Aurora — Object-Oriented Framework for Simulation and Analysis of Flow Networks

Alex Kurzhanskiy, Jaimyoung Kwon, Pravin Varaiya

June 2008

Abstract

Aurora object-oriented framework is a set of generic Java classes and interfaces that can be used as base objects in applications that model flow networks, such as road networks, networks of irrigation canals, or oil and gas pipelines. Any network consists of directed links, nodes and monitors. A substance (traffic, water, oil or gas, etc.) flows through the links that connect the nodes. Nodes can be simple, used to connect one or more input with one or more output links, and complex, which are networks themselves. Simple nodes host local controllers that control flows coming into a node from the input links. Complex nodes make the Aurora network structure hierarchical. Monitors are special objects whose purpose is to monitor the state of specified links and nodes and feed the observed data to complex controllers, whose purpose is the coordinated control of multiple nodes, or generate events. Events change network configuration or inputs. They can be generated by user and triggered at user-specified times, or by monitors and triggered based on the data observed by those monitors.

Aurora Road Network Modeler (RNM) is a set of tools for operational planning and management of travel corridors (road networks comprised of freeways and surrounding urban arterials), which is based on Aurora object-oriented framework. It consists of three graphical applications with intuitive user interface: Simulator runs macroscopic traffic simulations based on the CTM model, allowing the user to create simulation scenarios by means of events that change specific configuration parameters of the road network at specific times; Configurator is used to build road network configurations from scratch or edit the existing ones; and GIS Importer extracts road information from the GIS .shp and .dbf files and saves it in the XML format of Aurora configuration file.

Aurora RNM is open source and can be freely downloaded from the internet.

1 Introduction

This paper grew out of the authors’ participation in the Tools for Operational Planning (TOPL) project\(^1\), whose purpose is to provide tools for quantitative assessment of operational strategies designed to improve traffic conditions on congested freeways and surrounding arterials. The elements of such strategies are:

- demand management, which focuses on reducing excessive demand;

\(^1\)TOPL is supported by the California Department of Transportation through the California PATH program.
• incident management, which targets resources to alleviate accident hot spots;
• traveler information, which potentially reduces traveler buffer time; and
• traffic control, which implements aggressive ramp metering at locations where significant reductions in congestion are likely to occur.

The quick quantitative assessment provided by TOPL, can help rank a large set of operational strategies in terms of the benefits they will yield. Combined with a separate estimate of the cost of these strategies, TOPL can serve as the first step in selecting the most promising of them. This initial selection may be based on benefit/cost ratios or the magnitude of benefits.

The focus of TOPL is on operations in freeway corridors (road network comprised of freeway and surrounding arterials). A corridor is the smallest spatial unit that can be consistently analyzed as a self-contained system. Suppose, for example, that we wish to consider the impact of a promising new metering algorithm on some ramps on a given freeway. Evidently, this impact will depend on how other ramps on this freeway are metered. Furthermore, the impact of metering will affect (and be affected by) the signaling strategies on adjacent arterials. Thus, a good design of the metering algorithms and its proper assessment must take the entire freeway corridor into account. On the other hand, a major capacity expansion of a given freeway, such as the addition of a lane or the extension of the HOV facility, will significantly alter trip patterns. That is, the capacity expansion will have network-wide impact, which cannot be reliably assessed by studying the freeway alone. Thus, for traffic control, incident management, traveler information systems and demand management that TOPL seeks to assess, a corridor is the appropriate unit of analysis.

There exist two fundamentally different approaches to traffic modeling. The microscopic approach seeks to reproduce the behavior of an individual vehicle, as its driver responds to its environment by adjusting its speed and lane. Microscopic models typically involve variables such as vehicle position, speed and headway. The macroscopic approach ignores the dynamics of the individual vehicle and instead attempts to replicate the aggregate response of a large number of cars. These models represent traffic as a compressible fluid in terms of flow, density and speed. Traffic engineering has benefitted immensely from macroscopic models. They are widely used in the design of freeway facilities and they are present in nearly all model-based ramp metering designs. Because of its emphasis on quick and quantitative assessment, TOPL’s tools and procedures are based on macroscopic models that are easier to assemble, calibrate, and automate, as compared with their microscopic counterparts.

TOPL uses the macroscopic approach, as it is based on sound theory, is easy to implement in software, and the implementations are fast to run, allowing the user to simulate many different traffic situations in relatively short time. Our model of choice is the Cell Transmission Model (CTM) [20, 21] — a special case of Godunov discretization [27] of the Lighthill-Whitham-Richards (LWR) first order model [41], with triangular fundamental diagram. While simple, CTM adequately describes traffic flow on freeways, and the simulation results match well the measurement data provided by PeMS [1].

The first software package developed by TOPL was CTMSIM [38] — user friendly freeway traffic macro-simulator for MATLAB. Although it is a neat software package, useful for transportation researchers in their studies of freeway traffic, CTMSIM falls short of providing an appropriate toolset.

2Carrying out micro simulations for all plausible operational strategies is not practical. For example, a study uncovered more than 500 bottlenecks [39], the congestion caused by which could be mitigated by ramp metering. It is not possible to study all these opportunities by micro simulations.
for corridor management. It handles only one freeway and no arterial networks, cannot deal with HOV, has no notion of *event* that triggers certain configuration or input changes at given time or under given conditions allowing to program scenarios, and requires MATLAB which makes it unusable for the operations staff in organizations such as Caltrans.

The *Aurora*\(^3\) object-oriented framework \[^2\] overcomes the limitations of the CTMSIM. Its basic objects, nodes and links, allow the user to construct heterogeneous road networks. Various event classes make it possible to generate simulation scenarios. The monitor objects can keep track of the state at selected nodes and links, coordinate control actions at nodes, or generate events at nodes or links when the monitored states reach certain thresholds. Monitors and events enable the modeling of the impact of traveler information as well as incident management and the coordination of signal control on arterials with the ramp metering at freeway entries. The analysis module of Aurora, which is still under development, will address the issue of demand management: the goal is to solve the user equilibrium dynamic traffic assignment problem and evaluate various toll mechanisms.

The rest of the paper is organized as follows. Section 2 covers the theoretical background and reviews work done in control of conservation laws and in arterial modeling. Section 3 reviews the software tools used for macroscopic traffic modeling. Section 4 presents Aurora framework, its foundation principles and architecture. Section 5 describes Aurora Road Network Modeler (RNM), a suite of applications for road network creation, traffic simulation and analysis. Finally, Section 6 provides the roadmap for Aurora development.

## 2 Theoretical Background

### 2.1 Preliminaries

The Highway Capacity Manual 2000 \[^53\] provides the following definitions of the basic quantities. Symbols \(x\) and \(t\) represent position (measured in the direction of traffic flow) and time.

*Speed* \(v(x,t)\) is a rate of motion expressed as distance per unit of time. Depending on how it is measured, it is referred to as either *space mean speed* or *time mean speed*. Space mean speed is computed by dividing the length of a road by the average time it takes for vehicles to traverse it. Time mean speed is the average speed of vehicles observed passing a given point. The latter is easier to measure in the field, as it can be obtained directly from conventional sensing devices.

*Free flow speed* is the average speed of traffic measured under conditions of low volume, when vehicles can move freely at their desired speed.

*Flow* \(f(x,t)\) is the total number of vehicles that pass by the point \(x\) during a given time interval containing \(t\), divided by the length of the time interval. It is usually expressed as an hourly rate, and is easily measured with road sensors.

---

\(^3\)Aurora (short for polar aurora) — glow in the sky, seen often in a ring-shaped region around the magnetic poles (“auroral zone”) and occasionally further toward the equator. The name comes from an older one, “aurora borealis”, Latin for “northern dawn”, given because an aurora near the northern horizon (its usual location when seen in most of Europe) looks like the glow of the sky preceding sunrise. Also known as “northern lights”, although it occurs both north and south of the equator.
Density $\rho(x,t)$ is the number of vehicles occupying a length of road about point $x$ at time instant $t$. Its measurement is difficult because it requires the observation of a stretch of road. Instead, it is often approximated from measurements of flow and speed as

$$ \rho(x,t) = \frac{f(x,t)}{v(x,t)}. \tag{1} $$

Demand is the number of vehicles that desire to use a given facility during a specified time period.

Capacity is the maximal hourly rate at which vehicles reasonably can be expected to traverse a point or a uniform section of a lane or roadway during given time period under prevailing roadway, traffic and control conditions.

