Queueing Delays in Rate Controlled ATM Networks

Anindo Banerjea
CS Division, University of California, Berkeley, CA 94720
and International Computer Science Institute, Berkeley

Srinivasan Keshav
AT&T Bell Laboratories, 600 Mountain Ave., Murray Hill, NJ 07974

Abstract

This paper addresses the problem of finding the worst case end-to-end delay and buffer occupancy bounds in ATM networks with rate-controlled, non-work conserving servers. A theoretical framework is constructed to analyze such servers in isolation and in tandem. The analysis is based on a simple fluid model, but care is taken so that the computed delay and buffer occupancy values are upper bounds on actual values. A simple algorithm is presented to perform these calculations in linear time. Simulation results compare the computed worst case delays with the actual delays obtained on some simple network topologies. The algorithm is found to predict node delays well for bursty input traffic, but poorly for smooth input traffic. Buffer requirements are predicted well in both cases.

1. Introduction

Recent work has shown that *framed*, *non-work-conserving* servers can provide end-to-end delay bounds to users who need strict guarantees on network performance [15]. By *framed* we mean that the servers work on the basis of a fixed size interval called the *frametime*, during which they allocate a number of transmission slots to each channel being served. We assume that during each slot a fixed size packet or ATM *cell* is transmitted on the output trunk. By *non-work-conserving* we mean that given enough cells in the queue, the server will send out exactly the allocated number of ATM cells (the *chunk-size*) in each frame, even if this will leave the output trunk idle. The server is free to serve these cells during any portion of the frame. If there are fewer than *chunksize* cells in the queue for the channel, the server will send out the available cells, then use the free slots for non-real time traffic. Examples of such servers are those that obey the Hierarchical Round Robin (HRR) and the Stop-and-Go service discipline [8, 7]. These servers simultaneously serve a number of frames of different lengths so that a choice of frametimes is available during channel establishment.

While earlier work considered the behavior of a single server [8], and of 'smooth' traffic traversing a sequence of identical servers [7, 15], the general case, where a circuit carrying bursty traffic may be served at a different framing interval at each server, was not analyzed. This paper determines the upper bound on the end-to-end queueing delays and buffer requirements at each queueing point for a simplex channel established over any sequence of framed non-work conserving servers. We present a formal modeling of a tandem of such servers and use it to compute the worst case end-to-end delay and buffer requirements. These are then compared with simulation results.

2. Previous work

Traditionally, end-to-end delays have been computed for Poisson arrivals, and for the average case, since this makes queueing theoretic analysis easier. In contrast, we will assume parametrically constrained inputs and compute worst-case delays. This type of analysis was first proposed by Cruz who used it to analyze some work-conserving scheduling disciplines [2, 3]. In recent work, Cruz has also computed delay bounds for non-work conserving disciplines, but this analysis does not explicitly consider the effects of different frame sizes at different servers, resulting in weaker delay bounds [4]. Kurose has computed tight delay bounds not only for the maximum delay, but also for the delay distribution [10]. However, this work assumes a work-conserving service discipline, and a stronger characterization of input traffic than ours.

Parekh and Gallager [12, 13] have computed worst case delays assuming sources were leaky bucket compliant and the servers used the Packetized Generalized Processor Sharing (PGPS) scheduling discipline. This is a

A. Banerjea was supported by the National Science Foundation and the Defense Advanced Research Projects Agency (DARPA) under Cooperative Agreement NCR-8919038 with the Corporation for National Research Initiatives, by AT&T Bell Laboratories, Hitachi, Ltd., Hitachi America, Ltd., Pacific Bell, the University of California under a MICRO grant, and the International Computer Science Institute. The views and conclusions contained in this document are those of the authors, and should not be interpreted as representing official policies, either expressed or implied, of the U.S. Government or any of the sponsoring organizations.

work-conserving discipline, thus their analysis is necessarily more intricate than ours, since with non-work conserving disciplines, competing traffic streams interact minimally. On the other hand, we model the phenomenon of 'slippage' (to be described below) and this introduces complexities that are not found in their analysis. We have used their notion of 'greedy' sources extensively in our work.

