TrafMod Model Specifiction Report

Jack M. Keoshian, Jr.
Phd. Pre-Candidate

Department of Industrial & Operations Engineering
The University of Michigan
Ann Arbor, Michigan 48109-2117

Technical Report 96-8

July 1996

TrafMod Model Specification Report *

Jack M. Keoshian, Jr.
Phd. Pre-Candidate
Department of Industrial and Operations Engineering
The University of Michigan
Ann Arbor, Michigan 48109

July 18, 1996

Abstract

In this report we describe the models implemented in TrafMod, a simple discrete-event simulation of a deterministic traffic network under development at the University of Michigan. Although a general enough tool to be useful for many traffic simulation tasks, TrafMod is specifically designed to support dynamic route guidance research within the university's Intelligent Transportation Systems (ITS) community. This focus allows for simpler models and enhanced execution speed over more general purpose traffic simulators. The objective is to construct a simulator with sufficient realism to allow for algorithm development and testing. Evaluation of the resulting algorithm with respect to expected benefits can then be established using more realistic and computationally intensive simulators.

This report describes the models specified for the TrafMod simulation loop. The system event structure is discussed in detail. Only four events are required to completely describe the evolution of the traffic network over time: node arrival events, link entry events, new routing table events and incident events. In addition, computation

*This work was supported by the University of Michigan Research Center of Excellence in Intelligent Transportation Systems.

Contents

1 Introduction

TrafMod is a discrete-event simulation of a deterministic traffic network under development at the University of Michigan as part of ongoing research in Intelligent Transportation Systems (ITS). Although a general enough tool to be useful for many traffic simulation jobs, TrafMod is specifically designed to support dynamic route guidance research within the university's Traffic Modeling Group. By specifically targeting route guidance applications TrafMod can employ simpler, more focused, models than more general purpose traffic simulators. This in turn yields an implementation that can be smaller, execute more quickly and be easier to understand and maintain.

In the past, route guidance research at the University of Michigan has used the INTEGRATION-UM [6] traffic simulator. INTEGRATION-UM is a local modification of the popular traffic simulator INTEGRATION developed by Michael Van Aerde at Queen's University in Ontario, Canada [5]. By accepting the same input files and producing output in the same format as INTEGRATION-UM, TrafMod is "plug replaceable" for the older model in existing applications (e.g. SAVaNT [1] and ALLONS [3]).

1.1 Document Purpose

This report describes the models specified for the TrafMod simulation loop. It documents the final form of the models, as well as giving some justification and background for them. This report is not intended as a software design document. A detailed software design document will be issued in the future. Only the broadest details of the software design are included here to give a general context for the modeling issues.

The major substance of this document is contained in the sections on computing link travel time and node delay. These two issues are at the core of the TrafMod models. Significant discussion is also directed toward describing the system event structure and associated processing. A broad overview of the global structure of the TrafMod system is given and some of the key modeling assumptions are discussed.

increase in implementation complexity. The terms *discrete-event simulation* and *next-event time advance* are discussed in detail later in this report.

- Models in TrafMod are designed keeping their appropriateness to the route guidance task in mind. Functionality/complexity not required by the route guidance task is minimized. In particular, traffic movement along links is modeled in greater detail than through nodes.
- The software is developed using straight ANSI compliant C in the UNIX operating system using a minimum of system specific features. Portability, maintainability and available hardware and software upgrade paths are enhanced.
- The software is designed and implemented using a highly modular structured approach to enhance maintainability.

1.4 Key Modeling Assumptions

The following assumptions are made in developing the models implemented in TrafMod:

1. Only a single type of vehicle is modeled. That is, all vehicles are identical in every respect (e.g. maximum speed, etc ...), except for possible differences in guidance class. Note that the guidance class determines the route taken between fixed a origin and destination.
2. Vehicles are not allowed to pass leading vehicles on a link. Links are always considered to be first in first out (FIFO) in queuing discipline.
3. Vehicles are not allowed to pass other vehicles at intersections. Like links, nodes are considered to be FIFO in nature.
4. Interdependence of conflicting flow streams at intersections is ignored.
5. No explicit headway constraint is enforced on links or at nodes; i.e. multiple vehicles may arrive at a node or on a link at the same time.

