Space-Time Queues and Dynamic Traffic Assignment: A Model, Algorithm and Applications

Submission date: July 31, 2002
Word count: 6633 + (3 figures) * 250 words/figure = 7383

Michael Mahut, INRO Consultants, Inc., 5160 Decarie Blvd. Suite 610, Montreal H3X 2H9, Canada. Telephone: (514) 369-2023; fax: (514) 369-2026; e-mail: michaelm@inro.ca

Michael Florian, University of Montreal, P.O. box 6120, station centre-ville, Montreal H3C 3J7, Canada; and INRO Consultants, Inc. Telephone: (514) 369-2023; fax: (514) 369-2026; e-mail: mike@inro.ca

Nicolas Tremblay, INRO Consultants Inc. 5160 Decarie Blvd. Suite 610, Montreal H3X 2H9, Canada. Telephone: (514) 369-2023; fax: (514) 369-2026; e-mail: nicolas@inro.ca
The evaluation of on-line intelligent transportation system (ITS) measures, such as adaptive route-guidance and traffic management systems, depends heavily on the use of faster than real time traffic simulation models. Off-line applications, such as the testing of ITS strategies and operational planning studies, are also best served by fast-running traffic models due to the repetitive or iterative nature of such investigations. This paper describes a simulation-based, iterative dynamic-equilibrium traffic assignment model. The determination of time-dependent path flows is modeled as a master problem that is solved using the method of successive averages (MSA). The determination of path travel times for a given set of path flows is the network-loading sub-problem, which is solved using the space-time queueing approach of Mahut. This loading method has been shown to provide reasonably accurate results with very little computational effort. The model was applied to two versions of the Stockholm road network: one consists of 2080 links, 1200 nodes and 220 zones, representing over 5,000 turns; the other consists of 4342 links, 1980 nodes and 250 zones, representing over 11,000 turns. The results show that this model is applicable to medium-size networks with a very reasonable computation time.
INTRODUCTION

The functional requirements of a dynamic traffic assignment (DTA) model for ITS applications may be subdivided into two major modes of use: off-line and on-line. The off-line use of DTA is for the testing and evaluation of a wide variety of ITS measures before they are implemented in practice. In particular, iterative approaches to dynamic assignment that approximate (dynamic) user equilibrium conditions are generally restricted to off-line use due to the high computation times involved. The resulting assignments can also be interpreted as imitating drivers’ adaptation over time to changes in network topology or control, including the implementation of ITS measures. Due to the high number of iterations usually required, such applications are ideally suited for traffic models that have low computational requirements. On-line DTA can be used within a system that monitors and manages the network in real time. DTA and the embedded traffic models can play a key role in providing short-term forecasts of the system state that are used by adaptive traffic management, control and guidance systems. Due to the need to provide feedback in real time, on-line DTA poses rather stringent demands on the embedded models for maintaining low computation times.

The need to model the time varying network flow of vehicles for ITS applications has generated many contributions for the solution of dynamic traffic assignment methods. These contributions are varied and have been motivated by different methodological approaches. They may be classified according to the modeling paradigm underlying the temporal traffic model. In order to provide a common terminology to the various models, it is convenient to refer to two main components of any dynamic traffic model: the route-choice mechanism and the network-loading mechanism. The latter is the method used to represent the evolution of the traffic flow over the links of the network once the route choice has been determined.

There are two distinct approaches to modeling route-choice (i.e., assignment), namely en-route assignment and equilibrium assignment. In the en-route approach, route-choice decisions are made for drivers entering the network over an interval of time based on the link travel times on the network at the start of the interval. Drivers already on the network may also have their route choices modified as link travel times are updated after each interval. Because the link travel times used to evaluate potential paths describe the network conditions at a single point in time, en-route approaches are often said to be based on instantaneous travel time. This assignment method thus requires only one loading (or simulation) of the network.

The equilibrium approach is based on evaluating potential paths using actual, or experienced – as opposed to instantaneous – travel times. In this case, when potential routes are evaluated by the route-choice mechanism, the travel time that a specific link contributes to the total cost of a path depends upon the travel times of the links preceding it on the path. The input to the route-choice mechanism thus requires knowing the travel times on all links at all times. The solution takes the form of an iterative approach, where the route-choices made for one iteration are based on the (time-dependent) link travel times observed on the previous iteration. The equilibrium approach thus requires running the network loading and route-choice algorithm several times, typically at least 30 or 40 times, until some measure of convergence is satisfied.

