of the fundamental diagram. It can be computed from the parameters of Figure 5, \( w \Delta = \bar{f} \bar{v} / (\rho \bar{v} - \bar{f}) \).

The first term in the “\( \min \)” of Equation (22) applies to lead vehicles that travel with free-flow speed \( \bar{v} \). The second term involves \( h_i(t) \), the headway of the \( i \)th vehicle at time \( t \). This is defined, for all vehicles except the first vehicle in the lane group, as the distance to its leader: \( h_i(t) = |x_i(t) - x_{i-1}(t)| \). The first vehicle, however, does not have a leader in its lane group, and hence it is possible that its leader belongs to a different model. Computing the headway for the first vehicle may, therefore, involve requesting the position of the upstream-most vehicle in the lead vehicle’s next link. This is done by calling a model method provided by the modelling interface.

\[
\text{get\_distance\_to\_last\_vehicle} : (r) \rightarrow \eta \in \mathbb{R}^+
\]

(26)

This measures the distance from the upstream boundary of the link accessed by road connection \( r \) to the nearest vehicle in any of the lane groups accessible from \( r \). The third term in Equation (22) imposes a capacity constraint.

The lane change model that has been implemented for this model is simple. Vehicles that enter a link are placed directly into a lane group that connects to their target road connection, irrespective of whether the road connection they enter by actually connects to that lane group. In other words, they change lanes immediately and without obstruction into their target lane group. If the target lane group is already full, then the vehicle is placed in an additional FIFO waiting queue attached to that lane group, and enters as soon as space becomes available.

5.2. Spatial queueing model

This mesoscopic model is similar to the one reported in Varaiya (2013). Each lane group is equipped with two queues. The first is a transit queue, which delays every vehicle entering the lane group by the free-flow travel time. After leaving the transit queue, the vehicle enters the waiting queue, which is a FIFO queue serviced by a Poisson process. See Figure 10.

This model makes use of the three supply-side parameters of Figure 5. The speed limit \( \bar{v} \) is used to calculate the free-flow travel time. The capacity \( \bar{f} \) is the average discharge rate for the waiting queue. The jam density \( \bar{\rho} \) is used to compute the available supply for incoming vehicle packets. These packets are rejected whenever the combined number of vehicles in the transit and waiting queues reaches the maximum value. The lane-changing model for the queuing model is identical to that of the car-following model: an arriving vehicle is placed immediately into its target lane group unless it is full, in which case the vehicle is held in an additional FIFO waiting queue until space becomes available.

5.3. Cell-transmission model

The macroscopic model included with OTM is an adaptation of the cell-transmission model (CTM) of Daganzo (1994), with a lane change strategy that is similar to that of Laval and Daganzo (2006). Lane groups are divided into cells. Figure 11 shows a single link with three lane groups. All of the cells in a link are of equal size, which is computed as the
largest that yields an integer number of cells, without exceeding a user-defined maximum cell size. The simulation time step $\Delta t$ must comply with the Courant–Friedrichs–Lewy condition that no cell can be traversed in one time step by a vehicle travelling at maximum speed.

Each cell $i$ has up to two lateral neighbours $in(i)$ and $out(i)$ in adjacent lane groups. The state of a cell $i$ consists of the number of vehicles per state index $s$: $n_i^s$ (omitting the time index for convenience). $S^\ell$ is the set of all state indices that can use link $\ell$. The link shown in Figure 11 carries two states: one represented by a blue arrow, which is headed for link 1, and another represented by a red arrow, which is headed for link 2. Each lane group $g$ carries a map $\rho^g$ that returns, for each state index, the road connection $r$ that it must use to reach its next link.

$$\rho^g : S^\ell \to R$$  \hspace{1cm} (27)

In Figure 11 we have

$$\rho^1(\text{blue}) = 1 \quad \rho^1(\text{red}) = \text{null}$$

$$\rho^2(\text{blue}) = 2 \quad \rho^2(\text{red}) = 3$$

$$\rho^3(\text{blue}) = \text{null} \quad \rho^3(\text{red}) = 3$$

It is a requirement on the simulator that this map should return at most one road connection. That is, for each lane group $g$, the road connections that leave $g$ must all lead to different links. We define the target lane group set for a state $s$ within link $\ell$ as those from which its next link can be reached. In the figure, the target lane group set for the blue state is $\{1,2\}$, and for the red state, it is $\{2,3\}$. Each lane group also carries a map $\Phi^g$ indicating the direction in which a state must change lanes in order to reach its target lane group set.