- Origin and destination nodes
- Route guidance class

2.4 Signal

Under the assumption that signalization is second order in its effect on the route guidance problem, TrafMod adopts a simple model of traffic signals. Every node is assumed to have a traffic signal. Modeling of the effect of the signal on the progress of a vehicle is limited to delaying each vehicle by an expected delay which varies with traffic volume.

2.5 Intersection

Intersections are modeled as a collection of nodes and incident links. They have no explicitly defined maximum capacity. Turn pockets, protected and unprotected turning movements and other intersectional phenomena must be modeled with additional nodes and links.

2.6 Origins and Destinations

Sometimes identified as "sources" and "sinks" or collectively as "zone nodes", these are specified nodes where vehicles may begin or end their journey. No vehicle may enter or leave the network except at these designated nodes. Although the route a vehicle takes need not be fixed before the simulation run, the journey's start and end points always are.

3 Software Design Overview

This section describes the high level organization of the TrafMod traffic simulator. The intent is to give a context in which to consider the modeling issues, not describe the software design in great detail.

The TrafMod system is composed of three major logical components:

1. an initialization section
2. the simulation or event loop

- Schedule all network entry events on the event list.

The specifics of the algorithms implemented in the input processor will be detailed in a forthcoming document.

3.2 Simulation Loop/Event Processor

The models implemented in the TrafMod event processor (also referred to here as the simulation loop) are the primary focus of this memo. The state of the traffic network is moved forward in time by sequentially processing a series of "events" which modify the state of the network. The processing associated with each of these events is discussed later in this report.

The event loop is executed as long as there is at least one event scheduled on the event list and the current time is less than the user specified simulation end time. Processing in the event loop includes the following tasks:

- Get the next event from the event list.
- Do the processing associated with this event. One of five cases will apply:
  1. CASE OF "link entry event":
     - Compute smoothed flow onto link.
     - Compute link travel time.
     - Schedule node arrival event for this vehicle.
  2. CASE OF "node arrival event":
     - Compute smoothed node arrival rate.
     - Compute node delay time.
     - Output a record to the node arrival history file recording this node arrival event.
     - Schedule node exit event for this vehicle.
  3. CASE OF "node exit event":
     - Consult current routing table to determine link to take exiting the current node.
     - Schedule link entry event for this vehicle.

viewpoint. The "implementation" events define a smaller set of events which are significant with respect to the software implementation.

4.1 Definition of Terms

To be clear on our usage, the terms *state*, *event*, *discrete-event simulation* and *next-event time advance* are defined below. These definitions correspond to those given by Law and Kelton [2].

State: We define the state of the traffic network to be that collection of variables necessary to completely describe the network at any particular instant in time.

Event: An instantaneous occurrence that may change the state of the system.

Discrete-Event Simulation (DES): DES concerns the modeling of a system as it evolves over time by a representation in which the state variables change instantaneously at distinct points in time. These points in time are the ones at which an event occurs.

Next-Event Time Advance: With the next-event time advance approach, the simulation clock is initialized to zero and the times of occurrence of future events are determined. The simulation clock is then advanced to the time of occurrence of the most imminent (first) of these future events, at which point the state of the system is updated to account for the fact that the event has occurred. Our knowledge of the times of occurrence of future events is also updated. Then the simulation clock is advanced to the time of the (new) most imminent event, the state of the system is updated, etc .... Since all state changes occur only at event times, periods of inactivity are skipped over by jumping the simulation clock from event time to event time.

4.2 TrafMod "Model" Events

A collection of events have been identified for TrafMod which completely describe the evolution of the modeled traffic network. These events are referred

Link Incident Event: The link incident event is used to model some atypical non-recurring occurances on links which serve to slow traffic on the link. Traffic accidents and road construction which slow traffic could be modeled as link incidents. For each link incident event the start time, stop time, link and number of lanes blocked is specified by the user. The number of lanes blocked must be converted to a percent change in the link's practical capacity. The details of this parameter conversion are yet to be determined. Each incident event defined by the user is converted into two scheduled events within TrafMod; one event corresponding to the start of the incident and another corresponding to the end of the incident. Link incident events are processed as a special case of a general "incident" event in TrafMod.