Bottleneck is defined as any road element where demand exceeds capacity. Freeway bottlenecks sometimes appear near heavy on-ramps, where a localized increase in demand is combined with a decrease in capacity due to lane changing.

One of the early attempts to correlate freeway speed, density and flow was by Greenshields in 1934 [29]. He used photographic images to estimate aggregate vehicular speeds and densities on a straight two-lane roadway, and found that they could be reasonably well approximated by a straight line. Using (1) he derived parabolic relationship between flow and density as shown in Figure 1. Function $f = \Phi(\rho)$ is known as the fundamental diagram. Later researchers have suggested alternative shapes that provide a better fit to the measured data (see Figure 2). All of them share the following characteristics:

1. $\Phi(0) = \Phi(\rho_J) = 0$, where $\rho_J$ is called jam density.
2. Continuous portions of $\Phi(\rho)$ are concave.
3. Critical density $\rho_c$ can be defined where the maximum flow is attained. Then, $\Phi'(\rho) > 0$ for $\rho < \rho_c$ and $\Phi'(\rho) \leq 0$ for $\rho > \rho_c$.

Critical density $\rho_c$ splits the fundamental diagram into two regimes: free flow ($\rho \leq \rho_c$) and congestion ($\rho > \rho_c$) (see Figure 1). Measurements on the free flow side are usually well represented by a straight line, whereas measurements in congestion tend to be more scattered.

### 2.2 First Order LWR Model

The simplest continuous macroscopic model is the scalar one proposed by Lighthill and Whitham [41], and by Richards [49]. Hence, this model is called LWR. Lighthill and Whitham in 1955 were the first to pose a macroscopic dynamic model of traffic using Greenshields’ hypothesis of a static flow/density relationship. LWR is based on conservation of cars and is described by a single nonlinear hyperbolic equation, also known as conservation law:

$$\rho_t + (\Phi(\rho)x = 0, \quad (2)$$

where function $\Phi$ is the flow. In this model, the average speed $v$ is a function that depends only on density. The relation $\Phi(\rho) = \rho v(\rho)$ is a fundamental diagram and is classically assumed to be concave (does not need to be parabola, see Figure 3). It is defined for $\rho \in [0, \rho_J]$, where $\rho_J$ is the jam density and corresponds to the density at which traffic stops. The density $\rho_c$ for which the flow reaches maximum (the road operates at capacity), is the critical density. Traffic speed $v \geq \Phi(\rho_c)/\rho_c$ is called free flow speed. When the density exceeds critical, the road becomes congested: the traffic speed falls below free flow, $v < \Phi(\rho_c)/\rho_c$.

![Figure 3: Fundamental diagram.](image)

To include on- and off-ramps into the LWR model (Figure 4), we rewrite (2) in integral form and account for the on-ramp flow $r_i$ and off-ramp flow $s_i$:

$$\frac{d}{dt} \int_{x_L}^{x_R} \rho(x,t)dx = \Phi(\rho(x_L,t)) - \Phi(\rho(x_R,t)) + r_i - s_i,$$

which can be once again rewritten as

$$\int_{x_L}^{x_R} \rho_t(x,t)dx = \int_{x_L}^{x_R} \left((\Phi(\rho(x,t)))_x + \delta(x - \tilde{x}_1) r_i(t) - \delta(x - \tilde{x}_1) s_i(t) \right) dx, \quad (3)$$
where \( \delta(x) \) is a Dirac delta function.

For multiple on-ramps \( (N_{on} \geq 1) \) and off-ramps \( (N_{off} \geq 1) \) equation (3) generalizes to

\[
\rho_t(x, t) + (\Phi(\rho(x, t)))_x = N_{on} \sum_{i=1}^{N_{on}} \delta(x - \hat{x}_i) r_i(t) - N_{off} \sum_{i=1}^{N_{off}} \delta(x - \check{x}_i) s_i(t). \tag{4}
\]

Clearly, in the absence of ramps, equation (4) becomes (2).

### 2.3 Godunov Scheme and Cell Transmission Model

As conservation laws can have discontinuous solutions, they cannot be integrated numerically by standard methods such as finite differences or finite elements that create instabilities and wrong shock speeds. Among the numerical schemes for scalar and systems of conservation laws the Godunov scheme \[27\] is widely used. It is first order, correctly predicts shock propagations, is free of oscillating behavior and has physical interpretation. In this approach the time is discretized into intervals \([k\Delta t, (k + 1)\Delta t]\). The computational domain is divided into cells\(^4\), and at time \(k\Delta t\), the solution \(\rho\) of (2) is approximated by a piecewise constant function \(\tilde{\rho}\) (see Figure 5) defined as

\[
\tilde{\rho}(x, k\Delta t) = \rho_k^i, \quad \forall i, \forall x \in [x_{i-1}, x_i]. \tag{5}
\]

The computation of the approximation \(\tilde{\rho}(\cdot, (k + 1)\Delta t)\) using the approximation \(\tilde{\rho}(\cdot, k\Delta t)\) requires two steps.

1. Compute exact solution of (2) given the initial condition

\[
\rho(x, k\Delta t) = \tilde{\rho}(x, k\Delta t) = \rho_k^i, \quad \forall i, \forall x \in [x_{i-1}, x_i]. \tag{5}
\]

2. Take the average of \(\rho(\cdot, (k + 1)\Delta t)\) over every cell \([x_{i-1}, x_i]\):

\[
\rho_{i+1}^k = \frac{1}{\Delta x_i} \int_{x_{i-1}}^{x_i} \rho(y, (k + 1)\Delta t)dy. \tag{6}
\]

These two steps can be simplified as follows:

\[
\rho_{i+1}^k = \rho_i^k + \frac{\Delta t}{\Delta x_i} (f_i^{k+1} - f_i^k) \tag{6}
\]

\(^4\)Here the cell numbers increase in the direction of traffic flow: cell \(i\) is upstream of cell \(i+1\).
with

\[ f_i^k = \frac{1}{\Delta t} \int_{k\Delta t}^{(k+1)\Delta t} \Phi(\rho(x_i, s)) ds \]  

(7)

being the average flow crossing \( x_i \) from cell \( i \) to cell \( i + 1 \) during the time interval \([k\Delta t, (k + 1)\Delta t]\).

Finally, since function \( \Phi \) is concave, expression (7) can be replaced by

\[ f_i^k = \begin{cases} 
\min_{\rho_i^k \leq \rho \leq \rho_i^{k+1}} \Phi(\rho), & \text{if } \rho_i^k \leq \rho_i^{k+1}, \\
\max_{\rho_i^{k+1} \leq \rho \leq \rho_i^k} \Phi(\rho), & \text{if } \rho_i^k \geq \rho_i^{k+1}.
\end{cases} \]  

(8)

In summary, the Godunov scheme leads to a piecewise approximation of the state (density) \( \rho \) at each time step, whose evolution can be computed for small time intervals if we know the solutions of initial value problems with Heaviside initial conditions

\[ \rho(x) = \begin{cases} 
\rho^-, & x < 0 \\
\rho^+, & x > 0.
\end{cases} \]  

(9)

Such initial value problem is an abstraction of the problem (2), (5), and is called a Riemann problem. It can be solved analytically for scalar conservation laws, and in the system case, when there is no closed form solution, an approximate solver such as the Roe average method can be used. The Godunov scheme, consisting in solving a succession of local Riemann problems, is an effective method for simulating macroscopic traffic models.

The cell transmission model (CTM) proposed in [20] is a special case of the Godunov difference scheme where the fundamental diagram has triangular form with maximal flow \( F \), slope \( v > 0 \) for the free flow speed and slope \( -w < 0 \) for the congestion wave speed (see figure 6). In this framework, the Godunov scheme becomes

\[ \rho_i(t + 1) = \rho_i(t) + \frac{\Delta t}{\Delta x_i} (f_{i-1}(t) - f_i(t)), \]

where \( \Delta t \) is the sampling period, \( \Delta x_i \) is the length of the \( i \)th cell, and \( f_i \), the flow from cell \( i \) to cell \( i + 1 \), is given by

\[ f_i(t) = \min\{v\rho_i(t), w(\rho_J - \rho_i(t)), F\}. \]

Consequently, cell \( i \) can operate in one of two modes: free flow mode if \( f_{i-1} = \min\{v\rho_{i-1}, F\} \), or congested mode if \( f_{i-1} = w(\rho_J - \rho_i) \).

