WORKING PAPER

ITLS-WP-14-10

An efficient event-based algorithm for solving first order dynamic network loading problems

By
Mark P.H. Raadsen, Michiel C.J. Bliemer and Michael G.H. Bell

May 2014

ISSN 1832-570X

INSTITUTE of TRANSPORT and LOGISTICS STUDIES

The Australian Key Centre in Transport and Logistics Management

The University of Sydney

Established under the Australian Research Council's Key Centre Program.

NUMBER: Working Paper ITLS-WP-14-10

TITLE: **An efficient event-based algorithm for solving first order dynamic network loading problems**

ABSTRACT: In this paper we will present a novel solution algorithm for the Generalised Link Transmission Model (G-LTM). It will utilise a truly event based approach supporting the generation of exact results, unlike its time discretised counterparts. Furthermore, it can also be configured to yield approximate results, when this approach is adopted its computational complexity decreases dramatically. It will be demonstrated on a theoretical as well as a real world network that when utilising fixed periods of stationary demands to mimic departure time demand fluctuations, this novel approach can be efficient while maintaining a high level of result accuracy. The link model is complemented by a generic node model formulation yielding a proper generic first order DNL solution algorithm.

KEYWORDS: ***Dynamic network loading, first order, generalised link transmission model, event-based, fast approximate solution***

AUTHORS: ***Raadsen, Bliemer and Bell***

CONTACT: INSTITUTE OF TRANSPORT AND LOGISTICS STUDIES (C13)

The Australian Key Centre in Transport and Logistics Management

The University of Sydney NSW 2006 Australia

Telephone: +612 9114 1824

E-mail: business.itlsinfo@sydney.edu.au

Internet: http://sydney.edu.au/business/itls

DATE: May 2014

1. Introduction

Network modelling plays an important role in modern day transport planning. It all once started with empiric observations leading to the fundamental diagram and the identification of kinematic wave theory. Over time, the field has flourished and sparked enough interest to yield a broad range of models all with their own strengths and weaknesses. At the present day we roughly identify three major types of network models; firstly the traditional static models originally developed in the 1950s, secondly the dynamic models that emerged in the early 1990s and sitting in between the lesser known quasi-dynamic models that combine characteristics of both.

In this paper we will solely focus on dynamic models. In the literature this group of models is often referred to as Dynamic Traffic Assignment (DTA). In that case it encompasses both the propagation model, better known as Dynamic Network Loading (DNL) model and a, possibly iterative, route choice model. Over the years the DNL model paradigm branched off into three categories. For highly detailed, often smaller scale networks, microscopic models have been developed. This type of model simulates vehicles individually and is therefore potentially the most realistic. That said, they do come with their own set of challenges; they are known to be relatively computationally expensive as well as configuration hungry. Another known pitfall of microscopic models is their inherent unstable results, requiring more runs to get some form of 'representative' outcome. On the opposite end of the spectrum macroscopic models reside, adopting traffic flow theory in order to model traffic as a stream of vehicles. These models are traditionally utilised when modelling larger scale networks. They are faster to run, typically free of random components, easier to configure and relatively simple to interpret. However, they can lack some of the realism of their microscopic counterparts especially in an urban setting. The third category contains mesoscopic models, which adopt characteristics of the two others, usually modelling vehicles individually but propagating them according to macroscopic traffic flow theory (i.e., using an underlying fundamental diagram).

These DNL models can again be subdivided into first order and second order models. Second order models take "indirect behavioural" effects into account such as reacting to congestion downstream, or anticipating on merging effects taking place at motorway on- ramps for example. Even though this seems natural, from a theoretical perspective there is a lot of debate on the mathematical rigour of these models. First order models on the other hand only model direct effects such as flow (and thus speed) changes on a link based on a fundamental diagram, and are capable of representing most elemental traffic characteristics such as queue formation and propagation. They typically lack the additional complexity of the "indirect behavioural" component. These first order models have proven to be mathematically rigorous while still yielding sufficiently realistic results. In this paper we will concentrate on macroscopic first order DNL models for large scale networks.

1.1 Outline and Paper Contributions

In this paper we will propose a novel first order macroscopic DNL solution algorithm based on the general, and arguably, most elegant first order link model available to date: the Generalised Link Transmission Model (G-LTM), which extends the original link transmission model (LTM) to an arbitrary concave fundamental diagram instead of the simplified triangular fundamental diagram. In order to do this we will first discuss the current state of the literature on macroscopic DNL models before describing the mathematical framework of G-LTM. As each proper DNL model also has a node model we subsequently discuss the adopted first order node model. Although the node and link models presented are not a contribution of this paper as such, we have to introduce them thoroughly to be able to discuss our newly proposed algorithm and do justice to its subtleties. Also, we propose some reformulations in order to accommodate our algorithm in the best possible way.

In this paper we deviate from the typical time based algorithms that split the time horizon into small time steps, and propose an event based algorithm for solving the general first order DNL problem. The section on the proposed event based algorithm is broken up in a discussion regarding the basic steps and when needed expands on relevant details of sub steps. The next two chapters are dedicated to a comparison with the existing time discretised LTM solution algorithm and the discussion of two case studies (of a corridor network and a real world network (Amsterdam)) demonstrating its correctness, level of efficiency and applicability on large scale networks.

Reaching the conclusion we will have proposed to the reader a novel solution algorithm for the G-LTM. This novel solution algorithm owes its efficiency to the fact that it is truly event based which, to the best of our knowledge, is unique in macroscopic models to date. The proposed algorithm is capable of generating exact results (unlike time discretised counterparts). From a practical perspective we will have demonstrated that this algorithm can be adapted to support the option of weakening the exact result constraint, hereby significantly reducing the computational complexity even further. Furthermore, we will identify that the difference of the resulting approximation can be kept to a near negligible level compared to the original results. Consequently, we believe the proposed solution algorithm is a valuable contribution the current existing DNL models for real world large scale applications.

2. Dynamic network loading models

In this section we introduce the topic and its current state in the literature as well as notation and where needed some examples to clarify the mathematical foundations that are the basis of this paper. This section will not present new theory but is nonetheless essential in order to convey the principles that are behind the novel solution algorithm.

2.1 Macroscopic DNL in the literature

Macroscopic DNL models consist of a link model and a node model. We first provide a brief literature overview of the link models, and then of the node models.

The foundations for most modern day macroscopic DNL models were laid by kinematic wave theory. This theory entails the way (shock) waves of traffic traverse a road and is first described by Lighthill and Witham (1955) and Richards (1956). This model is often referred to as the LWR model. The conservation of flow defines the basic relationship between how a change in density *k* over time *t* relates to a change in flow rate *q* over space *x* in such a way that no vehicles can be lost in the process.

$$\frac{dk}{dt}+\frac{dq}{dx}=0 \tag{1}$$

In (1) the flow rate $q=f\left(k\right)$ is presented solely as a function of density, which in turn is nothing else than the well known fundamental diagram. The speed ϑ is a direct result ($\vartheta=q/k$) and is not defined separately. This all builds on the assumption that a road at position *x* at time *t* is in a stable state. This type of model is known as a first order model, in which changes in speed are assumed to be instantaneous. These initial first order models caught on and sparked further research. Payne (1973) for example designed a ramp metering system, where he incorporated an adapted version of the LWR model taken from Payne (1971). This extended version adopts an additional speed term derived from car following behaviour, and is referred to as a second order model. It is based on the argument that the assumption of a stable traffic state of the original model seldom presents itself in real life. This additional speed formulation as seen in equation (2) accounts for acceleration or deceleration given surrounding traffic states, i.e. speed changes are no longer assumed to be instantaneous. Here change in speed depends on the speed upstream (convection), the related current stable state speed weighed by relaxation constant *R* (relaxation) and the downstream situation (anticipation) with constant c. A time and

space discretised version of this model named METANET was developed by Messmer and Papageorgiou (1990) and adapted versions of this model like MaDAM have been applied extensively in practice (Raadsen et al. (2010)).

$$\begin{aligned} &\frac{dk}{dt} + \frac{d\left(k\vartheta\right)}{dx} = 0 \\ &\frac{d\vartheta}{dt} = \underbrace{-\vartheta\frac{d\vartheta}{dx}}_{\text{convection}} + \underbrace{\frac{f\left(k\right)-\vartheta}{R}}_{\text{relaxation}} - \underbrace{\frac{c^2}{k}\frac{dk}{dx}}_{\text{anticipation}} \end{aligned} \tag{2}$$

Second order models sparked of a lot of debate regarding their validity (see Daganzo (1995a), Papageorgiou (1998)), also possible solutions have emerged (e.g., Aw and Rascle, 2000). Both first and second order models adopt a fundamental diagram of some sort. Newell (1993a) proposed a simple triangular shaped fundamental diagram and was also the first (see Newell (1993b)) to mention boundary conditions as a way of keeping track of travel state. This proved very useful in converting theory to practice. Daganzo's (1995b) Cell Transmission Model (CTM) adopts Newell's simplified triangular fundamental diagram and proposes a pragmatic solution method by discretizing space (i.e., transforming all links into equally sized cells) and discretizing time (i.e., assuming a certain fixed time step size). Each cell in CTM has a single state defining its flow, speed and density for a given time step, again aimed at improving the feasibility of implementing these models in a practical (large scale network) context. The most recent development in link models is the emergence of link transmission models. These models step away from the cell based approach and instead utilise link boundary conditions to extract information on flows, densities and speeds. The Link Transmission Model (LTM) proposed by Yperman (2007) also adopts the triangular fundamental diagram in its basic formulation and extends the model of Newell (1993a) to a network context using discretized time. The LTM model can be solved more efficiently and more accurately than the CTM model as it does not require links to be split into smaller cells. The Generalised Link Transmission Model (G-LTM) proposed by Gentile (2011) describes a continuous time version of the model of Yperman (2007), but makes an important generalisation to supporting any concave fundamental diagram. While the triangular fundamental diagram allows very efficient solution algorithms as demonstrated in the CTM and LTM models due to the fact that there only exist two kinematic wave speeds (one forward and one backward), the extension to a more realistic general concave fundamental diagram essentially means there are an infinite number of possible kinematic wave speeds. This means that solving the G-LTM model is more challenging than solving the LTM model. Gentile (2011) solves the G-LTM model by again discretizing time and solving the model for each time interval.

Both the CTM and the (G-)LTM models are first order models, are mathematically rigorous and are among the most elegant link models found in the literature to date. Their main advantage over other models is their simplicity and potential efficiency in determining in and out flow rates while still yielding realistic results. All propose solution algorithms that rely on (fixed) time interval updates, which can prove to be inefficient as will be demonstrated in this paper.
The research on macroscopic node models has long remained a by-product; most of the aforementioned models for example only discuss merge and diverge nodes on highways like Daganzo (1995b). Bliemer (2007) formalised a demand constrained approach for any type of node configuration, an approach since long used in practice. This formulation however violates the invariance principle formulated by Lebacque and Khoshyaran (2005). In recent years more sophisticated first order node models have emerged that satisfy this constraint for merge nodes, such as Jin (2012), Jin and Zhang (2002) or Tampere et al. (2011). The latter proposing a more generic approach defined as a set of constraints and a solution algorithm that also works on nodes with multiple in and out links, i.e. cross nodes. When looking at internal node constraints it can be shown that even the node model proposed by Tampere can have trouble finding a unique solution as shown by Corthout (2012), this has caused researchers to look into even

more detailed, almost microscopic approaches in trying to solve this problem including turning movements on a lane level, see Flötteröd and Rohde (2011).

2.2 Theoretical framework

We will now introduce a mathematical representation of our transport system. We will adopt a transport network $G = (N, \mathrm{A})$, where each node in the network is part of set *N*, with $O \subseteq N$ and $D \subseteq N$ being the subsets containing the origin and destination nodes. Consequently, we can define an origin-destination pair (OD) pair as (*o*,*d*) with $o \in O, d \in D$. Links are represented by set *A*, each link only takes on traffic flows in one direction, making *G* a directed graph. Each link has a length via L_a (km) as well as a capacity Q_a (veh/h), jam density K_a (veh/km), free speed $\vartheta_a^{\max}$ and speed at capacity $\vartheta_a^{\mathrm{crit}}$. Let each path (route) that is constructed based on our transport network be defined via path $p \in P$.

