Gsn: a New Service Type for Integrated Services on the Internet*

ENRICO GREGORI, RICCARDO MARCANTONIO, FRANCESCO POTORTÌ
CNUCE, Institute of National Research Council, Via S. Maria 36-56126 Pisa, Italy
Phone: +39-050-593111 (operator), Fax: +39-050-904052
{E.Gregori,F.Potorti}@cnuce.cnr.it

Abstract. Network services with deterministic guarantees are based on a worst-case description of user-generated traffic. When designing a policing and scheduling algorithm for guaranteed services on the Internet, accuracy of description of the traffic profile has to be traded with simplicity of implementation. The result of this trade off is often expressed as the number of token buckets required by the service along with the choice of their parameters. The GS type of service proposed by the IETF uses two token buckets both for characterizing the traffic and for policing it. The choice of using only two token buckets is primarily driven by policing costs. In this paper we propose a method that allows the number of token buckets used for characterizing the traffic to be greater than what is actually needed to police it. This means we can obtain an accurate profile of the traffic while keeping policing simple. The method consists of computing a profile of the traffic which involves a number of token buckets of the order of ten, and then doing the policing using only the first token bucket, plus another one which is chosen depending on the delay requirements of the receivers. This paper shows that with this simple enhancement we obtain a guaranteed service whose performance closely approaches the theoretical limits of services with deterministic guarantees.

1 INTRODUCTION

The vast majority of Internet traffic is currently transported over TCP. Using IP as the network protocol, TCP provides a connection-oriented, reliable stream of data between two hosts. It ensures integrity of data by retransmitting packets when they are lost, an event which, in wired networks, is almost exclusively triggered by congestion.

With the advent of high speed packet networks, multimedia applications have recently been developed which do not work well with the timeout-based retransmission policy used by TCP, because of their strict deadlines on packet reception times. For example, applications which play out an audio or video stream typically discard any images or audio packets which arrive too late. This type of traffic is therefore different from traditional data traffic, since it requires some guarantees on the Quality of Service (QoS) that it receives from the network, e.g. a statistical or deterministic guarantee on the throughput, or on the drop rate. New technologies have been developed to deal with these application requirements. They allow the end host to reserve resources on the net in order to get a guaranteed QoS.

Since it is difficult to determine acceptable levels for the loss rate and missed deadlines [1], deterministic services are becoming more and more popular for VBR video transmission ([2], [3], [4]). A *deterministic service* ensures that no packet is dropped or delayed beyond the deadline requested by the application. The amount of resources to reserve for a video stream in order to offer a deterministic QoS is largely dependent on the traffic characterization method used to describe the stream. For a deterministic service we require a deterministic traffic characterization method that gives an upper bound on the traffic [5].

There are two conflicting requirements affecting traffic characterization: Accuracy and Simplicity.

Accuracy affects the utilization of the network for traffic with a guaranteed QoS. This means that the more accurately the traffic parameters are described, the more efficiently the network resources may be used. In [4, 6], some MPEG1 traces are studied, and it is shown that a source description with only two token buckets cannot be close to the optimal resource allocation for all delay

* This work has been carried out in the framework of the CNR project "Advanced applications for next generation packet switching networks".

Copyright 2001 by AEI. <URL: http://www.aei.it/>

bounds, but at least 8 token buckets are needed for end-to-end delays of less than 500ms.

Simplicity means that traffic characterization must be expressed in an easily controllable way. Indeed, a policing mechanism is needed which ensures that all traffic submitted to the network conforms to the declared traffic characterization. The most popular approach to traffic characterization is based on the token bucket mechanism [7]. With token buckets, the worst case traffic arrival in any interval of length *t* is described as a continuous piecewise-linear function of *t*. In real networks, a small number of token buckets are usually considered, for reasons of implementation efficiency. For example, the current IETF int-serv specifications only make provisions for a peak rate and a further token bucket for both GS (*Guaranteed Service*) and CL (*Controlled Load*) types of service [8]. Likewise, the ABR type of traffic in ATM networks uses two token buckets [9, 10].

Within this framework, this paper compares the theoretical limits of a guaranteed type of service using both the most stringent worst-case traffic characterization and the optimum scheduling algorithm, with the performance attainable with implementations that conform to the IETF's Guaranteed Service [2].

We show how characterizing the traffic with a *peak rate* plus a couple of parameters describing a *token bucket* can lead to a network utilization that is considerably lower than the theoretical limits. We then propose a modification to GS that can significantly narrow this gap at a low implementation cost.