Variations on the CTM theme can be found in literature. In [28] the asymmetric cell transmission model (ACTM) is presented. This work also describes a convex optimization problem whose solution is an optimal ramp metering strategy.
Figure 6: The fundamental diagram for CTM is characterized by the maximum flow $F$ and speeds $v, w$.

The linear hybrid system approach called *switching mode model* (SMM) based on CTM is introduced in [46].

### 2.4 Control of Conservation Laws

In the context of traffic flow applications, the goal of control is to improve the system efficiency by regulating the number of vehicles allowed to enter the freeway. Two fundamental performance measures are used to assess the system efficiency: the *total travel distance* (TTD) and the *total travel time* (TTT). TTD is defined as the sum of distances traveled by all vehicles of the system over a given time period. Equivalently, it is a product of the average trip length and the total number of vehicles, which can be computed as the integral of flow over time and space:

$$\text{TTD} = \int_X \int_T f(x,t) dt dx. \quad (10)$$

TTT is the sum of all trip times incurred by vehicles during a given time period, or the number of vehicles multiplied by the average trip time, which is computed as the integral of density over time and space:

$$\text{TTT} = \int_X \int_T \rho(x,t) dt dx. \quad (11)$$

The goal of control is regulating the number of vehicles entering the freeway either to maximize TTD or minimize TTT.

The optimal control theory of partial differential equations was initiated in the early 70’s by Lions [42]. The proposed approach consists in computing the necessary conditions of optimality in the form of the system equation, an adjoint equation of the same kind and a vanishing first variation condition. This analytic approach that was successfully applied to linear elliptic, parabolic and second order hyperbolic equations can be extended to nonlinear systems using gradient-based recursive algorithms. This method is widely used: in airfoil design [33, 35, 34]; fluid steering [13, 30, 17, 25]; gas steering [26]; control of water wave [50, 15]; air traffic control [11].

Very few attempts have been made to stabilize conservation laws using feedback control. In [22, 18], the authors propose a feedback controller for open channels but consider only smooth solutions and no shock waves. Krstic [37] proposed a feedback design for the Burgers equation with small viscosity parameter. Unfortunately, as the control law is inversely proportional to this parameter, the controller blows up in the nonviscous case. Successful control design using a finite dimensional discretization has been reported in [10] for parabolic partial differential equations. The main difficulty
in applying this method to hyperbolic conservation laws is that the classical finite difference scheme cannot be used for this class of equations due to possible presence of shock waves.

The problem of control of a system of conservation laws although addressed in the literature, remains difficult. A way around the problem, is to discretize the system first, then solve the control problem for the resulting dynamical system. An example of such approach is the multirate linear quadratic control with integral action (LQI) [52]. Jacquet [32] shows how Godunov discretization can be put in the form of a piecewise affine system if the fundamental diagram is approximated by a piecewise affine function, and suggests that constructive controller design methods proposed in [36, 19, 14] can be used to compute a set of static feedback gains for a switched controller.

2.5 Arterial Models

Ziliaskopoulos and Lee adapt CTM [21] for arterial modeling [55]. The cell length is generally much shorter for arterials than for freeways, hence the sampling period $\Delta t$ must be small enough to ensure

$$v \Delta t < l,$$

where $v$ is the free flow speed (Figure 6) and $l$ is the cell length.

Signalized intersections are modeled using diverging and merging cells\(^5\), and the signal phasing (red and green). The flow of the diverging cells is computed according to the CTM during the green phase and is set to zero during the red phase. In [54] CTM is used to formulate the system optimum dynamic traffic assignment problem\(^6\) as a linear programming (LP) problem.

In [43], Lo transforms CTM into a set of mixed-integer constraints and casts the dynamic signal-control problem\(^7\) to a mixed-integer linear program. As a dynamic platform, this formulation is flexible in dealing with dynamic timing plans and traffic patterns. It derives dynamic as well as fixed timing plans and addresses preexisting traffic conditions and time dependent demand patterns. Dynamic intersection signal control optimization (DISCO) that works with time-variant traffic patterns and derives signal timing plans is introduced in [44]. The authors compare DISCO with the platoon dispersion TRANSYT model [3] and conclude that timing plans generated by DISCO outperform those generated by TRANSYT by as much as 33% in delay reduction under a variety of demand patterns.

Feldman and Maher [24] investigate CTM applicability to the network of signalized arterials and compare it with the platoon dispersion TRANSYT model [3]. Modeling the arterial with a pair of traffic signals with both CTM and platoon dispersion model, the authors conclude that CTM yields similar or better results than the platoon based model does.

Amasri and Friedrich [9] also apply CTM to urban arterials and compare it with queueing models. They use genetic algorithm (GA) to find optimal signal timing plan having CTM as an underlying traffic flow model.

Alecsandru in [8] suggests modifications to CTM that include some microscopic features such as disaggregating the traffic flow by lanes and explicitly modeling the effects of individual lane-changing

---

\(^5\)Diverging are the cells with one predecessor and two or more successors. Merging are the cells with two or more predecessors and one successor.

\(^6\)See Section 6 for definition of the system optimum dynamic traffic assignment problem.

\(^7\)The problem of red/green signal phase assignment so as to minimize total travel time, number of slowed down vehicles (vehicles with speed below free flow speed), or maximize total out-flow of the system.
maneuvers; replacing some of the original parameters in the analyzed network with stochastic variables to capture the effect of the random driving behavior; and changes to the model equations that allow to keep track of different vehicle types. He also compares this modified CTM with CORSIM [31] microsimulation, and shows that the simulation outputs (traffic density and total network travel time) of these two models match well.

Nie [47] presents a polymorphic dynamic network loading (PDNL) framework for modeling road networks and solving the dynamic network loading problems. PDNL employs notions of links and nodes allowing different macroscopic traffic flow models to run on links while treating nodes as points of merge, diverge or general intersections, signalized or not.

Skabardonis and Geroliminis [51] propose an analytical model for travel time estimation on arterials. Their model is based on CTM, describes the spatial and temporal queuing at traffic signals and explicitly considers the signal coordination in estimating traffic arrivals at intersections. It estimates the travel time over an arterial link as the sum of free flow time and the delay at traffic signal.

In these works authors do not discuss computational complexity of the proposed models. The question how the size of a road network affects the efficiency of the proposed algorithms remains open.

3 Software Tools

3.1 METACOR

METACOR [23] is the first macroscopic simulation tool for corridor traffic, i.e., when freeways and arterials are modeled together in one network. It emerged as a fusion of METANET [45] (for freeways) and SSMT [40], a macroscopic model for urban networks. METACOR uses a discrete version of the Payne-Whitham second order model. It also includes control and dynamic traffic assignment modules to simulate ramp metering strategies and route information/guidance via changeable message signs. Currently, METACOR is being developed independently by the Technical University of Crete (TUC) and the Institut National de Recherche sur les Transports et leur Sécurité (INRETS). Individual copies of the former version can be obtained from M. Papageorgiou. He warns, though, that the user interface is rather primitive and there is no documentation in English. The latter version, known as PX-Metacor [4] is supposed to be commercially available. The advertisement says that additionally to Payne-Whitham, it supports LWR and ARZ models as well as micro- and hybrid (macro + micro) simulation, and has quality graphical user interface. We were unable to get hold of PX-Metacor. It is still under development and has not been released at the time of this writing.

3.2 TransModeler

TransModeler [5] is a heavy weight commercial software for traffic modeling. It provides three modes of traffic simulation for the user to choose — microscopic, in which traffic dynamics is captured by a car-following model; mesoscopic, in which traffic dynamics is determined from the speed-density relationship; and macroscopic, in which traffic is described by a static cost-flow relationship.

---

8 Problems that aim at obtaining the link cumulative arrival/departure curves (hence time-dependent link/path travel times) corresponding to a given set of temporal path flow rates on a congested network and over a fixed time period.
Since the mesoscopic model of TransModeler is the closest to Aurora RNM implementation of the CTM, we compared the two in the way they capture congestion using a simple simulation experiment. A single route consists of 6 sequential links with capacity 6000 vehicles per hour (vph), each 1 mile long. The upstream demand is constant, 4800 vph. The simulation time horizon is 1.5 hours. Between times 0.5 and 0.75 (hours) the capacity of the fifth link is reduced to 1500 vph, simulating an incident in that link. Flow, density and speed contour plots produced by Aurora RNM simulation are shown in Figure 7a, and by TransModeler — in Figure 7b. While Aurora RNM contours reflect the congestion propagation upstream of the fifth link, TransModeler contours indicate that it failed to capture the spill back of congestion. The underlying mesoscopic model of TransModeler does not employ the flow conservation condition, which is the key point of the CTM, and hence, fails to capture the variation and interaction of flow and density over time and space, although it does estimate the speed change.