The equilibrium approach can be thought of as mimicking drivers’ day-to-day learning and adaptation to experienced traffic conditions, and is useful for estimating the assignment and resulting traffic conditions that describe a typical day or peak period. The en-route approach is best suited to predicting drivers’ reactions to real-time information (such as en-route guidance information) given the route-choice for the typical day as a “do-nothing” alternative. When the network topology allows drivers to change routes frequently, a realistic assignment for the typical day can be obtained using the en-route approach with a naïve (“all-or-nothing”) do nothing alternative. However, this requires a highly connected network with closely-spaced nodes, and very short time intervals for updating the link travel times and route-choice decisions.

Calibrating a DTA model requires running the assignment and adjusting the inputs in order to obtain outputs that match a reference or ‘baseline’ data set. Calibration is thus an iterative process within which the chosen assignment method is repeated until a satisfactory convergence is obtained between the simulated and desired outputs. If using an iterative assignment method, the entire calibration process will typically involve loading the network hundreds of times.
Perhaps the most popular dynamic traffic models today are micro-simulation models based on the representation of the behavior of each driver regarding car following, gap acceptance and lane choice, such as CORSIM (1), INTEGRATION (2), AIMSUN2 (3), VISSIM (4), PARAMICS (5) and DRACULA (6). MITSIM (7) is an academic research model that has been used in several studies in Boston, Stockholm and elsewhere. There are many other micro-simulation models developed in universities and industrial research centers that use the same basic approach.

A traffic simulator combines some basic mathematical models, such as car-following and gap-acceptance models, with a set of heuristic rules that govern lane-changing, overtaking, aggressive merging (“gap-forcing”), as well as other aspects of driver behavior. The combination of multiple mathematical models with multiple rule-based (heuristic) models leads to a rather complex process that may exhibit high sensitivity to the values chosen for the numerous parameters involved. A successful calibration thus requires a thorough understanding of how changes in the inputs result in changes in the outputs, which will be will be at least partially specific to the simulator being used.

Due to the stochastic nature of the underlying processes, the proper use of traffic simulation requires the replication of runs. One network-loading step should thus be composed of several simulation runs, which would significantly increase the computation time of a single DTA, although this is not always done. Moreover, although micro-simulation can in principle be used in conjunction with an iterative assignment method, virtually all micro-simulators use an en-route approach. Nevertheless, the successful use of micro-simulations is commonly limited to relatively small size networks: their application to medium-to-large networks has been hindered by the relatively high computation time and effort required for a proper model calibration. This is not to say that the computations times are prohibitively long; it is perhaps rather a question of convenience. Micro-simulation models are nonetheless popular and their use is enhanced by traffic animation graphics that capture the attention of non-technical staff.

The aim of handling larger networks with lower computational times has led to the development of so-called “mesoscopic” approaches to traffic simulation, which are less precise in the representation of traffic behavior but are less cumbersome computationally. The aim is to obtain a traffic representation that still captures the basic temporal congestion phenomena, but models the traffic dynamics with less fidelity. One of the earliest examples of such an approach is CONTRAM (8) which is a commercially available package that has been used in England and elsewhere in Europe.

Recently, the development of mesoscopic simulation models for off-line dynamic traffic assignment has become an area of significant research activity, as witnessed by the United States Federal Highway Administration Dynamic Traffic Assignment Project (9). The development of DYNASMART (10) and DYNAMIT (11) are two significant developments. These mesoscopic models provide a path choice mechanism and a network loading method based on simplified representations of traffic dynamics. While CONTRAM represents traffic with continuous flow, as it has its roots in static traffic assignment models, DYNASMART and DYNAMIT move individual vehicles. CONTRAM and DYNASMART provide an iterative scheme for the emulation of dynamic user equilibrium, where all cars within the same departure interval for a given origin-destination pair experience the same travel time (approximately). The approach taken in DYNAMIT is to provide an “a priori” path choice and path set by using models based on random choice utility theory. Another approach to the network loading algorithm is that based on cellular automata theory (12), which has been implemented in the TRANSIMS software (13), developed recently by the Los Alamos National Laboratories in the USA. In this approach, the route choice is predetermined for each traveler and the network loading method loads the vehicles on a network where each lane of a link is divided into cells of equal size. The advance of vehicles is carried out by using local rules for each vehicle that determine the next cell to be occupied and the speed of the vehicle.