In this paper we assume Z finite periods of stationary demand, with $z = 1 \ldots Z$ and each period given by $\left[\mathrm{T}^{(z)}, \mathrm{T}^{(z+1)}\right]$. The end time of the entire DNL is denoted by *T*. Each time period has travel demand per path associated with it denoted by $f_p(z)$.

We will describe our novel DNL solution algorithm in terms of link inflow and outflow rates, without the need to calculate (intermediate) densities, travel speeds, and resulting link travel times. If the reader is interested in how to extract for example path and link travel times, densities and speeds from the simulation flow rates we would like to refer to the extensive literature existing on the topic by Newell (1993b), Daganzo (1993), or Szeto and Lo (2005). For comparisons between different approaches and their validity see for example Long et al. (2011).
Each proper DNL model consists out of a link model and a node model. Our node model formulation will be completely generic albeit in slightly restricted form because of the chosen link model. In this paper we will adopt the G-LTM formulation for the link model, which defines the class of first order link models adopting any concave fundamental diagram. The mathematical formulation of this model will be presented and discussed in the next section and are needed to understand the principles adopted for the novel event based algorithm formulation later on.

3. Link model formulation

We will start by defining the time dependent inflow and outflow rates for link *a*, denoted by $u_a(t)$ and $v_a(t)$ respectively. The cumulative inflow into link a at time instant *t* and the cumulative outflow out of link *a* at time instant *t* are respectively defined as

$$U_a(t) = \int_0^t u_a(w)\,dw \tag{3}$$

$$V_a(t) = \int_0^t v_a(w)\,dw \tag{4}$$

As stated our algorithm will be based on the G-LTM. The complexity of this model is such that we will first discuss the Link Transmission Model (LTM), which is a simplified version of the G-LTM adopting a triangular fundamental diagram, as presented in Figure 2(a) (for readability the subscript denoting link *a* is omitted here). We additionally will provide examples to explain the main principles that are behind the G-LTM as to date only a strictly mathematical formulation exists.

The traffic states at the link boundaries can be distinguished in terms of free-flow or no free-flow, and spillback or no spillback. Note that when the link boundary is in spillback state or no spillback state we could also group them under the no free flow state. Similarly when in free

flow state or no free flow state we can aggregate them under the term of no spillback. This grouping becomes useful later on when we will only be interested in link boundary traffic state changes between free flow and no free flow and spillback or no spillback. The described (valid) state transformations are also shown in Figure1.

	no spillback	spillback
free-flow	free-flow	N/A
no free-flow	congested	spillback

Figure 1 - State transitions as seen from either free flow link boundary state perspective (vertically), or spillback link boundary state perspective (horizontally)

Consider a forward kinematic wave arriving at the end of the link at a certain time instant *t*. This kinematic wave has encountered only free flow conditions if the kinematic wave departed at the beginning of the link L_a/γ_a earlier, where γ_a is the speed of forward kinematic waves, which in the triangular diagram is equal to the (maximum) free-flow speed $\vartheta_a^{\max}$. This is for example the case up to time t_1 in the example of Figure 2(b). More generally the following always holds in free flow state:

$$U_a\left(t-\frac{L_a}{\gamma_a}\right)-V\left(t\right)=0 \tag{5}$$

Using (3) and taking the time derivative of (5) we can formulate the outflow rate in terms of an inflow rate as shown in (6).

$$v_a\left(t\right)=u_a\left(t-\frac{L_a}{\gamma_a}\right) \tag{6}$$

We can now define the desired sending flow $s_a\left(t\right)$ in terms of $v_a\left(t\right)$, which will become relevant for the node model later on. The outflow rate $v_a\left(t\right)$ reflects the actual physical situation on the road whereas $s_a\left(t\right)$ represents the desired situation based on the traffic state on the link boundary. Equation (7) reflects the distinction between the free flow traffic state, in which the sending flow is equal to the inflow exactly L_a/γ_a earlier, and the no free flow traffic state, in which the sending flow reverts to capacity.

$$s_a(\mathrm{t})=\begin{cases} C_a, & \text{if } U_a\left(t-\frac{L_a}{\gamma_a}\right)-V\left(t\right)>0, \\ v_a\left(t\right), & \text{otherwise.} \end{cases} \tag{7}$$

The capping to capacity intuitively makes sense because a queue has formed in the no free flow state so the offered traffic fills all available outflow capacity in that case.

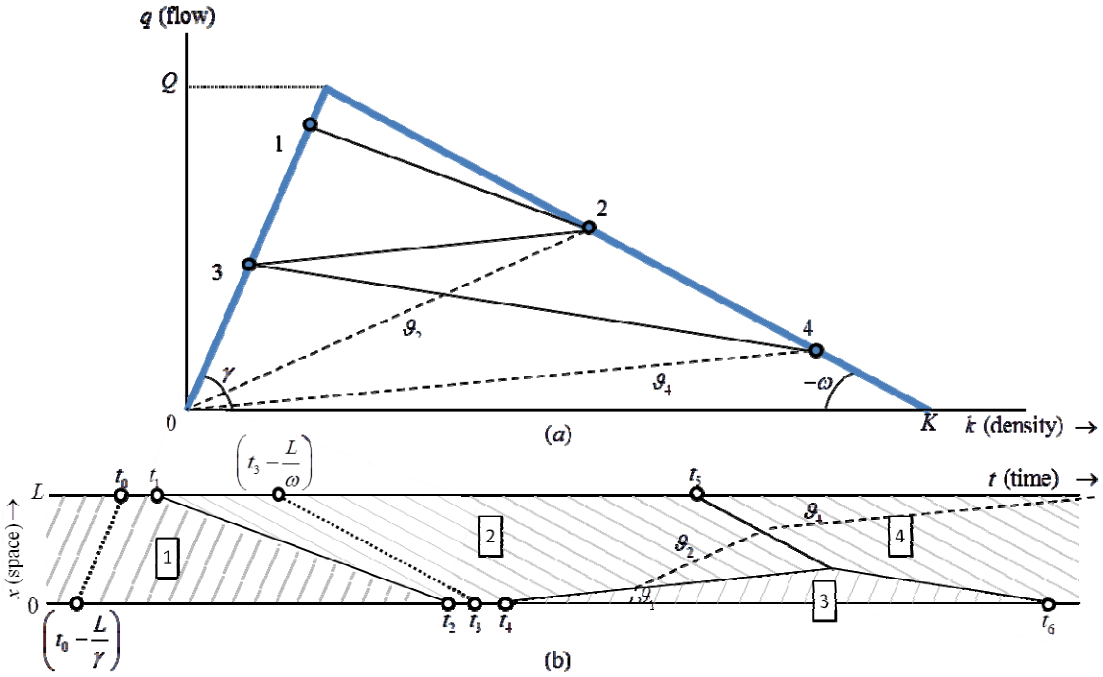


Figure 2 - (a) Triangular fundamental diagram with four example traffic states and transition shockwaves (b) Space time projection of (shock) waves and link traffic states on example link.

We can similarly define the spillback and no spillback traffic states. Consider a backward kinematic wave arriving at the start of the link at a certain time instant *t*. This kinematic wave has encountered only spillback conditions if the kinematic wave departed at the end of the link L_a/ω_a earlier, where ω_a is the speed of backward kinematic waves. Take for example time t_1 in Figure2(b), the link is no longer in free flow state (due to traffic state 2) and moves to spillback state at time t_2 .The $-\omega_a$ known as backward (spillback) kinematic wave speed and represents a negative speed to travel to the opposite border and find that in this time you have passed $K_a L_a$ vehicles.
This is expressed in (8) with the related time derivative expressing the inflow rate in (9)[1].

$$U_a(t) - V_a\left(t - \frac{L_a}{\omega_a}\right) = K_a L_a \tag{8}$$

$$u_a(t) = v_a\left(t - \frac{L_a}{\omega_a}\right) \tag{9}$$

With the definition of the inflow rate $u_a(t)$, we now define the possible receiving flow $r_a(t)$, where it reverts to the maximum intake of flow (i.e., the capacity) unless the link boundary is in spillback state. In the latter case the possible receiving flow is obviously capped to the related spillback state outflow rate. The receiving flow, like the sending flow is the main link between link and node model (discussed in section 4).

[1] The simplest way to picture this is to imagine a link with no outflow. Only after the whole link is filled with $K_a L_a$ vehicles it is in spillback as the situation at the end of the link (no outflow) has moved its way up to the start of the link.

$$r_a(t) = \begin{cases} C_a, & \text{if } U_a(t) - V_a\left(t - \dfrac{L_a}{\omega_a}\right) < K_a L_a, \\ u_a(t), & \text{otherwise.} \end{cases} \tag{10}$$

Finally, when both link boundaries are neither in free flow state or spillback state the link is said to be in congested state[2]. This means that one or more state changing shockwaves exist somewhere on the link. Note that the shockwaves associated with a change in traffic state take on the slopes of the lines drawn between the traffic states in the fundamental diagram. For instance, the traffic state at time t_6 is the result of interaction between three earlier traffic states, resulting in intersecting shockwaves (t_4 moving to free flow traffic state 3 and ending traffic state 2, while interacting with traffic state 4 that was introduced at t_5). For clarity a vehicle trajectory in terms of speed is depicted using a dashed line (e.g. $\vartheta_1, \vartheta_2, \vartheta_4$) to convey how vehicle speed relates to the existing (shock) waves. As can be observed the speed is defined by the slope between the origin and the current traffic state encountered.
As a final note on the congested state, the term can be slightly confusing to use from a link boundary perspective. Observe that the no free-flow state is not the same as congested (it can also be in spillback), while at the same time the no spillback state can also mean free flow (besides congested). Therefore we will refrain from using the term congested and use no free flow or no spillback instead, depending on the chosen perspective.

3.1 Cumulative vehicles and flow change moments

The model so far has been formulated in continuous time *t* (in contrast to the time discretised version of the link transmission model in Yperman, 2007). For our algorithm we are however only interested in the exact times changes in inflow or outflow rates occur. Let $\overline{t}_a^{(j)}$ therefore denote the (current) last event *j* at the downstream boundary of link *a*, i.e. a change in the outflow rate. In the same way, let $\underline{t}_a^{(i)}$ denote the (current) last event *i* at the upstream boundary of link *a*, i.e. a change in the inflow rate. The cumulative inflows and outflows are piecewise linear and can thus be expressed in reference to the changes in flow rates as shown in (11) and (12), where each piecewise linear section can be referenced via running variable *x* or *y* respectively.

$$U_a(t) = U_a\left(\underline{t}_a^{(i-x)}\right) + u_a\left(\underline{t}_a^{(i-x)}\right)\left(t - \underline{t}_a^{(i-x)}\right), \quad \underline{t}_a^{(i-x)} < t \le \underline{t}_a^{(i-x+1)} \wedge 0 \le x < i \tag{11}$$

$$V_a(t) = V_a\left(\overline{t}_a^{(j-y)}\right) + v_a\left(\overline{t}_a^{(j-y)}\right)\left(t - \overline{t}_a^{(j-y)}\right), \quad \overline{t}_a^{(j-y)} < t \le \overline{t}_a^{(j-y+1)} \wedge 0 \le y < j \tag{12}$$

We will now take a closer look at boundary conditions and how they will help in predicting (future) flow changes, utilising the introduced notation from this section.

3.2 Shockwave tracking vs link boundary conditions

In terms of defining a solution algorithm capable of exact results we could explicitly track shockwaves and impose flow changes when they arrive at a border. This however, results in a number of complications:

- How to find out where shockwaves will intersect?

[2] Observe that the only remaining combination of link boundary states; free flow on the one link boundary and spillback on the other border is physically impossible and can therefore be ignored as a viable traffic state

- How to verify if a shockwave is still valid or if it has become obsolete due to intersecting with another shockwave?
- How to keep track of all existing shockwaves?
- How to connect it (elegantly) to the traffic flow theory?