$$\Phi^g : S^\ell(g) \to \{in, out, none\}$$  \hspace{1cm} (28)

$\ell(g)$ is defined as the link that contains lane group $g$, and hence $S^\ell(g)$ is the set of states that use $g$. $\Phi^g(s)$ returns $in$, $out$, or $none$ depending on whether the state must change lanes inward, outward, or not at all to reach its target road connection. Figure 11 illustrates this map.

Every lane group managed by the fluid-dynamical model emits vehicle packets every $\Delta t$. These packets are assessed by the OTM node model, which involves calls to the supply calculation function. Following this, every lane group receives a set of vehicle packets from its incoming road connections, and proceeds to update its internal state. These three steps – vehicle packet generation, supply calculation, and state update – are described below.

### 5.3.1. Vehicle packet generation

This stage consists of five internal steps. First, the number of vehicles that change lanes during the upcoming time step is computed with Equations (29) through (31), and used to compute an intermediate state with Equation (32). The intermediate state is then used to evaluate the longitudinal demand, in step Equations (33). $C$ is the set of all cells.

**[step 1]** $\forall i \in C$, compute the total number of vehicles performing each of the lane change manoeuvres.

$$n^i_\mu = \sum_{\{s : \Phi^g(i)(s) = \mu\}} n^i_s \quad \mu \in \{in, out, stay\}$$  \hspace{1cm} (29)

Here, $g(i)$ is defined as the lane group for cell $i$, and therefore $\{s : \Phi^g(i)(s) = \mu\}$ is the set of states that perform manoeuvre $\mu$ in cell $i$.

**[step 2]** $\forall i \in C$, compute the total number of vehicles in the cell.

$$n^i = n^i_\text{in} + n^i_\text{out} + n^i_\text{stay} = \sum_s n^i_s$$  \hspace{1cm} (30)

**[step 3]** $\forall i \in C$, compute the scaling factor. $\xi^i \in [0,1]$ is a supply apportionment factor for lane change movements. The space available in cell $i$ for vehicles changing lanes is $\xi^i(n^i - n^i_\text{in})$, where $n^i_\text{in}$ is the maximum occupancy. As illustrated in Figure 12, the total number of vehicles entering cell $i$ is $n^i_\text{in} + n^i_\text{in}^\text{stay}$, and the scaling factor $\beta^i$ is

$$\beta^i = \min \left(1, \frac{\xi^i(n^i - n^i_\text{in})}{n^i_\text{in} + n^i_\text{in}^\text{stay}}\right)$$  \hspace{1cm} (31)

**[step 4]** $\forall i \in C$ and states $s$ in cell $i$ apply the scaling factors to the lane change demands and execute the manoeuvres. This results in an intermediate state $n^i_s$, which represents the number of vehicles for state $s$ after the lane changes have completed, but before vehicles have moved forward.

![Figure 11](image-url)
5.3.2. State update

Following the completion of the node model, reduced packets are passed between the lane groups. These are then translated into the macroscopic representation, and consolidated into two packets per lane group – one entering and one leaving. The content of the upstream and downstream packets for lane group \( g \) is the total number of vehicles per state \( s \): \( n^{\text{up}}_s \) and \( n^{\text{dn}}_s \). The densities per cell and state can then be updated using the law of conservation of vehicles. The incoming flow for the upstream-most cell is \( n^{\text{up}}_s \), the outgoing flow for the downstream-most cell is \( n^{\text{dn}}_s \). The flows across internal cell boundaries are given by the standard CTM formulas.

5.3.3. State update

Following the completion of the node model, reduced packets are passed between the lane groups. These are then translated into the macroscopic representation, and consolidated into two packets per-lane groups – one entering and one leaving. The content of the upstream and downstream packets for lane group \( g \) is the total number of vehicles per state \( s \): \( n^{\text{up}}_s \) and \( n^{\text{dn}}_s \). The densities per cell and state can then be updated using the law of conservation of vehicles. The incoming flow for the upstream-most cell is \( n^{\text{up}}_s \), the outgoing flow for the downstream-most cell is \( n^{\text{dn}}_s \). The flows across internal cell boundaries are given by the standard CTM formulas.

