TECHNICAL REPORT (UPC-DAC-RR-GEN-2020-1), NOVEMBER 2020

THE LOGISTIC QUEUE MODEL

FRANCO COLTRARO, MARC RUIZ, AND LUIS VELASCO

ABSTRACT. In this work we present a novel continuous queuing model called the *logistic queue model*. We show that in terms of queue size and outflow prediction our model is as precise as a discrete one with the additional advantage of speed in simulations. We prove mathematically that our proposed model has all the theoretical properties one should expect, e.g. positivity of queue, FIFO property. Finally, in contrast with other continuous models –e.g. *Vickrey's point-queue model*– our model can be easily integrated numerically and moreover it allow us to naturally explore multiple extensions to more general scenarios such as: finite queues, multiple servers, priority queues, etc.

CONTENTS

1. INTRODUCTION

The expected explosion of services offered in the cloud has revealed new paradigms in telecommunication networks. Traffic forecasts in [1] and [2] reveal that the exchange of data between datacenters (DCs) will reach 905 exabytes per year by 2019, which is induced by an increasing IP traffic that will be mainly coped by video services (80% of total IP traffic by 2019). Aiming at interconnecting DCs and services in DCs with end-users, connectivity from the transport network is required to carry traffic flows between network nodes, each grooming a mix of different services.

From the network perspective, transport networks are currently configured with big static fat pipes based on capacity over-provisioning aiming at guaranteeing traffic demand and Quality of Service (QoS). Cloud-ready transport network [3] was introduced as an architecture to handle the dynamic cloud and network interaction. The evolution towards cloud-ready transport networks (telecom cloud) is based on intelligent control architectures with applications requests for end-to-end connectivity provisioning (e.g. the

ABNO architecture standardized by the IETF [4]) and elastic data planes that can satisfy cloud requirements efficiently for both network and cloud operators [5].

The optimization of telecom cloud infrastructures includes network planning and reconfiguration by means of various mathematical and computational techniques [6]. Among them, dynamic network simulation is of paramount importance in order to evaluate the performance of new services and applications over the network. To that end, one of the key elements in the simulation is how to generate and model network traffic. Traffic generation is a useful technique that enables studying and evaluating the network performance through simulation, when real traffic traces are not available. Authors in [7] presented traffic generators to inject realistic traffic flows in telecom cloud simulated systems. Basically, different type of functions like piecewise linear, polynomial or trigonometric sine summation are used to model traffic flow average profiles that can be periodical and evolutionary, e.g. incremental. Besides, traffic is characterized by a random function around that average.

Simulations based on flow-based models present interesting features such as optimum efficiency and easiness of traffic parameter characterization. However, the mix of services that could be aggregated into a flow could be too much heterogeneous and variable to allow a good characterization following such a simple approach. Moreover, interesting outputs that could be measured in simulation such as end-to-end latency or node switching delay cannot be accurately obtained due to the inherent nature of flow-based models that hides any individual service behavior.

An alternative approach is to generate and simulate traffic at the packet level, i.e. characterizing the packet generation of an individual service and running a discrete event simulation that process the transition of the packet from source to destination to every intermediate node [8]. It is worth noting that the amount of information that could be obtained from a discrete event simulation based on a packet-based traffic generator is unbeatable. However, when high income bitrates e.g. in the order of dezens of Gb/s per flow, the computational cost of processing the resulting huge number of packets is prohibitive even for small networks. To this aim, in the last years several research works have focused on providing alternative simulation environments allowing a fine granular view of the system comparable with that of packet-based simulations with that much more efficiency and scalability of flow-based ones.

One of the most successful approaches for doing so has been the use of *fluid flow models* [9]. Basically the idea is to consider only changes in rates of traffic flows. This can result in large performance advantages, though information about the individual packets is lost. For this reason also hybrid models have been developed [10]. A fluid simulator records the changes in the fluid rate in the source and the queue, while a packet simulator records the events of all the packets in the system. The abstraction takes place when, packet flows with little *time slots* separations are considered to be in the same fluid flow with a constant fluid rate. Little time variations among packets are not considered, and in this way the number of events is reduced. Of course, a critical issue of this kind of models is how to choose the time slots when one is given an arbitrary flow. Moreover, recording the changes in the fluid rate can still be seen as being discrete in nature.

In the end, that type of models can be seen as discrete variations of the famous *Vickrey's point-queue model* [11]. This is a purely *continuous* model –we do not have to worry about time slots– formulated through a differential equation. It has been throughly studied, being one of the most notable works the one done in [11]. In that article the authors find an explicit solution to the previous differential equation and with that formula at hand they are able to prove many desirable properties of the point-queue model, e.g. positivity of the queue size and the FIFO property. Nevertheless there are some issues with the model. The most obvious one is that the right hand side of the differential equation is not continuous and hence in general there do not exist classical solutions. This is computationally and numerically problematic since we can not apply usual ODE integrators. Nevertheless it is worth to note that the authors give in [12] an easily applicable numerical algorithm. The main problem that remains is generalization. Since the numerical algorithm they derive relies in the exact solution they found, we can not expect to have numerical schemes for more general cases as: finite queues or priority queues.

In this work we present a novel model to solve the aforementioned problems. This new model, which we call the *logistic queue model* can be seen as a smooth formulation of the point-queue model. This is important, since in such a way we are able to use usual numerical integrators. We prove mathematically that our proposed model has all the theoretical properties one should expect:

1: Positivity of the queue.
2: Asymptotic behaviour: the queue gets empty if the inflow does not overflow.
3: FIFO property: the system satisfies a first in first out discipline.

Moreover, we validate the model comparing it with a discrete event simulator. This way we show that for many purposes our model is as precise as a discrete one with the advantage of speed in simulations. We compare simulation times and conclude that the logistic queue model is several orders of magnitude faster than a discrete one. Finally in contrast with the point-queue model our model allow us to easily explore multiple extensions to more general scenarios such as: finite queues, multiple servers, priority queues, etc.

2. Theoretical model

2.1. **Derivation.** Assume we have a system with an inflow of entities arriving to a server of constant speed μ with an infinitely long queue. In probabilistic terminology the system is denoted as $G/D/1$. We will derive an equation that relates the number of entities in queue C with the inflow to the system f and the outflow g.

Note that each quantity has the following units:

1. $[\mu]$ = entities/time.
2. $[C]$ = entities.
3. $[f] = [g]$ = entities/time.

We will assume that the amount of entities in queue $C(t)$ in the instant t is a real number. This will be a reasonable approximation if the inflow is large enough: $f \gg 1$, physically this means that *many* entities arrive to the system. Also, this will make us not to distinguish between entities in queue and entities in the system. Therefore, for a short

FIGURE 1. Example of a system with a queue and a server of speed μ.

period of time dt the amount of entities that have arrived to the system is approximately $f(t) \cdot dt$ and likewise the number of them that have abandoned it is $g(t) \cdot dt$. This gives us the following conservation law:

$$C(t + dt) = C(t) + f(t) \cdot dt - g(t) \cdot dt,$$

which in the limit when $dt \to 0$ reads:

$$C'(t) = f(t) - g(t). \tag{1}$$

2.2. **Outflow.** We shall assume that there is a functional relationship between f, g and C, and express the outflow as a function of the inflow and the queue size:

$$g(t) = g(f, C) = \mu + e^{-\alpha C(t)} \left[\min\{\mu, f(t)\} - \mu\right]. \tag{2}$$

where:

1. $\mu > 0$ is maximum outflow rate of the server.
2. α is a positive parameter taken to be $\frac{\rho}{\mu}$ where $\rho = \frac{\lambda}{\mu}$ is the average intensity,

$$\lambda = \tfrac{1}{t_1 - t_0} \int_{t_0}^{t_1} f(t)dt$$

is the mean inflow and $[t_0, t_1]$ is the interval of definition of f.

Remarks 2.1 Note that by construction we have:

1. If at t the queue is empty, i.e. $C(t) = 0$, then the outflow is just $\min\{\mu, f(t)\}$.
2. If the inflow is bigger than the maximum capacity, i.e. $f(t) > \mu$, then the outflow is just μ.
3. If $C(t) \to +\infty$ then the outflow tends to μ.

It is instructive to have a qualitative idea of how the outflow depends of the queue size. In Figure 2 we make a plot of $g = g(f_0, C_0)$ for a fixed time t_0 and varying $C_0 = C(t_0)$. In the picture we are assuming that $f(t_0) = 0.5$. It is important to stress that this figure reflects *all possible* outflows we may get at a fixed instant of time t_0 depending on the queue size C_0 we have. ■

Finally we put together (1) and (2) in the following definition:

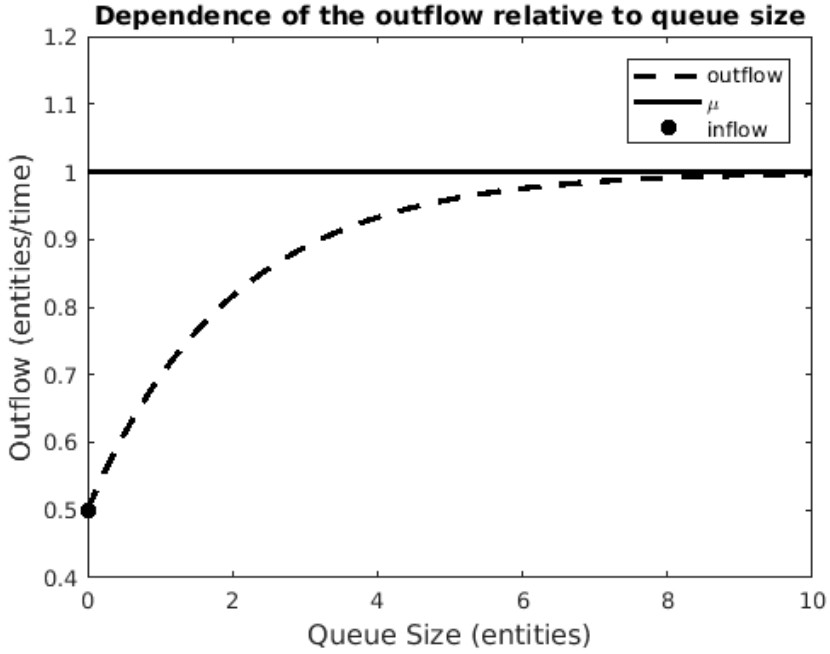

FIGURE 2. Possible outflows we may get at a fixed instant of time t_0 depending on the queue size C_0.