3.3 TRANSYT

TRANSYT [3] is widely used commercial software for determining and studying optimum fixed time and coordinated traffic signal timings in a network of roads, for which the average traffic flows are known. Its traffic network model computes a performance index, which is weighted sum of all vehicle delays and stops, and the optimization routine systematically alters signal offsets and/or green time allocations to search for the timings, which reduce this performance index to a minimum value. For modeling the traffic behavior within the network, TRANSYT offers two traffic models for the user to choose — the platoon dispersion model and the CTM.

TRANSYT is a suitable tool for modeling relatively small arterial networks for small time horizons.\(^9\)

\(^9\)Currently, maximum allowed number of time steps in TRANSYT is 500. Considering that arterial links can be very short, the recommended duration of a time step is one second.
and optimizing the signal timing plans, but it cannot be used for the simulation of large travel corridors.

4 Architecture of Aurora

The five foundation principles we tried to follow while developing the Aurora framework, are listed below.

1. **Multi-purpose** — basic structures and algorithms must be generic and not road traffic specific, making the framework reusable for other applications, such as irrigation canals, oil or gas pipelines, etc. Application specific classes inherit from these basic structures. This affects basic data structure definitions and general purpose algorithm development.

2. **Usability** — Aurora tools must be easy to handle: creating configuration files, running simulations, and calling analysis routines, must be intuitively clear. It is better to have several different lightweight applications for different tasks rather than one heavyweight application for multiple purposes.

3. **Interactivity** — simulation and analysis applications must provide clear and simple GUI with good data visualization. This and previous items affect the user interface and visualization: what data should be displayed and in what way.

4. **Scenario oriented** — user should be able to write scenarios: lists of events that change configuration or inputs, with times or conditions of their occurrence, and feed them to the simulator. This affects the way events are described and handled.

5. **Scalability** — user should be able to seamlessly add new subnetworks to already existing network configurations, or connect two or more networks with each other. This affects basic data structure definitions and the way configuration files are organized.


Aurora Road Network Modeler (Aurora RNM), the suite of applications for operational planning and management of travel corridors and analysis of their performance, is built on top of Aurora framework.

4.1 Basic Objects

The basic building block of the Aurora system is a network element with a unique integer ID. A network element can be a link representing a stretch of road (water canal, pipeline, etc.), simple node — point where links merge and/or diverge, complex node — network built out of network elements, or monitor — an object that monitors the state of specified links and nodes in a network.

A link has direction and length. It must have either of the two nodes, begin node or end node, or both of them, attached to it. Links with no begin nodes are source links. Source links provide input to a system. In the case of a road network (Aurora RNM), associated with source links are demand
<table>
<thead>
<tr>
<th>Link type</th>
<th>Admissible begin nodes</th>
<th>Admissible end nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>freeway</td>
<td>freeway</td>
<td>freeway</td>
</tr>
<tr>
<td>highway</td>
<td>highway</td>
<td>highway</td>
</tr>
<tr>
<td>HOV</td>
<td>freeway, highway</td>
<td>freeway, highway</td>
</tr>
<tr>
<td>interconnect</td>
<td>freeway, highway</td>
<td>freeway, highway</td>
</tr>
<tr>
<td>on-ramp</td>
<td>signal and stop junctions</td>
<td>freeway, highway</td>
</tr>
<tr>
<td>off-ramp</td>
<td>freeway, highway</td>
<td>signal and stop junctions</td>
</tr>
<tr>
<td>street</td>
<td>signal and stop junctions</td>
<td>signal and stop junctions</td>
</tr>
<tr>
<td>dummy</td>
<td>any</td>
<td>any</td>
</tr>
</tbody>
</table>

Table 1: Aurora RNM link types with corresponding admissible begin and end nodes.

values and queues. Links with no end nodes are destination links. In Aurora RNM, we assume that anything downstream of a destination link is in free flow.

Table 1 summarizes link types implemented in Aurora RNM together with begin and end nodes that each link type allows. Each of these link objects has associated with it a fundamental diagram and dynamics. Dynamics is an interface, i.e., any macroscopic traffic model can be used to compute the link state, namely, traffic density. Currently, only the CTM model is implemented. Density is implemented not as a simple numeric type, but as a generic object, allowing us to modify or extend the traffic model so it could deal with density as a vector of values (say, when traffic flows carry additional origin-destination information, or we want to distinguish vehicles by types, e.g. SOV, HOV, trucks), or more complex data structure. For each link, Aurora computes travel time, VHT\(^{10}\), VMT\(^{11}\), delay\(^{12}\) and productivity loss\(^{13}\).

<table>
<thead>
<tr>
<th>Node type</th>
<th>Admissible input links</th>
<th>Admissible output links</th>
</tr>
</thead>
<tbody>
<tr>
<td>freeway</td>
<td>freeway, HOV,</td>
<td>freeway, HOV,</td>
</tr>
<tr>
<td></td>
<td>interconnect, off-ramp</td>
<td>on-ramp, interconnect</td>
</tr>
<tr>
<td>highway</td>
<td>highway, HOV,</td>
<td>highway, HOV,</td>
</tr>
<tr>
<td></td>
<td>interconnect, off-ramp</td>
<td>on-ramp, interconnect</td>
</tr>
<tr>
<td>signal junction</td>
<td>street, off-ramp</td>
<td>street, on-ramp</td>
</tr>
<tr>
<td>stop junction</td>
<td>street, off-ramp</td>
<td>street, on-ramp</td>
</tr>
</tbody>
</table>

Table 2: Aurora RNM node types with corresponding admissible input and output links.

A simple node\(^{14}\) must have one or more input and one or more output links. Aurora RNM nodes are listed in Table 2 together with types of input and output links they admit. Local controllers (such as ALINEA [48]), if any, reside on nodes and are assigned to given input links, potentially restricting flows coming from these links. When there are multiple output links, nodes also carry information about what portions of which input flows must be directed to which output. Currently, for \(m\) inputs and \(n\) outputs in the node, it is implemented as an \(m \times n\) split ratio matrix, where elements are nonnegative and sum up to 1 in each row.

---

\(^{10}\)Vehicle Hours Traveled — for a given unit of time and a given link, the amount of time spent by all of the vehicles in the link.

\(^{11}\)Vehicle Miles Traveled — for a given unit of time and a given link, the sum of the miles of the link driven by each vehicle.

\(^{12}\)Difference between the actual VHT and the VHT that would be incurred if vehicles traveled at free flow speed. The value is positive when the road is congested, otherwise it is zero. Delay is measured in vehicle-hours.

\(^{13}\)Number of lane-mile-hours on the freeway lost due to reduced flow, while operating under congested instead of free-flow conditions. The value is positive only in congestion, otherwise it is zero.

\(^{14}\)We refer to it as node from now on, while referring to a complex node as a network.
Remark. Currently, we do not distinguish between freeways and highways. Highway objects are present in Aurora RNM following the road classification provided by HCM [53] and are reserved for future use.

While links and nodes physically form a network, a monitor is a special object whose purpose is to monitor the state of specified links and nodes, and feed the observed data to system wide controllers (such as SWARM [16]). Such monitors are called control monitors. The monitors used to generate certain events (such as split matrix change — to simulate traveler information affecting traffic flow directions) based on observed conditions (more about events in 4.2), are called emph monitors. There are also zipper monitors, which are used to connect two networks into a larger, more complex network by redirecting outputs of one network into the inputs of the other.

All described network elements — links, nodes and monitors — are always part of a complex node, a network. There is at least one network in any Aurora system — the top level complex node, to which all other links, nodes and monitors belong. Network objects are nodes, hence, networks can contain networks just as they contain simple nodes. It makes the Aurora structure hierarchical, allowing to create configurations out of building blocks that are more complex than links and simple nodes, which is faster and more convenient, and opens a door to parallel computation when simulation steps for different subnetworks can be computed concurrently by different processors. Another benefit of using a hierarchical structure is that different subnetworks may have different sampling periods, that is, simulation steps of different duration. It can save time if, for example, network consists of roads with long enough links that do not require a small sampling period, and roads with rather short links that do. Separating them into subnetworks with different sampling periods reduces computation time.

Remark. Sampling periods of subnetworks cannot be greater than sampling period of top level network.