The rest of the paper is organized as follows. Section 1 describes the deterministic service model we are going to consider, and its theoretical limits. Section 2 is a brief overview of GS. In section 3, the optimal choice of parameters for GS is analyzed. We give examples using VBR traffic obtained from MPEG-1 coding of movies, and we describe a method for maximizing the number of homogeneous connections on a given link by optimizing the traffic characterization parameters. Finally, in section 4 we propose a new type of service derived from GS, which we call GSn, which automatically chooses the best token bucket, on the basis of the traffic characterization and the receiver requirements.

2 MAXIMAL UTILIZATION FOR A DETERMINISTIC SERVICE

In this section we outline the main components required to offer a service with deterministic guarantees, that is, the traffic characterization and the scheduling policy, and we identify the maximum efficiency obtainable with deterministic services.

A key concept for a deterministic service is a priori knowledge of a deterministic traffic characterization. Let $A(t)$ denote the cumulative arrival traffic function, that is, the total traffic generated in the time interval [0, *t*], and $A[\tau, t+\tau]$ denote the arrivals in the time interval $[\tau, t+\tau]$. Then, a deterministic characterization is given by a *traffic constraint function* A^*, which provides an upper bound for the traffic generated in a given interval, i.e.

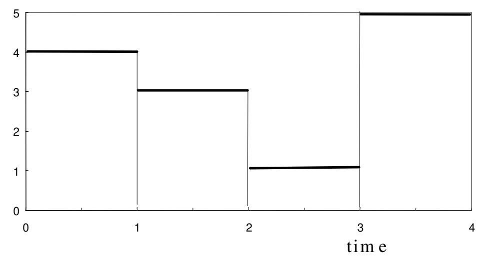
$$\mathrm{A}[\tau, \tau + \mathrm{t}] \le \mathrm{A}^*(\mathrm{t}) \qquad \forall \mathrm{t} \ge 0, \forall \tau \ge 0.$$

We call the minimum traffic constraint function an *empirical envelope* E^*, i.e.

$$E^*(t) \le A^*(t) \; \forall t \ge 0, \forall A^*(\cdot).$$

A common model for the function $A^*(t)$ is a series of n token buckets, each expressed as a (σ_i, ρ_i) pair, where ρ is the token rate and σ is the token bucket size. With this type of characterization, a traffic constraint function assumes the following form:

$$A^*(t) = \min_{1 \le i \le n} \{\sigma_i + \rho_i t\}. \qquad (1)$$



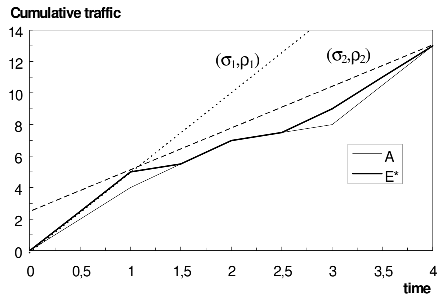


Figure 1a) An arrival function; 1b) E^* *and two token buckets for the arrival function in 1a).*

Not all empirical envelopes can be expressed in this form, both because it is a piecewise linear function, and because it is concave. If we assume that the arrival function $dA(t)/dt$ is piecewise constant (see for example Fig. 1a), the first issue is not a problem. The concavity issue can be tackled by applying a concavization algorithm to the empirical envelope, at the cost of losing some information. The resulting function is called the *Hull* of the empirical envelope. Without loss of generality, we can assume that the ρ_i decrease, while the σ_i

 Copyright 2001 by AEI. <URL: http://www.aei.it/>

increase with increasing i. Figure 1b) shows an example of how the empirical envelope E^* is built starting from an arrival function.

Note that in figure 1b) each line $\sigma_i + \rho_i t$ is a traffic constraint function, and

$$\mathrm{Hull}_{E^*}(t) = \min\{\sigma_i + \rho_i t\}$$

is the minimum concave traffic constraint function. Because the Hull loses some information with respect to the empirical envelope, it is usually described using a much lower number of parameters. For example, when considering the MPEG1 trace of the Jurassic Park movie [11], which is made up of 40000 video frames, the empirical envelope E^* needs 40000 parameters, while the corresponding Hull only needs 47 (σ_i, ρ_i) pairs, i.e. 94 parameters.

Another key component of a deterministic service is the scheduling policy. Some interesting proposals for schedulers are Weighted Fair Queuing (WFQ), Earliest Deadline First (EDF), Earliest Due Date (EDD) [12]. We now briefly summarize how an EDF scheduler works.