Other dynamic traffic assignment models have their roots in macroscopic traffic flow theory developed during the 1950’s (14) (15). The work of Papageorgiou (16) led to the development of the METACOR (17) and METANET (18), which has been used for the development of an iterative dynamic traffic assignment method (19). The route choice in this model is carried out by splitting proportions at nodes of the network, where only two arcs can originate at a given node. The network loading method is based on a second order (p.d.e.) traffic flow model.

Another line of research is that of analytical dynamic traffic assignment models, which has its roots in the mathematical programming approach to static network equilibrium models. This area is not covered in this contribution.
The dynamic assignment model presented in this paper is based on a traffic simulation model that was designed to produce reasonably accurate results with a minimum number of parameters and a minimum of computational effort \((20)\) \((21)\). However, the underlying structure of the model has more in common with microscopic than with mesoscopic approaches, as it is designed to explicitly capture the effects of car following, lane changing and gap acceptance. The simulation is a discrete-event procedure. Unlike discrete-time microscopic simulation models, where the computational effort per link is proportional to the total vehicle-seconds of travel, the computational effort per link required by this model is strictly proportional to the number of vehicles to pass through it, regardless of their travel times. As a result, the relative efficiency of this approach compared to microscopic methods increases with the level of congestion. A detailed description of the model and solution algorithm is provided by Mahut \((21)\).

The paper is structured as follows. The next section is dedicated to the exposition of approaches to dynamic traffic assignment. The algorithm used here for the dynamic traffic assignment in subsequently presented. This is followed by a section dedicated to the description of the network loading method (traffic simulator), which is a sub-problem of the dynamic traffic assignment problem. Applications of the model are then given and some conclusions end the paper.

**DYNAMIC TRAFFIC ASSIGNMENT**

As described briefly above, two different approaches are commonly used to emulate the path choice behavior of drivers: *en route* assignment and *equilibrium* assignment. In this work, the approach taken is to seek an approximate solution to the dynamic equilibrium conditions through an iterative approach.

**En-Route Assignment**

In the en route assignment problem, the routing mechanism is a set of behavioral rules that determine how drivers react to information received en route. Information may be available at discrete points in time (e.g. radio broadcasts), discrete points in space (e.g. variable message signs), or be continuously available in both space and time (e.g. traffic conditions visible to the driver). Some information may only be available to a certain class of vehicles, e.g., those equipped with vehicle guidance systems. Typically, the choice of what information is provided to the drivers, i.e., the information strategy, is an exogenous input. Drivers respond to information according some heuristic rules that may involve one or more parameters, such as the ‘penetration rate’. The output is the resulting (time-dependent) path choices given the time-dependent origin-destination demand. Another input to this problem is a suitable pre-trip assignment, i.e., path choices that represent the “do nothing” alternative and which are followed in the absence of any en route information. In many cases, an equilibrium assignment (discussed below) is used for this purpose. En route assignment thus only requires running a single dynamic (time-dependent) loading of the demand onto the network over the time period of interest.

**Equilibrium Assignment**

In the equilibrium assignment problem, only pre-trip path choices are considered. However, the path choices are modelled as a decision variable and the objective is to minimize each driver’s travel time. All drivers have perfect access to information, which consists of the travel times on all paths (used and unused) experienced on the previous iterations. All drivers furthermore attempt to minimize their own travel times, and the solution algorithm takes the form of an iterative procedure designed to converge to these conditions. The solution algorithm used here consists of two main components: a method to determine a new set of time-dependent path flows given the experienced path travel times on the previous iteration, and a method to determine the actual travel times that result from a given set of path flow rates. The latter problem is referred to as the “network loading problem”, and can be solved using any route-based dynamic traffic model. The algorithm furthermore requires a set of initial path flows, which are determined by assigning all vehicles to the shortest paths, based on free-flow conditions. The general structure of the algorithm is shown schematically in Figure 1.