There are two other basic objects. Object path describes route from a node to node as a sequence of adjacent links. For each path, Aurora RNM computes travel time, VHT, VMT, delay and productivity loss based on corresponding data from links forming the path. Object OD describes a pair of origin and destination nodes together with list of paths connecting the two. A complex node may contain a list of origin-destination pairs. For consistency, it is required that every link in every path of every origin-destination pair belongs to the same complex node as ODs in the list\textsuperscript{15}.

4.2 Events

Aurora RNM-specific events, summarized in Table 3, are derived from the generic Aurora event object and are handled by the Aurora event manager. Events that change fundamental diagrams can be used to simulate traffic incidents by reducing capacity. Changing demand coefficients and split ratio matrices help imitate special events, road closures, or effects of displaying traveler information. Controller and queue size changes may be part of complex ramp control strategies.

The event object carries the following information: activation time (in terms of simulation hours), ID of a network element where it must occur, new parameter values for this network element, and an activate() method that changes those parameter values at a network element while storing the

\textsuperscript{15}It may happen that both, origin and destination nodes, belong to the same subnetwork, while some links at certain paths connecting them are part of a different subnetwork. This is not a problem because top level network contains all the links present.
<table>
<thead>
<tr>
<th>Event: change in</th>
<th>Where occurs</th>
</tr>
</thead>
<tbody>
<tr>
<td>controller</td>
<td>nodes</td>
</tr>
<tr>
<td>split ratio matrix</td>
<td>nodes</td>
</tr>
<tr>
<td>demand coefficient</td>
<td>source links</td>
</tr>
<tr>
<td>queue size</td>
<td>source links</td>
</tr>
<tr>
<td>fundamental diagram</td>
<td>links</td>
</tr>
</tbody>
</table>

Table 3: Aurora RNM events.

old values coming from the network element in their place. An event list is an optional part of the configuration file, but the user can generate new events before or during the simulation run as well as disable those already in the list.

When the simulator reads a configuration file, it places all events listed there into a queue sorted by event activation time (this queue may be empty, if no events are specified). User generated events are added to this queue, their location in the queue being determined by their activation time also. Events are triggered by the event manager. Before each simulation step, it selects those events in the queue that are due (those whose activation time is smaller than the next simulation time step), activates them by invoking their activate() methods. At this point, each activated event records its actual activation time. Then, the event manager moves the activated events from the event queue to the end of the event history list in the order they occurred. Thus, events never get deleted. Later, when the user resets the simulation, events are rolled back, or activated in reverse sequence with reverse action, returning to network element parameters their original values. Such maintenance of event queue and history list potentially allows us to “rewind” simulation to any given point of its execution.

4.3 Computational Model

An object representing a network element contains the dataUpdate() method. It performs simulation step computations specific to the particular type of network element. The Aurora RNM recursive algorithm of dataUpdate() in a network is described next.

1. Check if at this time step any action is needed:

\[(k - k_0)\Delta t_0 < \Delta t,\]

where \(\Delta t\) is sampling period for this network, \(\Delta t_0\) is the sampling period for the top level network, \(k\) is the current time step, \(k_0\) is the time step at which the last action was performed. If \(k > 1\) and inequality (12) holds, then return without doing anything. Else, proceed to step 2.

2. For every monitor in the monitor list, call dataUpdate(). If present, each monitor has its own specific task—it may assign controller parameters, or generate events to be activated before the next simulation step or later, at prescribed time.

\(^{16}\)The actual activation time may be different from the user-specified activation time in cases when the user generates an event in the middle of a simulation making its activation time smaller than current simulation time.
3. For every node in the node list, call dataUpdate(). If the node is complex, start the algorithm from step 1 with respect to this node. Else (the node is simple), compute input and output flows based on demand from upstream and available capacity of downstream links. This can be done in many ways.

Daganzo in [21] introduces the concept of priorities for multiple input flows and the FIFO\textsuperscript{17} rule for multiple output flows.

In the Aurora RNM priorities are the fractions of the input flows accepted by the node, in case the upstream demand exceeds the downstream capacity (if the upstream demand is below the downstream capacity, priorities do not matter since all the vehicles from the upstream links can be accommodated by the downstream links). Different priority choices result in different flow values for the next simulation step. In the current Aurora RNM implementation we assume that the input priorities are proportionate to the input demands.

The FIFO rule means that if one of the output links cannot accommodate its allocation of flow, the total output flow is restricted\textsuperscript{18}. In the Aurora RNM the FIFO rule implies that the input-output flow relations defined by the split ratio matrix must be preserved.

To summarize, we compute the input and output flows based on the input demands, satisfying the downstream capacity restrictions by assuming the input priorities to be proportionate to the demands, while preserving the input-output flow relations defined by the split ratio matrix.

Given \( m > 0 \) input and \( n > 0 \) output links, computation proceeds as follows:

(a) Compute input demands

\[ \hat{d}_i(k) = \min(v_i, \rho_i(k_0), C(\rho_i(k_0)), F_i), \quad i = 1..m, \]  

where \( v_i \) is free flow speed, \( \rho_i(k_0) \) is the density at the input link \( i \); \( C(\rho_i(k_0)) \) denotes flow value suggested by a controller, if a controller is assigned to the input link \( i \); and \( F_i \) is the capacity of the input link \( i \).

(b) From the \( m \times n \) split ratio matrix \( \mathcal{B} \), and the \( m \)-dimensional demand vector \( \tilde{d}(k) \) we get the input-output flow relations defined by the split ratio matrix \( D(k) \),

\[ D_{ij}(k) = \mathcal{B}_{ij} \hat{d}_i(k), \quad i = 1..m, \quad j = 1..n, \]  

and output demands

\[ d_j(k) = \sum_{i=1}^{m} D_{ij}(k), \quad j = 1..n. \]  

(c) Compute available output capacities

\[ c_j(k) = \min(w_j(\bar{\rho}_j - \rho_j(k_0)), F_j), \quad j = 1..n, \]  

where \( w_j \) is congestion wave speed, \( \bar{\rho}_j \) is the jam density, and \( F_j \) is the capacity of the output link \( j \).

(d) Compute input-output demand matrix adjusted by the output link capacity restrictions, assuming the input priorities to be proportionate to the demands,

\[ \hat{D}_{ij}(k) = \frac{\min(d_i(k), c_j(k))}{d_j(k)} D_{ij}(k), \quad i = 1..m, \quad j = 1..n, \]  

\textsuperscript{17}First in, first out.

\textsuperscript{18}Vehicles unable to exit from the upstream link prevent all those behind, regardless of their destination, to continue.
and adjusted input demands
\[
\hat{d}_i(k) = \sum_{j=1}^{n} \hat{D}_{ij}(k), \quad i = 1..m. \tag{18}
\]
This step ensures that the adjusted input demand does not exceed the downstream capacity. More precisely,
\[
\sum_{i=1}^{m} \hat{D}_{ij}(k) \leq c_j(k),
\]
with equality being achieved if and only if \(d_j(k) \geq c_j(k)\).

**Remark.** Expression (17) makes sense only if \(d_j(k) \neq 0\). So, in case \(d_j(k) = 0\), we set \(\hat{D}_{ij}(k) = 0\).

(e) Compute input flows
\[
\tilde{f}_i = \hat{d}_i \min_j \left\{ \frac{\hat{D}_{ij}}{d_i B_{ij}} \right\}, \quad i = 1..m, \quad j = 1..n. \tag{19}
\]
In case \(\hat{d}_i = 0\) or \(B_{ij} = 0\) for all \(j = 1..n\), we set \(\tilde{f}_i = 0\).

(f) Compute output flows
\[
f_j(k) = \sum_{i=1}^{m} B_{ij} \tilde{f}_i, \quad j = 1..n. \tag{20}
\]
Steps (e) and (f) implement the FIFO rule: input and output flow values are assigned so as to maintain input-output relationship defined by matrix \(B\).

4. For every link in the link list, call `dataUpdate()`.

(a) Compute density and speed using model specific equations. For CTM, these are
\[
\rho(k) = \rho(k_0) + \frac{\Delta t}{\Delta x} (f_u(k) - f_d(k)), \tag{21}
\]
where \(\Delta x\) is the link length, \(f_u\) is the upstream flow (flow entering the link), \(f_d\) is the downstream flow (flow exiting the link), and \(V\) is the speed. If the link is a source link, \(f_u(k)\) equals current demand, otherwise \(f_u(k)\) is computed in the begin node of the link as one of its output flows. If the link is a destination link, \(f_d(k) = \min (v \rho(k_0), F)\),

where \(v\) is the free flow speed, and \(F\) is the capacity; otherwise \(f_d(k)\) is computed in the end node of the link as one of its input flows.