Instead it makes more sense to use the proposed mathematical framework and apply our knowledge on link boundary conditions to identify the exact moment a change in inflow or outflow rate will occur without the need for explicit shockwave tracking. To achieve this we will first derive the general formulation for a predicted (future) time $\overline{t}_a^{(j+1)}$ an outflow change occurs under or leading to free flow conditions. Utilising (5) and substituting the cumulative outflow with the just introduced piecewise linear formulation of (12) and doing the same for the cumulative inflow via (11) we are left with the decision what previous flow rate change relates to our input time *t*. Given that we are looking for $\overline{t}_a^{(j+1)}$ downstream it is easy to see that $\overline{t}_a^{(j-y)}$ can only revert to the event preceding $\overline{t}_a^{(j+1)}$, namely $\overline{t}_a^{(j)}$. Unfortunately this is not as trivial for the upstream situation. We only know it must hold that $\underline{t}_a^{(i-x)} < \left(\overline{t}_a^{(j+1)} - L_a/\gamma_a\right) \leq \underline{t}_a^{(i-x+1)}$ and due to piecewise linearity we choose $\underline{t}_a^{(i-x)}$ as out upstream reference time (do note that *x* is unknown at this point). We can now rewrite (5) in terms of $\overline{t}_a^{j+1}$ with adopted stationary inflow rate $u_a\left(\underline{t}_a^{(i-x)}\right)$ and stationary outflow rate $v_a\left(\overline{t}_a^{(j)}\right)$ yielding:

$$\overline{t}_a^{(j+1)} = \frac{U_a\left(\underline{t}_a^{(i-x)}\right) - V_a\left(\overline{t}_a^{(j)}\right) - u_a\left(\underline{t}_a^{(i-x)}\right)\left(\underline{t}_a^{(i-x)} + \frac{L_e}{\gamma_e}\right) + v_a\left(\overline{t}_a^{(j)}\right)\overline{t}_a^{(j)}}{v_a\left(\overline{t}_a^{(j)}\right) - u_a\left(\underline{t}_a^{(i-x)}\right)} \tag{13}$$

where it must hold that $\underline{t}_a^{(i-x)} + L_a/\gamma_a \leq \overline{t}_a^{(j+1)} < \underline{t}_a^{(i-x+1)} + L_a/\gamma_a$. It is important to notice that it is possible the correct relating upstream event is not necessarily the latest event $\underline{t}_a^{(i)}$ (as is demonstrated in Figure3) and also that due to shockwave interaction the up and downstream events do no need to be exactly L_a/γ_a apart, hence the fact that *x* in $\underline{t}_a^{(i-x)}$ is an unknown that we know exists but requires finding.
Along the same line of reasoning we can derive the (future) moment a spillback change occurs utilising the piecewise linear cumulatives formulations of (11) and (12) on the spillback traffic state formulation of (8).

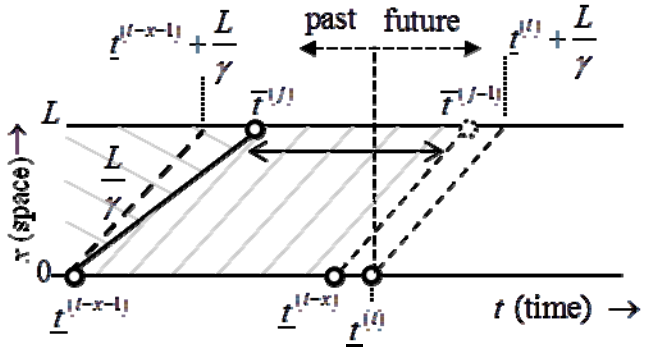


Figure 3- Schematic example of upstream and downstream flow rate change relation.

Upstream, due to piecewise linearity we can directly replace the unknown $\underline{t}_a^{(i-x)}$ from (11) with the previous event $\underline{t}_a^{(i)}$, because we know we're looking for $\underline{t}_a^{(i+1)}$, whereas downstream we have to keep the $\overline{t}_a^{(j-y)}$, with *y* remaining an unknown for now. This then gives us:

$$\underline{t}_a^{(i+1)} = \frac{K_a L_a - U_a\left(\underline{t}_a^{(i)}\right) + V_a\left(\overline{t}_a^{(j-y)}\right) + u_a\left(\underline{t}_a^{(i)}\right)\underline{t}_a^{(i)} - v_a\left(\overline{t}_a^{(j-y)}\right)\left(\overline{t}_a^{(j-y)} + \frac{L_a}{\omega_a}\right)}{u_a\left(\underline{t}_a^{(i)}\right) - v_a\left(\overline{t}_a^{(j-y)}\right)} \tag{14}$$

We will now extend this to G-LTM. The G-LTM can be formulated in a similar way albeit with more interdependencies.

3.3 Generalised link transmission model formulation

As G-LTM supports any concave fundamental diagram we have to revise our initial framework to support this. Let us consider the fundamental diagram as shown in Figure 4(a). Due to the fact that the fundamental diagram no longer needs to be linear; the speeds, wave speeds and shockwave speeds of a flow change within for example the free flow travel state no longer coincide (with $\gamma_a(0)$).This means that a vehicle exiting the link at t_0 did not by definition enter the link at $t_0 - L_a/\gamma_a(u_1)$ because of the quadratic nature of the chosen fundamental diagram. Instead it most likely entered later, as can be seen in Figure 4(b). The absolute difference in cumulative vehicles between the vehicle entering the link and $t_0 - L_a/\gamma_a(u_1)$ is defined by $\left|k_a^I(u_1)L_a\right|$. Furthermore, wave speed $\gamma_a(u_1)$ is now obtained via the tangent at inflow rate u_1.

$$k_a^I(q_a) = q_{I,a}^{-1} - \frac{q_a}{\gamma_a(q_a)} \tag{15}$$

$$k_a^{II}(q_a) = q_{II,a}^{-1} + \frac{q_a}{\omega_a(q_a)} \tag{16}$$

The density relating to a flow q_a in a free flow state is defined as $q_{I,a}^{-1}$ in (15), such that $k_a^I(q_a)$ represents the (negative) density of the discrepancy between vehicle speed and wave speed. Equation (16) does the same only for the spillback part of the fundamental diagram where $k_a^{II}(q_a) > K_a$, the wave speed $\omega_a(q_a)$ depends on flow q_a and $q_{II,a}^{-1}$ is the related spillback density. We will utilise this to generalise our earlier formulations.

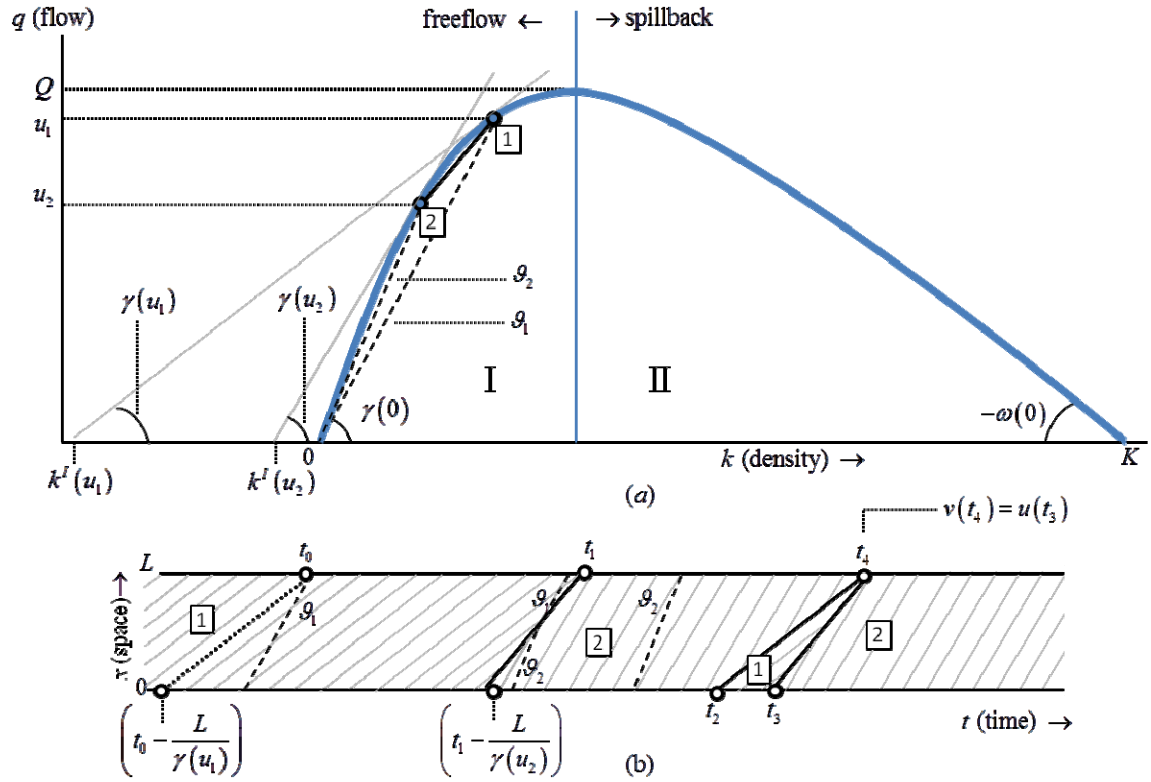


Figure 4 - (a) schematic general concave fundamental diagram depicting traffic state speeds, wave speeds (tangent) and shockwave speeds (b): Space time variant of depicted traffic state changes.

3.3.1 Generalising state and link boundary equations

Using this generalised form we are forced to redefine our free flow traffic state formulation presented in (5). Factoring in the density discrepancy and flow dependent γ_a leads us to (17). Note that we can either choose to define k_a^I and γ_a in terms of inflow rates or outflow rates at time $t - L_a/\gamma_a(\cdot)$ or t respectively. For now we choose outflow rates as this is more intuitive in terms of notation as it directly depends on t.

$$U_a\left(t - \frac{L_a}{\gamma_a\left(v_a(t)\right)}\right) - V(t) = k_a^I\left(v_a(t)\right)L_a \tag{17}$$

For the spillback traffic state formulation presented in (8) we incorporate the changes in a similar fashion leading us to (18).

$$U_a(t) - V_a\left(t - \frac{L_a}{\omega_a\left(u_a(t)\right)}\right) = k_a^{II}\left(u_a(t)\right)L_a \tag{18}$$

This in turn has an impact on the definition of desired sending and receiving flows as is shown in the updated versions of (7) and (10) leading to:

$$s_a(t) = \begin{cases} C_a, & \text{if } U_a\left(t - \dfrac{L_a}{\gamma_a\left(v_a(t)\right)}\right) - V(t) > k_a^I\left(v_a(t)\right)L_a, \\ v_a(t), & \text{otherwise.} \end{cases} \tag{19}$$

$$r_a(t) = \begin{cases} C_a, & \text{if } U_a(t) - V_a\left(t - \dfrac{L_a}{\omega_a\left(u_a(t)\right)}\right) < k_a^{II}\left(u_a(t)\right)L_a, \\ u_a(t), & \text{otherwise.} \end{cases} \tag{20}$$

We now turn back to the just updated (17) and (18) as they require a redefinition of the predicted flow change moments originally defined in equations (13) and (14). Starting with the prediction of the free flow traffic state we again are looking for $\overline{t}_a^{(j+1)}$ an outflow change occurs under free flow conditions. Due to piecewise linearity it is easy to see that when substituting (12) into (17), $\overline{t}_a^{(j-y)}$ will revert to $\overline{t}_a^{(j)}$ (the event preceding $\overline{t}_a^{(j+1)}$). For the opposite border $\underline{t}_a^{(i-x)}$ is the reference time. Because *x* is unknown at this stage $\underline{t}_a^{(i-x)}$ cannot be simplified. An additional problem is the fact that both k_a^I and tangent γ_a have been defined in terms of outflow rate but this outflow rate depends on $\overline{t}_a^{(j+1)}$ we're trying to obtain. To solve this we must rework k_a^I and γ_a to make it dependent on the inflow rate instead. Given that we already chose $\underline{t}_a^{(i-x)}$ as our (unknown) upstream reference point, we can conveniently utilise to achieve this. This then leads us to:

$$\overline{t}^{(j+1)} = \frac{-k^I\left(u\left(\underline{t}^{(i-x)}\right)\right)L + U\left(\underline{t}^{(i-x)}\right) - V\left(\overline{t}^{(j)}\right) - u\left(\underline{t}^{(i-x)}\right)\left(\underline{t}^{(i-x)} + \dfrac{L}{\gamma\left(u\left(\underline{t}^{(i-x)}\right)\right)}\right) + v\left(\overline{t}^{(j)}\right)\overline{t}^{(j)}}{v\left(\overline{t}^{(j)}\right) - u\left(\underline{t}^{(i-x)}\right)} \tag{21}$$

with,

$$\underline{t}^{(i-x)} + \frac{L}{\gamma\left(u\left(\underline{t}^{(i-x)}\right)\right)} \le \overline{t}^{(j+1)} < \underline{t}^{(i-x+1)} + \frac{L}{\gamma\left(u\left(\underline{t}^{(i-x+1)}\right)\right)}$$

where we dropped the reference to link *a* for brevity. We can obtain the spillback traffic state prediction in a similar fashion yielding (22). Now we are able to predict the future flow rate changes in a generalised form assuming stationary flows using equations (21) and (22). Observe that this new form is very similar to the simpler LTM model, but more versatile. The only aspect left to discuss on the link model part is the situation that occurs at t_4 in Figure4 (b), namely the situation seemingly leading to multiple solutions in terms of proposed flow rates for the same predicted future time t_4.

