Scheduling Announced Requests for Streamed Information

Danny De Vleeschauwer*†, Bart Feyaerts†, Herwig Bruneel†, Sabine Wittevrongel†
*Alcatel-Lucent, Bell Labs,
Copernicuslaan 50, B-2018 Antwerp, Belgium.
Phone: +32-3-2408196, Fax: +32-3-2411945
Email: {danny.de_vleeschauwer}@alcatel-lucent.com
†Ghent University, Dept. of Telecommunications and Information Processing,
SMACS Research Group,
Sint-Pietersnieuwstraat 41, B-9000 Ghent, Belgium.
Phone: +32-9-2648901, Fax: +32-9-2644295
Email: {bart.feyaerts,hb,sw}@telin.ugent.be

***Abstract*—We analyse a scheduling system in which users announce requests for information from a server some time before they actually need this information. Each constituent of the requested information has its own specific deadline, which is characteristic for applications that gradually consume a stream of information (e.g., video). For that purpose, a request fully specifies when the requesting user needs each constituent of this information. Since each user device is equipped with a buffer, the server can exploit this detailed timing information to deliver parts of the requested information upfront. The more the requests are announced in advance, the better the ability of the server to avoid deadline violations. We investigate for a given server capacity via semi-analytical techniques complemented with simulations how much the requests need to be announced in advance to essentially avoid violating all deadlines.**

***Keywords*—Information streaming; Scheduling; Deadline margin**

I. Introduction

We consider a system where users can issue a request for digital information some time before they actually need it. In a request a user announces when he or she will consume each constituent (e.g., bit, byte or packet) of this information, thus explicitly assigning a deadline for each such constituent. Specifying such a deadline profile, rather than associating one deadline for the complete information block the request pertains to (e.g., a file), is typical for streamed information (e.g., audio, speech, video). In [10] an example of such a system is described where the requests specify a byte range (of the memory of the server) and deadlines for the first and last byte of that range. Since in the system of that paper a user is assumed to consume the specified byte range at constant rate, the deadlines for all bytes (i.e., constituents) in that byte range can be determined by linear interpolation. In the present paper we will provide a mathematical description applicable to any deadline profile, but apply it to the system of [10].

Since we assume that each user has an (unlimited) buffer to temporarily store the information constituents until the user needs them, the server can start to serve information constituents as soon as the request is announced and possibly a long time before the information constituents are actually going to be consumed by the user. Since the server attempts to be fair to each user, it uses an EDF (earliest-deadline-first) scheduling discipline to serve the constituents of the already announced requests. The server serves information at a constant rate C and tries to avoid violating deadlines. In order to be able to avoid deadline violations, the requests need to be known a long enough time upfront. In fact, it is intuitively clear that if the requests are not known upfront, the server needs to have the capability to serve at the peak rate, while if the requests are known a very long time in advance, the server needs to serve only at the average rate. The question that we set out to answer in the present paper is how much the requests should be announced upfront for a given server rate C between the average and the peak rate in order to keep the probability of deadline violations below a given (very small) target value.

The rest of the paper is structured as follows. We review related work in Section II. Section III gives a mathematical description of the scheduling system under study and investigates the evolution of the deadline margin. Numerical results in Section IV show the impact of the time requests are announced upfront on the required server capacity to essentially avoid deadline violations. Finally, the paper is concluded in Section V.

II. Related work

In this section, we first point out how our paper fits in the long tradition of applying queueing theory to model the behaviour of traffic sources over a network. Then we concentrate on how our paper relates to existing work around schedulers.

Queueing models have been used for studying the multiplexing behaviour of traffic sources for a very long time. Back in the days of circuit switching the Erlang B formula [17] was used to assess the probability that a call would be blocked due to a switching node spuriously running out of circuits. With the advent of packet-based networks

various types of models were used to assess the delay or loss probability packets experience in network nodes [8], [17]. Models with a Markovian process as arrival process were studied first for various types of sources (voice [11], video [4]). Then it was discovered that for an aggregate of some types of sources the arrival of packets exposed self-similarity [19]. Subsequently, a model was studied that showed that this self-similarity has a large impact on the delay and loss characteristics [22]. All previous models essentially model traffic produced by sources that are not responsive to congestion signals. In the Internet of today, these sources are referred to as UDP (user datagram protocol) sources [23]. Congestion control was introduced in TCP (transmission control protocol) [2] to avoid that the Internet would collapse under congestion. A TCP source tries to grab a maximum, but fair, amount of the available resources by gradually increasing its rate and reacting to congestion signals. The multiplexing behaviour of TCP sources has been studied with queueing models as well, in particular with the GPS (general processor sharing) discipline [7] and with mean field theory [5]. Moreover, models to study the required queue size under load of TCP sources [3], [12] and AQM (active queue management) techniques that allow for better congestion feedback were investigated [1]. Our paper continues in this tradition: it studies an alternative to UDP and TCP, in which the sources (in contrast to UDP and TCP) provide timing information to the nodes in the network related to when the constituents of information they are requesting, are needed. In our paper we develop a model that captures this behaviour and explore how beneficial providing this timing information is.

The scheduler plays a central role in the system we study. Schedulers have received lots of attention in literature, mainly in the context of traffic differentiation. Since the traffic supported by the Internet has diverse characteristics and requirements, some protocols were designed to treat individual flows [24] or an aggregate of flows [6] differently. In that way some traffic sources or classes could be offered a lower delay or a smaller packet loss probability, without being hampered by the behaviour of other traffic sources or classes. Various schedulers were investigated in this context, e.g., strict priority or head-of-line scheduling, proportional-rate schedulers such as weighted round robin and weighted fair queueing, deadline-based schedulers and reservation-based schedulers. Overviews are given e.g. in [16] and [26]. EDF (earliest deadline first) was also considered in this context [25]. However, with EDF in this context the deadlines of the packets are given at the entrance of the network by a controller, and not by the source itself. The idea is that a network node may introduce a certain maximum delay, which sets the deadline the packet is given at the entrance, and that the use of EDF guarantees that the delay incurred in that node is smaller than that maximum delay as much as possible. Besides this use of EDF in the context of traffic differentiation, queues governed by an EDF service discipline have been studied in a theoretical context too, e.g., in [14] a system with Poisson arrivals is studied, in [21] the optimality of EDF is investigated in cases with exponential service times, a heavy traffic analysis for EDF based on a decomposition result is presented in [13] and in [18] a heavy load approximation is given in a general context. However, in these cases reported in literature one deadline is assigned to each arrival, which determines the time before which the entire job needs to be finished. This contrasts with our system as we provide a deadline profile that sets the deadline for each constituent of the request, rather than only one deadline for the entire job.