(b) Compute traffic speed
\[
V(k) = f_d(k)/\rho(k). \tag{22}
\]

(c) Compute travel time
\[
TT(k) = \Delta x/V(k). \tag{23}
\]

(d) Compute VHT
\[
VHT(k) = \rho(k)\Delta x\Delta t. \tag{24}
\]

(e) Compute VMT
\[
VMT(k) = V(k)\rho(k)\Delta x\Delta t. \tag{25}
\]
Compute delay

\[
Delay(k) = \begin{cases} 
0, & \text{if } \rho(k) \leq \rho_c \\
VHT(k) - VMT(k)/v, & \text{if } \rho(k) > \rho_c 
\end{cases}
\] (26)

where \(\rho_c\) denotes critical density.

\[
PL(k) = \begin{cases} 
0, & \text{if } \rho(k) \leq \rho_c \\
\left(1 - \frac{L_d(k)}{v}\right) \Delta x \Delta t, & \text{if } \rho(k) > \rho_c 
\end{cases}
\] (27)

5. Set \(k_0 = k\) and return.

It is required that the sampling period \(\Delta t\) associated with a network satisfies the condition

\[
\Delta t \leq \min \left(\frac{\Delta x}{v}\right),
\] (28)

where minimum is taken over all links in the network.

4.4 Configuration

From TOPL we learned that once the process of a freeway corridor study is established, the most tedious and time consuming task is putting together a configuration file with road network description. Being the least rewarding, this task requires attention to details and patience. Therefore, efficient configuration management was made one of the priorities in the development of Aurora.

General configuration file contains

- information about the network, which includes
  - list of nodes, describing their types, positions, split ratio matrices and local controllers;
  - list of links, describing their types, lengths, widths, fundamental diagrams and nodes they connect or are attached to;
  - list of monitors (if any are present), that refer to complex controllers or events;
  - list of origin-destination pairs, each with a list of feasible paths;
- demand profile for source links;
- event scenario — list of events describing what occurs, where and when.

Putting together the network description is the most difficult task, because there is no single source from which these data could be extracted. Eventually, PeMS intends to provide this information for arterials as well as for freeways. To date, however, PeMS only deals with freeways and it knows only about ramps and lanes where detectors are installed. Thus, we have to work with GIS data from regional planning agencies such as MTC (Bay Area) and SANDAG (San Diego), configuration files for different simulators used by other research groups, and ultimately consult Google maps. The procedure of network layout extraction is not well defined yet and requires a “human touch”, i.e., some manual checks and adjustments.
Calibration, i.e., computation of fundamental diagrams and split ratio matrices, and demand generation, with the lack of sufficient measurement data (especially, for arterials), is the second great challenge.

All this put together makes us realize that complete configuration files have significant value, so that establishing a repository of configurations makes sense.

Aurora configuration files use XML format whose syntax, data sets and validation rules as well as the schema, are described in the Aurora RNM User Guide that can be obtained at [2]. Types of links, nodes, monitors, controllers, events and dynamics are defined in the class attribute, which specifies what classes Aurora must instantiate upon reading the configuration. Configuration is modular. That is, origin-destination lists, demand profiles and event scenarios, are separate blocks that can be optionally added to a configuration file or stored on their own. This, plus the hierarchical structure of Aurora in which a network is just another complex node, make the manipulation of configuration building blocks relatively easy and efficient.

Another benefit of XML configuration is that it can be read by anyone (the technology is known and proven) and translated into other formats. That makes it a good candidate for an interchange format for road network descriptions.

5  Aurora Road Network Modeler

Aurora RNM consists of three applications: Simulator, Configurator and GIS Importer. In the typical workflow, first, a road network is extracted from GIS files and saved in the XML format understood by Aurora. This task is performed using GIS Importer. Then, the configuration is edited in the Configurator. Editing the configuration includes assigning fundamental diagrams and demand profiles to links, split ratios and controllers to nodes. Finally, the main application of the suite, the Simulator, is used to run simulations with given configuration files. In the Simulator the user can create simulation scenarios by means of events that change user-specified configuration parameters of the road network at user-specified times.

5.1  Simulator

Simulator is a graphical application with interactive user interface that runs traffic simulations on road networks. Figure 8 presents the face of Aurora Simulator. The application window is partitioned into 5 areas.

1. Status area displays the status of simulation: running, paused or stopped together with simulation step and time, and issues short instructions to the user.
2. Menu bar whose commands and options allow to load road network from file, start and stop simulation, turn control on and off, modify simulation settings, and save simulation.
3. Network tree displays network components: nodes, links, origin-destination pairs and paths in a hierarchical tree structure a la Windows Explorer. Special icons specify their types. Double clicking on a component brings up a subwindow in the main frame with details of that particular network element or path.
4. Main frame is used to host subwindows for selected network elements or paths.

- Network window has three tabs. Layout tab (Figure 9) displays the interactive network map. Placing the mouse cursor over a node or a link makes a tooltip with information about that particular network element to appear.
  1. File menu that allows the user to save simulation data plotted in the performance tab in a CSV file.
  2. Finger toggle button — when pressed, allows user to move nodes around and label nodes and links on the network map by clicking on them.
  3. Zoom-in button. Each time the user presses this button the scale of the network map increases by 10%.
  4. Zoom-out button. Each time the user presses this button the scale of the network map decreases by 10%.
  5. Color code mode checkbox — when checked, the links are colored using the speed values at each simulation step; otherwise, densities determine the link colors.

On the network map, nodes are depicted using icons corresponding to their types (freeway, highway, signal and stop junctions). White dots represent the ends of the source or destination links. Links change their colors as simulation runs. By default, they are colored using density values — the color changes from green to yellow to orange as density goes from zero to critical; and from to orange to red to black, as density goes from critical to jam. Alternatively, the user may choose to color the links using speed values — from green for maximum speed to yellow, to red, to black as speed drops to zero. Labels on selected network elements display the details together with computed simulation data for those particular network elements.

Performance tab (Figure 10a) shows the plot of total network delay, which is computed in vehicle-hours per display period. Plots of other aggregate values such as total VHT or total VMT can be added in the future.

Configuration tab (Figure 10b) displays configurable network parameters and provides facility for generating network events.

- Node window has two tabs.
  Simulation tab (Figure 11a) displays two plotting areas. The top one is used to plot the flows coming into the node from each input link and their sum — total input flow. The bottom one is used to plot the flows going out of the node to each of the output links and their sum — total output flow. The legend at the bottom of this tab explains which color on the plots corresponds to which flow. Total input flow always equals the total output flow, as nodes do not have any storage capacity.

Configuration tab (Figure 11b) is divided into the following sections.
  - Node description taken from a configuration file.
  - List of controllers corresponding to the incoming links.
  - Split Ratio Matrix displays which portions of incoming flows are directed to which out-links. The entries of this table can be edited by double-clicking on them. The user is responsible for the correctness of the values he/she enters. They must be in the range from 0 to 1, and it is generally required that the values in each row sum up to one.
  - Events section allows the user to generate the split ratio change or the controller change event at the node.

- Link window has three tabs.
  Simulation tab (Figure 12a) displays the following plotting areas.

\[19\text{Comma separated values file format that stores tabular data.}\]
Flow — plots the actual flow leaving exiting this link versus the link capacity.
Density — plots the actual traffic density in the link versus critical density. For source links, instead of density, the actual queue size versus queue limit are plotted.
Speed — plots the actual traffic speed in the link versus free flow speed.
Travel Time — plots the instantaneous travel time through the link versus minimal travel time through this link. For source links, instead of travel time, the demand versus link capacity are plotted.

Performance tab (Figure 12b) displays the following plotting areas.
- VMT — plots the actual VMT on the link in versus maximum achievable VMT if the link operated at capacity.
- VHT — plots the actual VHT on the link versus critical VHT had all the vehicles present in the link traveled with free flow speed.
- Delay — plots delay caused by congestion in the link or by the queue if it is a source link.
- Productivity Loss — plots productivity loss due to congestion at the link.
All these values are computed per display period, user adjustable simulation parameter specifying the frequency with which the simulation data display is updated. It should be larger or equal to the sampling period.
All the data displayed in the simulation and performance tabs can be saved by the user in a CSV file.

Configuration tab (Figure 12c) is divided into the following sections.
- General information contains the length, the number of lanes, and, in the case of a source link, demand coefficient\(^{20}\) and queue limit.
- Fundamental diagram shows the plot of a triangular fundamental diagram and displays the values of capacity, critical density and jam density as well as free flow and the congestion wave speeds.
- Events section allows the user to select among available link events and generate the selected event.