5.3.4. Upstream-most vehicle position

All models must respond to simulator requests for the position of the upstream-most vehicle in a lane group. The CTM does this based on the supply function of the upstream-most cell in the lane group. By assuming that density concentrates towards the downstream part of the cell, the distance from the upstream edge to the last vehicle is \( L'(n - n')/n' \), where \( L' \) is the length of the cell, \( n' \) is the maximum allowable vehicles, and \( n' \) is the total current vehicles in cell \( i \).

6. Control elements

We have thus far described the simulation of the hybrid open-loop traffic dynamics. We now describe how this dynamics can be influenced by a feedback algorithm or a controller. Controllers are implemented in OTM as plugins, similar to models. They do not interact directly with the traffic models, but instead through sensors and actuators. A sensor is any element that extracts information from the models and provides it to a controller. An actuator delivers the control command from the controller to the models. OTM provides a number of sensors and actuators that can
be used by the modeller to assemble the control infrastructure. Sensors and actuators are model agnostic: they operate using methods from the modelling interface, and are not concerned with the implementation details of those methods. Thus, the traffic models used in a scenario can be changed without modifying the control structures.

Each controller can register with one or more sensors and one or more actuators. An actuator, however, can only be assigned to a single controller. Each controller is prescribed a time step by the modeller. This time step need not be equal to (or a multiple of) the time steps of the underlying models. At every time step, the controller will read the measurements from its sensors, update its commands, and then transmit the commands to its actuators. Sensors and actuators have their own time steps, and again these are independent from the time steps of the controllers and models. Sensors use interface methods to extract information from the models at each sensor time step. For example, the fixed local sensor uses the `get_total_vehicles_in_lanegroup` method. OTM currently provides the following basic sensors:

- Fixed local sensor. This sensor is attached to the pavement, and can be used to extract density, flow, and speed information for a particular cross section.
- Fixed lane group sensor. This sensor is attached to a lane group, and can be used to obtain the total number of vehicles and their speeds.
- Probe. This sensor is attached to a vehicle and can be used to measure its speed and local environment.

The functionality required for the probe sensor is implemented in OTM’s basic vehicle class. This class can be extended for use in any vehicle-based model. When a probe vehicle enters a fluid-based model, OTM automatically creates a ‘virtual vehicle’, which it tracks through the fluid network using the local speed provided by the model.

In addition to these, there are also methods for querying exogenous information about the scenario. This includes information about past and future demands and split ratios, controller states, and geometric information of the links and lane groups.

These are the actuators that are currently available in OTM:

- Road connection blocking actuator. This actuator can be used to open and close road connections, and thus to mimic traffic signals.
- Variable speed limit. This actuator is used to change the speed limit of the link and its lane groups.
- Router. Used to change the route of a deterministically routed vehicle type.
- Demand modifier. Alter the profile of demand intensity for a source and vehicle type.
- Split ratio actuator. Alter the profile of split ratios for a given vehicle type at a junction.

In contrast to sensors, the implementation of these actuators does not make use of the modelling interface, but rather act by modifying the inputs to the models (e.g., road parameters).

### 7. Experiments

We apply the OTM software to a simple linear network with two models $m_u$ and $m_d$. Figure 13 illustrates the setup. The links are all of length 500 metres. They share per-lane characteristics: the capacity is 1,000 veh/hr/lane, jam density is 100 veh/km/lane, and the free-flow speed is 100 km/hour. Links 0 through 4 have two lanes, and hence their flow and holding capacities are 2,000 veh/hr and 100 vehicles, respectively. Link 5 has half of these values (1,000 veh/hour and 50 vehicles). Links 0, 1, and 2 are managed by $m_u$; links 3, 4, and 5 by $m_d$. A source of 1,500 veh/hr is applied to link 0. This causes congestion to accumulate upstream of link 5. This congestion propagates over the modelling interface, and dissipates after the demand is removed. This experiment is meant only to demonstrate the backward propagation of congestion with different combinations of models. It does not exercise other features of the program, such as vehicle types, lane changing, the node model, and control structures.