Definition 2.2 (**Logistic Queue Model**). Given an initial condition $C(t_0) = C_0$, a continuous and positive inflow function $f : [t_0, +\infty) \to [0, +\infty)$ and a maximum outflow rate $\mu > 0$, the *logistic queue model* is the following ordinary differential equation:

$$\begin{cases} C'(t) &= f(t) - \left[\mu + e^{-\alpha C(t)} \left(\min\{\mu, f(t)\} - \mu\right)\right] \text{ for } t > t_0, \\ C(t_0) &= C_0. \end{cases} \tag{3}$$

Remark 2.3 If we let $\alpha \to +\infty$ we get the famous *Vickrey's point-queue model*:

$$C'(t) = f(t) - \begin{cases} \min\{\mu, f(t)\} & \text{if } C(t) = 0, \\ \mu & \text{if } C(t) \neq 0. \end{cases} \tag{4}$$

Note that the right hand side of the ODE is not continuous in C. This causes many computational and theoretical difficulties. In the next section we will prove that the logistic queue model has all good properties of the point-queue model discussed in the introduction with the additional advantage that it is easy to integrate numerically because the ODE is smooth.

2.3. **Properties.** In this section we prove theoretical properties of the model. Let's start with existence, uniqueness and positivity.

Proposition 2.4 (**Existence, Uniqueness and Positivity**). *Given an initial condition $C(t_0) = C_0 \geq 0$, a continuous and positive inflow function $f \geq 0$ and a maximum outflow rate $\mu > 0$, we have that there exist a unique continuous differentiable solution $C(t)$ to*

the system (3) defined in an interval $[t_0, T_{max})$ *for* $T_{max} \leq +\infty$*. Moreover such solution is always greater or equal than zero.*

Proof. Existence, uniqueness and differentiability of $C(t)$ in an interval $[t_0, T_{max})$ for $T_{max} \leq +\infty$ is an easy consequence of the fact that the right hand side of (3) is a smooth function of the variable C and it is continuous in t, hence we can apply *Cauchy-Lipschitz theorem*. Let us prove now positiveness. Assume that for some time $t_- > t_0$ we have that $C(t_-) < 0$. Then, since $C(t)$ is continuous and $C_0 \geq 0$, there must be a t_+ and a t_* such that $t_+ < t_* < t_-$ where $C(t_+) \geq 0$ and $C(t_*) = 0$. See Figure 3. Also, after reducing the interval $I = [t_+, t_-]$ if necessary, we may assume that $C(t)$ is strictly decreasing there. Therefore we have:

$$C'(t) \leq 0 \text{ in } I = [t_+, t_-].$$

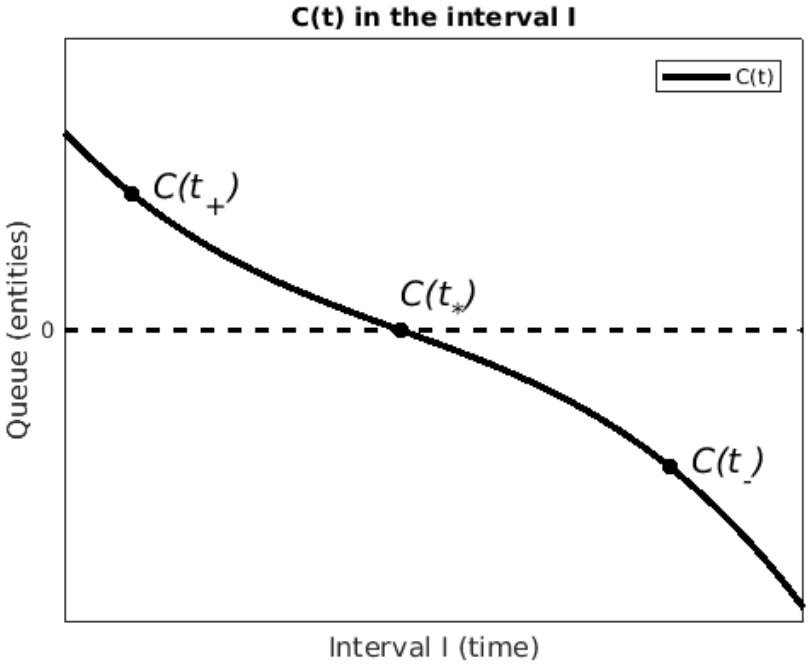

FIGURE 3. C is decreasing in $[t_+, t_-]$.

Let's see how this implies that $f(t) \leq \mu$ in I. Indeed if $f(t) > \mu$ for some $t \in I$ we would have using (3) that $C'(t) = f(t) - \mu > 0$, which is a contradiction. Hence we deduce that $\tilde{C}(t) \equiv 0$ is a solution of

$$\begin{cases} Q'(t) &= f(t) - \left[\mu + e^{-\alpha Q(t)} \left(\min\{\mu, f(t)\} - \mu\right)\right] \text{ for } t \in I, \\ Q(t_*) &= 0. \end{cases} \tag{5}$$

Which is impossible because we would have two different solutions C and $\tilde{C}$ of (5) in I. □

Remark 2.5 As a consequence of the previous proposition we deduce that if the queue starts empty, i.e. $C(t_0) = 0$, and the inflow is always smaller than the maximum outflow rate:

$$f(t) < \mu \text{ for all } t \geq t_0,$$

then the queue remains empty for all times and the outflow is equal to the inflow.

Now we are ready to prove some asymptotical properties of the queue:

Proposition 2.6 (**Asymptotical behaviour**). *Let $C(t_0) = C_0 \geq 0$ be an initial condition, $f \geq 0$ a continuous and positive inflow function and $\mu > 0$ a maximum outflow rate. Then the solution of (3) is defined for all times, i.e. it is defined in the interval $[t_0, +\infty)$. Moreover if the inflow satisfies:*

$$f(t) \leq f_\infty < \mu \text{ for all } t \geq t_f \tag{6}$$

for some time $t_f > t_0$ then

$$\lim_{t \to +\infty} C(t) = 0$$

exponentially fast.

Proof. Note that we can rewrite (3) as:

$$C'(t) = f(t) - \mu + e^{-\alpha C(t)} \left(\mu - \min\{\mu, f(t)\}\right), \tag{7}$$

but since $C(t) \geq 0$ we have that $e^{-\alpha C(t)} \leq 1$, and moreover $\mu - \min\{\mu, f(t)\} \leq \mu$. Hence we deduce that:

$$C'(t) \leq f(t) - \mu + \mu = f(t),$$

so finally we get that:

$$C(t) \leq C_0 + \int_{t_0}^{t} f(s)ds.$$

Therefore $C(t)$ is defined for all times since f is continuous and hence integrable. Assume now that f satisfies (6). Then, using (7), we get:

$$\begin{aligned} C'(t) &= (1 - e^{-\alpha C(t)}) \cdot (f(t) - \mu) \\ &\leq (1 - e^{-\alpha C(t)}) \cdot (f_\infty - \mu), \quad \text{for all } t \geq t_f. \end{aligned}$$

Now, it is well known that the exponential satisfies:

$$e^{\alpha C} = \sum_{n=0}^{\infty} \frac{(\alpha C)^n}{n!} \geq 1 + \alpha C, \quad \text{for all } C \geq 0.$$

Hence,

$$e^{-\alpha C} \leq \frac{1}{1 + \alpha C} \Leftrightarrow 1 - e^{-\alpha C} \geq 1 - \frac{1}{1 + \alpha C} = \frac{\alpha C}{1 + \alpha C}.$$

Now, since $(f_\infty - \mu) < 0$ we get the following inequality:

$$C' \leq \frac{\alpha C \cdot (f_\infty - \mu)}{1 + \alpha C}.$$

Calling $\beta = \mu - f_\infty > 0$ and rearranging terms we obtain:

$$(1 + \frac{1}{\alpha C}) \cdot C' \leq -\beta \Leftrightarrow \frac{d}{dt}\left(C + \frac{\log C}{\alpha}\right) \leq \frac{d}{dt}\left(-\beta t\right).$$

Therefore, integrating from t_f to t,

$$C(t) + \frac{\log C(t)}{\alpha} \leq -\beta t + k, \quad \text{for all } t \geq t_f$$

where $k = C_f + \frac{\log C_f}{\alpha} + \beta t_f$ and $C_f = C(t_f)$. Applying the exponential in both sides of the inequality one gets:

$$C^{1/\alpha} \cdot e^{C} \leq e^{-\beta t + k}. \tag{8}$$

Finally, last equation implies that

$$C(t) \leq e^{\alpha(k-\beta t)}, \quad \text{for all } t \geq t_f.$$

So the queue gets empty exponentially fast as stated. □

Remark 2.7 Equation (8) gives us an estimate of how fast the queue goes to zero. Assuming (6) and given a fixed tolerance $\epsilon > 0$, if we impose that:

$$C(t) \leq e^{\alpha(k-\beta t)} \cdot e^{-\alpha C(t)} \leq \epsilon,$$

then we find that the **emptying time** $T_\epsilon(C_f) \geq t_f$ needed to empty the queue within a tolerance $\epsilon > 0$ satisfies the following bound:

$$T_\epsilon(C_f) \leq t_f + \frac{C_f - \epsilon}{\mu - f_\infty} + \frac{1}{\alpha \cdot (\mu - f_\infty)} \cdot \log\left(\frac{C_f}{\epsilon}\right). \tag{9}$$

Note that the previous equation has the following expected *physical* properties:

1. The fastest way to empty the queue is by stopping the inflow, i.e. taking $f_\infty = 0$. On the other hand, if $f_\infty \to \mu$ then the bound goes to infinity as expected.
2. If $\alpha \to +\infty$ then we get that the bound is equal to the one of the *point-queue model*.
3. The bound is optimal because if we take $\epsilon = C_f$ then we deduce that $T_\epsilon(C_f) = t_f$.