Node Incident Event: The node incident event is used to model some atypical occurances at nodes which serve to slow traffic through the node. For each node incident event the start time, stop time, node approach (i.e. link) affected and percent change in service rate (μ) is specified by the user. Analogous to the situation for the processing of link incident events, each node incident event specified by the user is represented by two scheduled events within TrafMod. An event scheduled for the start time of the node incident modifies the service rate as specified by the user. A second event scheduled for the end time of the node incident returns the service rate to its original value.

Note that previously an event corresponding to a change in the rate at which departures were leaving an origin had been considered. Need for this event has been eliminated by requiring that all network entries for the entire TrafMod run be scheduled at the start of the run.

4.3 TrafMod "Implementation" Events

As described above, the state of the traffic network modeled by TrafMod changes only at discrete points in time, known as event times. The evolution of the network can be completely described by a set of system model events, which are defined to occur at these event times. Since some of these events are related to one another TrafMod actually implements a smaller set of events which we will refer to as the "implementation" events. Each of

4.5.1 Link Entry Event

The processing associated with a vehicle at node n bound for destination node d is as follows:

1. Consult the current routing table to determine the appropriate next link l for a vehicle at n bound for d.
2. Update value of smoothed flow onto link l via the exponential smoothing procedure discussed in detail later.
3. Assign the link travel time via the BPR function.
4. IF (This vehicle will arrive at the end of the link ahead of vehicles which entered the link earlier.)
   THEN (Modify the travel time such that this does not occur.)
5. Schedule a node arrival event for this vehicle at the node at the head of link l. The vehicle will arrive at this node at the current time plus the travel time computed above.
6. Save values of relevant parameters.

4.5.2 Node Arrival Event

Processing for an arrival at node n from link l is as follows:

1. Update value of smoothed arrival rate to node n from link l via the exponential smoothing procedure described in detail later.
2. Assign the node delay time via Webster's equation.
3. IF (This vehicle will exit the node ahead of vehicles which arrived at the node earlier.)
   THEN (modify the node delay time so this does not occur).
4. Write a record in the output simulation history file recording the vehicle ID, link being exited, link entry time, link travel time, delay at this node and a flag indicating whether this node is the vehicle's ultimate destination or not.
5. IF (Node n is not the destination node for this vehicle.)
   THEN
   (Schedule a link entry event for this vehicle. The

5 Major Algorithms

This section provides details of the major algorithms implemented in TrafMod. Computation of link travel time, node delay time and an exponential smoothing procedure are all discussed at some length.

5.1 Link Travel Time

The link travel time for a vehicle is computed as part of the link entry event. The "standard" BPR impedance function developed by the U.S. Bureau of Public Roads [4] is used to generate link travel times which are a function of dynamic demand on the link.

5.1.1 The BPR Function

Sheffi [4] gives the BPR function for delay on a link as follows

$$d_i = t_l^0 \left[1 + \alpha \left(\frac{F_{i-1}}{c_l'} \right)^{\beta} \right] \tag{1}$$

where,

$$\begin{aligned}
t_i &= \text{the time of the } i^{th} \text{ link entry event} \\
d_i = d_l(t_i) &= \text{estimated travel time on link } l \text{ at time } t_i \text{ (in seconds)} \\
F_{i-1} = F_l(t_{i-1}) &= \text{estimated flow on link } l \text{ at time } t_{i-1} \text{ (in vehicles/second)} \\
t_l^0 &= \text{free flow travel time on link } l \text{ (in seconds)} \\
c_l' &= \text{"practical capacity" of link } l \text{ (in vehicles/second)} \\
\alpha, \beta &= \text{model parameters}
\end{aligned}$$

The reference indicates that typical values for the "adjustable" model parameters are $\alpha = .15$ and $\beta = 4.0$. Note that these values imply that the *practical capacity* of a link is the flow at which the travel time is 15% higher than the free-flow travel time. The capacity of a road is sometimes defined to be the maximum possible flow through a link, but this is not the intended usage here. Indeed, there is no upper bound on flow in this case. The lack of a flow constraint significantly simplifies its use within optimization algorithms.

the time remaining until the previous vehicle to enter the link arrives at the end of the link; i.e.

$$d_i \leftarrow \max(d_i, t_{i-1} + d_{i-1} - t_i).$$

In this way, the entering vehicle is assured of not exiting the link ahead of vehicles which are already on the link, thus maintaining FIFO (First In First Out) order.