III. Mathematical description of the system

A. Evolution equations

We consider a server with capacity C that uses the EDF discipline. Such a server has the property that it simultaneously serves (infinitesimal) constituents (of the already announced requests) that have the same deadline. At times τ_i, referred to as announcement times, a request for information is announced with a deadline profile $A_i(t)$, where $A_i(t)$ designates the amount of (infinitesimal) constituents that the user will consume between time t and $t+\mathrm{d}t$ (with $\mathrm{d}t$ an infinitesimal increment), and where $A_i(t)$ is 0 for t outside the interval $[\tau_i + \Delta_i, \tau_i + \Delta_i + \theta_i]$. Here Δ_i indicates how long the request is announced upfront and θ_i is the duration of the request, see Figure 1. In this paper, we will assume that the announcement times τ_i are governed by a Poisson process and $A_i(t)$, Δ_i and θ_i (all ≥ 0) by statistical laws to be detailed below. Besides being 0 outside the interval $[\tau_i + \Delta_i, \tau_i + \Delta_i + \theta_i]$ the deadline profile $A_i(t)$ can, in principle, assume any form, while in [10] it is constant over that interval.

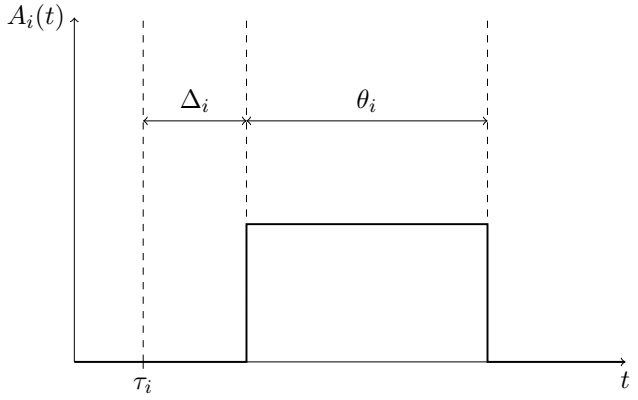

Fig. 1. Example of a deadline profile of a request with announcement time τ_i.

To analyse this system we first consider how it evolves over a period between two announcement times and then concentrate on what happens at an announcement time.

First, we describe the system evolution between two announcement times, τ_i and τ_{i+1}. Say that just after the i-th announcement at time τ_i the server is serving constituents (of all known requests) with deadline υ_i. All information constituents with a deadline earlier than υ_i were already served and the remaining work has a deadline larger than υ_i. Define $W_i(t)$ as the sum of all yet to be served constituents (of all requests known just after τ_i) that the users will consume at time t. Notice that by definition $W_i(t) = 0$

for $t \leq \upsilon_i$. It will turn out to be useful to also define

$$\omega_i(\upsilon) = \frac{1}{C} \int_{\upsilon_i}^{\upsilon} W_i(t)\,\mathrm{d}t + \tau_i\,. \tag{1}$$

Notice that $\omega_i(\upsilon) = \tau_i$ for $\upsilon \leq \upsilon_i$.

To see how the deadline evolves, we express that during the interval $[\tau_i, \tau[$ (with $\tau \geq \tau_i$) the EDF server processes all constituents (of all known requests) with deadline between $[\upsilon_i, \upsilon(\tau)[$. Concretely, the amount of information served by the server in the interval $[\tau_i, \tau[$, i.e., $C \cdot (\tau - \tau_i)$, is equal to the information of the constituents with a deadline in the interval $[\upsilon_i, \upsilon(\tau)[$, i.e., $\int_{\upsilon_i}^{\upsilon(\tau)} W_i(t)\,\mathrm{d}t$. This yields an equation for the evolution of the deadline $\upsilon(\tau)$ for τ in the interval $[\tau_i, \tau_{i+1}[$:

$$C\,(\tau - \tau_i) = \int_{\upsilon_i}^{\upsilon(\tau)} W_i(t)\,\mathrm{d}t\,,$$

or, in view of definition (1),

$$\tau = \omega_i(\upsilon(\tau))\,. \tag{2}$$

Essentially this means that

$$\upsilon(\tau) = {\omega_i}^{-1}(\tau)\,,$$

for τ in the interval $[\tau_i, \tau_{i+1}[$. Notice that if $\omega_i(\upsilon) < \tau_{i+1}$ for all υ, then the system becomes empty (i.e., has no more work to do) prior to the next announcement time τ_{i+1} (and, it can be considered that, in some finite interval prior to that announcement time τ_{i+1}, the server is serving constituents with deadline infinity).

Since $\omega_i(\upsilon)$ is non-decreasing, as its derivative $W_i(t)$ is non-negative, and does not change as long as there are no new requests announced, equation (2) uniquely determines $\upsilon(\tau)$ in the interval $[\tau_i, \tau_{i+1}[$. Notice that $\upsilon(\tau)$ can assume the value infinity over some interval prior to τ_{i+1}.

Secondly, we consider what happens when the next request is announced at time τ_{i+1} ($\geq \tau_i$). At that time τ_{i+1} a new deadline profile is announced, bringing additional work to the server, and consequently $\omega_i(\upsilon)$ needs to be updated. Immediately prior to τ_{i+1}, all work up to the associated deadline $\upsilon_{i+1}^- = \upsilon(\tau_{i+1})$ was already performed. At that time the remaining work is given by $W_{i+1}^-(t) = W_i(t)$ for $t > \upsilon_{i+1}^-$, and $W_{i+1}^-(t) = 0$ for $t \leq \upsilon_{i+1}^-$. Using equation (1) it can be seen that immediately prior to τ_{i+1} the time by which the server is serving constituents (of all requests known just after τ_i) with deadline υ is given by

$$\begin{aligned}\omega_{i+1}^-(\upsilon) &= \int_{\upsilon_{i+1}^-}^{\upsilon} W_{i+1}^-(t)\,\mathrm{d}t + \tau_{i+1} \\ &= \max\{\omega_i(\upsilon), \tau_{i+1}\}\,.\end{aligned}$$

The work $A_{i+1}(t)$ announced at the time instant τ_{i+1}, which is only non-zero in an interval of length θ_{i+1} starting at $t = \tau_{i+1} + \Delta_{i+1}$, still needs to be added. Similarly as in equation (1) we define

$$\alpha_{i+1}(\upsilon) = \frac{1}{C} \int_{-\infty}^{\upsilon} A_{i+1}(t)\,\mathrm{d}t\,. \tag{3}$$

Remark that $\alpha_{i+1}(\upsilon) = 0$ for $\upsilon < \tau_{i+1} + \Delta_{i+1}$ and is constant for $\upsilon > \tau_{i+1} + \Delta_{i+1} + \theta_{i+1}$. This additional work is added to the work that remained just prior to the announcement time τ_{i+1}, which yields

$$\omega_{i+1}(\upsilon) = \max\{\omega_i(\upsilon), \tau_{i+1}\} + \alpha_{i+1}(\upsilon)\,. \tag{4}$$

Equations (2) and (4) completely describe the system evolution. The function $\omega_i(\upsilon)$ represents the system state: according to equation (2), $\omega_i(\upsilon)$ is the only information needed to describe the evolution between two consecutive announcement times, τ_i and τ_{i+1}; and $\omega_i(\upsilon)$ is updated according to equation (4) at announcement times τ_{i+1}.