The mathematical statement of the dynamic equilibrium problem is in the space of path flows \( h_k (t) \), for all paths \( k \) belonging to the set \( K \), for an origin-destination \( i \in I \), at time \( t \). The time-varying demands are denoted
The path flow rates in the feasible region $\Omega$ satisfy the conservation of flow and non-negativity constraints for $t \in T_d$, where $(0, T_d)$ is the period during which the temporal demand is defined. That is

$$\Omega = h(t) : \sum_{k \in K_i} h_k(t) = g_i(t), i \in I; h_k(t) \geq 0$$

for almost all $t \in T_d$

The definition of user optimal dynamic equilibrium is given by the temporal version of the static (Wardrop) user optimal equilibrium conditions, which are:

$$s_k(t) \begin{cases} = u_i(t) & \text{if } h_k(t) > 0 \\ \geq u_i(t) & \text{otherwise} \end{cases}$$

for all: $k \in K_i, i \in I$, for almost all $t \in T_d$

where: $h_k \in \Omega, u_i(t) = \min_{k \in K_i} s_k(t)$ for almost all $t \in T_d$ and $s_k(t)$ is the path travel time determined by the dynamic network loading. Friesz et al (22) showed that these conditions are equivalent to a variational inequality problem, which is to find $h^* \in \Omega$ such that

$$\left(S(h^*), h - h^*\right) \geq 0, \forall h \in \Omega$$

The continuous time problem (1)-(3) is usually solved by using some time discretization scheme.

**THE ALGORITHM**

The solution approach adopted for solving the dynamic network equilibrium model (1)-(3) is based on a time discretization into discrete time periods $\tau = 1, 2, ..., \lfloor \frac{T_d}{\Delta t} \rfloor$, where $\Delta t$ is the chosen duration of a time interval. This results in the discretized model

$$s_k^\tau \begin{cases} = u_i^\tau & \text{if } h_k^\tau > 0 \\ \geq u_i^\tau & \text{if } h_k^\tau = 0 \end{cases}$$

for all $k \in K_i, i \in I, \tau = 1, 2, ..., \lfloor \frac{T_d}{\Delta t} \rfloor$

where the feasible set of time dependent flows $h_k^\tau$ belong to

$$\Omega^\tau = h_k^\tau : \sum_{k \in K_i} h_k^\tau = g_i^\tau, i \in I, all \ \tau ;$$

$$h_k^\tau \geq 0, k \in K_i, i \in I, all \ \tau$$

which can be shown to be equivalent to solving the discretized variational inequality.

$$\sum_{\tau} \sum_{k \in K} s_k^\tau (h_k^\tau - h_k^\tau) \geq 0$$
where \( K = \bigcup_{i \in I} k_i \)

The path input flows \( h^*_k, k \in K \) are determined by the method of successive averages (MSA), which is applied to each O-D pair \( i \) and time interval \( \tau \). The iterative algorithm updates the path inflow vectors at each iteration \( l \), until either the convergence criteria is below some threshold, \( \varepsilon \), or until a maximum number \( (L_0) \) of iterations has been achieved. While no formal convergence proof can be given for the algorithm, since the network loading map does not have an analytical form, a measure of gap, inspired from that used in static network equilibrium models may be used for qualifying a given solution. It is the difference between the total travel time experienced and the total travel time that would have been experienced if all vehicles had the travel time (over each interval \( \tau \)) equal to that of the current shortest path. Hence:

\[
R \text{Gap}^\tau(l) = \frac{\sum_{i \in I} \sum_{k \in k} h^*_i(l)s^*_i(l)}{\sum_{i \in I} g^*_i(l)}
\]  

(7)

Where \( u^*_i(l) \) are the lengths of the shortest paths at iteration \( l \). A relative gap of zero would indicate a perfect dynamic user equilibrium flow. Clearly this is a fleeting goal to aim for with any dynamic traffic assignment.

The algorithm can be stated as follows:

**Step 0** Initialization: \( l=1 \).

Compute dynamic shortest paths for each interval \( \tau \) based on the free-flow travel times and load the demands \( g^*_i, i \in I \), to obtain an initial solution.

**Step 1** \( l=l+1 \).

If \( l<N \): compute a new dynamic shortest path for each time interval \( \tau \), for each O-D pair \( i \in I \), and assign to each path \( k \in K \) the input flow \( g^*_i/l \), for all \( \tau \).

Else: identify the shortest among the used paths and redistribute the flows as follows:

\[
h^*_i(l) = \begin{cases} 
    h^*_i(l-1) \left( \frac{l-1}{l} \right) + \frac{g^*_i}{l} & \text{if } s^*_i(l-1) = u^*_i(l-1) \\
    h^*_i(l-1) \left( \frac{l-1}{l} \right) & \text{otherwise}
\end{cases}
\]  

(8)

for \( k \in K_i, i \in I \) and all \( \tau \)

Load the demands onto the network (execute traffic simulation) to obtain new time-dependent link travel times.