Suppose we have a set J of flows described by $\{A^*_j, d_j\}_{j\in J}$, where, for each connection j, A^*_j is the traffic constraint function, and d_j is the maximum tolerable delay for the delivery of packets belonging to flow j. An Earliest Deadline First scheduler orders the sending queue so that packets with shorter deadlines are transmitted first. The aim of the scheduler is to send packets within their respective deadlines, without preemption: if d_j is the delay relative to connection j, and a packet for that connection arrives at time t, then it will be transmitted within its assigned deadline $t+d_j$. This implies that the sending queue needs to be reordered for each incoming packet.

Assume, for the sake of simplicity, that the link capacity is equal to 1, and deadlines increase with their index, i.e. $i < j \Rightarrow d_i \le d_j$. Then $\{A^*_j, d_j\}_{j\in J}$ is schedulable without preemption if and only if the following holds:

$$\forall t : d_1 \le t < d_{|J|}, \quad t \ge \sum_{j\in J} A^*_j\left(t - d_j\right) + \max_{k: d_k > t} s_k\,, \qquad (2)$$

where

$$\max_{t<d_k} s_k \equiv 0$$

if $t > d_{|J|}$ and each s_k is the maximum transmission time for a packet from connection k [13].

EDF scheduling is shown to be optimal in [13], in the sense that if any packet scheduling method can meet a set of connection delay constraints, so can EDF.

We conclude that, from a transmission resources standpoint, the most efficient deterministic service using token buckets is obtained using the most accurate token bucket description of the traffic, i.e. the Hull, together with the optimal scheduling policy, i.e. EDF. For this reason, EDF/Hull based service will hereafter be called TLDS (Theoretical Limit for a Deterministic Service), and will use it as a benchmark for estimating the efficiency of a guaranteed service.

3 DETERMINISTIC SERVICES ON THE INTERNET

This section briefly introduces the Internet Guaranteed Service (GS) [2]. GS is a type of service which guarantees a requested bandwidth and, if the traffic conforms to the specifications, ensures that no packets are lost due to buffer overflows in the routers, and that the end-to-end delay upper bound is known. This type of service is suitable for applications with real-time requirements, such as audio and video.

Since current GS implementations are based on the RSVP protocol, we also use RSVP to clarify service implementation details. RSVP is a resource reservation protocol designed for the Internet, which supports integrated services [14]. A primary design goal of RSVP was to support multicasting, with receivers being able to add themselves to a multicast session at will. Thus RSVP is a receiver-initiated protocol, with the resource reservations being made by the receivers. Here we outline only the reservation aspects of RSVP since they are fundamental for understanding the protocol modifications proposed in the next section. Hereafter the term *flow* will refer to a stream of data traffic that is transported from a sender to a receiver.

With RSVP, *Path* messages are sent from the sender to all the receivers in the distribution list along the default routing path of the Internet. These messages contain information about the flow, hereafter *Tspec*. A *Tspec* describes the flow's characteristics in terms of two token buckets. Specifically, a *Tspec* contains:

1. maximum packet size, *M*
2. peak rate, *p*
3. token bucket size, *b*,
4. the token accumulation rate, *r* and a few more parameters which are not relevant for this paper.

In addition the path messages contain an *Adspec*, which describes the path characteristics using two delay parameters that can be modified by the routers traversed by the *Path* message.

The receiver uses the information in the sender's *Tspec* and in the *Adspec* to decide the level of resources it needs to reserve. The *Resv* message sent by the receiver retraces the path of the *Path* message and establishes the required reservation. The amount of resources that need to be reserved are a function of:

User characteristics: This is related to the *Tspec* and the end-to-end delay requirement of the receiver.

Copyright 2001 by AEI. <URL: http://www.aei.it/>

Network characteristics: These include factors such as the number of hops on the path, the scheduling policy employed at each hop, and the end-to-end latency that is present. These factors affect the *Adspec*.