Applying equation (4) recursively k times yields:

$$\begin{aligned}\omega_i(\upsilon) = \max\Bigg\{ &\omega_{i-k-1}(\upsilon) + \sum_{\ell=0}^{k} \alpha_{i-\ell}(\upsilon), \\ &\max_{0\leq m\leq k} \left\{ \tau_{i-m} + \sum_{\ell=0}^{m} \alpha_{i-\ell}(\upsilon) \right\} \Bigg\}.\end{aligned} \tag{5}$$

If we assume that (e.g., for stability reasons) the system has been empty at some time in the past, there exists a value of k for which the first element of this set is never the maximum, so that

$$\omega_i(\upsilon) = \sup_{k\geq 0} \left\{ \tau_{i-k} + \sum_{\ell=0}^{k} \alpha_{i-\ell}(\upsilon) \right\}. \tag{6}$$

This equation shows how the statistics of all announced deadline profiles in the past and the announcement instants prior to announcement instant τ_i, determine the statistics of the system state $\omega_i(\upsilon)$.

We define the deadline margin $T(\tau)$ as $T(\tau) = \upsilon(\tau) - \tau$. In principle the deadline margin $T(\tau)$ can take positive or negative values, but, as we will describe below, we will impose the restriction that the deadline margin has to remain (essentially) positive for a well-behaving system. Notice that the statement that the deadline margin $T(\tau)$ remains a positive function for all τ is equivalent to the statement that all deadlines (of all constituents) are being met.

B. Graphical illustration

Figures 2-5 illustrate the evolution of the deadline margin graphically. The thick line represents $\omega(\upsilon)$; a solid thick line indicates how the system evolved in the past, while a dotted thick line indicates how it will evolve in the future under the currently known workload. The horizontal difference between the diagonal and the solid thick line indicates how the deadline margin $T(\tau)$ evolved in the past; and between the diagonal and the dotted thick line how it will evolve in future with the currently known work. This dotted thick line represents the system state: 1) only this information is needed to describe the system evolution in the immediate future as long as no new requests are

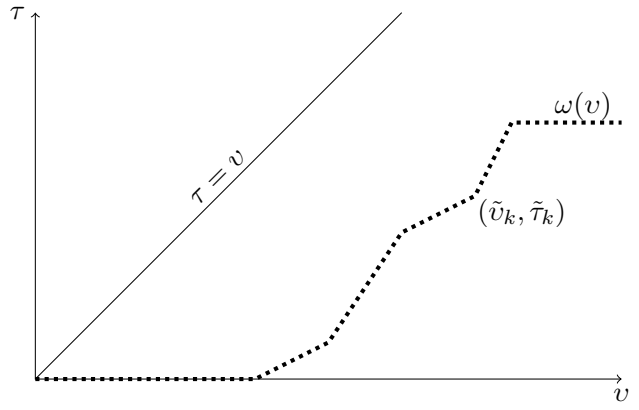

Fig. 2. The system at time $\tau = 0$.

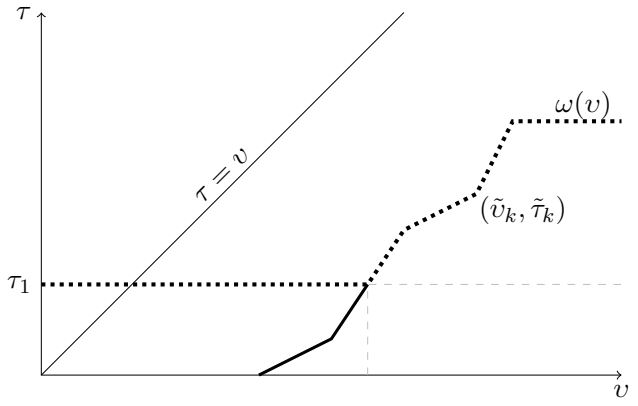

Fig. 3. The system evolution up to, but just prior to, τ_1.

announced and 2) it is updated when a new request is announced.

In the figure we illustrate the case in which requests ask for a constant information rate over some interval. In that case $A_i(t)$ is constant over the specified interval and $\alpha_i(\upsilon)$ is linear over that interval, while before $\tau_i + \Delta_i$ it is 0 and it is constant after $\tau_i + \Delta_i + \theta_i$. As a consequence $\omega(\upsilon)$ is piecewise linear and can be fully specified by its inflection points $(\tilde{\upsilon}_k, \tilde{\tau}_k)$.

Figure 2 shows the system at time $\tau = 0$. Figure 3 shows the evolution (according to equation (2)) up to, but just prior to, the time τ_1 the next request is announced. Remark that the dotted thick line becomes solid and this part represents the actual evolution of the deadline margin $T(\tau)$ up to the new announcement time. At the moment τ_1 a new request is announced, the remaining work needs to be updated (according to equation (4)) and, as a consequence, the future evolution of the deadline margin, i.e., the system state represented by the thick dotted line, changes. Figure 4 shows the system just after the time τ_1 the new request was announced. In particular, it illustrates the case where a new workload arrives with a start deadline smaller than the deadline that was already served just prior to the arrival of the announcement. Because the new workload $\alpha_1(\upsilon)$ is added, the deadline margin exposes a sharp drop. After that new announcement the deadline margin again follows the thick dotted line until the next request is announced at time τ_2. As illustrated in Figure 5 at the announcement instant τ_2 the system was empty with an infinite deadline margin. The system state represented by the thick dotted line is in this case given by (see equation (4)) the sum of τ_2 and the new load $\alpha_2(\upsilon)$. From Figures 2-5 it can be concluded that, in case of requests for constant information rate, the system's evolution (i.e., its past and its future evolution) can be completely described by maintaining the list of inflection points $(\tilde{\upsilon}_k, \tilde{\tau}_k)$.