Path window has three tabs.
Layout tab (Figure 13a) displays the interactive map of a path that connects given origin with given destination. This tab has the same functionality as the Layout tab in the network window.

Performance tab (Figure 13b) displays the following plotting areas.
- Travel Time — plots the instantaneous travel time through the path versus minimal travel time through this path.
- VMT — plots the actual VMT on the path versus maximum achievable VMT if all the links of the path operated at capacity.
- VHT — plots the actual VHT on the path versus critical VHT had all the vehicles present in the path traveled with free flow speed.
- Delay — plots delay caused by congestion in the path and/or by the queue at the first link of the path.
- Productivity Loss — plots productivity loss due to congestion at the path.

Contour plots (Figure 13c) shows flow, density and speed contours. Here it is assumed that traffic moves from left to right. Horizontal axes show miles, and vertical axes show simulation time increasing from from the bottom up.

\(^{20}\)Coefficient by which the demand value is multiplied before being processed as simulation runs.
All the data displayed in the performance tab and in contour plots can be saved by the user in a CSV file.

5. Scenario frame is divided into two tabs. One lists the events of the current simulation scenario: their type, description, activation time. The user cannot delete an event from the list, only edit it, or disable it. The event editor window pops up when the user double clicks on an event. Example of an event editor is shown in Figure 14. One of the key features Aurora offers to the user is the ability to create simulation scenarios. This is done by means of events that change configuration parameters at given network elements at user-specified times. A scenario is a list of events generated by the user. The user can generate events at any point before or during the simulation. If the activation time specified by the user is smaller than current simulation time, then the event will be activated before the next simulation step. Events with activation time 0 fire before the first simulation step. Once generated, events show up in the event list and cannot be deleted, but they can be disabled.
The other tab is a console for the Simulator output.

5.2 Configurator

Configurator is a graphical application with interactive user interface whose purpose is to produce XML configuration files for the Simulator. It can be used to build road networks from scratch; edit existing road networks by deleting or reassigning network components or adding new network components; provision road parameters such as fundamental diagrams and split ratios; and input demand profiles. The application window is partitioned into four areas (Figure 15).

1. Menu bar whose commands allow the user to start new; open existing configuration files; filter links based on their type; validate the configuration to make sure that links are connected to nodes, nodes have both inputs and outputs, and that connected links and nodes have compatible types; edit simulation settings; and save the configuration in a file.

2. Network tree displays the network elements in the same way it does in the Simulator. The user can make multiple selections and open networks, nodes and links for editing.

3. Main frame is used to host editor subwindows for selected nodes and links.
   - Network editor opens individually for every network and has two tabs.
     Layout tab (Figure 16a) displays interactive network map. Placing the mouse cursor over a node or link makes a tooltip with information about that particular network element to appear. It also allows to add new or delete the existing network elements and to redirect the existing links. On the network map, nodes are depicted using icons corresponding to their types (freeway, highway, signal and stop junctions). White dots represent the ends of the source or destination links. Right-clicking on a node or link brings up a menu allowing the user to edit or delete this node or link. The following buttons can help the user in the network editing.
       1. Fix button fixes node positions as they are currently displayed. Node positions affect only how the nodes are displayed and nothing else.
       2. Filter toggle button — if pressed, the filter is applied to the displayable links: only links of selected types together with the adjacent nodes are displayed.
3. Finger toggle button — when pressed, allows user to select nodes and links and move selected nodes around.

4. Zoom-in button. Each time the user presses this button the scale of the network map increases by 10%.

5. Zoom-out button. Each time the user presses this button the scale of the network map decreases by 10%.

Configuration tab (Figure 16b) allows the user to modify network attributes — ID, name description, sampling period, whose upper bound is determined by (28); and to turn the control on or off.

- When the user selects multiple nodes and clicks enter, these nodes get grouped so that nodes with the same number of in- and out-links belong to one group. For every group of nodes an editor is opened. The node editor allows the user to set parameters for all nodes in the group at once. Only the parameters actually modified by the user will be provisioned to the nodes when the user clicks OK button. Those parameters untouched will remain unchanged on all the edited nodes. If a certain parameter has been modified by the user, an asterisk appears in front of its name. Any changes made by the user can be discarded by clicking Cancel button.

   The node editor has three tabs.

   - Nodes tab (Figure 17a) lists the currently edited nodes and allows to change their type.
   - General tab (Figure 17b) allows to modify the name and description. If a single node is being edited, the user can also change its ID.
   - In/ out tab (Figure 17c) has two parameters.
     - List of input controllers contains the incoming links and types of the mainline and queue controllers assigned to them. The user can change the mainline controller for a particular in-link. If a controller is assigned to a particular in-link, the user can edit its parameters. Queue controllers are changed as other controller parameters. No queue controllers can be assigned if no mainline controller is assigned.
     - Split ratio matrix displays which portions of incoming flows are directed to which out-links. The entries of this table can be edited. They must be in the range from 0 to 1, and it is generally required that the values in each row sum up to one.

- When the user selects multiple links and clicks enter, the single link editor subwindow opens in the main frame allowing the user to modify the parameters on all of these links at once. Only the parameters actually modified by the user will be provisioned to the links when the user clicks OK. Those parameters untouched will remain unchanged on all the edited links. If a certain parameter has been modified by the user, an asterisk appears in front of its name. Any changes made by the user can be discarded by clicking Cancel button.

   The link editor has four tabs.

   - Links tab (Figure 18a) lists the currently edited links and allows to change their types.
   - General tab (Figure 18b) allows to modify the length, the number of lanes, the initial density and the queue limit. The last parameter makes difference only for the source links and only in the case when queue control is turned on.
   - Fundamental diagram tab (Figure 19a) displays the fundamental diagram per lane, which the user can modify by adjusting capacity, critical density and jam density parameters. Instead of critical and jam densities, the user can adjust free flow speed and congestion wave speed if it is more convenient. The fundamental diagram parameters cannot be set arbitrarily. It is required that the fundamental diagram always remains a triangle. The editor would prevent the user from assigning negative values to any of these parameters.
or making jam density less than critical density. When the user would change capacity or critical density, free flow and congestion wave speeds would adjust automatically. Changing jam density triggers automatic adjustment of the congestion wave speed. And the other way around, if the user changes free flow speed, critical and jam densities adjust automatically. Change in the congestion wave speed results in automatic adjustment of the jam density. When provisioned to the link, the fundamental diagram (capacity, critical and jam densities) will be multiplied by the number of lanes.

Demand tab (Figure 19b) shows parameters that make sense only for the source links. These are: demand profile — an array of demand values; demand sampling interval, which specifies the frequency of switching between the values in the demand profile; and demand coefficient — the number by which the demand value is multiplied before being processed by the system, allowing the user to increase or decrease the demand from its profile values. Manipulating with the demand coefficient the user can generate scenarios such as what if the demand increases or decreases by 5%.

4. Error frame is divided into two tabs. One is used as a console for the Configurator output. The other one lists configuration errors after validation if there are any. Clicking on an error directs the user to the node or link where the error was reported.

5.3 GIS Importer

GIS Importer is a simple graphical application that reads road network information in a GIS file and convert it to XML format understood by Aurora. Figure 20 presents the face of GIS Importer. The application window has the following components:

1. Menu bar whose commands allow the user to open GIS files; filter links based on their type; simplify network edges; and save the road network as an aurora XML file.
2. Four buttons corresponding to the main commands.
3. Log window shows the helpful information as each command is executed.

The “Type Filter...” command lets the user export only road segments of certain types (e.g., freeway and major arterials) to Auroral xml files. To use this function, the database schema of the GIS file itself needs to have an attribute specifying the type (sometimes called “functional class”) of each road segment. In SANDAG GIS data, for example, such attribute is called “CCSTYLE” and in TeleAtlas GIS data for routing (Dynamap), the attribute is called “ACC”. The user needs to refer to the manual of the GIS file to find out the name of the road type/class attribute.

In GIS Importer, when “Type Filter...” command is executed, it opens up “Road Type Selection Window” (Figure 21). This window consists of the following components:

1. “Road Type Attribute” list (left) shows all attributes in the database schema of the GIS file. When the user select an attribute in this window, “Attributes to Use” list is populated.
2. “Attributes to Use” list (middle) shows all unique road type values appearing in the first 1,000 road segments in the GIS file. When the user selects the value(s) in this list and press “OK” button on the right, type filtering is performed.