The user characteristics consist of the *Tspec*, which includes the M, p, b and r parameters, and of the *Rspec*, which indicates the level of resources that have to be reserved for this flow. For now it suffices to say that the *Rspec* is a rate R. Each router traversed by the flow is assumed to behave according to a fluid flow model with a rate R associated with this flow. This assumption is optimistic and for this reason in the *Adspec* each router exports two parameters C and D, which indicate the deviation of the router scheduler from a fluid flow model operating at a rate R. For each router i, the C_i and D_i parameters are best described assuming the following behavior of the router: the router first delays each arriving bit of $C_i/R + D_i$ time units and then serves it exactly at the reserved rate R. This means that the delay experienced by any bit of the flow in router i will not exceed $C_i/R + D_i + 1/R$. One advantage of this type of specification is that it allows a simple computation of the end-to-end delay bound, given the *flowspec*, the *adspec*, and the reserved rate R.

As will be shown in the next section, this reservation procedure may result in a significantly suboptimal performance, with respect to the fundamental limits of a deterministic service.

4 USE OF THE GS TYPE OF SERVICE

Every router participating in the GS service allocates a bandwidth R and a buffer space B for each flow that requests a GS type of service. Using a fluid model approximation of the traffic, one can say that the service provides an end-to-end bandwidth R, and any flow that conforms to a token bucket with rate r and depth b experiences a delay equal to b/R, provided that $B \geq b$ in the first router.

In practice, the deviations from the fluid model should be accounted for, namely packetization effects, router and link latencies, and line speed. Considering all these factors yields the following formula [2] for the end-to-end delay bound of a traffic flow advertising a token bucket (b, r), a peak rate p, and a maximum packet size M, to which a service rate R is granted:

$$d(R) = \frac{(b-M)(p-R)}{R(p-r)} + \frac{M + C_{tot}}{R} + D_{tot} \tag{3}$$

with $r \leq R < p$

The C_{tot} and D_{tot} parameters represent the total effect of the deviation from the fluid flow model. The former is the rate-dependent delay introduced by the network (essentially, the packetization delays), the latter is the rate-independent term. From (3) we have that $d(R)$decreases as R increases in $[r, p]$, and its values at the boundaries of this range are

$$d_{\min} = d(p) = \frac{M + C_{tot}}{p} + D_{tot},$$
$$d_{\max} = d(r) = \frac{b + C_{tot}}{r} + D_{tot}. \tag{4}$$

We call them $d_{\min}$ and $d_{\max}$ because requesting $R<r$ would lead to an unbounded delay, while requesting $R>p$ can only increase the rate at which a traffic unit is delivered, so it is generally of no practical interest.

4.1 CHOOSING THE TOKEN BUCKET PARAMETERS

In this section we show that if the sender chooses the token bucket parameters (M, p, b and r) without knowing the delay requested by the receivers, the resulting allocated rate R may be far from optimal. We begin by studying the bandwidth utilization obtainable with GS, then we compare it with TLDS.

Suppose we know the empirical envelope of the traffic generated by the sender, and we have computed its Hull:

$$\text{Hull}_{E^*} = \min_{1 \leq i \leq m} \{\sigma_i + \rho_i t\}.$$

The M and p parameters depend on the packet size and the minimum packet spacing, which in turn depends on the empirical envelope and the link speed. The token bucket parameters r and b can be tuned, for a given delay bound, in order to minimize the allocated bandwidth R. A careful choice of the token bucket parameters can make the performance of GS approach the theoretical limit for a deterministic service as defined in section 1.

For $d \in [d_{\min}, d_{\max}]$, equation (3) is invertible, so we can compute $R \in [r, p]$ as:

$$R = \frac{(bp - rM) + C_{tot}(p - r)}{(d - D_{tot})(p - r) + (b - M)}. \tag{5}$$

If we use

$$\text{Hull}_{E^*} = \min_{1 \leq i \leq m} \{\sigma_i + \rho_i t\}$$

to characterize the traffic, the GS parameters should be set to $p = \rho_1$ (provided it is no greater than the link speed), $b = \sigma_i$ and $r = \rho_i$, where $2 \leq i \leq m$. Since, as in section 1, $\sigma_i < \sigma_j$ and $\rho_i > \rho_j$ for $i < j$, we also have $p > r$.

Let us show with an example how equation (5) can be used. We consider a link with 7.5 Mbps[1] capacity, over which a number of identical flows are transmitted, each

[1] We chose this value because, in its default configuration, the software of the router we used for some laboratory experiments allowed RSVP to reserve a maximum of 75% of the capacity of our 10 Mbit/s Ethernet link.