Example 2.8 (**Constant inflow**). Take $f(t) \equiv f_\infty < \mu$, some $C_0 > 0$ and $t_0 = 0$. Then we have that (3) reduces to:

$$\begin{cases} C'(t) &= f_\infty - \left[\mu + e^{-\alpha C(t)}\left(f_\infty - \mu\right)\right] \text{ for } t > 0, \\ C(0) &= C_0. \end{cases} \tag{10}$$

This can be easily integrated, so we get the following explicit formula:

$$C(t) = \frac{1}{\alpha} \log\left(1 + k e^{\gamma t}\right) \tag{11}$$

where $\gamma = \alpha \cdot (f_\infty - \mu) < 0$ and $k = e^{\alpha C_0} - 1 > 0$. From the formula above is clear that $C(t) \geq 0$ and that $\lim_{t \to +\infty} C(t) = 0$ as expected.

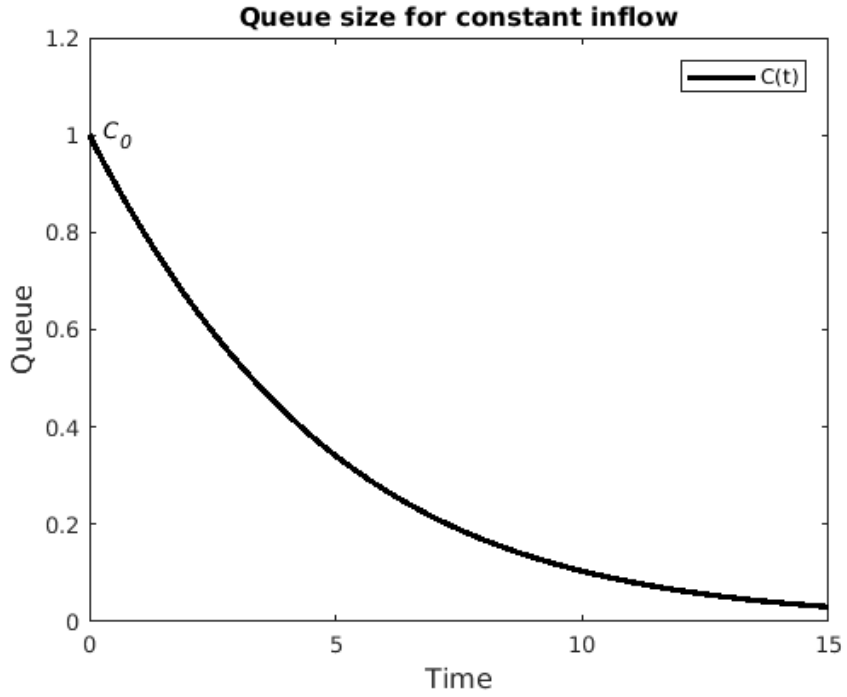

FIGURE 4. Plot of analytical solution (11) for $\mu = 1$, $f_\infty = 0.5$, $\alpha = \frac{1}{2}$ and $C_0 = 1$.

Let's now compare the bound we got in previous remark for the emptying time:

$$\hat{T}_\epsilon(C_0) = \frac{C_0 - \epsilon}{\mu - f_\infty} + \frac{1}{\alpha \cdot (\mu - f_\infty)} \cdot \log\left(\frac{C_0}{\epsilon}\right), \tag{12}$$

with the exact one we get with the analytical solution:

$$T_\epsilon(C_0) = \frac{1}{\alpha \cdot (\mu - f_\infty)} \cdot \log\left(\frac{e^{\alpha C_0} - 1}{e^{\alpha\epsilon} - 1}\right). \tag{13}$$

Note that we have the following:

1. $\hat{T}_\epsilon(C_0) \geq T_\epsilon(C_0)$ as we already know.
2. In the limit:
$$\lim_{\alpha \to +\infty} T_\epsilon(C_0) = \frac{C_0 - \epsilon}{\mu - f_\infty}.$$
So we recover again the the emptying time of the *point-queue model*.
3. When both C_0 and ϵ are small, we have that $\frac{e^{\alpha C_0} - 1}{e^{\alpha\epsilon} - 1} \approx \frac{C_0}{\epsilon}$. Hence T and $\hat{T}$ only differ by the linear term $\frac{C_0 - \epsilon}{\mu - f_\infty}$.

To conclude this example we make a plot of T and $\hat{T}$ with respect to C_0 and for $\mu = 1$, $f_\infty = 0.5$, $\alpha = \frac{1}{2}$ and $\epsilon = 0.05$.

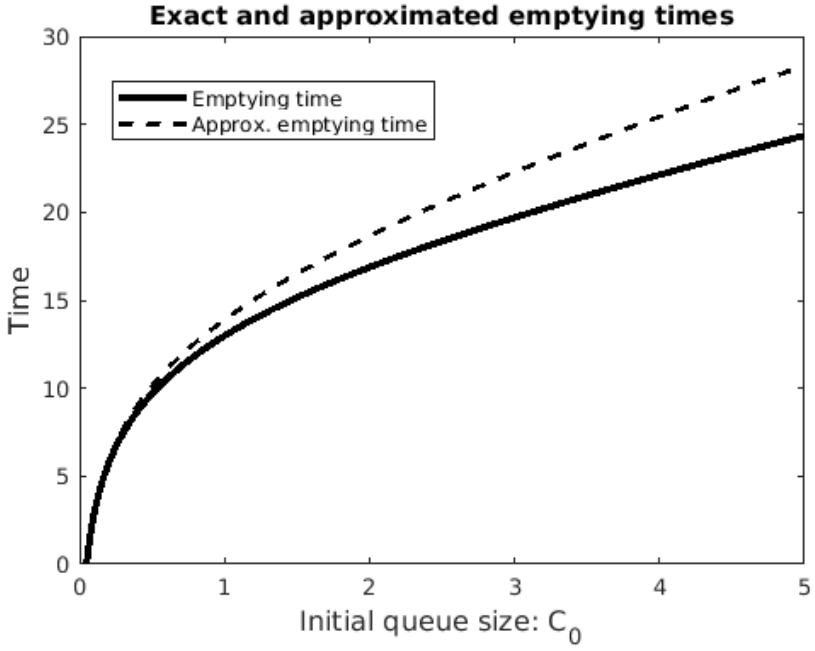

FIGURE 5. Comparison of T_ϵ and $\hat{T}_\epsilon$ for $\mu = 1$, $f_\infty = 0.5$, $\alpha = \frac{1}{2}$ and $\epsilon = 0.05$.

To end this section we will show that our model satisfies the FIFO (first-in first-out) property, this is, that the entity that arrives first to the queue is also the one that gets out first. To accomplish this, we introduce the **exit time** of an entity that has arrived in the instant t as:

$$\Lambda(t) = t + \frac{C(t)}{\mu}. \tag{14}$$

Remark 2.9 The rationale behind this definition is the following: $C(t)/\mu$ has units of time and it represents approximately the amount of time needed for emptying the queue if no more entities arrived after t.

We have the following proposition:

Proposition 2.10 (**FIFO property**). *Let $C(t_0) = C_0 \geq 0$ be an initial condition, $f \geq 0$ a continuous and positive inflow function and $\mu > 0$ a maximum outflow rate. Then*

$$\text{If } t < s \text{ we have that } \Lambda(t) < \Lambda(s).$$

In other words the exit time of an entity that has arrived in the instant t is smaller than other that arrives at $s > t$.

Proof. We have that $\Lambda(t) < \Lambda(s)$ is equivalent to:

$$\frac{C(s) - C(t)}{s - t} > -\mu. \tag{15}$$

Since C is differentiable, by the *Mean-Value theorem* we know that there exists a $\xi \in (t, s)$ such that

$$C'(\xi) = \frac{C(s) - C(t)}{s - t}.$$

Now, we distinguish two cases:

1. $f(\xi) < \mu$. There are two sub-possibilities:
   1.1 $C(\xi) = 0$. In this case $C'(\xi) = f(\xi) - f(\xi) = 0 > -\mu$, so (15) holds.
   1.2 $C(\xi) > 0$. In this case the outflow g satisfies $0 < g(\xi) < \mu$, hence
   $$C'(\xi) = f(\xi) - g(\xi) \geq -g(\xi) > -\mu.$$
2. $f(\xi) \geq \mu$. In this case $C'(\xi) = f(\xi) - \mu > -\mu$, so (15) holds.

□

2.4. **Variations.** In this section we present variations of the model that allow us to address more general scenarios than the one assumed until now. Namely:

- The case in which we have a probabilistic distribution on service times, i.e. a G/G/1 system.
- The case in which we have a finite queue.
- The case in which we have k servers, i.e. a G/D/k system.
- The case in which two flows join in a server and then they separate following different paths.
- The case in we have priority queues.

Note that we may also have combinations of the previous cases.

2.4.1. ***Variable service times***. Until this point we have assumed that μ is constant. Nevertheless it is very easy to see that all results that we have proved are still valid even when $\mu = \mu(t)$ depends on time as long as $\mu(t) \geq 0$ for all t.