Figure 14 shows the result when $m_u$ is a macroscopic model and $m_d$ is mesoscopic. The left-hand plot shows time series plots for each of the links. The right-hand plot shows a contour of vehicle densities in the time/space plane. Vehicles immediately begin to queue in link 5. The number of vehicles in link 5 reaches 50 after about 400 seconds, and then the queue spills into link 4. The queue reaches the model boundary at around 1,750 seconds and propagates into the macroscopic model. The queueing density and speed in the

![Figure 13. Experimental setup.](image-url)
macroscopic portion are 55 vehicles and 9.09 km/hr, according to the hydrodynamic theory. Notice that the congested density in the fluid model is significantly lower than in the mesoscopic model (55 vehicles versus 100 vehicles). As a consequence, the speed of propagation of congestion is higher. After the demand is removed (at 2,500 s) the congestion dissipates, again at a faster pace in the macroscopic model than in the mesoscopic model. Observe that the source link (link 0) is represented in the macroscopic model with a single cell, as opposed to the five cells of links 1 and 2. This particular macroscopic model assigns a simplified dynamics to source links.

Figure 15 shows the result when $m_u$ is macroscopic and $m_d$ is microscopic. For the microscopic model, the capacity reduction of link 5 is applied throughout its length, and not only at the downstream edge of the link, as with the macroscopic and mesoscopic models. This can be seen on the upper-right-hand side of Figure 15 as vehicles increase their spacing when they enter link 5. This causes congestion to form more quickly than in the previous case. The congestion wave reaches the model boundary after approximately 540 seconds. Measuring from the plot, the speed of propagation of congestion in the microscopic region is approximately 8.4 km/hour, which is slower than in the macroscopic model, and faster than in the mesoscopic model. The queueing density in the microscopic model is about 34 vehicles in 500 metres. The congestion reaches the upstream boundary of the segment at around 1,400 seconds. After that, the demand is turned off and congestion dissipates.

Figure 16 shows the result when $m_u$ is macroscopic and $m_d$ is microscopic. This is the simplest case since both models are vehicle-based. The downstream microscopic side is similar to Figure 15. On the mesoscopic side, the grey area is proportional to the combined number of waiting vehicles in links 0, 1, and 2. The green area is proportional to the total number of vehicles in transit in links 0, 1, and 2. We can see that the mesoscopic queue accumulates more slowly than the microscopic queue, and at a higher density (100 vehicles versus 34 vehicles in 500 metres). The demand, in this case, was removed after 1,400 seconds. Figure 17 shows the result when $m_u$ is microscopic and $m_d$ is macroscopic. This experiment demonstrates the translation of flux packets from a fluid-based model into a vehicle-based model. Observe that the sink link in the macroscopic model is represented by a single cell, as opposed to five cells in links 3 and 4. The macroscopic model does not track detailed dynamics in sink links.

8. Conclusion

In the experiments of the previous section, it can be observed that congestion propagates much slower in
the mesoscopic model than in the microscopic and macroscopic models. This despite the fact that they all have the same capacity, free-flow speed, and jam density. To join the mesoscopic and macroscopic models in a transition zone, one would have to adjust the parameters of one or both in order to achieve consistency (equal congestion wave speeds). However, the parameters that achieve this depend on the bottleneck capacity, which may vary depending on the characteristics of the bottleneck. This essential problem has hindered the development and applicability of hybrid simulators that combine fluid-based and vehicle-based models.

The goal of this work has been to develop a general solution for hybrid traffic simulation, meaning one that is capable of combining arbitrary pairs of models chosen from a wide class. This was achieved by replacing the transition zone with a protocol that allows models to negotiate the sizes of flux packets that are passed from one to another. The protocol is closely analogous to Godunov’s numerical method for solving conservation equations, and can therefore be applied to the LWR model as well as the GSOM class of second-order models.