$$\underline{t}^{(i+1)} = \frac{k^{II}\left(v\left(\overline{t}^{(j-y)}\right)\right)L - U\left(\underline{t}^{(i)}\right) + V\left(\overline{t}^{(j-y)}\right) - u\left(\underline{t}^{(i)}\right)\underline{t}^{(i)} - v\left(\overline{t}^{(j-y)}\right)\left(\overline{t}^{(j-y)} + \dfrac{L}{\omega\left(v\left(\overline{t}^{(j-y)}\right)\right)}\right)}{u\left(\underline{t}^{(i)}\right) - v\left(\overline{t}^{(j-y)}\right)} \tag{22}$$

with,

$$\overline{t}^{(j-y)} + \frac{L}{\omega\left(v\left(\overline{t}^{(j-y)}\right)\right)} \le \underline{t}^{(i+1)} < \overline{t}^{(j-y+1)} + \frac{L}{\omega\left(v\left(\overline{t}^{(j-y+1)}\right)\right)}$$

3.3.2 Unique flow rate identification under multiple solutions

Consider two forward shockwaves, the first shockwave enters the link at t_2 as shown in Figure4 (b) and is predicted to exit at t_4. At time t_3 another forward shockwave with $u_3 < u_2$ occurs. This shockwave is also predicted to exit at exactly t_4. Consequently, there are now two valid $\underline{t}^{(i-x)}$ when solving (21) for t_4. The problem with this is not so much the time prediction (it is uniquely identified by t_4 even with two solutions), but which of the two inflow rates is dominant and should be chosen as the outflow rate for $t \geq t_4$ providing a unique and physically meaningful solution? It can be shown that from all solutions $\underline{t}^{(i-x)}$ the only correct solution adopts the minimum outflow rate (which is the same as the minimum $u\left(\underline{t}^{(i-x)}\right)$). This is due to the concave nature of the fundamental diagram and is known as the Newell-Luke Minimum Principle (for the exact description of this principle we refer to Newell 1993b and Gentile 2011). Do observe that this is exactly the same as choosing the maximum $\underline{t}^{(i-x)}$ from all eligible options. This principle also holds for the spillback state in case multiple waves coincide at the upstream link boundary.

4. Node model formulation

All inflow and outflow rates mentioned in the link model section are usually sanctioned by a node model. In this section we will formulate a class of first order node models that is suitable for our framework and proposed algorithm.

Besides the node model each DNL adopts a way of directing route flows on the network via splitting rates at the node level. We will propose splitting rates that depend on a period of stationary flow *z* and demonstrate that although this is a simplification of the reality it is adequate for the purpose of this paper. Let each node *n* have a number of in links part of set A_n^{in} as well as a set of out links denoted by A_n^{out}. We can then define the splitting rate $\varphi_{ab}(t)$ from in link *a* in the direction of out link b at time instant *t* as:

$$\varphi_{ab}(t) = \frac{\sum_{p \in P} \delta_{ap} f_p(z)}{\sum_{p \in P} \delta_{ap}\delta_{bp} f_p(z)}, \quad \mathrm{t} \in \left[T^{(z)}, T^{(z+1)}\right], a \in A_n^{in}, b \in A_n^{out}, n \in N, \tag{23}$$

with a path incidence indicator δ_{ap} taking on a value of 1 if path *p* contains link *a*, otherwise it reverts to 0. Finally, $f_p(z)$ is the travel demand on path *p* for the relevant period *z*. As can be seen the splitting rates are fixed per period instead of dynamically determined by the optimal route flows towards the destinations at every possible *t*. This is not ideal from a theoretical perspective. The reason to nevertheless adopt this approach is the fact that our focus in the remainder of this paper is on both the DNL algorithm and comparing it to the LTM solution algorithm and not so much on optimal route choice or ultimate realism in output. We do want to stress it would of course be easy to alter the model formulation of (23) to accommodate the path based approach, but this would require additional verbosity on our other link and node model notation that we chose to avoid.

On a more practical side, note most real world time discretised CTM implementations like METANET or MaDAM do not offer path based splitting rates as it does not scale to large networks, let alone in combination with en-route route choice (i.e. continuous route choice while traversing the route). The LTM does adopt the path based splitting rates in the theoretical

description, but as far as the authors of this paper are aware, no implementation of LTM exists that adopts this approach in practice (for large scale applications).

Most importantly, we would like to point out that this approach regarding splitting rates is not expected to have a significant positive or negative effect on the number of events generated compared to the path based approach[3]. The reason for this is the fact that the number of events will mainly determined by fluctuations of demand at the origin dictated by our periods of stationary demand. When we assume the average length traversed by the combined path flows does not vary too much between periods we can see that it does not matter if periodically initiated flow is directed in one direction or the other, in any case either events are generated, or not. Furthermore one might argue that the computational complexity and memory demands of the path based time dependent splitting rates are much higher in terms of the continuous (dis)aggregation of data and storing it separately. This of course is true, but this argument holds equally well for time discretised implementations of for example LTM and G-LTM, so comparisons as presented in our case studies can still be made as long as we make sure the assumptions imposed regarding the splitting rates are equal for all DNL models compared .
Continuing with our node model formulation we now define the desired sending flow for a specific turn $s_{ab}(t)$ as the offered outflow rate from link *a* to link *b* or more formally:

$$s_{ab}(t) = \varphi_{ab}(t) s_a(t), \quad a \in A_n^{in}, b \in A_n^{out}, n \in N \tag{24}$$

Our node model then takes in the desired sending flows per turn as well as the possible receiving flows and constructs the accepted turn based flows $v_{ab}(t)$ that jointly account for $v_a(t)$ and also lock the $u_b(t)$ at the same time. This node model can be formulated in a very generalised and turn based form as presented in equation (25).

$$\left[v_{ab}(t)\right]_{a \in A_n^{in}, b \in A_n^{out}} = \Gamma_n\left(s_{a'b'}(t), r_{b'}(t), \forall a' \in A_n^{in}, \forall b' \in A_n^{out}\right), \quad \forall \mathrm{n} \in \mathrm{N} \tag{25}$$

A special case arises when dealing with an origin node. In that case the transfer/generation of traffic flow is slightly different because it is directly linked to the path flow of the current period *z* as can be observed in (26)[4]

$$u_b(t) = \begin{cases} \sum_{p \in P} \delta_{bp} f_p(z) & ,\text{if } b \in A_o^{out}, \mathrm{o} \in O, \mathrm{t} \in \left[T^{(z)}, T^{(z+1)}\right], \\ \sum_{a \in A_n^{in}} v_{ab}(t) & ,\text{else } b \in A_n^{out}, n \in N. \end{cases} \tag{26}$$

Since we adopted a FIFO based link model in the form of the G-LTM we can only keep the link and node model consistent if our node model also satisfies FIFO. This constraint and a number of additional constraints were specified by Tampere et al. (2011) to obtain a generic class of first order node models. It also proposed a solution algorithm that we is adopted in the case studies of this paper as our node model of choice. In practice this means that if the resulting $v_a(t)$ is smaller than the original $s_{ab}(t)$, all $v_{ab}(t)$ are scaled back proportionally ensuring FIFO is never violated.

[3] As long as no continuous en-route route choice is adopted; which in real world applications on a macroscopic scale to date is a given.
[4] The underlying assumption here is that the combined path flows do not exceed the capacity of the origin out link.

5. Event Based Solution algorithm

As shown in the previous section we are able to predict a future change in flow rate at any time in the simulation regardless of the present state of the link. We also observed that the in and outflow cumulatives (and flow rates) remain piecewise linear even in G-LTM. This makes an event based algorithm a very promising approach because, in contrast to discretised (fixed) time step approaches, it can be exact. Furthermore, in case less flow changes occur than the number of fixed updates on the link in a time discretised model; the event based model requires less computational power and has a smaller memory footprint.

To make sure that we can exploit these advantages to the fullest but allow the user to decide on the balance between realism and computational complexity we propose a predefined number of fixed periods with stationary demand. We will also demonstrate that it is possible to reduce calculation time further when adopting a method of approximation. This extension is optional, but useful as is demonstrated in the real world case study.

5.1 Trigger events/release events

We will make a distinction between two types of events in our algorithm. We differentiate between trigger events and release events. A trigger event is defined as an event that triggers a flow rate change on a link as a result of the calculation of the node model (or an origin node flow, which is a special case). Note that trigger events can entail either a forward or backward shockwave depending on where the shockwave originated (at the upstream or downstream link boundary).

A release event is an event that marks the end of a shockwave when it arrives at a link boundary. Its source was an earlier trigger event. When positively validated it will enforce a flow rate change and possibly (but not necessarily) a link state change.

Each flow rate change at time *t* caused by the *l*-th processed event is denoted by $e_a^{(l)}(t) \in E_a$, with E_a being the overall set of processed events on link *a*. This set is the combination of its upstream and downstream sub-sets: $E_a = \underline{E}_a \bigcup \overline{E}_a$. A further distinction can be made on the event type; when $e_a^{(l)}(t) \in E_a^\varsigma$, with $E_a^\varsigma = \underline{E}_a^\varsigma \bigcup \overline{E}_a^\varsigma$ it denotes a trigger event (on one of the two link boundaries). Similarly, when $e_a^{(l)}(t) \in E_a^\rho$, with $E_a^\rho = \underline{E}_a^\rho \bigcup \overline{E}_a^\rho$ it denotes a release event.

Unprocessed but already scheduled triggered events will be made part of set $X^\varsigma = \underline{X}^\varsigma \bigcup \overline{X}^\varsigma$. Similarly, unprocessed (but future scheduled) release events are to be temporarily stored in set $X^\rho = \underline{X}^\rho \bigcup \overline{X}^\rho$.

5.2 Basic algorithm

The algorithm is in essence an infinite update process where each new trigger event can spawn possible release events. In turn, whenever a release event is classified as valid, it triggers an update on the node model via the boundary of the link it is accepted on. The node model, faced with a change in desired sending or receiving flow, computes the (possibly) updated accepted flow rates based on the change. The algorithm then generates new trigger events for the changed flows. These events are scheduled and processed and the whole system is set in motion again from the beginning.

Some of the basic algorithm steps will refer to a specific section discussing their internal process flow. Refer to those sections for a detailed description of the inner workings of this step in the algorithm.

Input: $u_a(0) = v_a(0) = 0, U_a(0) = V_a(0) = 0$ for all $a \in A$.

There are no events scheduled yet: $X^{\rho} = X^{\varsigma} = E = \varnothing$.

Step 0: Pre-schedule period based origin flow change trigger events:

$$e_b^{(l)}\left(T^{(z)}\right) = \sum_{p \in P} \delta_{bp} f_p(z), \quad b \in A_o^{out}, \forall o \in O, \forall z \in Z \text{ analogous to (26)}$$

All events are added to $\underline{X}^{\varsigma}$.

Step1: *Events present?*

In case $X^{\rho} = X^{\varsigma} = \varnothing$ finish algorithm, otherwise continue to Step2.

Step2: *Get the next earliest release event:*[5]

In case $X^{\rho} = \varnothing$, go to Step6.

Collect $e_a^{(l)}(t)$, with $t \le t^* \wedge l \le l^* \ \forall e_{a^*}^{(l^*)}(t^*) \in X^{\rho}$.

Step3: Classify validity (possibly update) the candidate release event, see section 5.7:

When $e_a^{(l)}(t)$ is classified as invalid go to Step8, otherwise proceed

Step4: Update node model, see section 5.6:

$X^{\rho} = X^{\rho} \setminus e_a^{(l)}(t)$

Step5: *Get the next earliest trigger event:*

In case $X^{\varsigma} = \varnothing$, go to Step8.