2.4.2. ***Finite queue***. Assume we are only able to store a finite number of entities $C_{max} > 0$ in queue. We would like to add this restriction to the model. The idea is to annihilate the inflow when the queue size overflows the maximum storage capacity. Ideally we would replace f for $\tilde{f}$ where:

$$\tilde{f}(C(t), t) = H(C(t)) \cdot f(t)$$

and H is a *Heaviside function*:

$$H(C) = \begin{cases} 1 & \text{for } C \leq C_{max}, \\ 0 & \text{for } C > C_{max}. \end{cases}$$

The only problem with this approach is that we would be introducing discontinuities to C' and thus we would not be able to apply existence theorems and numerical integration schemes. Hence, we use the *logistic function* as a smooth approximation of H:

$$H_k(C) = \frac{1}{1 + (\frac{1}{H_0} - 1) \cdot e^{k(C - C_{max})}}$$

where $H_k(C_{max}) = H_0 > 0$. Note that:

$$\int_{-\infty}^{+\infty} (H(x) - H_k(x))^2 dx = \int_{-\infty}^{C_{max}} (1 - H_k(x))^2 dx + \int_{C_{max}}^{+\infty} (H_k(x))^2 dx$$
$$= 2 \cdot \int_{C_{max}}^{+\infty} (H_k(x))^2 dx \leq \frac{1}{k}.$$

Hence $H_k \to H$ as $k \to +\infty$ in the L^2 norm. Therefore we obtain the following *logistic*

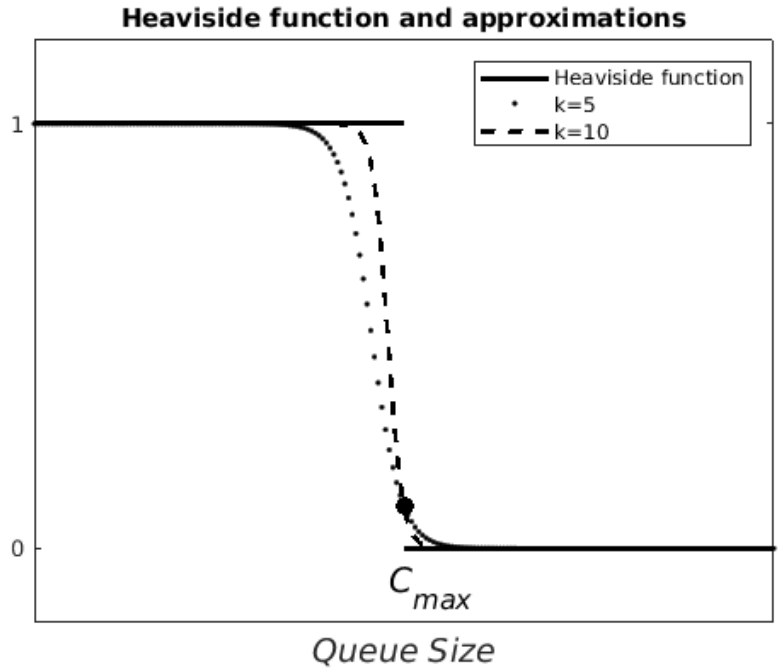

FIGURE 6. Approximation of the Heaviside function by the logistic function.

finite-queue model:

$$\begin{cases} C'(t) & = \tilde{f}(C(t), t) - \left[\mu + e^{-\alpha C(t)} \left(\min\{\mu, \tilde{f}(C(t), t)\} - \mu\right)\right] \text{ for } t > t_0, \\ C(t_0) & = C_0. \end{cases} \tag{16}$$

where

$$\tilde{f}(C(t), t) = H_k(C(t)) \cdot f(t).$$

Theorem 2.11. *Let $C(t_0) = C_0 \geq 0$ be an initial condition, $f \geq 0$ a continuous and positive inflow function, $\mu > 0$ a maximum outflow rate and $C_{max} > 0$ a maximum queue capacity. Then we have that the system (16) satisfies the following properties:*

1. *There exists a unique an positive solution defined for all times.*
2. *If the inflow is strictly smaller than the maximum outflow rate then the queue gets empty exponentially fast.*
3. *FIFO: the exit time of an entity that has arrived in the instant t is smaller than other that arrives at $s > t$.*

4. *If $C_0 < C_{max}$ and the inflow satisfies that*

$$f(t) \leq M_f \text{ for all } t \geq t_0 \text{ for some } M_f > 0, \tag{17}$$

then there exists a $H_0 > 0$ and a $k > 0$ such that H_k approximates H as precisely as desired (in the L^2 sense) and moreover

$$C(t) \leq C_{max} \text{ for all } t \geq t_0.$$

Proof. Properties 1,2 and 3 follow because the right hand side of (16) is *locally Lipschitz continuous* so we can apply again existence and uniqueness, and afterwards repeat exactly the same arguments given before. Let's prove property number 4. Assume there is a $t' > t_0$ such that $C(t') > C_{max}$. Since $C(t_0) < C_{max}$, there must exist a t_* such that $C(t_*) = C_{max}$. We shall assume that in a sufficiently small interval I around t_* the function $C(t)$ is strictly increasing, so that $C'(t) > 0$ for $t \in I$. Analogously as in the proof of Proposition 2.4 we must have that $\tilde{f} > \mu$ in I. Hence

$$\begin{aligned} C'(t_*) &= \tilde{f}(C(t_*), t_*) - \mu = H_0 \cdot f(t_*) - \mu \\ &\leq H_0 \cdot M_f - \mu = 0 \end{aligned}$$

where we have taken $H_0 = \frac{\mu}{M_f}$. Whence we get a contradiction, so the queue remains bounded by C_{max} for all times. □

2.4.3. ***Multiple servers***. The key modification now is to make $\mu = \mu(C(t))$ to depend on the queue size. Assume we have k servers of speed μ_0. We take:

$$\mu(C(t)) = \begin{cases} \mu_0 k & \text{if } C(t) \geq k-1, \\ \mu_0 \left(1 + C(t)\right) & \text{if } C(t) \leq k-1. \end{cases} \tag{18}$$

2.4.4. ***Separation of flows***. Assume f_1 and f_2 are two inflows that join in a system with a server and a queue. They are processed in that system but afterwards they follow different paths. Denote by g_1 and g_2 the corresponding outflows. We can obtain them with a very simple argument. If we denote by $f = f_1 + f_2$ and process this inflow, we get an aggregated outflow g. Then for each t we should have that:

$$\frac{f_1(t)}{f(t)} = \frac{g_1(t)}{g(t)},$$

hence

$$g_1(t) = \tfrac{g(t)}{f(t)} \cdot f_1(t).$$

Likewise,

$$g_2(t) = \tfrac{g(t)}{f(t)} \cdot f_2(t).$$

Note that we have that $g_1(t) + g_2(t) = g(t)$ as expected.

2.4.5. ***Priority queues***. Assume again f_1 and f_2 are two inflows that join in a system with a server of fixed speed μ. The only difference is that this time we assume that the flow f_1 has a priority over f_2, i.e. entities coming from f_1 will be served first than entities from f_2. We will model separately the queues formed by each flow:

$$\begin{cases} C_1'(t) &= f_1(t) - \left[\mu_1 + e^{-\alpha C_1(t)} \left(\min\{\mu_1, f_1(t)\} - \mu_1\right)\right], \\ C_2'(t) &= f_2(t) - \left[\mu_2 + e^{-\alpha C_2(t)} \left(\min\{\mu_2, f_2(t)\} - \mu_2\right)\right], \end{cases} \tag{19}$$

where we impose that $\mu_1 + \mu_2 = \mu$. The key idea now is to take:

$$\mu_2(C_1(t)) = \frac{f_2(t)}{f(t)} \mu e^{-\alpha C_1(t)}$$

where $f = f_1 + f_2$. Also we take $\mu_1(C_1(t)) = \mu - \mu_2(C_1(t))$. Observe that:

1. If $C_1 = 0$ then $\mu_1 = \frac{f_1}{f}\mu$ and $\mu_2 = \frac{f_2}{f}\mu$ so each speed is proportional to its flow weight.
2. If $C_1 \to +\infty$ then $\mu_1 \to \mu$ and $\mu_2 \to 0$ so just the priority one entities are served and the others are not.

2.5. **Applying the model.** When we apply the logistic model with *real data* there are several considerations that must be taken into account. We discuss some of them in the following.

2.5.1. ***Discrete inflow***. As already stated the inflow must be given to the logistic model in a continuous fashion. However in practice we only see discrete entities arriving to the system. Hence we need to approximate our ideal continuous inflow. In order to do this, we simply count how many entities have occurred each dt seconds and then add them. In more mathematical terms, in a point of the form $t_{i+1} = (i+1) \cdot dt$ we put

$$f(t_{i+1}) \approx \frac{1}{dt} \sum \left(\text{entities occurring in } \; [i \cdot dt, (i+1) \cdot dt]\right).$$

It may also be the case that we are already given the inflow at a discrete set of times, in that case we do not need to worry about entities at all.

2.5.2. ***Integrating the ODE***. Next we will need to solve the ODE (3). In order to do this we can use very standard mathematical algorithms for integrating differential equations. For example Runge-Kutta methods. The only caution one needs to take care of is the interpolation of the inflow between two successive points t_i and t_{i+1}. A linear interpolation usually suffices.