**Step 2** Convergence test: if \( R\text{Gap}^\tau(l) \leq \varepsilon \), STOP.

Otherwise, return to Step 1.

Up to iteration \( N \), the time-dependent link travel times after each loading are used to determine a new set of dynamic shortest paths (for each interval) that are added to the current set of paths. Thus, there are up to \( N \) different paths available between each O-D pair for each departure interval \( i \). for At each iteration \( l, l \leq N \), the volume assigned as input flow to each path in the set is \( g^*_i/l \), \( i \in I \), all \( \tau \). After that, for \( l>N \), only the shortest among used paths is identified and the path input flow rates are redistributed as shown in equation (8) above.
NETWORK LOADING

As mentioned above, the input to the network-loading problem is the set of time-dependent path flows, while the output is the set of time-dependent path travel times (as a function of the departure time from the origin). Any network loading model will simultaneously yield the time-dependent link flows, travel times and densities.

The network loading model used here moves discrete vehicles on a network defined at the level of individual lanes. The underlying mechanism of congestion in the model is the crossing, merging and diverging – collectively referred to as conflicts – of vehicle trajectories. Simply stated, whenever two vehicles pass the same point in space, there must be a minimum time separation between them. How to propagate the resulting delays upstream from one vehicle to the next in a realistic way is a problem of traffic dynamics.

Before these delays can be propagated, the conflicts themselves must identified, which requires the drivers to make choices about which lanes they will use on the links of their pre-assigned paths. Although the paths are assigned a priori, the lanes are chosen as the vehicle proceeds along its path, using a set of behavioral rules. Once the conflicts are identified, they must subsequently be resolved: one vehicle must lead, while the other must follow, and the following vehicle incurs some amount of delay. The delay is then propagated upstream according to a simplified car following relationship. The delay experienced by the first vehicle at a traffic control device is propagated in the same way.

Simulation Approach

Most microscopic simulators (2) (3) (4) (7) and mesoscopic simulators (10) (11) use a discrete-time (fixed time step) procedure. The simulation period is discretized into small time intervals, \( \Delta t \). After each \( \Delta t \), all the vehicles that are present on the network are moved, which implies the computation of the new position of each vehicle. This usually implies two scans of all the vehicles: one to determine the possible movements and the other to move the vehicles. The network is updated at each clock tick \( t = n(\Delta t) \), where \( n \in \{0, 1, \ldots, T/\Delta t\} \), and \( T \) is the duration of the loading period.

The solution algorithm for the model used here is a discrete-event (“event-based”) procedure. In a discrete-event simulation, each temporal process modeled is associated with a specific sequence of events, and each event is associated with a real-valued point in time. For instance, an event may be associated with a change of signal phase at a controlled intersection, or the arrival time of a vehicle to a link. Event based algorithms are typically used for modeling queuing systems. An event-based approach may be very efficient if one can minimize the number of events modeled and still obtain valid results.

As described below, a special property of this model is that the traffic dynamics are modelled without the (longitudinal) discretization of links into segments or cells. As a result the procedure only explicitly calculates the time at which each vehicle crosses each node on its path. This leads to a drastic reduction in computational effort relative to microscopic discrete time approaches, where the computational effort is a function of the total travel time experienced by the drivers.

Network Representation

The network definition required for this DTA model requires somewhat more information than that required for static network equilibrium models, yet somewhat less than is generally required for micro-simulation traffic models. Since the underlying traffic model moves individual vehicles on discrete lanes, each link must be defined by a number of lanes. Each lane furthermore is defined by an access code that determines which classes of vehicles may use the lane (e.g., taxi, bus, HOV, etc…). A length and speed limit furthermore define each link. At each node (intersection) of the network, a turn is defined for each permitted movement from an incoming link to an outgoing link. Each turn is defined by an access code and a saturation flow rate per lane. Unlike micro-simulation models, the network definition does not require geometrical information such as lane width, turning angles, and the dimensions of intersections.
Lane Choice

In contrast to continuum traffic models and static assignment models, traffic simulators model the movement of vehicles on individual lanes. How drivers utilize the available lanes of a roadway can have a significant, even drastic impact on both the total delays experienced and how these delays are distributed (spatially and temporally) in the network. Naturally, these effects will only be captured if the traffic model employed is sensitive to the effects of lane-changing activity on the effective flow capacity of a link. In this case, the pre-trip path information must be complemented by a set of lane choice rules in order to provide the necessary information to identify conflicts between vehicle trajectories. As mentioned above, such conflicts are the principal mechanism of traffic congestion in the model.