Collect $e_a^{(l)}(t)$, with $t \le t^* \wedge l \le l^* \ \forall e_{a^*}^{(l^*)}(t^*) \in X^{\varsigma}$.

Step6: Update link model see section 5.4:

$X^{\varsigma} = X^{\varsigma} \setminus e_a^{(l)}(t)$

Step 7: Go back to Step1

There are three sub algorithms to discuss. We will start with the link model update in Step6 as it is the most straightforward.

5.3 Update link model

A trigger event is by definition valid as it is a direct result of (usually) the node model. The main task of the update lies in the scheduling of a (possible) future release event related to the trigger.

Step6.1 Update link:

if $e_a^{(l)}(t) \in \underline{X}^{\varsigma}$ proceed to Step6.2a

else proceed to Step6.2b

Step6.2a Update link (upstream trigger event)

Update cumulatives according to (11) and set $u_a(t) = e_a^{(l)}(t)$

Update event set: $\underline{E}_a^{\varsigma} = \underline{E}_a^{\varsigma} \cup e_a^{(l)}(t)$

Proceed to Step6.3a

Step6.2b Update link (downstream trigger event)

Update cumulatives according to (12) and set $v_a(t) = e_a^{(l)}(t)$

Update event set: $\overline{E}_a^{\varsigma} = \overline{E}_a^{\varsigma} \cup e_a^{(l)}(t)$

Proceed to Step6.3b

[5] In practice Step2 and Step3 should be repeated until all release events scheduled at the exact same time *t* have been processed. Only then proceed to Step4. This to make sure the node model is provided with all relevant flow changes and only needs to be calculated once.

At this point $e_a^{(l)}(t)$ is accepted and becomes the most recent event on the relevant link boundary. Given that we now want to see if this trigger event leads to a release event on the opposite border we clearly should utilise the most recent event both up and downstream in the derived prediction equations (21) and (22), greatly simplifying the prediction process described in Step6.3:

Step6.3a Predict, validate future free flow release event for upstream trigger event:
Solve free flow state equation (21) with $\underline{t}_a^{(i-x)} = \underline{t}_a^{(i)} = t$ yielding a $\overline{t}_a^{(j+1)}$.
Verify $\overline{t}_a^{(j+1)}$: it must hold $\overline{t}_a^{(j+1)} \geq \underline{t}_a^{(i)} + L_a \Big/ \gamma_a\left(u_a\left(\underline{t}_a^{(i)}\right)\right)$.
When valid create: $e_a^{(l+1)}\left(\overline{t}_a^{(j+1)}\right) = e_a^{(l)}(t)$, otherwise proceed to Step6.4.
Schedule it: $\overline{X}^{\rho} = \overline{X}^{\rho} \bigcup e_a^{(l+1)}\left(\overline{t}_a^{(j+1)}\right)$ and exit sub-algorithm.

Step6.3b Predict, validate future free flow release event for downstream trigger event:
Solve spillback state equation (22) with $\overline{t}_a^{(j-y)} = \underline{t}_a^{(j)} = t$ yielding a $\underline{t}_a^{(i+1)}$.
Verify $\underline{t}_a^{(i+1)}$: it must hold $\underline{t}_a^{(i+1)} \geq \overline{t}_a^{(j)} + L_a \Big/ \omega_a\left(v_a\left(\overline{t}_a^{(j)}\right)\right)$.
When valid create: $e_a^{(l+1)}\left(\underline{t}_a^{(i+1)}\right) = e_a^{(l)}(t)$, otherwise proceed to Step6.4.
Schedule it: $\underline{X}^{\rho} = \underline{X}^{\rho} \bigcup e_a^{(l+1)}\left(\underline{t}_a^{(i+1)}\right)$ and exit sub-algorithm.

Step6.4 Recalculate eligible opposite border events, see section 5.5:

Figure5 shows two examples where there is no valid predicted release time for $e_a^{(l)}(t)$ due to shockwave interaction. However, even when the prediction is invalid; $e_a^{(l)}(t)$ itself is accepted and must have indirectly impacted another pending release event. Hence this other release event is eligible for an update via its original opposite border trigger event. This is what Step6.4 refers to and is discussed in the next section.

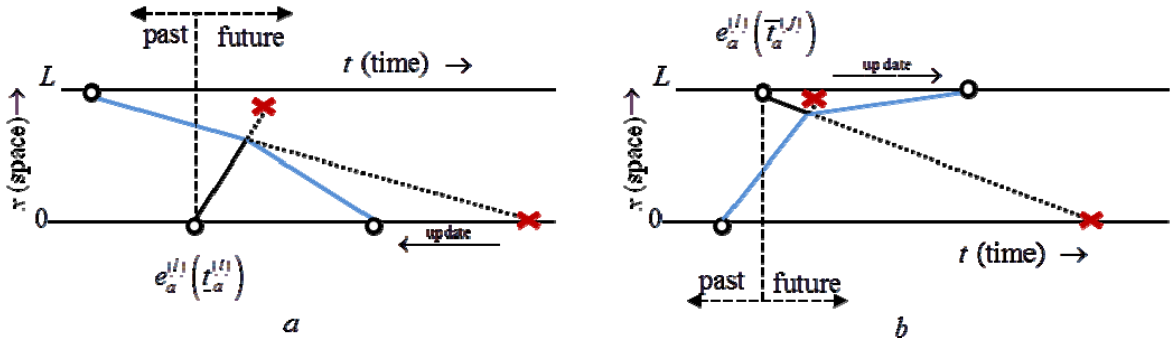


Figure5 - Examples of invalid direct predictions of expected free flow release events (a) instead causing spillback or (b) causing different moment of free flow traffic state arrival.

5.4 Recalculate eligible opposite border events

Whenever entering this sub-process we've established that there is additional information on the opposite border that could affect existing shockwaves (see Figure5 for examples). We need to re-evaluate all eligible pending release events. Eligibility is established via Step6.4.3(a/b) and is

illustrated (for an update on possible spillback release events) in Figure 6(a)[6]. In this figure all opposite border trigger events of a certain time t where it holds that $\underline{t}_a^{(i-1)} - L_a/\omega_a(Q_a) \le t_a \le \underline{t}_a^{(i-1)}$ could be impacted. This is due to the fact that a flow rate change in that period could cause shockwave interaction with $e_a^{(l)}\left(\underline{t}_a^{(i)}\right)$. The maximum of this catchment period is set to $L_a/\omega_a(Q_a)$ because we adopted a concave fundamental diagram. In this diagram Q_a gives us the least steep tangent (thus the slowest possible wave speed) that can cause an interaction between different spillback states. Moreover, observe that the first earlier opposite border event with $t_a < \underline{t}_a^{(i-1)} - L_a/\omega_a(Q_a)$ could still cause interaction with $e_a^{(l)}\left(\underline{t}_a^{(i)}\right)$ in case it is a link state changing shockwave (allowing an even less steep shockwave slope) and therefore also needs to be considered.

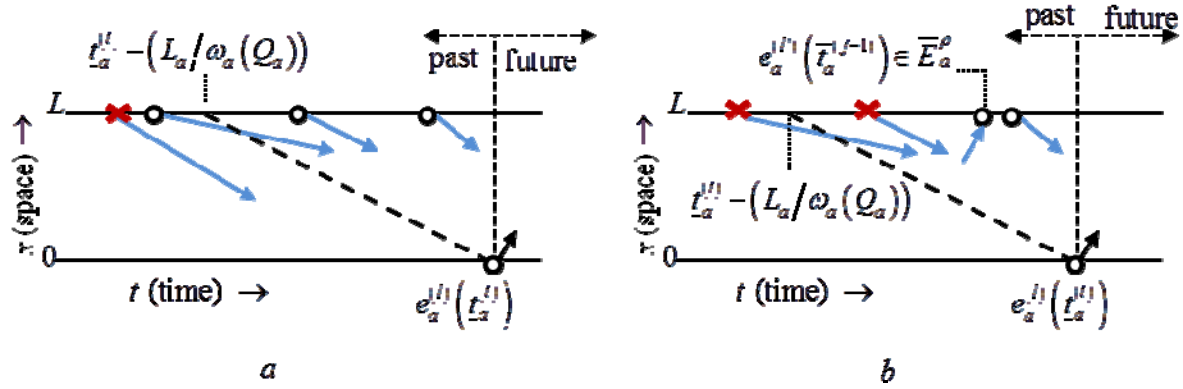


Figure 6 - Schematic representation of opposite border eligibility constraints in case of an inflow rate change represented by $e_a^{(l)}\left(\underline{t}^{(i)}\right)$ with (a) looking back one beyond catchment area and (b) due to earlier release event only partial consideration of catchment area.

Finally, note that in Figure6 (b) the release event $e_a^{(l')}\left(\overline{t}^{(j-1)}\right)$ cuts off two of the three triggers preventing interaction, leaving only the most recent one eligible, demonstrating an example of the second constraint placed on verifying eligibility in Step6.4.2(a/b).

Step6.4.1 Determine type of opposite border update given the current event :
if $e_a^{(l)}(t) \in \underline{E}_a^{\varsigma} \cup \underline{E}_a^{\rho}$ proceed to Step6.4.2a
else go to Step6.4.2b

Step6.4.2a Get most recent opposite border event for current upstream event :
Collect $e_a^{(m)}\left(\overline{t}_a^{(j-y)}\right)$, with $\overline{t}_a^{(j-y)} \ge t^* \wedge m \ge m^*$, $\forall e_a^{(m^*)}\left(t^*\right) \in \overline{E}_a^{\varsigma} \cup \overline{E}_a^{\rho}$
Proceed to Step6.4.3a

Step6.4.2b Get most recent opposite border event for current downstream event :
Collect $e_a^{(m)}\left(\underline{t}_a^{(i-x)}\right)$, with $\underline{t}_a^{(i-x)} \ge t^* \wedge m \ge m^*$, $\forall e_a^{(m^*)}\left(t^*\right) \in \underline{E}_a^{\varsigma} \cup \underline{E}_a^{\rho}$
Proceed to Step6.4.3b

Step6.4.3a Is opposite border (downstream) event eligible for an update?
$e_a^{(m)}\left(\overline{t}_a^{(j-y)}\right)$ is eligible when $\overline{t}_a^{(j-y+1)} > t - L_a/\omega_a(Q_a)$ and $e_a^{(m)}\left(\overline{t}_a^{(j-y)}\right) \in \overline{E}_a^{\varsigma}$,
If eligible go to Step6.4.4a otherwise exit sub-algorithm.

Step6.4.3b Is opposite border (upstream) event eligible for an update?

[6] Depending on the chosen fundamental diagram $L_a/\omega_a(Q_a)$ might vary or even be infinite (as is the case in our example fundamental diagram in Figure4).

$e_a^{(m)}\left(\underline{t}_a^{(i-x)}\right)$ is eligible when $\underline{t}_a^{(i-x+1)} > t - L_a/\gamma_a\left(Q_a\right)$ and $e_a^{(m)}\left(\underline{t}_a^{(i-x)}\right) \in \underline{E}_a^{\varsigma}$

If eligible go to Step6.4.4b otherwise exit sub-algorithm.

Step6.4.4a Predict and schedule future spillback release event time(if valid):

Solve spillback equation (20) with $\underline{t}_a^{(i-x)} = t$, and $\overline{t}_a^{(j-y)}$ yielding $\underline{t}^{(i+1)}$.

$$\underline{X}^{\rho} = \underline{X}^{\rho} \bigcup e_a^{(l+1)}\left(\underline{t}^{(i+1)}\right), \quad \text{if } \underline{t}^{(i+1)} > t$$

Update to next earlier opposite border event $y = y - 1$,

Return to Step6.4.3a

Step6.4.4b Predict and schedule future free flow release event time(if valid):

Solve free flow equation (19) with $\underline{t}_a^{(i-x)}$, and $\overline{t}_a^{(j-y)} = t$ yielding $\overline{t}^{(j+1)}$.

$$\overline{X}^{\rho} = \overline{X}^{\rho} \bigcup e_a^{(l+1)}\left(\overline{t}^{(j+1)}\right), \quad \text{if } \overline{t}^{(j+1)} > t$$

Update to next earlier opposite border event $x = x - 1$,

Return to Step6.4.3b

5.5 Update node model

This section describes Step4 of the basic algorithm in more detail. It utilises the (updated) in and/or outflow rate(s) in the node model computation resulting in new accepted inflow and outflow rates. In case multiple release events arrive at the same time *t* at a node *n*, the node model should be able to process them simultaneously. For brevity the sub algorithm presented in Step4 will focus on the single event situation only $e_a^{(l)}\left(t\right)$, but note that in case of multiple events it just means that Step4.1 and Step4.2(a/b) should be repeated; once for each event.