Our formal modelling of a message stream and of a server as a transformation on a message stream is based on the recent work of Low and Varaiya [11]. While they use their model mainly to analyze burstiness, we use it for exact transient analysis of a tandem of rate-controlled servers.

Our server model is based on the HRR servers described in [8]. The Stop-and-Go discipline proposed by Golestani [7] is closely related to this discipline, and the techniques we propose can be used to analyze Stop-and-Go as well. As currently described in the literature, the delay and delay jitter bounds for Stop-and-Go are valid only when a connection is allocated the same frametime at all servers along the path, and the source is assumed to be smooth at the time scale of a frametime. Our analysis can be trivially extended to provide delay and delay jitter bounds for Stop-and-Go when a connection can be assigned to different frametimes at different servers, or when it is not smooth at the time scale of a frametime.

3. Network model

In our model a channel is associated with a static path through the network and has state and reserved resources at each switch along this path. Since a rate-controlled server isolates the traffic flowing along each channel from the traffic on other channels, we focus our attention on the performance of a single channel in the network. We first characterize the input traffic, then present a series of definitions, lemmas and theorems leading to an algorithm to compute the worst case delay and buffer bounds.

3.1. Input traffic model

We characterize the traffic entering the channel with three parameters as suggested in [5, 6]:

$x_{\min}$ The minimum cell inter-arrival time.

x_{ave} The minimum average cell inter-arrival time, over **any** time interval of length I.

I The averaging interval for calculating x_{ave}.

Recent work has shown that this model accurately describes many types of rate-controlled traffic expected in high-speed networks [14]. We assume that the user of the channel is required to obey these restrictions on the input traffic, and the delay and buffer values calculated need only hold if these restrictions are not violated.

3.2. Definitions

Our notation draws from, and extends, that used in Reference [11]. We define a *message* as a temporal flow of data, represented by a bounded non-negative function $m(t)$, $0 \le t \le T$, where $m(t)$ is the instantaneous rate at time t in cells/sec. The interval $[0, T]$, $T < \infty$, is called the *time space* and is denoted by *TS*. The *cumulative message curve* $M(t) = \int_0^t m(t)$ is the amount of the message seen till time t. Its inverse function $M^{-1}(m_0)$ is defined as the *earliest* time at which the prefix of the message up to m_0 has been seen. The interval $[0, M(T)]$ is the *message space* and will be denoted by *MS*.

A *server* is a device that takes an arriving message $m_{in}(t)$ and transforms it into a departing message. The arriving message at a server is defined by its arrival curve $A(t) = \int_0^t m_{in}(t)$ and the departing message is defined by the service curve $S_A(t) = \int_0^t m_{out}(t)$. Every server is associated with a queue. A message is stored in the queue from the time that it arrives till the time it is served. The instantaneous amount of data in the server (*queue size*) is denoted by $Q_A(t) = A(t) - S_A(t)$. The *maximum queue size* Q^* is defined as $\underset{t \in TS}{Max}\, Q_{A(t)}$. The *queueing delay* $D_A(m_0) = S_A^{-1}(m_0) - A^{-1}(m_0)$ is the time between the arrival and departure of the m_0 prefix of the message. The *maximum queueing delay* D^* is $\underset{m \in MS}{Max}\, D_A(m)$.

A cumulative message curve $M(t)$ will be called *piecewise linear constrained (PLC)* if it is made of piecewise linear segments. A *greedy* source is one that sends data at the maximum rate possible and at the earliest moment possible. A *parametrically constrained source (PCS)* is a source that obeys the following constraints:

(a) the rate of message transmission $m(t) \le 1/x_{\min}$

(b) over all intervals of time of length I and all time t, $\int_t^{t+I} m(t)dt \le I/x_{ave}$,

where $x_{\min}$, x_{ave} and I are the three parameters constraining the source. A *parametrically constrained greedy source (PCGS)* will send at rate $1/x_{\min}$ until I/x_{ave} packets are sent, then will remain idle until time I. After this, the behavior will repeat, until the entire message has been sent. Clearly, a PCGS is PLC. The cumulative message curve $M(t)$ for a PCGS source is shown by the dashed line in Figure 1.