Figures 6-8 illustrate the closed-form solution of equation (6). Figure 6 depicts three deadline profiles that are announced at announcement times τ_1, τ_2 and τ_3, respectively. In particular, the figure shows the functions $\alpha_1(\upsilon) + \tau_1$, $\alpha_2(\upsilon) + \tau_2$ and $\alpha_3(\upsilon) + \tau_3$. We focus on the case where the system was empty just before announcement time τ_1 and use equation (6) to determine $\omega_3(\upsilon)$. Figure 7 shows the three functions $\tau_{3-k} + \sum_{\ell=0}^{k} \alpha_{3-\ell}(\upsilon)$, for $k = 0, 1, 2$ over which the maximum needs to be taken to determine $\omega_3(\upsilon)$. These functions depend solely on the announced deadline profiles. Figure 8 shows the evolution of $\omega(\upsilon)$ to just after announcement time τ_3, with the thick dotted line being $\omega_3(\upsilon)$. It can be seen that the thick dotted line in Figure 8 is indeed the maximum of the three functions of Figure 7.

C. Comment on the used methodology

Describing a queueing system via a "state", i.e., a variable or a set of variables that suffice to describe the system between two arrivals (in our paper, announcements) and that needs to be updated at arrival times is a common analysis technique in queueing theory. The simplest example is Lindley's approach to tackle the G/G/1 queue [20], [17] (Section 8.1). In this way a recursive relation between the "state" at consecutive arrivals is obtained, in our case given by equation (4). Solving this equation by recursive substitution is also a well-known step, and leads in our case to the solution of equation (6). The difference is that in our case the state consists of a function, while in, e.g., Lindley's case it is one variable. For that reason it is not straightforward in our case to write an integral

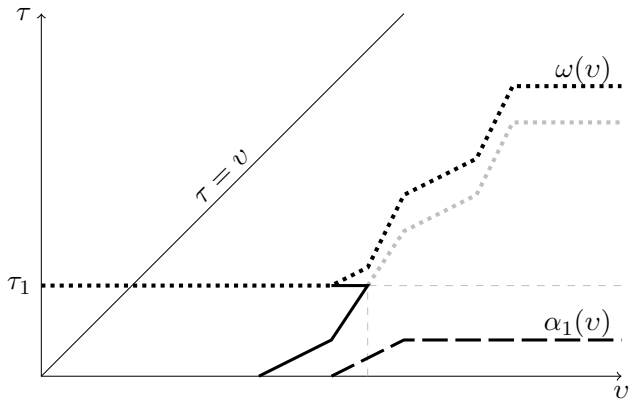

Fig. 4. The system just after announcement time τ_1.

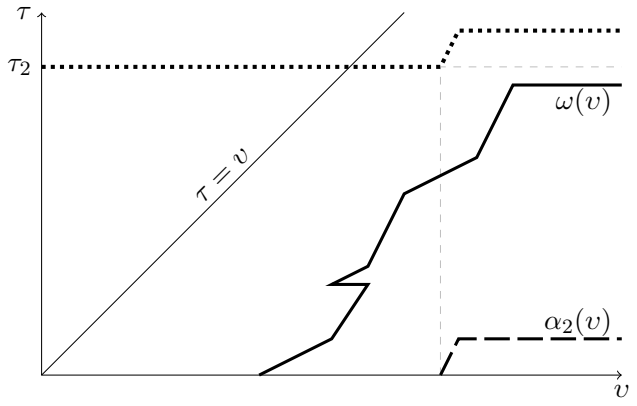

Fig. 5. The system just after announcement time τ_2.

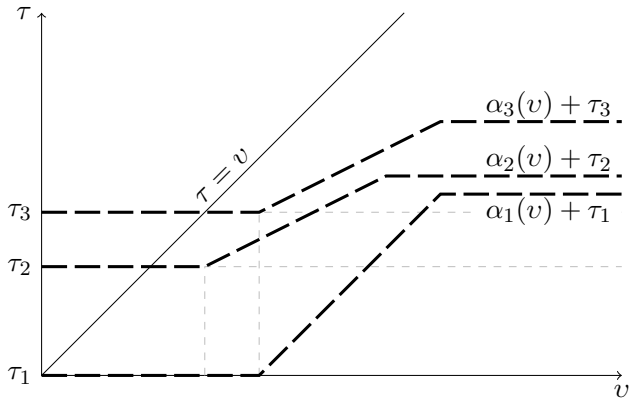

Fig. 6. Illustration of equation (6): deadline profiles are announced at times τ_1, τ_2 and τ_3.

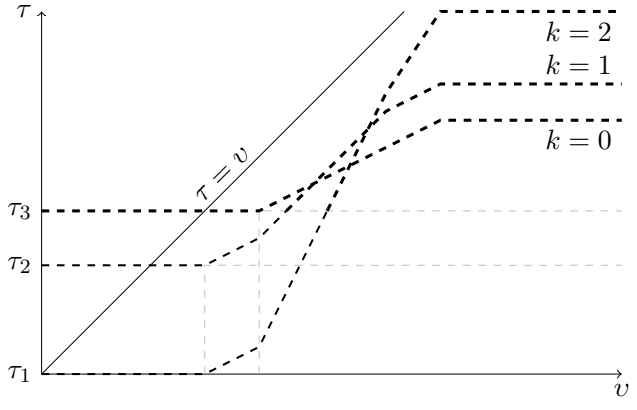

Fig. 7. Illustration of equation (6): the three functions over which the maximum needs to be taken to determine $\omega_3(v)$.

equation, analogue to Lindley's integral, that the statistics of the state obey. For that reason we revert to simulation to obtain numerical results.

D. Keeping the deadline margin essentially positive

The restriction that we impose on a well-behaving system is that the deadline margin $T(\tau)$ should hardly ever become negative. For that purpose we mark the announced

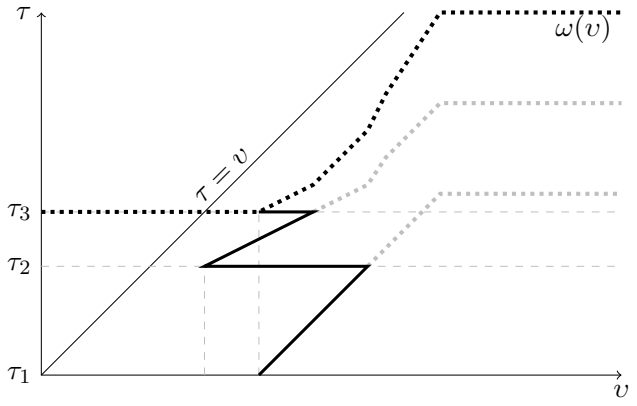

Fig. 8. Illustration of equation (6): the dotted thick line is $\omega_3(v)$.

requests that make the future deadline margin negative. These marked requests are considered to be requests that overload the system. Concretely we calculate the minimal future deadline margin d_i at the i-th announcement time τ_i as

$$d_i = \min_{\tau > \tau_i} \left\{ {\omega_i}^{-1}(\tau) - \tau \right\} \tag{7}$$

and assess the probability $\mathrm{Prob}[d_i < 0]$. The parameters of the system should be tuned so that this marking probability is kept smaller than a very small target value P (e.g., 10^{-5}).