A second goal of this work has been to provide an implementation of the proposed hybrid simulator that allows users to arbitrarily assign models to links in a traffic network. A network representation was proposed that is sufficiently detailed to sustain all three types of models considered. The representation is more fine-grained than typical single-pipe graphs used in macroscopic modelling, but coarser than the lane-by-lane specifications used in many microscopic models. Instead, the unit is the lane group, which aggregates one or more lanes that are expected to advance at similar speeds. This grouping is deduced from the user-supplied road connections. Models are allowed to assign a single instantiation of the longitudinal dynamics to each lane group. This corresponds to a row of vehicles for the microscopic model, a pair of FIFO queues for the mesoscopic model, and a differential equation for the macroscopic model.

Apart from a generic description of the road, the goal of hybrid modelling also requires specifications of the demands and control systems that are independent of the model dynamics. The hybrid simulator recognises two types of demands: deterministically and probabilistically routed. All other vehicle characteristics are captured by the vehicle type. Models must conserve these characteristics. This is done by tracking a vehicle’s state index. Control algorithms are isolated from the model dynamics with actuators and sensors. The program provides a host of useful actuators and sensors which communicate with user-supplied models and control algorithms through well-defined interfaces.
Open Traffic Models (OTM) is an implementation of this scheme. The software includes the three example models described here. OTM allows traffic researchers to easily test new models by implementing the interface, which consists of methods for executing the hybrid protocol, interacting with actuators and sensors, and computing performance metrics. All other aspects of network simulation (e.g., the node model and traffic control) are provided by OTM. The present paper has focused on a description of the approach and has provided a very limited demonstration of the capabilities of the software. We intend in future publications apply OTM to several problems in traffic management: signal control, ramp metering, HOV and HOT lane analysis, and path planning. We intend also to deploy the software in a distributed computing environment such as a high performance computing cluster. OTM is open-source; it can be obtained from https://github.com/ggomes/otm-sim.

**Disclosure statement**

No potential conflict of interest was reported by the author.

**ORCID**

Gabriel Gomes [http://orcid.org/0000-0002-3258-0623](http://orcid.org/0000-0002-3258-0623)

**References**

Bourrel, E., & Lesort, J. B. (2003). Mixing Microscopic and Macroscopic Representations of Traffic Flow: Hybrid Model Based on Lighthill-Whitham-Richards Theory. *Transportation Research Record*, 1852(1), 193–200.

Burghout, W., Koutsopoulos, H., & Andréasson, I. (2005). Hybrid mesoscopic–microscopic traffic simulation. *Transportation Research Record*, 1934(1), 218–225. [https://doi.org/10.1177/0361198105193400123](https://doi.org/10.1177/0361198105193400123)

Chandler, R., Herman, R., & Montroll, E. (1958). Traffic dynamics: Studies in car following. *Operations Research*, 6(2), 165–184. [https://doi.org/10.1287/opre.6.2.165](https://doi.org/10.1287/opre.6.2.165)

Daganzo, C. (1994). The cell transmission model: a dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B*, 28B(4), 269–287. [https://doi.org/10.1016/0191-2615(94)90002-7](https://doi.org/10.1016/0191-2615(94)90002-7)

Greenshields, B. (1935). A Study of Traffic Capacity. *Highway Research Board*, 14, 448–477. Retrieved from https://trid.trb.org/view/120649

Hale, D. (2019). Quick scan of TRB 2019 papers with a tie-in to traffic simulation. *SimSub Annual Report*, 11–16. Retrieved from https://www.simsub.bagley.mststate.edu/Files/Newsletters/2017%20-%202019/SimSubAnnualReport-Jan2019.pdf

Intelligent Transportation Systems. (2018). *Transportation Research Record*. 2672(19). Retrieved from https://journals.sagepub.com/toc/trra/2672/19

Korčec, P., Sekanina, L., & Fučík, O. (2011). Cellular automata based traffic simulation accelerated on GPU (pp. 395–402). Institute of Automation and Computer Science FME BUT. Retrieved from https://www.fit.vut.cz/research/publication/9555

Laval, J., & Daganzo, C. (2006). Lane-changing in traffic streams. *Transportation Research Part B: Methodological*, 40(3), 251–264. [https://doi.org/10.1016/j.trb.2005.04.003](https://doi.org/10.1016/j.trb.2005.04.003)

Lebacque, J. P., Mammar, S., & Haj-Salem, H. (2007). Generic second order traffic flow modelling In R. E. Allsop, M. G. H. Bell, and B. G. Heydecker (Eds.), 17th international symposium on transportation and traffic theory. Amsterdam: Elsevier.