For a given set of parametric constraints $x_{\min}$, x_{ave}, and I a specific interval $[t_0, t_0+I] \subseteq TS$ is denoted by *IS* and is completely specified by choosing t_0, the origin for the interval. A PCS obeys the average rate rule over any interval *IS*.

A server is called a *fixed rate server* with service rate ρ if its output message

$m_{out}(t) = \Phi(m_{in})(t)$
$= m_{in}(t)$ if $m_{in}(t) \le \rho$ and $Q_A(t) = 0$
$= \rho$ if $m_{in}(t) > \rho$ or $Q_A(t) > 0$.

3.3. Exact analysis of a tandem of servers

We now state some theorems about the behavior of fixed rate servers with PCS inputs. We first look at a single server, then consider servers operating in tandem, and finally introduce, define and analyze servers with slippage. For reasons of space, we defer proofs to Reference [1].

3.3.1. Single server

Lemma 1.1: If traffic generated by a PCS $M_{in}(t)$ is input to a fixed rate server with rate $\rho \ge 1/x_{ave}$, then any interval of time during which $Q_{M_{in}}(t) > 0$ continuously has to be of length $< I$.

Lemma 1.2: If the input to a fixed rate server is PCS then so is the output.

Theorem 1.1: For a fixed rate server of rate $\ge 1/x_{ave}$, the largest possible queueing delay D^* and the largest possible queueing size Q^* over all possible PCS input messages are achieved by a parametrically constrained greedy source (PCGS).

This tells us that for a single fixed rate server, the worst case queueing behavior is observed on sending in a PCGS input to the server, and occurs within the first I time units. Thus, we can compute D^* and Q^* simply by studying the behavior of a PCGS traffic input over a single time interval of duration I.

3.3.2. Tandem of servers

A tandem of servers is defined to be a set of servers numbered $1 \cdots N$ such that the output of server i is the input to server $i+1$. There may be a fixed propagation delay in between servers. In this analysis, the delay is irrelevant and is assumed to be 0. For a tandem of fixed rate servers, the service curve of server i is the arrival curve for server $i+1$.

Lemma 2.1: If the input to a fixed rate server is from a PCGS, then the output is PCS. If $\rho > 1/x_{\min}$ then output is identical to the input, otherwise it looks like a PCGS with $x'_{\min} = 1/\rho$ and $x'_{ave} = x_{ave}$.

Theorem 2.1: All of the fixed rate servers in a tandem achieve the largest possible delay and the largest possible queue size when the input to server 1 is PCGS as compared to any other PCS source with the same parameters. Moreover, this occurs within the first I time units of the PCGS stream.

Theorem 2.1 extends the result from the case of a single server to that of a tandem of servers. Thus, even for a tandem, we can calculate bounds by studying a single traffic input for a limited period of time.

3.3.3. Slippage

Consider a source that has been allocated a service rate of, say, 1 cell per frame at some server. A source may try to fool the server in the following way: it sends one cell at the start of the frame time, and when it knows that the that cell has been served, send another cell within the same frame time. If the server schedules the VCI for service immediately upon the arrival of the second cell, then the source will be able to have two cells served per frame time whereas it would have paid for only one. To prevent this, some servers maintain two sets of VCI service lists [8]. If a cell arrives on a VCI that has an idle queue, the VCI is placed in the 'idle' service list. At the end of the frame time, the 'idle' and 'active' service lists are swapped. This prevents sources from over-using the server, but at the cost of added delay for the first cell that arrives to an idle VCI. We call this additional delay *slippage*. Slippage may also occur if a switch stores state information for inactive VCIs in secondary storage, so that when a cell arrives to an idle VCI, there may be some time lost in fetching VCI state.

After the slippage time, the flow is continuous until the queue becomes empty. The channel is then again made inactive. In a HRR server, the channel remains inactive up to one frametime, which is the slippage for the server. A *bounded slippage fixed rate server* with slippage bound s and rate ρ has the following properties.