It turns out that it is instructive to acquire the CCDF (complementary cumulative distribution function) of d_i for the following reason. Suppose that *all* instances of $A_i(t)$ are horizontally shifted over a time δ (i.e., *each* value of Δ_i is replaced by $\Delta_i + \delta$). It can be seen from equation (6) (or alternatively from equation (4)) that, as a consequence, the resulting $\omega_i(v)$ are horizontally shifted by δ, and hence, that a value δ is added to all minimum deadline margins d_i. So, the marking probability in case all workloads would have been shifted over a time δ is given by $\mathrm{Prob}[d_i + \delta < 0]$ or $\mathrm{Prob}[d_i < -\delta]$. This property allows us to considerably reduce the number of simulations we have to perform to investigate how the required server capacity evolves in terms of the time that requests are announced upfront.

IV. Results

We have implemented the system evolution (governed by equations (2) and (4)) in a simulator. The simulator generates requests. Although the theory is more generic, we consider the system with requests for a constant information rate over some interval (see [10]). Each request is determined by a quadruple $(\tau_i, \Delta_i, \theta_i, R_i)$, where i is the sequence number of the request, τ_i is the announcement time, $\tau_i + \Delta_i$ is the deadline of the first byte and $\tau_i + \Delta_i + \theta_i$ the deadline of the last byte and R_i is the rate at which the user is going to consume the information. Loading the system consists in generating a list of requests. The simulator maintains and updates the function $\omega_i(v)$ from announcement time to announcement time given a server capacity C and calculates the CCDF of the minimal future deadline margin d_i (see equation (7)) at each announcement time. The (measured) CCDF of d_i provides enough information to determine the additional time δ the requests need

to be known upfront to keep the marking probability below a target value P (given the server capacity C).

A. Loading the system

We generate the announcement times τ_i according to a Poisson process. The time unit is a day, such that the integer part $\lfloor \tau_i \rfloor$ of τ_i corresponds to the day of announcement and the fractional part $\{\tau_i\} = \tau_i \bmod 1 = \tau_i - \lfloor \tau_i \rfloor$ of τ_i corresponds to the time of day of the announcement. We consider two cases. In the first case the announcement times are generated by a stationary Poisson process with arrival rate λ.

We also want to investigate the impact of fluctuations that have a periodic, daily-repeating pattern besides random fluctuations. Therefore, in the second case, we generate announcement times according to a non-stationary Poisson process with periodic arrival rate. In order to generate such announcement times we first generate stationary Poisson arrivals ${\tau_i}'$ (with arrival rate λ). We then adjust the fractional parts $\{{\tau_i}'\}$ of these announcement times according to

$$\{\tau_i\} = \int_0^{\{{\tau_i}'\}} f(x)\,\mathrm{d}x\,, \tag{8}$$

where the function $f(x) = 1 + c_1 \sin(2\pi x) + c_2 \sin(4\pi x)$ describes the diurnal pattern to be repeated. The actual announcement times τ_i can then be constructed as

$$\tau_i = \lfloor {\tau_i}' \rfloor + \{\tau_i\}\,. \tag{9}$$

Generating non-stationary Poisson announcement times via this kind of time-warping was introduced in [9] (Corollary 7.8) and extended in [15].

Choosing the coefficients $c_1 = -0.580$ and $c_2 = -0.473$ has the effect that the announcement times τ_i follow a realistic pattern with little activity between midnight and morning hours, moderate activity around noon and high activity at evening times, as depicted in Figure 9. Note that for $c_1 = c_2 = 0$, no adjustment is made to the announcement times, such that this case boils down to a stationary Poisson process.

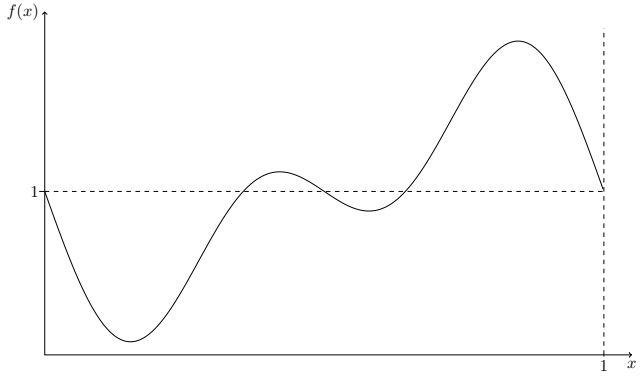

Fig. 9. Evolution of the arrival intensity during the course of a day.

The duration θ_i of each flow is set to a constant Θ equal to $\frac{1}{24} = 0.04167$ day (1 hour). The rate R_i of each flow is set to 1 unit per time unit. The capacity C is expressed in the same units as R_i. We perform experiments with all $\Delta_i = 0$, but as explained in Section III-C the CCDF of d_i allows us to determine which (constant) value Δ_i should take to yield a marking probability below a target value.

We now take a look at the effect of diurnal patterns in the arrival process on the minimal deadline margin. Therefore, we generated two streams of announcement times, one from a stationary Poisson process, and one originating from a non-stationary Poisson process as described above. In both cases, the average arrival rate λ is set to 1000 flows per day, the duration Θ is set to 1 hour, δ is set to 0 and the available bandwidth allows for the simultaneous transmission of 45 flows. For the non-stationary Poisson process, the parameter c_1 is set to -0.580 whereas c_2 is chosen to be -0.473. Figure 10 then shows for each flow the announcement time τ_i on the horizontal axis and the corresponding minimal deadline margin d_i as described in (7) on the vertical axis.