5.2 Node Delay

The node delay time for a vehicle is computed as part of the node arrival event. A formula due to Webster which assumes Poisson arrivals and constant maximum departure rate is used to generate the node delay times. An exponential smoothing procedure is used to estimate the current arrival rate required in Webster's formula. The delay time generated by Webster's formula is modified to assure that vehicles do not pass one another at the node.

5.2.1 Webster's Delay Function

Sheffi [4] gives a node delay function credited to Webster which assumes a M/D/1 queue at the intersection and which is quoted to "be within 5% of the actual delay" in widespread empirical studies. Webster's formula for the delay at a node n arrived at via link l at node arrival time t_j follows. Note that j counts the arrival events at the head of link l and is used to indicate the time varying nature of some of the parameters below. For the sake of readability all subscripts except j, which counts arrivals, have been suppressed in the equation below. The node delay according to Webster's formula is then,

$$\bar{w}_j = \frac{r^2}{2c(1-\rho_j)} + \frac{R^2}{2\lambda_j(1-R)} - .65\left(\frac{c}{{\lambda_j}^2}\right)^{1/3} {R_j}^{2+5g/c} \qquad (2)$$

where,

t_j = time of the j^{th} node arrival event associated with node at head of link l (in seconds)

The smoothed arrival rate at time t_j is then given by,

$$\lambda_j = \begin{cases} (1-\alpha^{t_j - t_{j-1}})\hat{\lambda}_j + \alpha^{t_j - t_{j-1}}\lambda_{j-1}, & j = 1,2,3,\ldots \\ 0, & j = 0 \end{cases}$$

where α is a smoothing factor, $0 \leq \alpha < 1$, and assuming $t_j \neq t_{j-1}$. Note that the initial arrival rate is taken to be zero.

In the case of simultaneous arrivals of two or more vehicles the update above cannot be used. In such cases, for all but the "first" of the simultaneous arrivals $t_j = t_{j-1}$, causing the calculation of $\hat{\lambda}_j$ to become unbounded. In practice, numerical instability becomes a concern whenever t_j is sufficiently close to t_{j-1}. For cases where $|t_j - t_{j-1}| < \epsilon$, where ϵ is a floating point tolerance, the following arrival rate update is used

$$\lambda_j = -\ln \alpha + \lambda_{j-1}.$$

5.2.3 Special Processing as Utilization Ratio Approaches Unity

As the utilization ratio, $\rho = \lambda/\mu$, approaches unity the first term of Webster's formula becomes unbounded. This problem is avoided by enforcing an upper limit on the allowed values for the computed arrival rate, λ. The desired upper limit on λ is specified via a user supplied parameter α_{lim} as follows,

$$\lambda \leftarrow \min(\lambda, \alpha_{lim} \cdot \mu)$$

where $0 < \alpha_{lim} < 1$.

5.2.4 Maintaining Vehicle Order at Node

Just as on links, it is desirable to maintain vehicle ordering at nodes. A vehicle arriving at a node must exit the node (enter the following link) after all vehicles which arrived earlier. This is accomplished in much the same manner as described above for maintaining order on links. Again, no explicit queue is maintained at the node. Instead, the delay time for a vehicle is set to the larger of the delay generated by Webster's formula and the delay remaining until the scheduled link entry event of the previous vehicle to arrive at this node on this link. That is,

$$\bar{w}_j \leftarrow \max(w_j, t_{j-1} + \bar{w}_{j-1} - t_j).$$

In this way, the arriving vehicle is assured of not exiting the node ahead of vehicles which arrived previously.