(a) If $Q_{M_{in}}(t) = 0$ then $m_{out}(t) = 0$

(b) If $Q_{M_{in}}(t_1) = 0$ and $Q_{M_{in}}(t_1+^{\dagger}) > 0$ then $\exists_{t_2 \in TS}\{t_2 \mid t_2 \le t_1+s\}$ and $\exists_{t_3 \in TS}\{t_3 \mid t_3 > t_2\}$ such that $m_{out}(t) = 0$ in $[t_1, t_2)$, $m_{out}(t) = \rho$ in $[t_2, t_3)$ and $Q_{M_{in}}(t_3) = 0$.

In other words the slippage (the length of the interval during which the server remains inactive after $Q_{M_{in}}(t)$ becomes non-zero) varies from 0 to s.

A *fixed slippage fixed rate server* with fixed slippage s and rate ρ is a server of the above type such that in (b) above $t_2 = t_1+s$. This server is deterministic and the slippage is always s.

The following theorems and lemmas are useful when $s \ll I$. We expect this to be the common case.

Lemma 3.1: If the message curve $M_{in}(t)$ generated by a PCS is input to a bounded slippage fixed rate server (ρ, s)

† $f(t_1+)$ is the limit from the right of f(t) at t_1. That is, the queue becomes non zero after t_1.

having $\rho > \frac{I}{x_{ave}(I-s)}$ then any interval of time during which $Q_{M_{in}}(t) > 0$ continuously has to be of length $< I$

Lemma 3.2: If the message curve $M_{in}(t)$ generated by a PCS is input to a bounded slippage fixed rate server (ρ,s) having $\rho > \frac{I}{x_{ave}(I-s)}$ then the output is PCS.

Theorem 3.1: The queueing delay and queuesize of a *bounded* slippage fixed rate server (ρ,s) having rate $\rho > \frac{I}{x_{ave}(I-s)}$ over all possible PCS inputs are bounded by the D^* and Q^* achieved by a PCGS input to a fixed rate server with *fixed* slippage s.

By Lemma 3.1 and the periodic nature of a PCGS D^* and Q^* occur within one interval of length I. Thus, we only need analyze the system for the first I time units, with a PCGS source, to find the delay and queue size bounds.

3.3.4. Tandem of fixed rate servers with slippage

We now introduce a relative definition of greediness. A message curve $M(t)$ is said to be greedy with respect to some constraint set *CS* where *CS* is a set of message curves if $\forall_{N(t)\in CS} \; \forall_{m\in MS} \; M^{-1}(m) \le N^{-1}(m)$.

A server maps a message curve $M(t)$ into a output message curve $S_M(t)$. Thus it can also be thought to map sets of messages corresponding to all legal inputs ($\in CS$) to sets of messages corresponding to all the outputs to the messages in the input set *CS*. Loosely, we can also use the symbol Φ to denote this transformation.

Lemma 4.1: If the input to a fixed slippage fixed rate server is greedy with respect to the set *PCS* of message curves which are PCS, the output is greedy with respect to the set of message curves $\Phi(PCS)$ which are the outputs for the message curves in the input set.

Theorem 4.1: The queueing delay and queuesize of a tandem of *bounded* slippage fixed rate servers (ρ_i,s_i) having rate $\rho_i > \frac{I}{x_{ave}(I-s)}$ over all possible PCS input messages are bounded by the D^*s and Q^*s achieved by a PCGS input to a tandem of *fixed* slippage fixed rate servers (ρ_i,s_i).

From the theorems above, we can compute upper bounds on the queueing delay and queue size for a tandem of bounded slippage fixed rate servers (with $\rho_i > \frac{I}{x_{ave}(I-s)}$) for any parametrically constrained input by assuming PCGS input to server 1 and assuming fixed slippage at each of the servers. This is a simplified and deterministic system with fixed and computable curves $M_i(t)$ at the output of each server i. Moreover, by Lemma 3.1, we only need to look at any one interval *IS*.

The system can be analyzed graphically in a manner similar to [13] as follows: We start with $M_0(t)$ as the greedy input curve. Apply the series of transformations Φ_i as defined by the definition for fixed slippage fixed rate servers and the parameters (ρ,s), to this curve for $i = 1...N$, the number of servers in the tandem, to get the series of intermediate message curves $M_i(t)$.