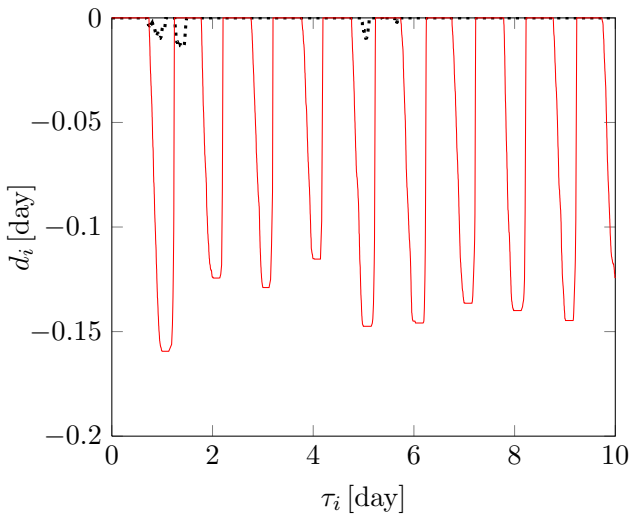

Fig. 10. Minimal deadline margin as a function of the announcement time for the case $\lambda = 1000$, $\Theta = \frac{1}{24}$, $R = 1$ and $C = 45$, for a stationary Poisson process (dotted line) and a non-stationary Poisson process (full line).

From Figure 10, the impact of the diurnal patterns becomes manifest. Flows created by the stationary Poisson process (dotted line) are distributed more or less evenly in time, resulting in modest peaks in the bandwidth requirement. This seldom results in a negative deadline margin and when it does, the lag sustained by the system is small. In case of the non-stationary Poisson process (full line), the lag builds up during each evening time peak. This results in a diurnal pattern for the minimal deadline margin, which on top is much more distinct, reflecting the fact that the peak bandwidth requirement for the non-stationary arrival process is much larger.

In the example of Figure 10, we consider the case where requests are not announced upfront ($\Delta_i = \delta = 0$). Note that we defined the minimal future deadline margin d_i only on announcement times, as we only need d_i at those times to check if a new request would violate a deadline or not. Hence, we calculated d_i only at those times in the simulation. As on announcement times there is always immediate work to do (since $\Delta_i = 0$), the minimal deadline

margin d_i shown in Figure 10 is never positive in this case. We could have defined the minimum future deadline margin for any other time, other than announcement times. In that case the curve in Figure 10 could have become positive in between some announcement times. We opted not to calculate the minimum future deadline at intermediate times as it would have considerably slowed down our simulation.

B. Time the requests need to be known upfront

1) Stationary Poisson process: We first consider the case of a stationary Poisson process. The average arrival rate λ is either 1000, 10000 or 100000 flows per day, the duration Θ is set to $\frac{1}{24}$ day, the coefficients c_1 and c_2 are 0 (since it is a stationary Poisson process) and simulations are run over 100 days. We repeated each simulation 10 times to assess the statistical fluctuations on the time δ the requests need to be announced upfront.

Figures 11-13 show the results. In each of the figures we show the minimal time the requests need to be known upfront for the marking probability to be below a target value $P = 10^{-5}$ given a certain capacity. Alternatively the figure shows the minimal capacity that is needed to keep the marking rate below $P = 10^{-5}$ as function of the time δ the requests are announced upfront. To be able to easily compare the various cases, the vertical axes are scaled with λ. It is intuitively clear that if $\delta = 0$, the capacity is the $(1 - P)$-quantile of number of requests that arrive in an interval of duration Θ and that if $\delta = \infty$ the required capacity is the average arrival rate, i.e., $\lambda\Theta$. From the figures we can determine the value of δ for which the capacity is substantially lower than the peak and close enough to the average. The figures show that, in case of a stationary Poisson process, if the average number of simultaneous users (i.e., $\lambda\Theta$) increases, the potential gain in capacity decreases, where the capacity gain is defined as the ratio between the required capacity when the requests are not known upfront to the required capacity when the requests are known an infinite time upfront. For a small number of simultaneous users (e.g., 41.67) there is still a considerable gain, while for a large number of simultaneous users (e.g., 4167) the gain is marginal. In case there is a capacity gain, in order to attain this gain, the requests need to be known upfront a multiple of Θ (e.g., 2Θ).

2) The impact of diurnal patterns: Next we consider the case of a non-stationary Poisson process in which the arrival rate varies according to realistic diurnal patterns. We use the same parameters as before, but set $c_1 = -0.580$ and $c_2 = -0.473$. Figures 14-16 show the results. Again we simulated 100 days and ran each simulation 10 times to assess the statistical fluctuation. As in the case above we show the minimal capacity that is needed to keep the marking rate P below 10^{-5} as function of the time δ the requests are known upfront. The average rate (and hence, the rate that the required capacity should tend to as $\delta \to \infty$) is still $\lambda\Theta$. The figures show that, in case of a non-stationary Poisson process, there is a considerable gain for all values of the average number of simultaneous users (i.e., $\lambda\Theta$). The capacity gain in all cases is about the same and almost equal to the maximal value of $f(x)$. In order to attain this capacity gain, the requests need to be known upfront a considerable fraction of the day (e.g., 0.20 day).

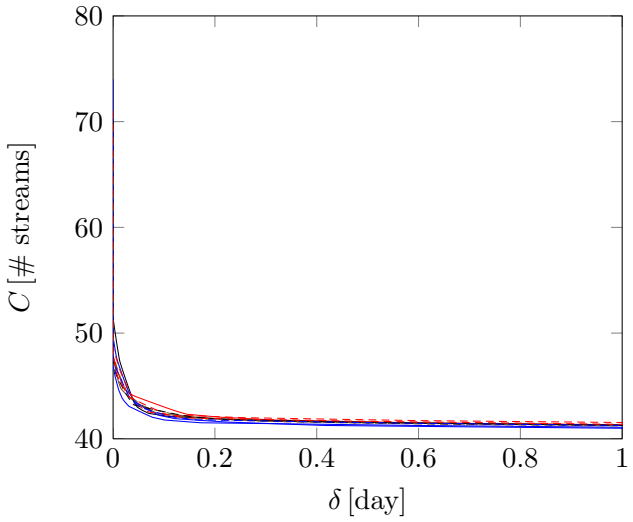

Fig. 11. Required server capacity as a function of the time the requests are known upfront for the case $\lambda = 1000$, $\Theta = \frac{1}{24}$, $R = 1$, $P = 10^{-5}$ and $c_1 = c_2 = 0$.