2.5.3. ***Queue error due to aggregation time***. Approximating the inflow as discussed introduces an error in the model that we can not avoid but which we can estimate. Denote by $C_{log}(t)$ the queue size given by the logistic queue model, and by $C_{disc}(t)$ the real queue size. An entity that enters to the system at $t \in [i \cdot dt, (i+1) \cdot dt]$ will experience a delay of $C_{disc}(t)/\mu$ seconds, nevertheless aggregating each dt seconds the model can not account the fact that the packet got in at t, hence:

$$\left|\frac{C_{disc}(t)}{\mu} - \frac{C_{log}(t)}{\mu}\right| \leq dt.$$

If the traffic is really intense, i.e. if $\rho = \frac{\lambda}{\mu}$ is near one, the best we can expect for is:

$$|C_{disc}(t) - C_{log}(t)| \sim \mu \cdot (1 - \rho) \cdot dt. \tag{20}$$

3. Validations and results

3.1. **Notations.** In this section we recall all our notation and its meaning. They will be used throughly in the following sections.

- *Maximum outflow rate*: μ. Its units are entities/time. It will be assumed to be constant, for example, $\mu = 1$ Gb/s. It is given, so that it is an input.
- *Queue size at t*: $C(t)$. Its units are entities. It varies over time. Our goal is to predict this quantity.
- *Inflow to the system at t*: $f(t)$. Its units are entities/time. It varies over time. It is given, so that it is an input.
- *Outflow of the system at t*: $g(t)$. Its units are entities/time. It varies over time. Our goal is to predict this quantity.
- *Mean inflow*: λ. Its units are entities/time. It is the mean of the inflow f.
- *Mean intensity or occupancy*: ρ. It has no units. It is defined as $\frac{\lambda}{\mu}$. This quantity gives us an idea of the average use of the server.
- *Aggregation time*: dt. Its units are time. When a flow is given in discrete form, we will assume it is given each dt seconds. Usually $dt = 60$.
- *Logistic model parameter*: α. It is defined as $\frac{\rho}{\mu}$.

3.2. **Error measures.** In the following sections we will compare the queue size given by the logistic queue model, which we denote $C_{log}(t)$, with the queue size given by a discrete packet simulator, which we denote $C_{disc}(t)$. In order to compare them we need some error measures. Most of the time those quantities are given at discrete instants of time. So in the end we will only have two vectors $\mathbf{C_{disc}}, \mathbf{C_{log}}$ evaluated at a finite number of times n. Since the logistic model will not react to small fluctuations in the queue, we will be mostly interested in checking if the model captures big queues. These are the important ones because they affect the outflow the most. We define:

1: *Error relative to the maximum*:

$$\frac{\|\mathbf{C_{disc}} - \mathbf{C_{log}}\|}{\|\mathbf{1} \cdot \max(\mathbf{C_{disc}})\|}$$

where $\mathbf{1}$ is vector of ones of length n.

2: *Maximum occupancy error*:

$$\frac{|\max(\mathbf{C_{disc}}) - \max(\mathbf{C_{log}})|}{|\max(\mathbf{C_{disc}})|}.$$

Likewise, we will want to compare the outflow given by the logistic queue model, which we denote $g_{log}(t)$, with the outflow given by the discrete Simulink simulator, which we denote $g_{disc}(t)$. Again, in order to compare them we need some error measures. Most of the time those quantities are given at discrete instants of time. So in the end we will only have two vectors $\mathbf{g_{disc}} = (g^1_{disc}, \ldots, g^n_{disc})$ and $\mathbf{g_{log}} = (g^1_{log}, \ldots, g^n_{log})$ evaluated at a finite number of times n. We define:

3: *Mean relative outflow error*:

$$\frac{1}{n}\sum_{i=1}^{n}\frac{|g^i_{disc}-g^i_{log}|}{|g^i_{disc}|}$$

4: *Global relative error*:

$$\frac{\|\mathbf{g_{disc}}-\mathbf{g_{log}}\|}{\|\mathbf{g_{disc}}\|}$$

3.3. **User service modelling.** We will model the inflow created by a video user in a discrete way, i.e. packet by packet. In order to do this, we will make some statistical assumptions about the packets generated while using a video service:

(1) All packets are assumed to be of the same constant size.
(2) Packets are assumed to come in *bursts* of variable size. We will assume that the size of each burst follows a *Normal distribution*.
(3) Bursts are assumed to be separated by variable intervals of time. We will assume that the size of an interval of time separating two bursts –*Interburst time*– follows an *Exponential distribution*.
(4) Finally, packets are assumed to be separated by variable intervals of time inside a burst. We will assume that the size of an interval of time separating two packets inside a burst –*Interpacket time*– also follows an *Exponenetial distribution*.

Real video flows were generated and the parameters of each distribution were estimated using standard statistical methods. For characterizing video streaming, an on-demand video file was served from a set of HTTP servers to a single end user based on the MPEG-DASH v1.4 standard. On the server-side, two virtual machines each running an Apache HTTP server instance were responsible for serving the audio and the video components, respectively. The video was served at HD 720p and its duration was 10 minutes. We obtained the results of Table 1.

Packet size (Bytes)	1464
Burst size (mean)	1714
Burst size (variance)	278
Interburst time (seconds)	5.56
Interpacket time (seconds)	0.00345

TABLE 1. Estimation of parameters for a video user.

Next we want to model the use of video service. Since we are interested in simulating more than a few hours, we cannot assume that each video user is watching video continuously without stopping. Hence we assume that *uses of the service* are separated by an interval of time –*Interuse time*– following an *Exponential distribution* with mean 45 minutes. So in mean, a typical user watches some videos each forty five minutes. Finally, we know from experience that not all videos are of the same length, and even if they were, it is possible and usual to watch more than one. Therefore we also assume a probability distribution on *video consumptions* listed in Table 2 .

Video use (minutes)	*Probability*
5	0.4
15	0.3
30	0.25
120	0.05

TABLE 2. Length of video consumptions and their probabilities.

Using all previous assumption we can simulate all packets generated by a video user. Its easier to picture all the previous concepts with a figure. In Figure 7 we show a simulated packet flow using the previously estimated parameters.

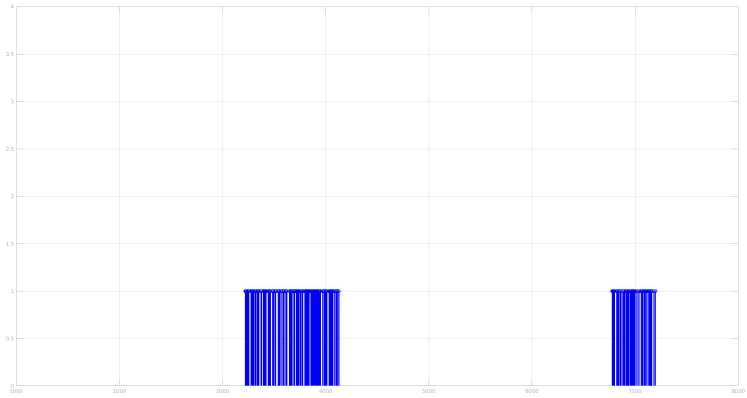

FIGURE 7. Simulation of packets generated by a video user during two hours. Each blue line represents a packet.

Now we want to compute the flow f_v it generates. In order to do this, we simply count how many packets have occurred each $dt = 60$ seconds and then add them. In more mathematical terms, in a point of the form $t_{i+1} = (i+1) \cdot dt$ we put

$$f_v(t_{i+1}) \approx \frac{1}{dt} \sum (\text{packets occurring in } [i \cdot dt, (i+1) \cdot dt]) .$$

In Figure 8 we plot the flow generated by a typical video user. As it is expected, the service is not being used constantly, but rather sparely as wanted.

3.4. **Illustrative example.** In order to have an intuitive understanding of the model and all the things involved with it, we will first give an illustrative example of its performance. Later we will make a more exhaustive performance analysis. We assume the following particular scenario:

(1) We aggregate 10 video users in a server of maximum velocity $\mu = 11.33$ Mb/s during 2 days.
(2) The queue is assumed to be large enough so that we do not lose any packets.

For generating the video flows we use the method described in the previous section. The goal is to predict the queue formed under this scenario and the outflow we get.

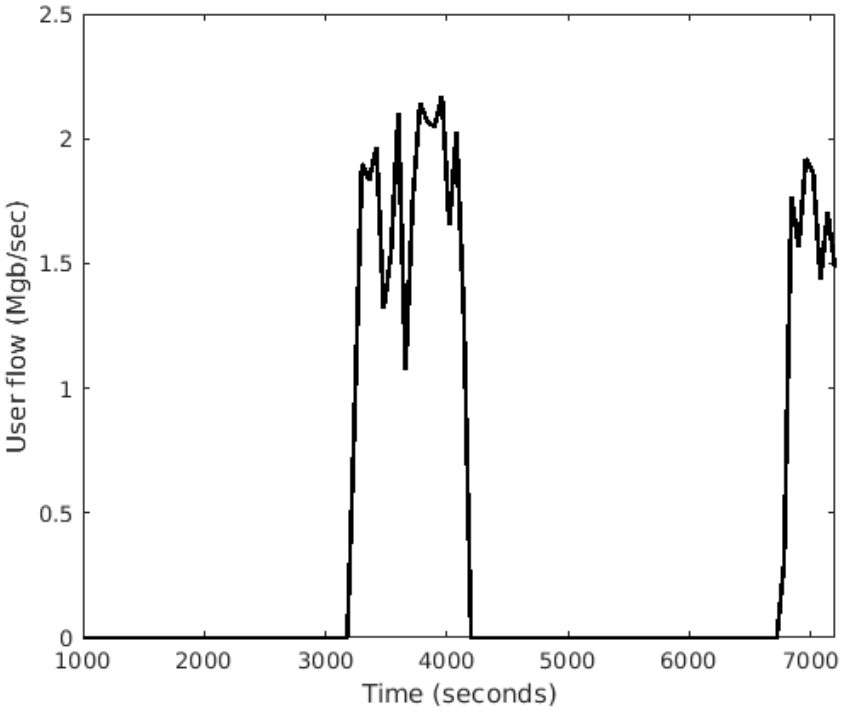

FIGURE 8. Flow f_v generated by one video user during two hours.