Step4.1 Update link:

If $e_a^{(l)}\left(t\right) \in \underline{X}^{\rho}, \quad a \in A_n^{out}, n \in N$ proceed to Step4.2a

else proceed to Step4.2b

Step4.2a Update link (Upstream release event)

Update cumulatives according to (11) and set $u_a\left(t\right) = e_a^{(l)}\left(t\right)$

Update event set: $\underline{E}_a^{\rho} = \underline{E}_a^{\rho} \bigcup e_a^{(l)}\left(t\right)$

Proceed to Step4.3

Step4.2b Update link (Downstream release event)

Update cumulatives according to (12) and set $v_a\left(t\right) = e_a^{(l)}\left(t\right)$

Update event set: $\overline{E}_a^{\rho} = \overline{E}_a^{\rho} \bigcup e_a^{(l)}\left(t\right)$

Step4.3 Run the node model:

Determine the new turn flows via

$$\left[v_{ab}\left(t\right)\right]_{a \in A_n^{in}, b \in A_n^{out}} = \Gamma\left(s_{a'b'}\left(t\right), r_{b'}\left(t\right), \forall a' \in A_n^{in}, \forall b' \in A_n^{out}\right), n \in N$$

Step4.4 *Create new trigger events:*

Whenever the inflow rate changed:

$$e_b^{(l+1)}\left(t\right) = \sum_{a \in A_n^{in}} \alpha_{ab}\left(t\right) s_{ab}\left(t\right) \quad , \sum_{a \in A_n^{in}} v_{ab}\left(t\right) \neq u_b\left(t\right) \forall b \in A_n^{out}, n \in N$$

and schedule it: $\underline{X}^{\varsigma} = \underline{X}^{\varsigma} \bigcup e_b^{(l+1)}\left(t\right)$.

Whenever the outflow rate changed:

$$e_a^{(l+1)}\left(t\right) = \sum_{b \in A_n^{out}} \alpha_{ab}\left(t\right) s_{ab'}\left(t\right) \quad , \sum_{b \in A_n^{out}} v_{ab}\left(t\right) \neq v_a\left(t\right) \forall a \in A_n^{in}, n \in N$$

and schedule it: $\overline{X}^{\varsigma} = \overline{X}^{\varsigma} \bigcup e_a^{(l+1)}(t)$.

exit sub-algorithm

5.6 Release event validation (and possible update)

In the previous section each $e_a^{(l)}(t) \in X^{\rho}$ considered is validated in Step3 of the algorithm. This section discusses Step3 in detail. Let Figure5 serve as a reminder why this validation process is important as it shows how a release event might become invalid over time. Note that this step is similar to Step6.4, but there are a few important differences. For example, we shall now utilise equations (21) and (22) for validation instead of prediction purposes. Also, during validation we will continuously keep track of the nearest valid future flow rate change $e_a^{(l+1)}\left(t_{\min}^{\rho}\right)$ computed. When done and $e_a^{(l+1)}\left(t_{\min}^{\rho}\right)$ matches $e_a^{(l)}(t)$ in time and flow rate, we can consider it valid. If it is not considered valid but $t_{\min}^{\rho} > t$ we can immediately reschedule the event to a future moment in time.

Step3.1 Determine type of release event:
Initialise $e_a^{(l+1)}\left(t_{\min}^{\rho}\right) = t_{\min}^{\rho} = \infty$.
If $e_a^{(l)}(t) \in \underline{X}^{\rho}$ proceed to Step3.2a,
Otherwise go to Step3.2b

Step3.2a Collect most recent downstream event:
Collect $e_a^{(m)}\left(\overline{t}_a^{(j-y)}\right)$, with $\overline{t}_a^{(j-y)} \geq t^* \wedge m \geq m^*$, $\forall e_a^{(m^*)}\left(t^*\right) \in \overline{E}_a^{\varsigma} \bigcup \overline{E}_a^{\rho}$
Proceed to Step3.3a

Step3.2b Collect most recent upstream event:
Collect $e_a^{(m)}\left(\underline{t}_a^{(i-x)}\right)$, with $\underline{t}_a^{(i-x)} \geq t^* \wedge m \geq m^*$, $\forall e_a^{(m^*)}\left(t^*\right) \in \underline{E}_a^{\varsigma} \bigcup \underline{E}_a^{\rho}$
Proceed to Step3.3b

Step3.3a Is current downstream event possibly compatible with our upstream event?
If $e_a^{(m)}\left(\overline{t}_a^{(j-y)}\right) > e_a^{(l)}(t) - L_a / \omega_a\left(e_a^{(l)}(t)\right)$ it is too recent, go to Step3.5a,
else if $\overline{t}_a^{(j-y+1)} > t - L_a / \omega_a(Q_a)$ and $e_a^{(m)}\left(\overline{t}_a^{(j-y)}\right) \in \overline{E}_a^{\varsigma}$, it is eligible go to Step3.4a,
otherwise go to Step3.6.

Step3.3b Is current upstream event possibly compatible with our downstream event?
If $e_a^{(m)}\left(\underline{t}_a^{(i-x)}\right) > e_a^{(l)}(t) - L_a / \gamma_a\left(e_a^{(l)}(t)\right)$ it is too recent, go to Step3.5b,
else if $\underline{t}_a^{(i-x+1)} > t - L_a / \gamma_a(Q_a)$ and $e_a^{(m)}\left(\underline{t}_a^{(i-x)}\right) \in \underline{E}_a^{\varsigma}$, it is eligible go to Step3.4b,
otherwise go to Step3.6.

Step3.4a Update minimum release event time:
Solve spillback equation (22) with $\underline{t}_a^{(i-x)} = t$, and $\overline{t}_a^{(j-y)}$ yielding $\underline{t}^{(i+1)}$.
If $\underline{t}^{(i+1)} \geq t$: update $e_a^{(l+1)}\left(t_{\min}^{\rho}\right)$ with $t_{\min}^{\rho} = \min\left(\underline{t}^{(i+1)}, t_{\min}^{\rho}\right)$
Proceed to Step3.5a

Step3.4b Update minimum release event time:
Solve free flow equation (21) with $\underline{t}_a^{(j-y)} = t$, and $\overline{t}_a^{(i-x)}$ yielding $\overline{t}^{(j+1)}$.
If $\overline{t}^{(j+1)} \geq t$: update $e_a^{(l+1)}\left(t_{\min}^{\rho}\right)$ with $t_{\min}^{\rho} = \min\left(\overline{t}^{(j+1)}, t_{\min}^{\rho}\right)$
Proceed to Step3.5b

Step3.5a *Get the next earlier downstream event:*
$y = y - 1$, go back to Step3.3a

Step3.5b *Get the next earlier upstream event:*
$x = x - 1$, go back to Step3.3b

At this stage all eligible events within the temporal boundaries of the opposite border catchment area of $e_a^{(l)}(t)$ have been processed. So we now can validate, update or discard current event $e_a^{(l)}(t)$.

Step3.6 *Validate current release event:*
When $t_{\min}^{\rho} = t \wedge e_a^{(l+1)}\left(t_{\min}^{\rho}\right) = e_a^{(l)}(t)$ event $e_a^{(l)}(t)$ is valid, go to Step3.8,
else if $t_{\min}^{\rho} = \infty$, $e_a^{(l)}(t)$ is discarded go to Step3.9,
otherwise $e_a^{(l)}(t)$ is not valid, but we found an updated future release time, go to Step3.7.

Step3.7 Schedule updated release event:

$$\underline{X}^{\rho} = \underline{X}^{\rho} \bigcup e_a^{(l+1)}\left(t_{\min}^{\rho}\right), \quad e_a^{(l)}(t) \in \underline{X}^{\rho},$$

$$\bar{X}^{\rho} = \bar{X}^{\rho} \bigcup e_a^{(l+1)}\left(t_{\min}^{\rho}\right), \quad \text{otherwise.}$$

Go to Step3.9

Step3.8 Recalculate eligible opposite border events, see section 5.5.

Step3.9 exit sub algorithm

In case the release event is marked valid in Step3.6, additional information has become available on the link (analogous to when a new trigger event is accepted when updating the link model, but no opposite border release event could be found (described earlier in Step6). In that case we allow for an update on pending release events as referred to in Step3.8. Figure7 (a) shows a situation where due to too little information at $t=t_0$ a predicted release at $\underline{t}^{(i+2)}$ proves invalid. It is only in Figure7 (b) after $\underline{t}^{(i)}$ is validated that we can correctly predict the next release event to $\underline{t}^{(i+1)}$ due to the additional information now present on the link border.

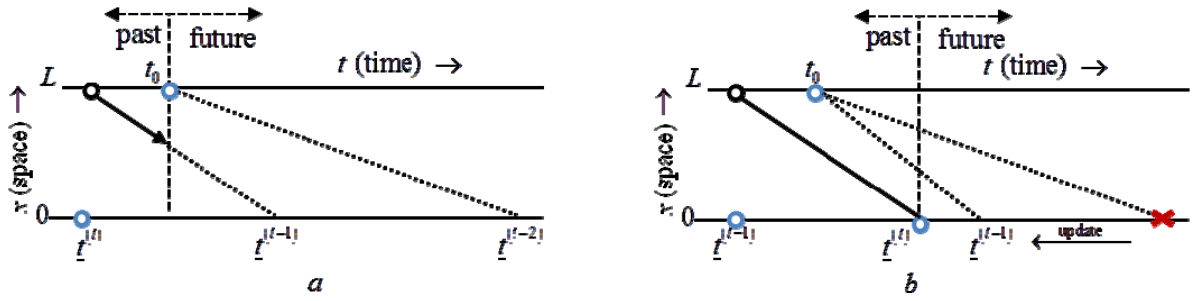


Figure7 - Example of the necessity of updating opposite border eligible trigger events after release event validation (a) original release $\underline{t}^{(i+2)}$ is generated, but (b) shows it is actually invalid and can only be correctly predicted after the validation of $\underline{t}^{(i)}$.

6. Compared to LTM Solution algorithm

Here we will compare our novel algorithm to the time-step based LTM Solution Algorithm (LTM-SA) as proposed by Yperman (2007). Given that we do not have access to an implementation of this algorithm we will only compare based on known properties regarding computational complexity.
LTM-SA can almost be considered a special (limited) case of G-LTM-SA. Both the node model and fundamental diagram(s) supported by LTM-SA can be adopted by G-LTM-SA. The fundamental difference between the two algorithms is found in the approach; G-LTM-SA is truly event driven whereas the LTM-SA is based on the more conventional time discretised approach.

6.1 Discrete time steps versus event based

As stated before, the LTM-SA is based on discretised time steps of a predefined interval Δt, resulting in a different link model updating procedure compared to our G-LTM-SA. The desired sending and receiving flows ultimately feeding the node model do not relate to flow rate changes directly, but instead focus on difference in cumulative vehicles within a fixed time window in combination with applying the Newell-Luke principle:

$$\begin{aligned} s_a(t) &= \min\left(Q_a \Delta t, U_a\left(t+\Delta t-\frac{L_a}{\gamma_a}\right) - V_a(t)\right) \\ r_a(t) &= \min\left(Q_a \Delta t, \mathrm{V}_a\left(t+\Delta t-\frac{L_a}{\omega_a}\right) + K_a L_a - U_a(t)\right) \end{aligned} \tag{27}$$

In effect this approach yields an average of the related opposite border flow(s) for a period Δt as it does not keep track of the exact flow rate changes.

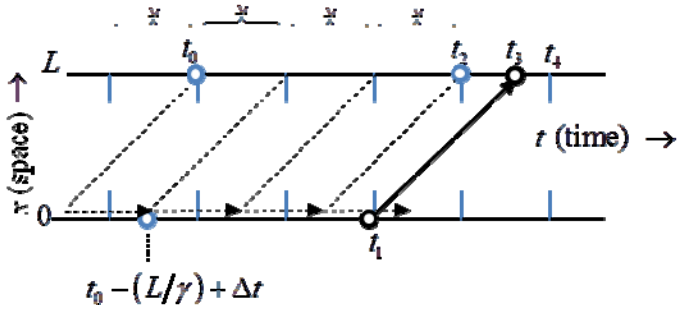


Figure 8 - Drawbacks of LTM-SA discretised time step algorithm, updates happen without additional information available and averaging out the effects flow changes (at t3).