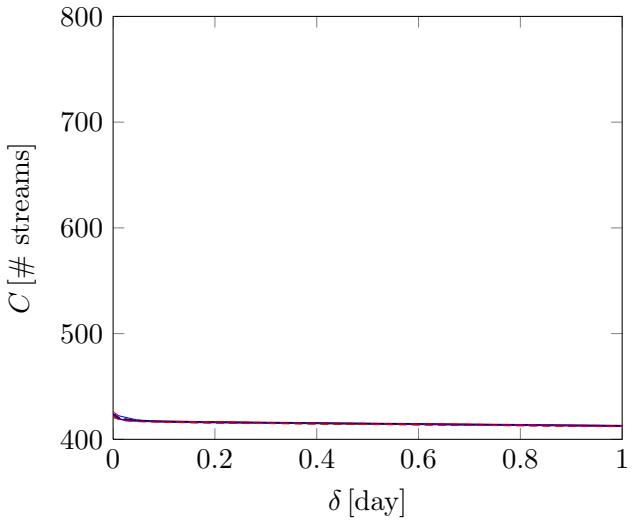

Fig. 12. Required server capacity as a function of the time the requests are known upfront for the case $\lambda = 10000$, $\Theta = \frac{1}{24}$, $R = 1$, $P = 10^{-5}$ and $c_1 = c_2 = 0$.

V. Conclusions

We have studied a system in which users can announce requests for information upfront by specifying a deadline profile. In contrast to studies previously reported in literature, in which one deadline is associated with a complete job, the deadline profile defined in this paper precisely indicates by when a user needs each constituent of the requested information. Specifying such a deadline profile is typical for streaming (multimedia) services where the information is not all needed at once, but can trickle in at a predetermined (possibly variable) rate. The server can use this detailed timing information provided by the clients to

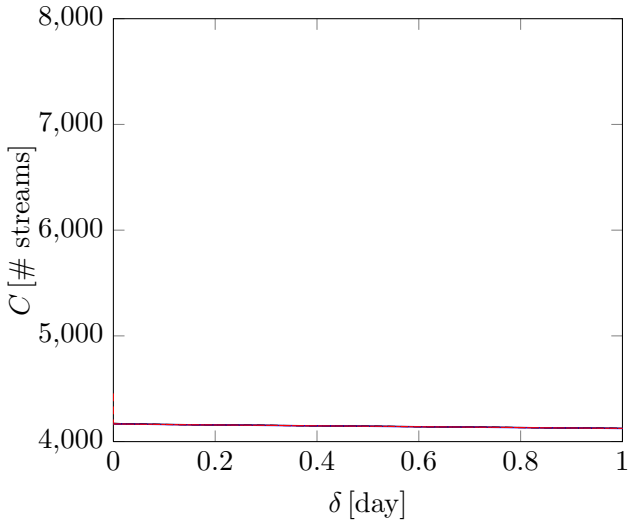

Fig. 13. Required server capacity as a function of the time the requests are known upfront for the case $\lambda = 100000$, $\Theta = \frac{1}{24}$, $R = 1$, $P = 10^{-5}$ and $c_1 = c_2 = 0$.

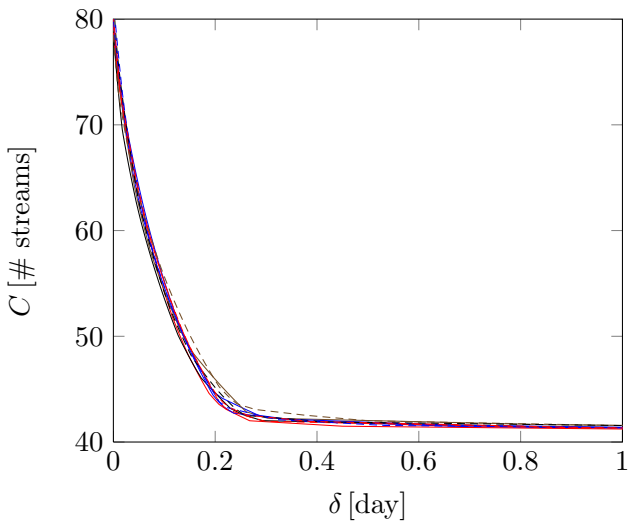

Fig. 14. Required server capacity as a function of the time the requests are known upfront for the case $\lambda = 1000$, $\Theta = \frac{1}{24}$, $R = 1$, $P = 10^{-5}$, $c_1 = -0.580$ and $c_2 = -0.473$.

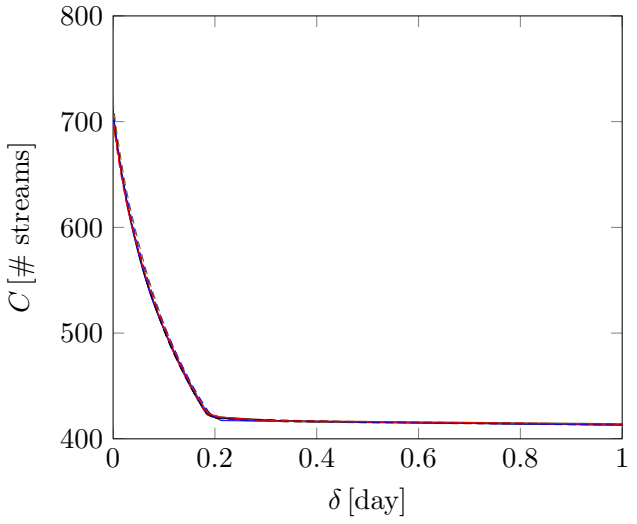

Fig. 15. Required server capacity as a function of the time the requests are known upfront for the case $\lambda = 10000$, $\Theta = \frac{1}{24}$, $R = 1$, $P = 10^{-5}$, $c_1 = -0.580$ and $c_2 = -0.473$.

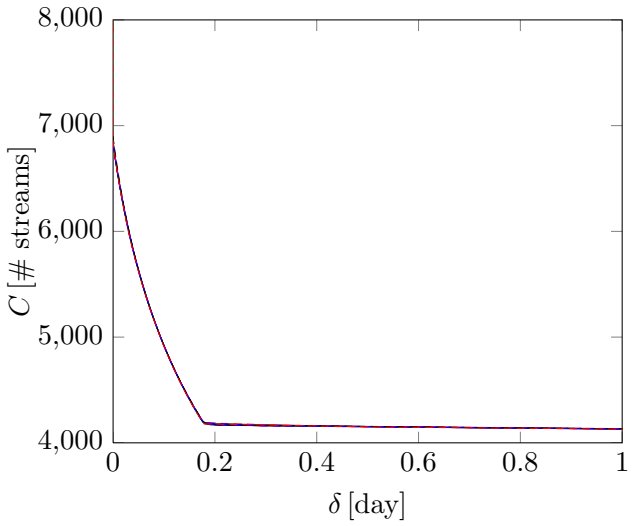

Fig. 16. Required server capacity as a function of the time the requests are known upfront for the case $\lambda = 100000$, $\Theta = \frac{1}{24}$, $R = 1$, $P = 10^{-5}$, $c_1 = -0.580$ and $c_2 = -0.473$.