One way to do this would be to construct $M_0(t)$ as a set of points in the $MS\times IS$ plane, using enough points to ensure accuracy, then apply Φ_1 to each of them to get the output from the first server, and so on for each Φ_i. This is computationally expensive, and also requires too much communication in case the computation needs to be done in a distributed fashion (all the points in $M_i(t)$ need to be carried across from server i to i+1). We need some short hand to describe $M_i(t)$.

3.3.5. Envelope of the curve

We define the envelope of $M_i(t)$, $\mathcal{E}_i(t)$, by construction as follows. $\mathcal{E}_0$ is the PCGS message curve input to server 1, as shown by the dashed line in Figure 1. $\mathcal{E}_i$ the envelope of the output from server i (with rate ρ_i and slippage s_i) is constructed from $\mathcal{E}_{i-1}$ the envelope of the output from server $i-1$ as follows :

Let:

$t_0 = Min\{t \in TS \mid \mathcal{E}_{i-1}(t) = 0 \text{ and } \mathcal{E}_{i-1}(t+) > 0\}$

$t_1 = Min\{t \in TS \mid \mathcal{E}_{i-1}(t) = \rho_i(t-t_0-s_i)$

Then: $\mathcal{E}_i(t)$

$= 0$ for $t < t_0+s_i$

$= \rho_i(t-t_0-s_i)$ for $t_0+s_i \le t < t_1$

$= \mathcal{E}_{i-1}(t)$ for $t \ge t_1$

$\mathcal{E}_i(t)$ can be thought of as an upper bound on $M_i(t)$. At t_0 server i receives the first cell into an empty queue, at t_0+s_i it becomes active, and Q goes to zero at time t_1. In Figure 1 for server 1, t_0 is the origin, and t_1 is the x-coordinate of point **A**.

Figure 1 shows how $\mathcal{E}_1$ (drawn in bold lines) is defined from $\mathcal{E}_0$ (in the dashed lines). Figure 2 shows $\mathcal{E}_2$ in bold and $\mathcal{E}_1$ in the dashed lines.

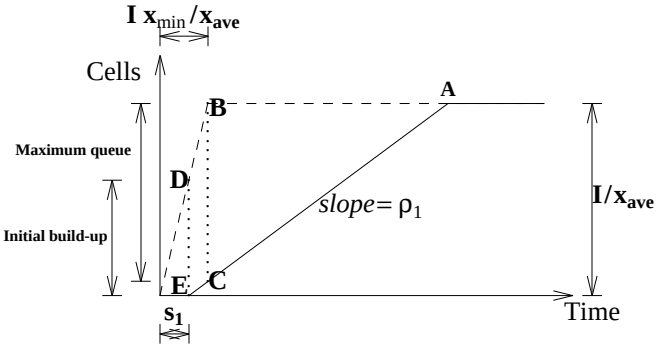

Figure 1. Calculating $\mathcal{E}_1$ from $\mathcal{E}_0$.

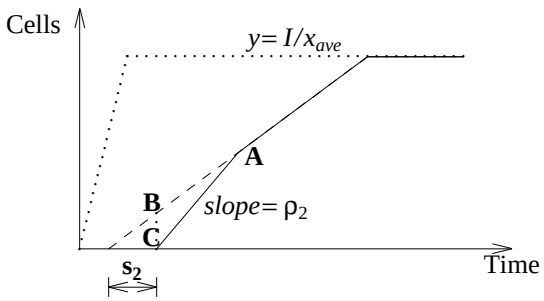

Figure 2. Calculating ε_2 from ε_1.

For the rest of this section $M_i(t)$ refers to the message curve output from server i if the input to server 1 is PCGS.

Lemma 5.1: The maximum queueing delay D^* and the maximum queuesize Q^* for server i with input $M_{i-1}(t)$ occur in the interval (t_0, t_1). (t_0 and t_1 are from the definition of $\varepsilon_i(t)$.

Theorem 5.1: If ε_{i-1} is the input to server i, the resulting $Q^*_{\varepsilon_{i-1}}$ and $D^*_{\varepsilon_{i-1}}$ are upper bounds on $Q_{M_{i-1}}(t)$ and D_M