3.4.1. *Inflow.* Adding the 10 video flows we get the inflow of Figure 9. In other words we put

$$f(t) = \sum_{v=1}^{10} f_v(t).$$

The mean of the inflow is $\lambda = 6$ Mb/s, hence the average intensity is $\rho = 53\%$. Therefore we take $\alpha = \frac{\rho}{\mu} = 0.05$. As we see in the picture, during some periods of time, the inflow is bigger than the maximum outflow μ, therefore in those moments we expect the queue size to increase.

3.4.2. *Comparison of queues.* We feed the inflow f to both the Simulink discrete model and to the logistic queue model and compare the queue results we get in both cases. Note that f is given to the discrete simulator in discrete form, i.e. packet by packet as it was generated, whereas to the logistic model the inflow is given in a continuous fashion, i.e. aggregated each 60 seconds and then linearly interpolated when needed. Just looking at Figures 10 and 11 of the queues formed is impossible to tell the difference, hence we need some metrics to compare them. We will use the two measures introduced before:

- *Error relative to the maximum*:

$$\frac{\|\mathbf{C_{disc}} - \mathbf{C_{log}}\|}{\|\mathbf{1} \cdot \max(\mathbf{C_{disc}})\|} = 0.63\%$$

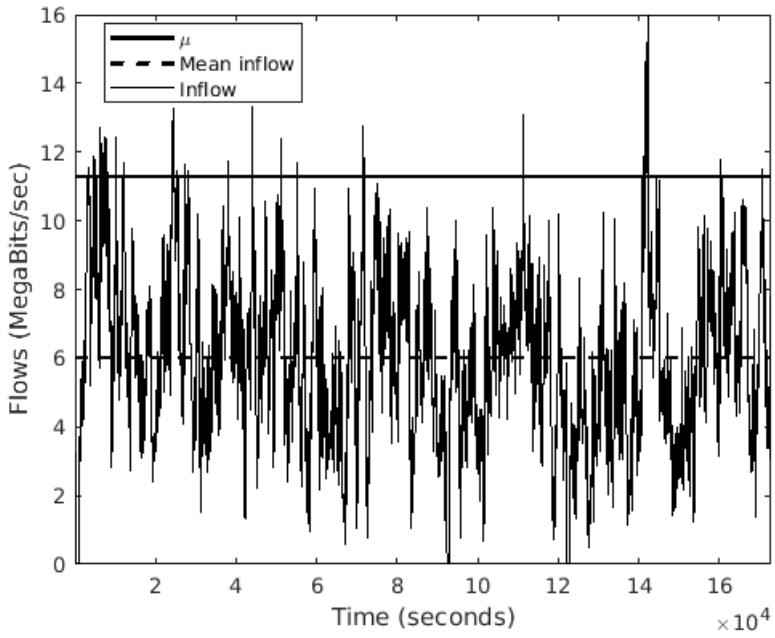

FIGURE 9. Inflow to the system. Its standard deviation is 2.53 Mb/s.

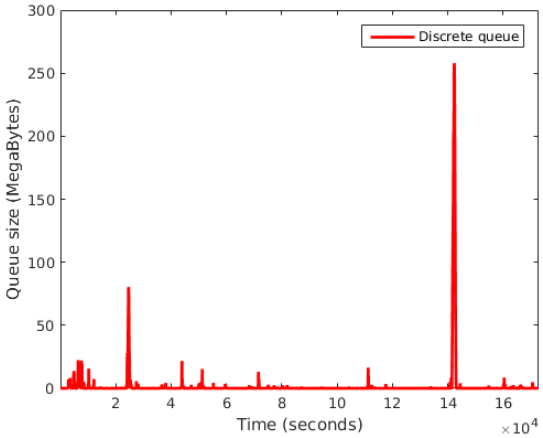

FIGURE 10. Queue of the discrete simulator.

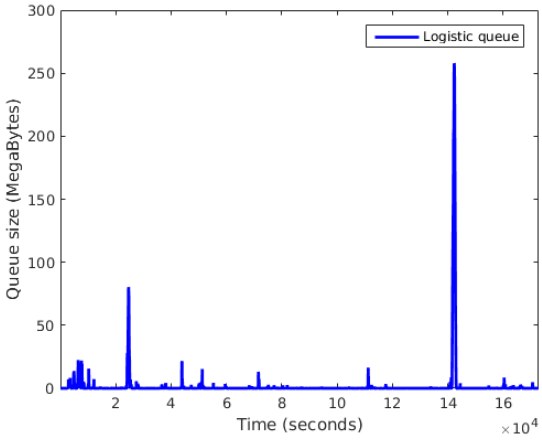

FIGURE 11. Queue of the logistic model.

- *Maximum occupancy error*:

$$\frac{|\max(\mathbf{C_{disc}}) - \max(\mathbf{C_{log}})|}{|\max(\mathbf{C_{disc}})|} = 2.08\%$$

It is worth to examine both queues superposed in a neighborhood of the maximum occupancy.

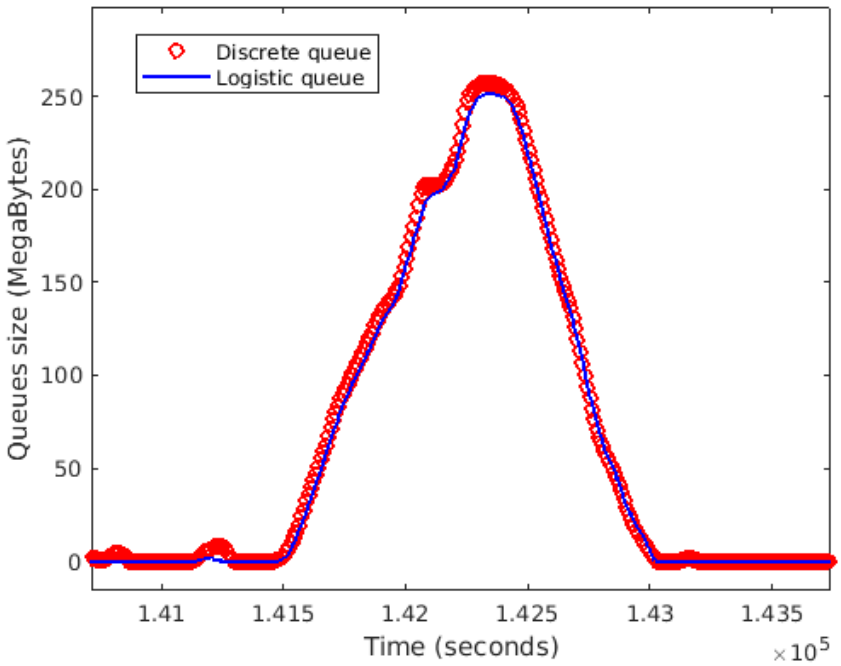

FIGURE 12. Superposition of logistic and discrete queues in a neighborhood of the maximum queue occupancy.

3.4.3. *Queue error due to aggregation time.* Recall that if the traffic is intense, i.e. if $\rho = \frac{\lambda}{\mu}$ is near one, the best we can expect for is:

$$\left| \frac{C_{disc}(t)}{\mu} - \frac{C_{log}(t)}{\mu} \right| \sim (1-\rho) \cdot dt.$$

In our example we actually have that

$$\max_{t \in [t_0, t_1]} \left| \frac{C_{disc}(t)}{\mu} - \frac{C_{log}(t)}{\mu} \right| = 15.71 \text{ seconds},$$

so even the estimate with $(1-\rho) \cdot dt = 28.18$ seconds holds.

3.4.4. *Comparison of outflows.* We can also compare both outflows, the one given by the discrete simulator and the other computed from the logistic queue. As expected both flows do not exceed the maximum capacity μ. Just looking at Figures 13 and 14 is again impossible to say anything. We again use the two measures already introduced:

- *Mean relative outflow error*:

$$\frac{1}{n} \sum_{i=1}^{n} \frac{|g^i_{disc} - g^i_{log}|}{|g^i_{disc}|} = 0.67\%$$

- *Global relative error*:

$$\frac{\|\mathbf{g_{disc}} - \mathbf{g_{log}}\|}{\|\mathbf{g_{disc}}\|} = 1.58\%$$

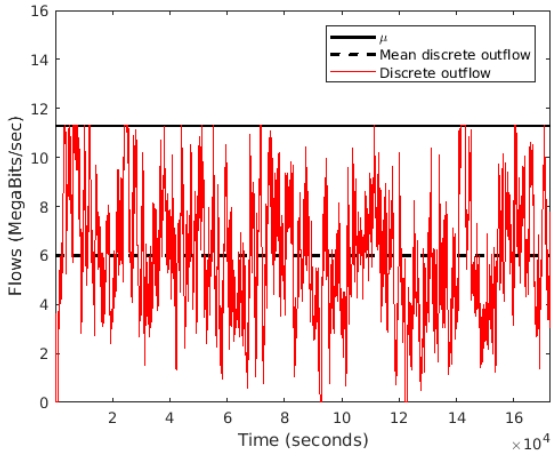

FIGURE 13. Outflow of discrete simulator.

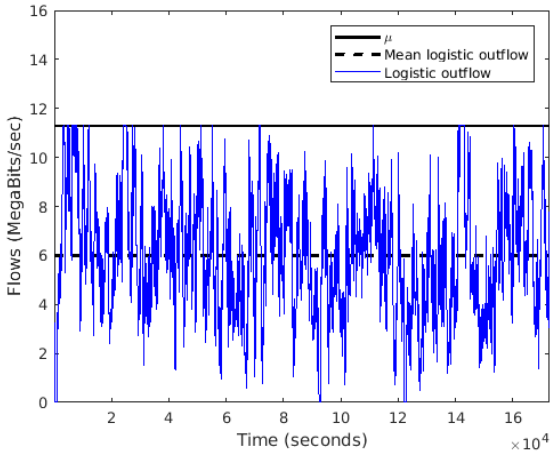

FIGURE 14. Outflow of logistic model.