schedule the work better. By shifting work from future busy periods to quiet periods, the server requires less capacity. In fact the server can operate at mean rate rather than at peak rate, if the requests are known upfront long enough. We described the scheduling system making use of this detailed timing information in a mathematical way, by identifying the system state and by providing an evolution equation for this state. We defined a well-behaving system as a system where practically none of the deadlines are violated. We proved a shifting property that allowed us to calculate the deadline violation probability for a whole range of scenarios in one simulation, which considerably reduces the required number of simulations. Finally, we identified via simulations how much the requests need to be known upfront to reduce the required capacity from peak to mean rate in case the announcement times are governed by either a stationary Poisson process or a non-stationary Poisson process in which the arrival rate varies according to diurnal patterns observed in reality.

ACKNOWLEDGEMENT

This research has been funded by the Interuniversity Attraction Poles Programme initiated by the Belgian Science Policy Office.

REFERENCES

[1] R. Adams, Active queue management: a survey, IEEE communications surveys & tutorials, vol. 15, no. 3, 2013, pp. 1425-1476.

[2] M. Allman, V. Paxson, E. Blanton, TCP congestion control, IETF RFC 5681, September 2009.

[3] G. Appenzeller, I. Keslassy, N. McKeown, Sizing router buffer, ACM SIGCOMM Computer Communication Review, vol. 34, no. 4, 2004, pp. 281-292.

[4] Z. Avramova, D. De Vleeschauwer, K. Laevens, S. Wittevrongel, H. Bruneel, Modelling H.264/AVC VBR video traffic: comparison of a Markov and a self-similar source model, Telecommunication Systems, vol. 39, no. 2, 2008, pp. 91-102.

[5] F. Baccelli, A. Chaintreau, D. De Vleeschauwer, D.R. McDonald, A mean-field analysis of short lived interacting TCP flows, Proceedings of ACM SIGMETRICS, New York (US), 12-16 June 2004, pp. 343-354.

[6] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, An architecture for differentiated services, IETF RFC 2475, December 1998.

[7] O. Brun, A.A. Sheikh, J. Garcia, Flow-level modelling of TCP traffic using GPS queueing networks, Proceedings of the 21st International Teletraffic Congress (ITC), Paris (France), 15-17 September 2009, pp. 1-8.

[8] J. Choe, N.B. Shroff, A central-limit-theorem-based approach for analyzing queue behavior in high-speed networks, IEEE/ACM Transactions on Networking, vol. 6, no. 5, 1998, pp. 659-671.

[9] E. Çinlar, Introduction to Stochastic Processes, Prentice-Hall, Englewood Cliffs, NJ 1975.

[10] K. De Schepper, B. De Vleeschauwer, C. Hawinkel, W. Van Leekwijck, J. Famaey, W. Van de Meerssche, F. De Turck, Shared Content Addressing Protocol (SCAP): optimizing multimedia content distribution at the transport layer, Proceedings of the IEEE network operations and management symposium (NOMS), Hawaii (USA), 16-20 April 2012, pp. 302-310.

[11] D. De Vleeschauwer, A. Van Moffaert, J. Janssen, M.J.C. Büchli, G.H. Petit, B. Steyaert, Determining the number of packet-based phones that can be supported by one access node, Proceedings of the 14th ITC Specialists Seminar on Access Networks and Systems (ITCSS01), Girona (Spain), 25-27 April 2001, pp. 197-204.

[12] A. Dhamdhere, H. Jiang, C. Dovrolis, Buffer sizing for congested Internet links, Proceedings of IEEE INFOCOM 2005, vol. 2, Miami (US), 13-17 March 2005, pp. 1072-1083.

[13] B. Doytchinov, J. Lehoczky, S. Shreve, Real-time queues in heavy traffic with earliest-deadline-first discipline, The Annals of Applied Probability, vol. 11, no. 2, 2001, pp. 332-378.

[14] V.G. Abhaya, Z. Tari, P. Zeephongsekul, A.Y. Zomaya, Performance analysis of EDF scheduling in a multi-priority preemptive M/G/1 queue, IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 8, 2014, pp. 2149-2158.

[15] I. Gerhardt, B.L. Nelson, Transforming renewal processes for simulation of nonstationary arrival processes, INFORMS Journal on Computing, 2009, pp. 1-11.

[16] M. Gidlund, G. Wang, Uplink scheduling algorithms for QOS support in broadband wireless access networks, Journal of communications, vol. 4, no. 2, 2009, pp. 113-142.

[17] L. Kleinrock, Queueing systems, Volume 1: Theory, Wiley Interscience, New York, 1975.

[18] Ł. Kruk, J. Lehoczky, K. Ramanan, S. Shreve, Heavy traffic analysis for EDF queues with reneging, The Annals of Applied Probability, vol. 21, no. 2, 2011, pp. 484-545.

[19] W.E. Leland, M.S. Taqqu, W. Willinger, D.V. Wilson, On the self-similar nature of Ethernet traffic (extended version), IEEE/ACM Transactions on Networking, vol. 2, no. 1, 1994, pp. 1-15.

[20] D.V. Lindley, The theory of queues with a single server, Proceedings Cambridge philosophical society, vol. 48, 1952, pp. 277-289.

[21] P. Moyal, On queues with impatience: stability, and the optimality of earliest deadline first, Queueing Systems, vol. 75, no. 2-4, 2013, pp. 211-242.

[22] I. Norros, A storage model with self-similar input, Queueing Systems, vol. 16, no. 3-4, 1994, pp. 387-396.

[23] J. Postel, User Datagram Protocol, IETF RFC 768, August 1980.

[24] S. Shenker, J. Wroclawski, General characterization parameters for integrated service network elements, IETF RFC 2215, September 1997.

[25] W. Sun, K.G. Shin, Coordinated aggregate scheduling for improving end-to-end delay performance, Twelfth IEEE International Workshop on Quality of Service (IWQOS 2004), Montreal (CA), 7-9 June 2004, pp. 77-86.

[26] S. Wittevrongel, S. De Vuyst, C. Sys, H. Bruneel, A reservation-based scheduling mechanism for fair QoS provisioning in packet-based networks, Proceedings of the 26st International Teletraffic Congress (ITC), Karlskrona (Sweden), 9-11 September 2014, pp. 1-8.