As illustrated in Figure8, there are two main drawbacks to this approach. Firstly when the link is in a stable state, extrapolation of cumulatives would suffice due to piecewise linearity, instead unnecessary calculations take place every Δt. Secondly the (average) flow rate adopted for each next Δt might originally have consisted out of multiple different flow rates. For example at t_2 the new average flow rate over $[t_2 - L/\gamma, t_2 - L/\gamma + \Delta t]$ will be adopted. This sending flow rate is based on two different inflow rates. In reality the new flow rate will be adopted after t_3 .Thus, LTM-SA applies a flow rate change too early and with a different flow (average) than actually is taking place[7]. Finally, observe that all changes in flow that in reality happen within a Δt will

[7] Also note that with LTM-SA at t_4 another flow rate change is in order as now the link has stabilised and the flow of t_1 is adopted for the whole period.

snap to either end of a period so the change in flow at t_1 probably occurred at a different time on the upstream link, but this information indeed has already been lost.
LTM-SA does propose an "event-based" alternative, but this does not relate to the actual definition of events as known in the field of the computer sciences and agent based systems. Instead, it only aims to optimise the Δt period per node by making it as large as possible given the in link lengths of the node. This reduces the amount of calculations needed, but at the same time increases the error in the resulting approximation. It is therefore an optimisation that still exhibits the aforementioned drawbacks. Obviously, the G-LTM-SA in this example only requires two events in total (at t_1 and t_3) yielding both an exact as well as a correct result. Note that when mentioning G-LTM-SA we refer to the proposed novel event based approach and not the solution algorithm proposed by Gentile (2011), that solution algorithm exhibits the exact same problems as LTM-SA.

6.2 Fundamental diagram support

The LTM-SA also proposes an extension beyond the Newell triangular fundamental diagram boasting a more sophisticated piecewise linear version. In that case (27) is reformulated into the slightly more complicated (28). As can be seen LTM-SA now checks each possible piece wise linear option *h* or *h'* (available via set *H* for the free flow part and *H'* for the congested part respectively), using the corresponding wave speed ($\gamma_{h,a}$, $\omega_{h,a}$) and tangent density ($k^{I}_{h,a}$, $k^{II}_{h,a}$) to obtain the correct result. Observe that when the fundamental diagram is chosen in the continuous form of Figure4 (a) the number of elements in both sets becomes infinite and the LTM-SA approach is no longer feasible.

$$s_a(t) = \min\left(Q_a\Delta t, \min_{h\in H}\left(U_a\left(t+\Delta t-\frac{L_a}{\gamma_{h,a}}\right)-k^{I}_{h,a}L_a-V_a(t)\right)\right)$$
$$r_a(t) = \min\left(Q_a\Delta t, \min_{h'\in H'}\left(V_a\left(t+\Delta t-\frac{L_a}{\omega_{h,a}}\right)+k^{II}_{h,a}L_a-U_a(t)\right)\right) \quad (28)$$

7. Solution Algorithm Practicalities

In ascertaining the efficiency of G-LTM-SA we discuss two degrees of freedom: the number of periods with stationary flow within the simulation and the level of accuracy of the result. Both have an impact on the runtimes of the algorithm on large scale networks which is investigated in our case studies.

7.1 Periodic stationary flows in DNL

The more periods Z of stationary flow are defined the more accurate travel demand patterns can be simulated. On the other hand a large number of Z causes more fluctuations in flow rates, in turn leading to more events. Consequently, the event based approach gets less efficient compared to fixed time step algorithms such as LTM-SA the more periods are defined. To what extent this observation has a practical implication will be discussed in the case study of a large scale network.

G-LTM-SA offers the choice to run and guarantee exact results, resulting in no approximations at all. For large scale networks however this can potentially get inefficient. To make G-LTM-SA of practical relevance the user should be able to define the trade-off between exactness and computational efficiency. We will therefore include an event flow acceptance threshold. During algorithm execution this threshold is verified against every flow rate change proposed by the node model. Whenever the proposed flow rate change exceeds the threshold it will be processed, otherwise it is discarded. The bigger the threshold the fewer events occur and the smaller the computational effort required; this comes at the cost of the accuracy of the results. In

our large scale network we will experiment with different settings of this threshold to demonstrate the impact of this threshold in practice.

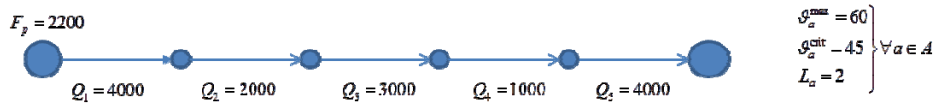
8. Case Studies

This chapter discusses two case studies: a synthetic corridor network and a real world network (city of Amsterdam). The first will serve as a demonstration of the principle working of the algorithm, whereas the second shows its applicability on larger scale networks. For G-LTM-SA we will adopt the Quadratic-Linear fundamental diagram as proposed by Bliemer et al. (2013) and shown in (29). This fundamental diagram has a quadratic free flow part and a linear congested part; providing a fair trade off between realism and computational efficiency.

$$q(k_a)=\begin{cases}\vartheta_a^{\max}\left(1-\dfrac{1}{k_a(Q_a)}\left(1-\dfrac{\vartheta_a^{\text{crit}}}{\vartheta_a^{\max}}\right)k_a\right)k_a, & \text{if } 0\le k_a\le k_a(Q_a),\\ C_a\left(1-\dfrac{k_a-k_a(Q_a)}{K_a-k_a(Q_a)}\right), & \text{if } k_a(Q_a)<k_a\le K_a,\end{cases} \tag{29}$$

8.1 Corridor Case Study

Let us consider the simple corridor network with a single one hour period of stationary flow depicted in Figure9 (a); links have a length of 2 km, a free speed $\vartheta_a^{\max}$ of 60 km/h and speed at capacity $\vartheta_a^{\text{crit}}$ of 45 km/h. The results of running G-LTM-SA, with a path flow demand of $F_p = 2200$, are presented in Figure9 (b). Note that the presented results are exact and all flow rate changes relate to an event on a one-on-one basis.



(a)

t_1	u_1	U_1	t_2	u_2	U_2	t_3	u_3	U_3	t_4	u_4	U_4	t_5	u_5	U_5
0	2200	0	136	2000	0	296	2000	0	436	1000	0	596	1000	0
3600	0	3000	2531	1000	1331	2043	1000	971	8356	0	2200	8516	0	2200
			5660	0	2200	6468	0	2200						

t_1	v_1	V_1	t_2	v_2	V_2	t_3	v_3	V_3	t_4	v_4	V_4	t_5	v_5	V_5
136	2200	0	296	2000	0	436	2000	0	596	1000	0	722	1000	0
136	2000	0	2043	1000	971	436	1000	0	8516	0	2200	8643	0	2200
2531	1000	1331	6468	0	2200	8356	0	2200						
5660	0	2200												

(b)

Figure 9 - (a) Corridor network with (b) single period G-LTM-SA simulation results.

Note that only 27 events[8] are needed. If we would compare this to LTM-SA with a rather generous fixed time step of let's say a 100 seconds, the difference is already $\left(2\cdot 5\cdot\left(3600/100\right)-24\right)=336$ calculations. With a more realistic time step of 10 seconds,

[8] Observe that the first outflow events on link 1 and 3 are redundant and are shown only for clarity: whenever a release event is accepted with a different flow than originally proposed we should only create the resulting trigger event with the accepted flow. Also there are three more release events that have been generated but became invalid during the simulation due to intersecting shockwaves and eventually were not accepted (all on link 1 and not shown in table)

causing less carry over accuracy problems, the difference becomes even more pronounced (3574 calculations).

8.2 Amsterdam Case Study

In this case study we will use the Amsterdam network kindly provided by the Amsterdam City Council (via Goudappel Coffeng B.V.). Although the network has signalised intersections they are ignored as this requires a more complex node model and this is beyond the scope of this paper. The G-LTM-SA is implemented as a separate DNL model in the StreamLine DTA framework (see Raadsen et al. (2010)) and the results are presented in OmniTRANS. We adopt all non-DNL components from this framework for our case study. Furthermore, this case study is not conducted to analyse the situation in Amsterdam. Instead, the aim is to provide insights in the complexity and practical feasibility of the algorithm on a representatively scaled real world network. We will conduct separate simulations for a number of different periods of stationary demand to assess the impact on performance. Clearly, the more periods we consider the more precise the DUE algorithm can direct traffic and the more realistic the results get. On the other hand each period with different flow rates will cause a wave of new events increasing the cost of running the simulation. Besides varying the number of periods we will also conduct runs experimenting with different flow rate epsilon thresholds. We do this to find out how this impacts both the performance and/or accuracy of the results. Based on this we will then draw some conclusions on the applicability of the G-LTM-SA.

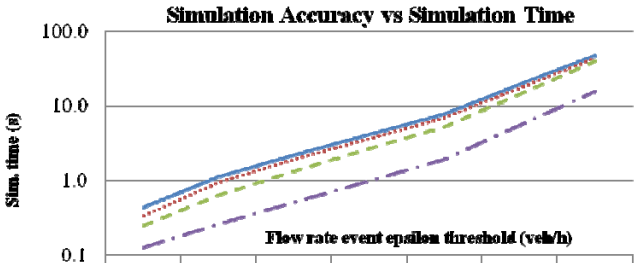
Network:
279 origins, 279 destinations:
2037 connect nodes
865 cross nodes
66 merge nodes
71 diverge nodes
3868 links
Routes:
174.279 routes from predefined route set
Simulation:
3 hours (10800 seconds)
OD Matrix:
Synthetic evening peak period demand
Route Choice:
MNL (spread=0.2 iteration 1, else spread=0.14)
MSA smoothing over iterations
Route Cost:
Trajectory travel time at t=6480s (60%)

Figure 10: Result of G-LTM-SA simulation run with 9 periods, and an event flow threshold epsilon of 5 veh/h. Shown are the average densities during period [100-110] minutes.

Figure11 shows the various runtimes (in seconds) of G-LTM-SA on the Amsterdam network[9]. Figure12 shows the same results but now plotted against the number of generated events[10]; it is clear that the computation time relates in a linear way to the number of events generated.

[9] 32 bit C++ MS Visual Studio 2008 SP1 running on desktop: 16GB RAM, Intel Xeon 3.1GHz, Windows7 SP1

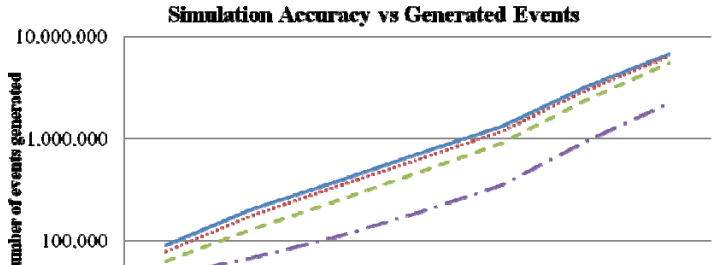
[10] This includes all events (even if they eventually proved to be invalid), as this takes up computation time. Therefore it would be unfair to only compare against valid/accepted events.



	100	40	20	10	5	2	1
9 Periods	0.4	1.1	2.2	4.1	7.8	19.3	46.8
6 Periods	0.3	0.9	1.9	3.6	6.9	17.4	43.1
3 Periods2	0.2	0.6	1.3	2.6	5.2	14.2	39.3
1 Period	0.1	0.3	0.5	1.0	1.9	5.4	15.3

Figure 11 - Impact of various flow rate thresholds on different stationary flow periods expressed in simulation time for a single iteration (average taken over ten iterations)

It can also be concluded that with a linear increase of the flow rate epsilon the computational complexity decreases exponentially (logarithmic scale used on the vertical axis in Figures 11 and 12). Finally we observe that increasing the number of periods leads to an increase of simulation time as well. However, this effect is much less pronounced than the influence of the flow rate threshold epsilon.