3.4.5. *Comparison with no queue model.* In order to have a reference to compare with, we can compute the previous error measures with the inflow to the system. This would be the simplest model in which we assume that the outflow equals the inflow and the queue has no effect whatsoever. In this example the mean relative error is:

$$\frac{1}{n}\sum_{i=1}^{n}\frac{|g^i_{disc}-f^i|}{|g^i_{disc}|}=0.75\%,$$

while the global relative error is:

$$\frac{\|\mathbf{g_{disc}}-\mathbf{f}\|}{\|\mathbf{g_{disc}}\|}=5.27\%.$$

As expected mean-wise the relative errors are very similar. This is due to the fact that most of the time the inflow is smaller than μ and hence the outflow g is equal to f. On the other hand the global relative error is sensibly bigger for the no queue model since this measure penalizes more point-wise errors.

3.4.6. *Discussion of example.* As already stated the goal of this example was to introduce in an intuitive fashion all concepts involved with the model in a practical setting. The results are very promising in the sense that our model captures very well the queue generated and hence also the outflow. At the end of the day, one may wonder why it is worth introducing a continuous model (the logistic queue model) if we already can make discrete simulations and get quite accurate results. The key is of course **scalability**. In this illustrative example the simulation time of the discrete simulator was around 60021 seconds, which is about 16 hours. On the other hand to the logistic queue model the whole process only took about 29 seconds. So the continuous model is around 2000 times faster! This is expectable since the discrete simulator had to process around $2.45\cdot 10^7$ entities. In the next section we will make a more comprehensive analysis of the performance of the model.

3.5. **Permormance Analysis.** In this subsection we analyze how the error measures introduced in the previous examples evolve as the intensity of the inflow is varied. Again we assume the same scenario as in the example:

(1) We aggregate 10 video users in a server of maximum velocity $\mu = 11.33$ Mb/s during 2 days each 60 seconds.
(2) The queue is assumed to be large enough so that we do not lose any packets.

We vary inflow intensities ρ from approximately 45% to 85%. In order to do so we simply reduce the *Interuse time* introduced before when modeling a video user. Recall that this parameter represented the amount of time between two uses of video service. Hence reducing it, we only make our 10 users watch videos more often. In total 21 simulations were launched both in the discrete event simulator implemented in Simulink and using the logistic queue model implemented in MATLAB. Of the 21 simulations 3 of them had not finished after more than a month in the discrete simulator, and therefore they were discarded. Note that in each simulation we are generating around 25 million packets, so in total in 21 simulations we generated about 500 million packets which is roughly the amount of Spanish speaking people in the world.

3.5.1. ***Comparison of queues***. We first compare the maximum queue size given by both models. In Figure 15 bellow we can see a plot of both curves for different intensities. As we see both curves are quite close. In dashed lines we have plotted the error bound (20).

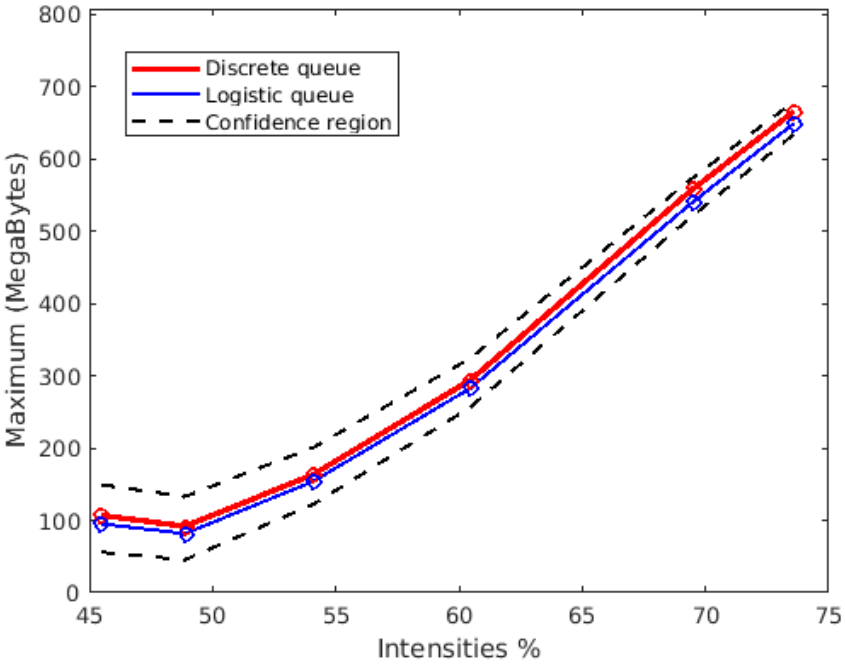

FIGURE 15. Comparison of maximum queue size for both models varying the inflow intensity.

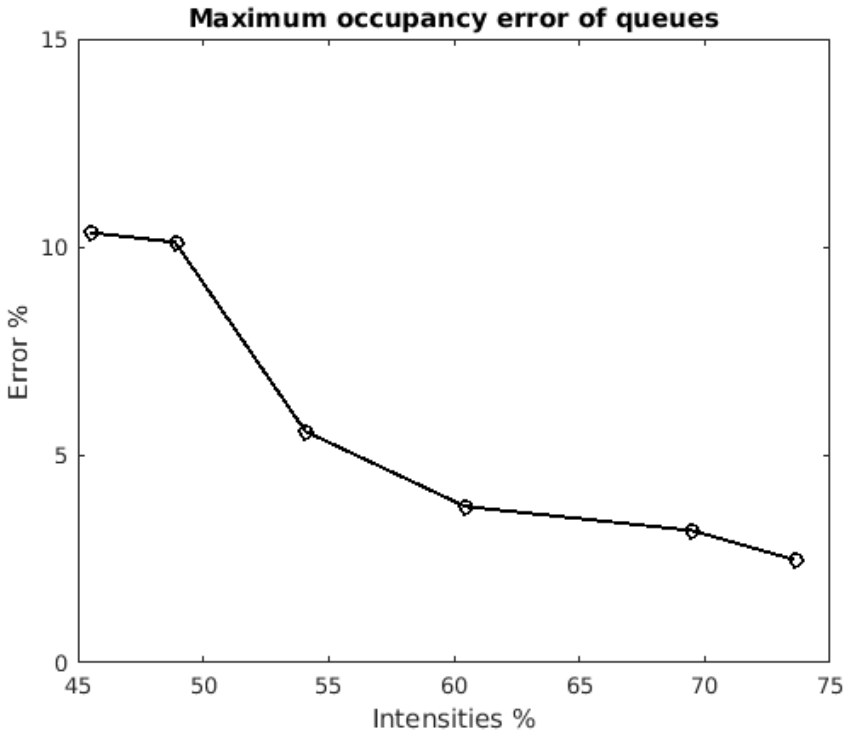

FIGURE 16. Maximum occupancy error of the logistic model varying the inflow intensity.

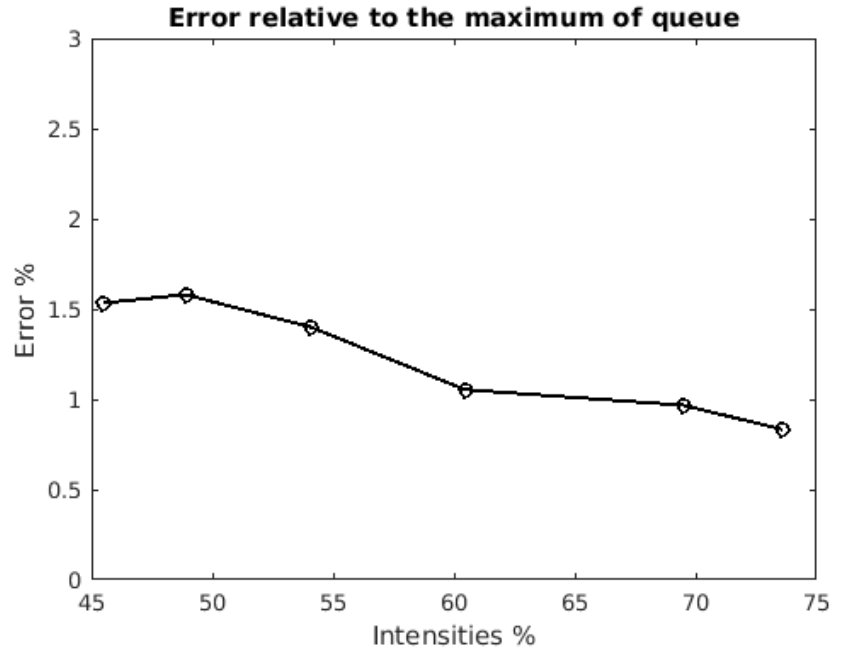

FIGURE 17. Error relative to the maximum of the logistic model varying the inflow intensity.

It is also interesting to see how the maximum occupancy error evolves as the intensity increases. We see in Figure 16 how the error decreases as the intensity increases. Finally in Figure 17 we plot the error relative to the maximum introduced earlier. Again the error reduces as the intensity increases. This is again in accordance with (20).