A common example of the impact of lane utilization is a congested off-ramp from a highway. Even if the ramp is only one lane wide, delays may be incurred on more than one lane of the highway. Some drivers will inevitably miss the back of the queue, intentionally or not, and then begin queueing in the neighboring lane(s) as they look for an opportunity to merge into the lane leading onto the ramp. The degree to which the queue spills over onto the neighboring lanes depends to a great extent on driver behavior. Specifically, if the queue spills back upstream over several links on a daily basis, drivers may be able to recognize the source of congestion as they reach the end of the queue several links upstream of the ramp. Thus, those drivers who are destined for the off-ramp may decide to join the back of the queue immediately, while those remaining on the highway may choose to avoid the queue. Drivers may often make such decisions even though they are still several links upstream of the off-ramp itself, which is the physical location of the bottleneck that is influencing this decision.

By joining the back of the queue immediately, drivers destined for the off-ramp will not delay drivers remaining on the highway; i.e., the amount of queue spill-over is reduced. Conversely, the amount of queue spill-over could well be unrealistically high if drivers were unaware of which lane exited the highway until they were on the last link before the ramp. In the traffic simulation literature, heuristics that take into account non-local (beyond the next link or turn) information about a driver’s intended path are often called “look-ahead” rules. The addition of look-ahead rules to existing heuristics based strictly on local information has been shown to significantly improve the reality of the model outputs for some specific though not uncommon network topologies (23) (24).

In the model used here, vehicle trajectories along links are modelled implicitly, rather than explicitly. Specifically, each driver chooses the lanes by which he/she will enter and exit a link just before actually arriving to the link and, once on the link, the choice cannot be re-considered. The principal argument behind using such an approach is that it is sufficient to model only mandatory lane changes in order to reproduce the general congestion patterns resulting from a given set of path flows. Mandatory lane changes are those that must be made in order to exit and enter each link on the lanes permitted for the associated turns.

The permitted lanes over a sequence of downstream turns are considered here when some of the lanes immediately downstream of the driver are queueing and some are not. This logic allows a driver to join the queue if necessary, or to by-pass it if his/her path does not go through the head of queue. Preliminary tests with this look-ahead feature have indicated a significant reduction in the amount of queue spill-over, as well as total delay, in the case of a congested off-ramp as discussed above.

Conflicts and Precedence

Given the network, path flow rates and lane-choice rules, conflicts may arise between vehicle trajectories at nodes and along links. A conflict between two vehicles exists when, given their positions at one moment in time, their desired arrival times to the same downstream position violates a constraint that specifies the minimum time separation between vehicles at that point (such as a specified saturation flow rate). Conflicts can arise both at nodes and on multi-lane links. In order to satisfy a minimum headway constraint, it must be decided which vehicle is to precede the other, and thus which vehicle is to be delayed. It is these delays that are the underlying mechanism of congestion in the model. The process of deciding precedence between two conflicting vehicles is referred to here as conflict resolution.

In reality, which vehicle precedes the other depends to some extent on human behavior. The question is typically resolved in a traffic simulation model by gap-acceptance rules (2) (3), which are based on one of the two
vehicles having priority over the other, and the specification of a time-gap parameter. In continuum traffic models, the approach is to specify the maximum low-priority flow as a function of the prevailing high-priority flow ($\delta$). In the model used here, a relatively simple gap-acceptance model has been implemented to determine precedence between vehicle conflicts at nodes, while a FIFO (first-in-first-out) rule is applied on links.