	100	40	20	10	5	2	1
9 Periods	90,222	200,474	370,489	703,803	1,316,908	3,208,535	6,763,693
6 Periods	78,749	175,207	332,195	620,645	1,176,905	2,928,946	6,374,484
3 Periods	63,312	127,421	242,135	464,353	906,706	2,389,044	5,485,792
1 Period	45,292	67,658	106,872	187,184	353,262	933,943	2,221,254

Figure12 - Impact of various flow rate thresholds on different stationary flow periods expressed in generated (taken from reference iteration ten).

Assuming a minimum flow rate threshold of just a 1 (veh/h) results in almost exact results, we will demonstrate how well the results hold up when decreasing the accuracy (thus increasing the flow rate epsilon). The most detailed (nine period) simulation run was taken as a reference and "lost vehicle hours" were used as a measure of accuracy. This seems a natural choice to be able to compare performance on a network level. The results are depicted in Figure13 (a). As expected the "lost vehicle hours" decrease over iterations as we're moving towards equilibrium.

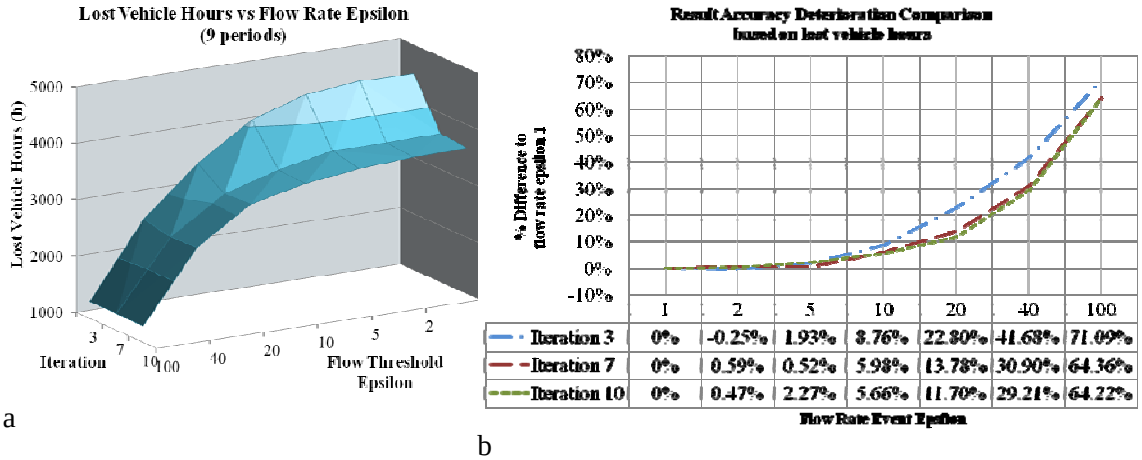


Figure 13 - (a) Comparing result validity under different flow rate epsilons and iterations via lost vehicle hours. (b) Percentual difference (of different flow rate epsilons) compared to 1veh/h flow rate epsilon.

Furthermore, result accuracy deteriorates rapidly beyond a flow rate epsilon of 10 (veh/h). Observe, that adopting a flow rate epsilon of 5 (veh/h) seems a very good trade off between computation speed and accuracy (only 0.5-2.2% difference in lost vehicle hours as depicted in Figure13 (b)).

The most important observation however is the fact that significant computational benefits can be achieved with near negligible effects on the result validity. Unfortunately, we cannot compare this G-LTM-SA to the LTM-SA directly as we do not have an implementation available in the current StreamLine framework. Instead, we will take our most detailed but complex simulation run of nine periods of stationary flow with a flow rate epsilon of 5 (veh/h) and compare the number of generated events to the number of calculations needed in the LTM-SA. It is hard to say if this is truly fair as updating events in G-LTM-SA also takes some time while on the other hand for LTM-SA we have little idea of the approximation error it generates, the impact of having to do many more node update computations, the appropriate time step and the fact that it adopts a simpler fundamental diagram. Nevertheless, we think it is an interesting thought experiment to at least give a rough idea on how the two relate in terms of computational complexity.

Given that no link on our network exceeds 0.3km and quite a few are less than 0.1km, we probably have to assume a time step of no longer than 2 seconds for LTM-SA. In that case the LTM-SA would then require $\left(2 \cdot 3868 \cdot \left(10800/2\right)\right) = 41,774,400$ updates versus the G-LTM-SA's 1,316,908 generated events. We therefore can conclude that G-LTM-SA only requires about 3.1% of the number of updates compared to LTM-SA. Moreover, G-LTM-SA has the advantage of the more realistic QL fundamental diagram over a triangular Newell diagram which adds value in terms of result realism.

9. Conclusion and future research

In this paper we proposed a novel solution algorithm for the Generalised Link Transmission Model (G-LTM) that is unique in a number of ways. Due to the (truly) event based nature it is capable of generating exact results for any concave fundamental diagram adopted on the link model, as far as we know it is the only algorithm in the literature that can do this. All existing (efficient) solution algorithms always yield approximate results at best. Like G-LTM it sports a generic node model making it flexible and (hopefully) future proof as one can adopt any node model that is deemed suitable.

Besides being precise the G-LTM-SA, due to its events based nature, can be made very efficient in terms of run times allowing it to simulate an iteration of a medium to large networks such as

the Amsterdam model within a matter of seconds. A comparison based on computational complexity properties indicate that the G-LTM-SA should be orders of magnitude faster than for example the existing LTM-SA which adopts an even simpler fundamental diagram. That said; further research is needed to obtain conclusive empirical evidence on how much faster it actually is compared to existing first order algorithms as we unfortunately had no implementation of the LTM-SA at our disposal at the time of writing.

By supporting an option to let go of exact results via a proposed flow rate change threshold property, the efficiency of the algorithm can be increased even further. In our case study we conclusively showed that when this threshold is chosen sensibly the price one pays in terms of result accuracy can be kept to a minimum while boosting the performance by a factor 5 to 10.
Realism of simulations is increased when embedded with precise departure time flow patterns. This is supported by introducing periods of stationary flow. The effects of such periods on run times have been assessed. The Amsterdam case study shows that while adding more periods does have a negative impact on run times, this effect is far less pronounced than we initially expected. Especially in combination with the aforementioned threshold it can become very subtle indeed, this of course is good news in terms of practicality.

In order to make a completely sound comparison between G-LTM-SA and LTM-SA and possibly other (second order) models we should have them at our disposal within the same framework. This would be the fairest way to objectively compare the run times, efficiency and memory usage. Furthermore when comparing such models the same fundamental diagram should be used. So G-LTM-SA should adopt a triangular or piece-wise linear diagram to correctly compare results to LTM-SA.

Another observation we can make is the fact that with the increase of the flow rate change threshold the lost vehicle hours drop (Figure12 (a)), this can be explained by the fact that when traffic builds up, it builds up less with a bigger threshold, consequently it gets less congested and we have less lost vehicle hours. This is an unwanted effect in terms of realism, so it might prove interesting to find a different way of defining this threshold or keep track of the error once en event is ignored and compensate for this.

We also found that one of the reasons of the exponentially increasing simulation time is the scheduling of eventually invalid release events and the cost of evaluating all of them in due time. It would be very interesting to do some research on finding a way to identify invalid future scheduled events in a simpler way than running them through the full validation process. This could potentially reduce computation times drastically especially on large networks when running them in concert with small flow rate thresholds.

References

Aw A., Rascle M. (2000). Resurrection of "Second Order: Models of Traffic Flow. SIAM Journal on Applied Mathematics. Vol. 60 issue 3, February-March 2000, pp 916-938.

Bliemer M.C.J. (2007). Dynamic Queuing and Spillback in an Analytical Multiclass dynamic Network Loading Model. Transportation Research Record 2029. 2007, pp14-21.

Bliemer M.C.J., Raadsen M.P.H., Smits E-S., Zhou B., Bell M.G.H. (2013). Capacity constrained stochastic static traffic assignment with residual point queues incorporating a proper node model. ITLS Working Paper.

Corthout R., Flotterod G., Viti F., Tampere C.M.J. (2012). Non-Unique Flows in Macroscopic First-Order Intersection Models. Transportation Research, Part B: methodological. Volume 46B issue 3, March 2012, pp343-359.

Daganzo C.F. (1993). A simplified theory of kinematic waves in highway traffic, Part II: Queuing at freeway bottlenecks. Transportation Research, Part B: methodological. Volume 27B issue 4, August 1993 pp289-303.

Daganzo C.F. (1995a). Requiem for second-order fluid approximations of traffic flow. Transportation Research, Part B: methodological. Volume 29B issue 4, August 1995 pp277-286.

Daganzo C.F., (1995b). The Cell Transmission Model, Part II: Network Traffic. Transportation Research, Part B: methodological. Volume 29B issue 4, August 1995 pp79-93.

Flötteröd G., Rohde J. (2011). Operational Macroscopic Modeling of Complex Urban Road Intersections. Transportation Research, Part B: methodological. Volume 45B issue 6, July 2011, pp903-922.

Gentile G. (2011). The General Link Transmission Model for Dynamic Network Loading and a comparison with the DUE algorithm. In book: New Developments in Transport Planning: Advances in Dynamic Traffic Assignment, Chapter 8. April 2011.

Jin W.L., Zhang H.M. (2002). On the Distribution Schemes for Determining Flows Through a Merge. Transportation Research, Part B: methodological. Volume 37B issue 6, July 2003 pp521-540.

Jin W.L. (2012). A kinematic wave theory of multi-commodity network traffic flow. Transportation Research, Part B: methodological. Volume 46B issue 8, September 2012 pp1000-1022.

Lebacque, J. Koshyaran, M. (2005). First Order Macroscopic Traffic Flow Models: Intersections Modeling, Network Modeling. Transportation and Traffic Theory. Flow, Dynamics and Human Interaction. 16th International Symposium on Transportation and Traffic Theory. 2005.

Lighthill M., Whitham G., (1955). On Kinematic waves II: A theory of traffic flow on long crowded roads. Proceedings of the royal society of London, Part A 229 (1178), May 1955, pp281-345.

Long J., Gao Z., Szeto W.Y. (2011). Discretised Link Travel Time Models Based on Cumulative Flows: Formulations and Properties. Transportation Research Part B: Methodological. Volume 45 issue 1, January 2011, pp232-254.

Messmer A., Papageorgiou M., (1990). METANET: a macroscopic simulation program for motorway networks, Traffic Engineering and Control. Volume 31 issue 9, 1990, pp466–470.

Newell G.F. (1993a). A Simplified Theory of Kinematic Waves In highway Traffic, Part II: Queuing At Freeway Bottlenecks. Transportation Research Part B: methodological. Volume 27B issue 4, 1993, pp289-303.

Newell G.F. (1993b). A Simplified Theory of Kinematic Waves In highway Traffic, Part I: General Theory. Transportation Research Part B: methodological. Volume 27B issue 4, 1993, pp281-287.

Papageorgiou M., 1998. Some remarks on macroscopic traffic flow modelling. Transportation Research part A: policy and practice. Volume 32 issue 5, September 1998, p323-329.

Payne H.J., Models of Freeway Traffic and Control (1971), Mathematical Models of Public Systems, Simulation Councils Proceedings. Volume 1 issue 1, 1971, pp. 51-61.

Payne H.J., Thompson W.A., Isaksen L., (1973). Design of a traffic-response control system for a Los Angeles freeway. Systems, Man and Cybernetics, IEEE Transactions on. Volume 3 issue 3, May 1973, pp. 213-224.

Raadsen M.P.H., Mein H.E., Schilpzand M.P., Brandt F, (2010). *Implementation of a single dynamic traffic assignment model on mixed urban and highway transport networks including junction modelling*. DTA Conference, Takayama Japan, July 2010.

Richards P.I. (1956), Shock waves on the highway, Operations. Research. Volume 4, 1956, pp42-51.

Szeto W.Y., Lo H.K., (2005). Properties of Dynamic Traffic Assignment with Physical Queues. Journal of the Eastern Asia Society for Transportation Studies, Volume 6, 2005, pp2108 – 2123.

Tampere. C.M.J., Corthout R., Cattrysse D., Immers, L.H. (2011). A Generic Class of First Order Node Models for Dynamic Macroscopic Simulation of Traffic Flows. . Transportation Research Part B: methodological. Volume 45B issue 1, 2011, pp289-309.

Yperman I. (2007). The LinkTransmission Model for Dynamic Network Loading. PhD Thesis Katholieke Universiteit Leuven, June 2007.