3.5.2. ***Comparison of outflows***. Next we compare the outflow given by the logistic queue model with the one given by the discrete simulator. To have a reference to compare with we also plot the error of the no-queue model as discussed before. Recall that this just assumes that the queue has no effect whatsoever so the outflows just equals the inflow. We begin by plotting the global relative error in Figure 18. Note how the error of

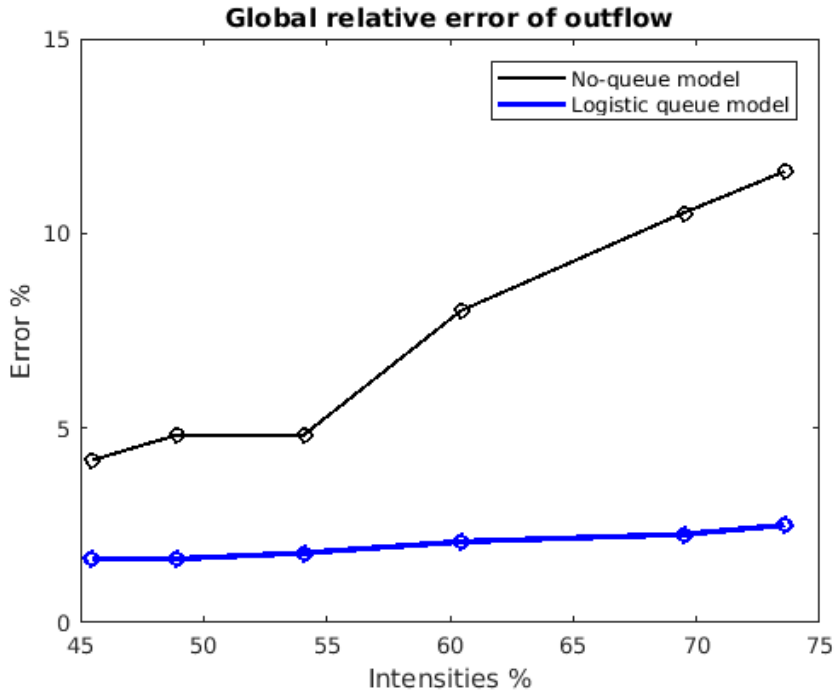

FIGURE 18. Global relative error of the outflows given by logistic model and the no-queue model varying the inflow intensity.

the no-queue model increases as the intensity increases. This is to be expected because increasing the intensity increases the queue size and therefore makes the outflow differ more from the inflow. On the other hand the error of the logistic queue model seems quite stable, its mean is around 2%. Finally we compute the mean relative error. In Figure 19 we observe that until $\rho = 55\%$ there is no big difference between both models, both errors are quite low. This again logical since we know that the outflow of the system is almost identical to the inflow when there are not a big queues.

3.5.3. ***Scalability: simulation times***. Now we want to compare the simulation time of both models. Since the mean simulation time of the logistic model is 26.3 seconds but the mean of the discrete simulator is 183.01 hours we take logarithms. As we see in Figure 20, the logistic queue model is about 8 orders of magnitude faster than the discrete simulator.

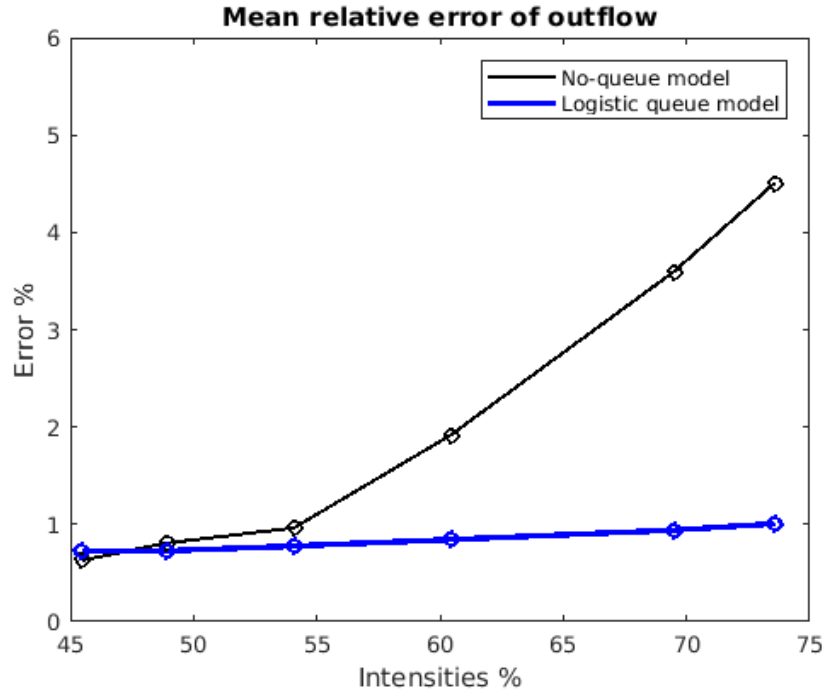

FIGURE 19. Mean relative error of the outflows given by logistic model and the no-queue model varying the inflow intensity.

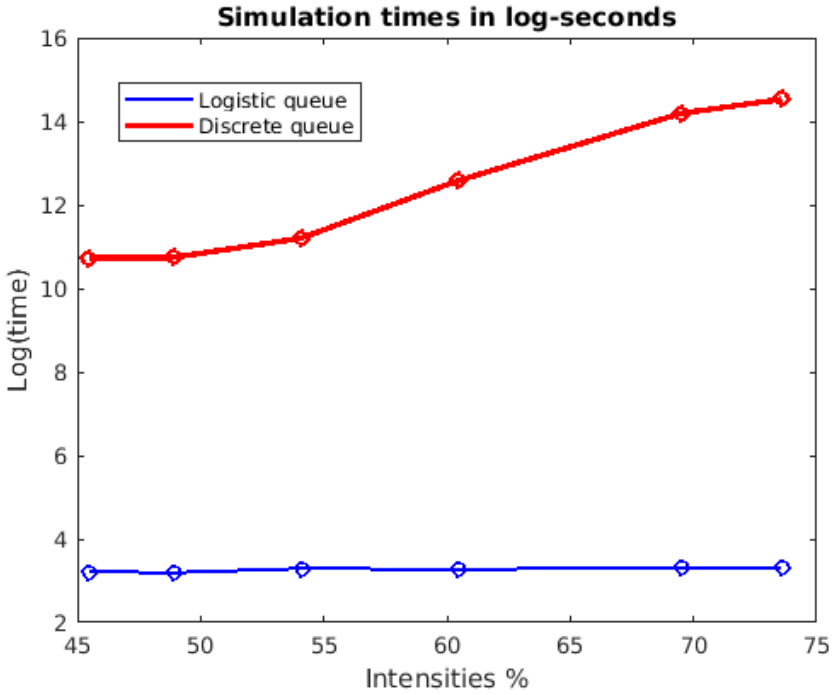

FIGURE 20. Simulation times of logistic and discrete queue model varying the inflow intensity.

4. Conclusions

In this work we have presented a novel queuing model called the *logistic queue model*. We have proven mathematically that our proposed model has all the theoretical properties one should expect:

1: Positivity of the queue size.
2: Asymptotic behaviour: the queue gets empty if the inflow does not overflow.
3: FIFO property: the system satisfies a first in first out discipline.

Moreover, we validated the model comparing it with a discrete event simulator. We showed that in terms of queue size and outflow prediction our model is as precise as a discrete one with the advantage of speed in simulations. We compared simulation times and concluded that the logistic queue model is several orders of magnitude faster than the discrete one. Finally in contrast with the point-queue model our model allow us to easily explore multiple extensions to more general scenarios such as: finite queues, multiple servers, priority queues, etc.

References

[1] 1
CISCO Global Cloud Index (GCI), 2015.

[2] 1
CISCO Visual Networking Index (VNI), 2016.

[3] L. M. Contreras, V. L¬ópez, O.Gonz¬ález, A.Tovar, F.Mu¬ñoz, A.Aza¬ñ¬ón, J.P.Fern¬ández−Palacios, andJ.Folgueira : *Towards cloud-ready transport networks.*1
IEEE Communications Magazine, vol. 50, pp. 48-55, 2012.

[4] D. King and A. Farrel: *A PCE-based architecture for application-based network operations.* 2
CIETF RFC7491, 2015.

[5] V. Lopez and L. Velasco: *Elastic Optical Networks: Architectures, Technologies, and Control.* 2
Ed. Springer, 2016.

[6] L. Velasco, A. Castro, M. Ruiz, and G. Junyent: *Solving Routing and Spectrum Allocation Related Optimization Problems: from Off-Line to In-Operation Flexgrid Network Planning.* 2
IEEE/OSA Journal of Lightwave Technology (JLT), vol. 32, pp. 2780-2795, 2014.

[7] A.P. Vela, A. V¬ía, F.Morales, M.Ruiz, andL.Velasco :*Traffic generation for telecom cloud-based simulation.*2
IEEE International Conference on Transparent Optical Networks (ICTON), 2016.

[8] 2
OMNeT++ Discrete Event Simulator: http://www.omnetpp.org/

[9] P. S. Barreto, T. F. Bento, Paulo H. P. de Carvalho: *Multimedia Network Simulation with a Fluid Flow Model.* 2
IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.6, June 2010.

[10] C. Kiddle, R. Simmonds, C. Williamson, and B. Unger: *Hybrid Packet/Fluid Flow Network Simulation.* 2
Proceedings of the Seventeenth Workshop on Parallel and Distributed Simulation (PADS√ï03)1087 − 4097/03, *IEEE*2003.Vickrey, W.S:*Congestion theory and transport investment.*3
The American Economic Review 59 (2), 251√ê261.

[1[1]1] Ke Han, Terry L. Friesz, Tao Yao: *A partial differential equation formulation of Vickrey√ïsbottleneckmodel, partI : Methodologyandtheoreticalanalysis.*3
Transportation Research Part B Methodological. March, 2013.

[12] Ke Han, Terry L. Friesz, Tao Yao: *A partial differential equation formulation of Vickrey√ïsbottleneckmodel, partII : Numericalanalysisandcomputation.*3
Transportation Research Part B Methodological. March, 2013.