Leclercq, L. (2007). Hybrid approaches to the solutions of the “Lighthill–Whitham–Richards” model. *Transportation Research Part B: Methodological*, 41(7), 701–709. [https://doi.org/10.1016/j.trb.2006.11.004](https://doi.org/10.1016/j.trb.2006.11.004)

Lieberman, E. (2004). Brief history of traffic simulation. In *Traffic and transportation simulation - looking back and looking ahead: Celebrating 50 years of traffic flow theory, a workshop* (pp. 17–29). Transportation Research Board.

Lighthill, M. J., & Whitham, G. B. (1955). On kinematic waves II. A theory of traffic on long crowded roads. *Proceedings of the Royal Society A*, 229(1178), 317–345. [https://doi.org/10.1098/rspa.1955.0089](https://doi.org/10.1098/rspa.1955.0089)

Lopez, P. A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flöterodt, Y. P., Hilbrich, R., ... & Wießner, E. (2018). Microscopic traffic simulation using sumo. In 2018 21st International Conference on Intelligent Transportation Systems (ITSC) (pp. 257–2582). IEEE.

Nagel, K., & Schreckenberg, M. (1992). A cellular automaton model for freeway traffic. *Journal De Physique I*, 2(12), 2221–2229. [https://doi.org/10.1051/jp1:1992277](https://doi.org/10.1051/jp1:1992277)

Newell, G. (2002). A simplified car-following theory: A lower order model. *Transportation Research Part B: Methodological*, 36(3), 195–205. [https://doi.org/10.1016/S0191-2615(00)00044-8](https://doi.org/10.1016/S0191-2615(00)00044-8)

Prescan by TASS International. (2020). [https://tass.plm.automation.siemens.com/prescan](https://tass.plm.automation.siemens.com/prescan)

Richards, P. (1956). Shock waves on the highway. *Operations Research*, 4(1), 42–51. [https://doi.org/10.1287/opre.4.1.42](https://doi.org/10.1287/opre.4.1.42)

Tampère, C., Corthout, R., Cattrysse, D., & Immers, L. H. (2011). A generic class of first order node models for dynamic macroscopic simulation of traffic flows. *Transportation Research Part B: Methodological*, 45(1), 289–309. [https://doi.org/10.1016/j.trb.2010.06.004](https://doi.org/10.1016/j.trb.2010.06.004)

Van Wageningen-Kessels, F., van Lint, H., Vuijk, K., & Hoogendoorn, S. (2015). Genealogy of traffic flow models. *EURO Journal on Transportation and Logistics*, 4(4), 445–473. [https://doi.org/10.1007/s13676-014-0045-5](https://doi.org/10.1007/s13676-014-0045-5)

Varaiya, P. (2013). Max pressure control of a network of signalized intersections. *Transportation Research Part C: Emerging Technologies*, 36, 177–195. [https://doi.org/10.1016/j.trc.2013.08.014](https://doi.org/10.1016/j.trc.2013.08.014)

Wright, M., Gomes, G., Horowitz, R., & Kurzhanskiy, A. (2017). On node models for high-dimensional road networks. *Transportation Research Part B: Methodological*, 105, 212–234. [https://doi.org/10.1016/j.trb.2017.09.001](https://doi.org/10.1016/j.trb.2017.09.001)

Wright, M., & Horowitz, R. Generic second-order macroscopic traffic node model for general multi-input multi-output road junctions via a dynamic system approach. arXiv:1707.09346 [cs, math], July 2017.

Zhang, Y., Cassandras, C., Li, W., & Mosterman, P. A. (2018). Discrete-event and hybrid simulation framework based on SimEvents for intelligent transportation system analysis. *IFAC PapersOnline*, 51(7), 323–328. [https://doi.org/10.1016/j.ifacol.2018.06.320](https://doi.org/10.1016/j.ifacol.2018.06.320)

Zhou, X., Taylor, J., & Pratico, F. (2014). DTAlite: A queue-based mesoscopic traffic simulator for fast model evaluation and calibration. *Cogent Engineering*, 1(1). [https://doi.org/10.1080/23311916.2014.961345](https://doi.org/10.1080/23311916.2014.961345)