**Traffic Dynamics**

Once a conflict has been identified and resolved, and the appropriate delay has been calculated, this delay (or a residual portion of it) may propagate recursively over a sequence of vehicles against the direction of the traffic flow. The propagation of delay occurs in this model much the same way as in a normal queueing model. In fact, in the case of an isolated intersection, the amount of delay propagated from one vehicle to the next is exactly as would be determined by a standard queueing approach. What is different is where and when a vehicle in queue experiences each of the delays (or residuals thereof) that are propagated from downstream. This difference is due to the fact that the model employed here rigorously respects the finite speed at which delays propagate in actual traffic, sometimes called the negative wave speed. In a standard queueing model, delays propagate upstream instantaneously. In the model employed here, in the case of a general network where links have finite lengths and storage capacities, delays propagate from the exit position of a link to the link entrance through traffic densities that can be as low as the critical density (the density corresponding to the maximum flow rate), exactly as would be predicted by the hydrodynamic theory ($14, 15$). The model is derived from the following simplified car-following relationship, which can be shown to yield a two-linear-segment ("triangular") flow-density relationship ($21$):

\[
x(t, n) \leq x(t - \tau, n) + \tau V \\
x(t, n) \leq x(t - \tau, n - 1) + \lambda
\]

where:

$t$ = time

$n$ = vehicle number by order of arrival to the lane

$x(t, n)$ = position of vehicle $n$ at time $t$

$V$ = maximum speed on the lane (speed limit)

$\lambda$ = effective vehicle length (inverse of jam density)

$\tau$ = driver/vehicle response time

This relationship thus respects the maximum speed of the link, as well as the response time and effective vehicle length in a collision avoidance rule. This relationship can be manipulated to yield the following model for a one-lane link:

\[
t(n, L) = \max \left[ t(n, 0) + \frac{L}{V}, t(n - 1, L) + \left( \frac{\lambda}{V} \right) t^S(n, L) \right]
\]

(10)

\[
t(n, 0) = \max \left[ t^D(n, 0), t(n - 1, 0) + \left( \frac{\lambda}{V} \right) t(n - X, L) + X \tau \right]
\]

where:

$t(n, x)$ = the time of departure of vehicle $n$ from position $x$

$L$ = link length

$X$ = maximum occupancy of a lane ($L/\lambda$)

$t^S(n, L)$ = "supply time" of vehicle $n$ at the link exit

$t^D(n, 0)$ = "demand time" of vehicle $n$ at the link entrance
The “supply time” is the earliest time that a vehicle may exit the link as a function of the conditions at the downstream node and on the next downstream link, while the “demand time” is the earliest time that a vehicle may enter a link due to conditions at the upstream node and on the next upstream link. Because the times at which vehicles will enter and exit links according to this model are based strictly on other entrance and exit times, the solution algorithm calculates these times directly without explicitly modeling the trajectories of the vehicles on the links themselves. A thorough exposition of the above relationships, along with the multi-lane version that was used in this work and the solution algorithm that correctly solves the model under all possible traffic conditions is beyond the scope of this paper but can be found elsewhere (21). A small yet challenging benchmark test of the model, along with comparison results obtained with microsimulation and with a macroscopic model, can be found in the published literature (20).

Traffic Control

The implementation used in this work also permits the specification of detailed traffic control information such as (pre-timed) signal timing and ramp metering plans. Traffic control specifications furthermore require the number of lanes associated with each turning movement, and the lanes (on both the upstream and downstream links) that may be used for executing a turn. These data may vary with the signal phase rather than being fixed for each turn.

Vehicle Classes

Vehicle attributes (or parameters) can be broken down into two distinct categories: physical attributes and routing attributes. The physical attributes are the effective length (based on vehicle spacing at jam density), and the driver/vehicle response time. Together, these parameters yield the jam density and negative wave speed associated with each vehicle class. Routing attributes include the vehicle class identifier, which determines which lanes and turns of the network may be used by the class, and identifies any class-based routing strategy that may be defined. For instance, the class car uses different routing rules than the class bus, which travels along fixed itineraries and has mandatory stops. A demand matrix by class contains the flow in vehicles per hour for each origin-destination pair. The matrices are “time-sliced” in the sense that flow rates may be specified for given time intervals.

APPLICATIONS

This dynamic traffic assignment model was coded in C++ using an object-oriented approach. The original design was carried out on a SUN Workstation under Solaris 2.8. The code also runs and on an Intel PC under Linux and Windows 2000. The tests were run on an 1.2 GHz Intel PC with 512 Mb of RAM, running the Windows 2000 operating system. The Swedish Road Administration provided the authors with two versions of the Stockholm road network. The first network consisted of 1191 nodes and 2080 links representing about 11,000 turns. The demand for this network was defined over 114 zones and four 20-minute ‘time-sliced’ origin-destination (O-D) matrices. The second network consisted of 1980 nodes and 4342 links representing some 11,000 turns. This network included 250 zones, for which five consecutive 20-minute O-D matrices were defined. The total number of car trips defined for each of the networks was on the order of 120,000.