The “Simplify Edges” command, as the name suggests, removes nodes that have only one incoming link and one outgoing link. This reduces redundancy in road network geometry but could result in loss of information.
6 Development Roadmap

Aurora Road Network Modeler is a project in development. Our goal is to make it an end user product consisting of three major modules: configuration, simulation and analysis.

6.1 Simulation Module

Simulation module is a centerpiece of Aurora RNM. Significant progress has been made in the Simulator development, but several items remain unfinished.

- Currently, we are working on a number of control algorithms for the ramps and arterials. These include system-wide ramp algorithms such as SWARM [16], signal control with adaptive rate and coordinated signals on arterials, and coordinated arterial signals and ramp metering. Until signals are implemented, we make trivial assumption that arterial traffic always moves with free flow speed (25 – 30 mph) and there are no delays at signal junctions.

- Currently, we use demand values at source links as inputs to the system, while split ratio matrices at MIMO\textsuperscript{21} nodes determine how traffic flow is divided between different links. Alternative form of input data can be used: origin-destination flow matrices. That is, instead of demand profile and split ratios at junctions, there are origin-destination (OD) matrices generated with given period (say, every 5 or 15 minutes) that specify how many vehicles started from given source to given destination during that period. Input data in the OD form is useful if we solve dynamic trip assignment (DTA) problem. Our goal is to make Aurora RNM capable of processing input in OD form. This would change the way the state of the link is represented. Currently the state of the link is defined by the vehicle density, it is a scalar value. We would like to define the state as a vector of densities marked by the OD pairs, needed to solve the DTA problem. For our current input (demand and split ratio profiles), the vector-state of a link would allow us to classify vehicles by their types (HOV, SOV, trucks, etc.).

- So far, “travel time” through a link or path refers to the instantaneous travel time as opposed to actual travel time. Instantaneous travel time is the travel time that would be experienced if the traffic speed in each link of the path were to stay constant assuming values at current time step. It can be computed every time step as simulation runs. Actual travel time can be only computed after the whole simulation data becomes available. Suppose, we start at a source link 0 at time step \( k_0 \) and this link has a queue \( q(k_0) \). The time step at which we arrive at link 1 is

\[
k_1 = k_0 + N_0,
\]

with

\[
N_0 = \arg\max_k \left\{ \sum_{k'=0}^{k-1} f_d(k_0 + k') \Delta t \leq q(k_0) \right\},
\]

where \( \Delta t \) is a sampling period, and \( f_d(k_0 + k') \) is the flow leaving link 0 at time step \( k_0 + k' \). If we arrive at link \( i \) (\( i > 0 \)) of our path at time step \( k_i \), the actual travel time through this link will be

\[
T_i(k_0) = N_i \Delta t,
\]

\textsuperscript{21}Multiple input, multiple output.
with

\[ N_i = \arg\max_k \left\{ \sum_{k' = 0}^{k - 1} V_i(k_i + k') \Delta t \leq \Delta x_i \right\}, \]  

(32)

where \( \Delta x_i \) is the length of \( i \)-th link, and \( V_i(k_i + k') \) is the average traffic speed on \( i \)-th link at time step \( k_i + k' \).

Given (29)-(32), arrival time at link \((i + 1)\) is \( k_{i+1} = k_i + N_i \). Then the total travel time over the path through links \( i = 0, 1, \cdots M \) is

\[ T(k_0) = \Delta t \sum_{i=0}^{M} N_i, \]  

(33)

with \( N_i \) determined from (30) and (32).

### 6.2 Configuration Module

The configuration module is a cornerstone of Aurora RNM, without which the task of creating the configuration files would be unmanageable. Eventually, PeMS will be collecting data from arterials as well as from freeways, with road configurations coming directly from GIS databases. For California, it will then become a unique source of road geometry together with density-flow data needed for fundamental diagram estimation, and demand profiles and split ratios — everything needed for Aurora configuration files, enabling the automatic creation of configuration files. At this point, however, it is impossible to make the construction of configuration files completely automatic. The main reason is the large variety of configuration sources, each with data in its own format: PeMS with its configuration and measurement data; regional planning agencies with their GIS databases; census and demographic data that determines origin-destination travel patterns; and configuration files for other (microsimulation) packages used by different research groups.

Despite the necessary manual intervention in the process of configuration building, we try to automate it where possible. The focus is on GIS databases from regional planning agencies (SANDAG, MTC) as the most consistent and comprehensive sources of information about road geometry. GIS Importer extracts road networks from the GIS files and saves them in XML format suitable for Aurora. Once the road network is in place, the system must be calibrated. Fundamental diagrams for freeway links can be estimated from PeMS data. For arterials, some best guess default values have to be used. Split ratios are estimated from PeMS, census and survey data. Configurator is used to effectively set these values to given links and nodes. Finally, demand profiles should be generated using PeMS, census and survey data and added to the configuration file.

In the future development we plan to improve the structure of the configuration file so that demand profiles and event scenarios could be stored separately from the network geometry. This would allow to keep rarely updated and frequently updated configuration portions apart. The other objective is to simplify the process of building complex networks from the existing subnetworks just by referencing the corresponding configuration files. Lastly, configuration module must provide support for specifying the input in terms of OD matrices as opposed to demand and split ratio profiles. This would also require changes in the structure of configuration file and additional editing functionality in the Configurator.
6.3 Analysis Module

Analysis module, currently nonexistent, can be thought of as a collection of special purpose traffic applications relying on simulation data. Properly organized database, that stores the simulation results and provides easy access to them, is a key factor in the effective performance analysis of road networks. This module could include

- performance comparison between two or more simulations;
- shortest path calculation based on actual travel time;
- demand management;
- analysis of stochastic demands and capacities;
- bottleneck identification;
- fee computation for tolled lanes or roads; etc.

One of the goals in TOPL is to implement the DTA application. We have a model of corridor comprising freeways and arterials. The corridor is modeled as a dynamical system (CTM). Underlying the dynamical system is a road network consisting of nodes and links. A subset of node-pairs is identified as a set of OD pairs. Associated with each OD pair are two entities:

1. demand profile — function of time that gives for each $t$ the flow of vehicles that start at the origin at $t$ and wish to travel to the destination;
2. set of paths or routes through the network that start at the origin and end at the destination that a vehicle is likely to take.

A trip assignment is an assignment of all OD demand profiles to paths. In other words, for each OD pair and time $t$, the assignment specifies how many of the vehicles will travel over each path associated with the OD pair.

Two types of trip assignments are important.

- **User equilibrium (UE)** — if no individual vehicle can reduce its travel time given that everyone else follows the trip assignment (Wardrop’s first principle).
- **System optimal (SO)** — if it minimizes the total travel time summed over all demand profiles (Wardrop’s second principle).

In general, UE and SO are different, and

$$T_{UE} \geq T_{SO},$$

where $T_{UE}$ denotes total travel time under UE, and $T_{SO}$ denotes total travel time under SO.

Currently, in the literature we can find description of the standard trip assignment problem [12], in which the demand is stationary (does not depend on time) and there are no dynamics: the delay on a link is simply a function of flow on that link. For such problems, both, UE and SO trip assignments are computed. Trip assignment problems using dynamical system (CTM) as a model are presented in [54, 43], but they focus solely on SO, because computing UE is much more difficult.

We would like to be able to dynamically compute both, UE and SO trip assignments, obtaining “cost of anarchy” as $(T_{UE}/T_{SO})$. 

27
References


Figure 8: Aurora RNM Simulator — look and feel.
Figure 9: Network window: layout tab.

Figure 10: Network window: (a) performance tab; (b) configuration tab.
Figure 11: Node window: (a) simulation tab; (b) configuration tab.

Figure 12: Link window: (a) simulation tab; (b) performance tab; (c) configuration tab.
Figure 13: Path window: (a) layout tab; (b) performance tab; (c) contour plots.

Figure 14: Editor widow for the link event that changes fundamental diagram.
Figure 15: Aurora RNM Configurator — look and feel.
Figure 16: Network editor: (a) layout tab; (b) configuration tab.

Figure 17: Node editor: (a) nodes tab; (b) general tab; (c) in/ out tab.
Figure 18: Link editor: (a) links tab; (b) general tab.

Figure 19: Link editor: (a) fundamental diagram tab; (b) demand tab.
Figure 20: Aurora GIS Importer — look and feel.

Figure 21: Aurora GIS Importer Type Selection Window.