A dynamic traffic assignment was run on the first network for 40 iterations, each requiring roughly 1.5 minutes, for a total of one hour of computation time. The loading duration was subdivided into eight departure intervals for which the convergence measure (relative gap) was calculated after each iteration, as shown in Figure 2. A relative gap of zero indicates a user-optimal dynamic equilibrium. Gap values ranging from 1.4 to 8.0 percent were obtained by the last iteration. The gaps were generally increasing with each time interval, i.e., 1.4% was obtained for the first interval and 8.0% for the last. This increasing trend can be attributed to the fact that each driver’s decision is based on the travel times experienced in the previous iteration only, and does not consider the decisions made by earlier departing drivers in the current iteration. In this sense, the previous iteration serves as a prediction of the traffic conditions that will be encountered on the next iteration. As any given iteration (simulation) advances in time from $t = 0$ to $t = T$, the quality of this prediction degrades due to the increasing number of “unforeseen” decisions (those made for the current iteration before time $t$) that are affecting the actual traffic conditions on the network. The results are very promising and indicate that a reasonable level of convergence is attainable for a medium-sized network with an acceptable amount of computing time. The second network was run for 60 iterations requiring some 2.1 minutes each, for a total of 2.1 hours of computation time. Relative gap values for eight departure intervals ranged from 2.0 to 13 percent.
The convergence measure is an indication of the difference between the average travel time and the best travel time for the iteration. This should not be interpreted as the difference between the current average travel time and that corresponding to a perfect equilibrium. A better guess of how much improvement in travel times can still be attained might be half of the gap, i.e., it might be expected that the difference between the current travel times and the true equilibrium solution for the first network is on the order of 0.7% to 4.0% (depending on the departure interval).

Network statistics were collected over 10-minute intervals. A snapshot of the first network at 8:00 a.m. of the last iteration is shown in Figure 3. The widths of the links indicate the average link outflow rates over the 10-minute interval starting at 8:00 a.m. The colour indicates the relative density (occupancy) on each link. (blue: 0-10%; green: 10%-20%; yellow-green: 20%-40%; yellow: 40%-60%; orange: 60%-80%; red: 80%-100%).

CONCLUSIONS

A dynamic traffic assignment model, which uses the method of successive averages (MSA) to determine pre-trip dynamic equilibrium path choices combined with an event-based traffic simulation model, was successfully applied to two medium-sized networks of the city of Stockholm: one consisting of 1191 nodes and 2080 links, the other consisting of 1980 nodes and 4342 links. Convergence measures (relative gaps) for the smaller network ranging from 1.4% to 8.0% percent (by time interval) were obtained with 1 hour of computation time on an 1.2 GHz PC. Relative gaps for the larger network ranging from 2.0% to 13% were obtained in 2.1 hours of computing time. These results can be interpreted as indicating an approximate difference between the current travel times and the true equilibrium solution of 0.7% to 4.0%, and 1.0% to 6.5%, for the smaller and larger networks, respectively.

The results are very promising in two ways. Firstly, the computation time for the traffic simulation model indicated a real-time speed-up of about 100x for medium-sized networks using affordable hardware. Secondly, the number of iterations required to obtain acceptable levels of convergence implied a reasonable amount of total computation time for the dynamic traffic assignment. The method has excellent potential for use in practice for operational planning studies and off-line testing of ITS measures, such as lane management strategies and the evaluation of alternative control scenarios in response to incidents. The model may have potential for further development as an on-line tool, due to the low computation times and memory requirements.

ACKNOWLEDGEMENTS

This work was partially sponsored by a Post-Doctoral Industrial Research Fellowship of the Natural Sciences and Engineering Council of Canada (NSERC).

REFERENCES


**LIST OF FIGURES**

1 Structure of solution algorithm to DTA problem.
2 Convergence criteria by departure time interval.
3 Link flows colored by density at 8:00 a.m.
Figure 1  Structure of solution algorithm to DTA problem.
Figure 2  Convergence criteria by departure time interval.
Figure 3  Link flows colored by density at 8:05 a.m.