Copyright 2001 by AEI. <URL: http://www.aei.it/>

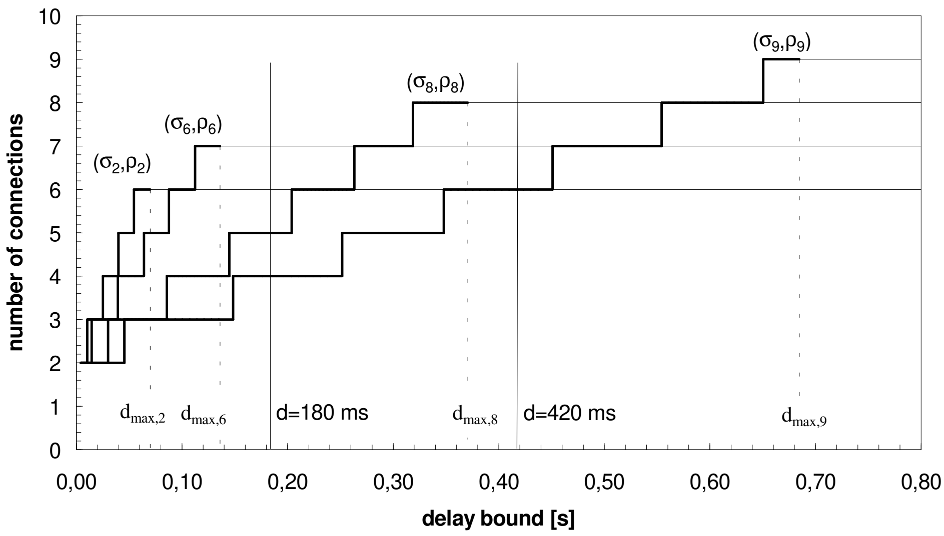


Figure 2: Number of connections using four different choices for (r, b).

with the characteristics of the Jurassic Park MPEG1 trace [11]. We assume that we will make a number of reservation requests for identical GS flows, until one is refused, for different (b, r) pairs. In figure 2 the number of successful reservations, computed using equation (5), is plotted versus the required delay bound, for (b, r) set to the first four significant (σ_i, ρ_i) chosen from those which compose the Hull. Indexes i greater than 9 were not considered, because they add nothing to the results in the delay bound range of [0, 800] ms. Among the token buckets with an index less than 9, four were not plotted because the corresponding number of successful reservations was practically equal to one of the selected ones. Each piecewise-linear function relative to (σ_i, ρ_i) covers the delay bound range [$d_{\min}$, $d_{\max,i}$].

Let us now assume that the sender knows the Hull of the traffic it is going to transmit, and has to choose the *Tspec* parameters. We consider two possible situations. In the first, which we label *GSkd* (GS known delay), we assume that the sender knows the delay requested by the receiver. In the second, which we label *GSmd* (GS maximum delay), we assume that the sender only knows an upper bound for the delay that the receiver can request.

4.1.1 GSkd case

If we know the delay bound d for the flow, we can identify the (σ_i, ρ_i) couple which minimizes the allocated rate R:

$$R = \min_{2 \le i \le m} R_i , \tag{6}$$

where

$$R_i = \frac{(\sigma_i p - \rho_i M) + C_{tot}(p - \rho_i)}{(\bar{d} - D_{tot})(p - \rho_i) + (\sigma_i - M)},$$

being

$$\bar{d} = \min[d, d_{\max,i}], \; d_{\max,i} = \frac{\sigma_i + C_{tot}}{\rho_i} + D_{tot} .$$

Note that while $d_{\min}$ does not depend on the choice of the token bucket (σ_i, ρ_i), we have $d_{\max,i} < d_{\max,j}$ for $i < j$.

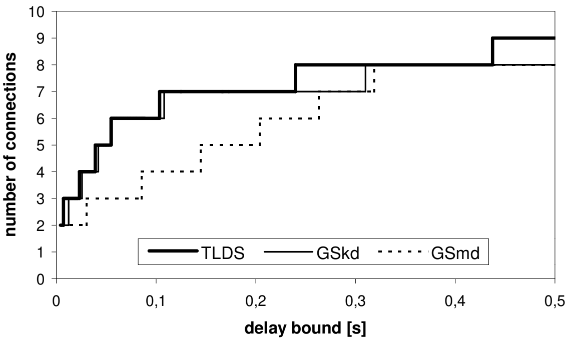
As an example, suppose that the user needs a service with a delay bound guarantee of 420 ms. Using equation (6) to choose the optimal (σ_i, ρ_i) for the cited MPEG1 trace, we obtain $r = \rho_8$, $b = \sigma_8$, $R = r$. In figure 2 we see that the maximum number of connections for the given link is 8, which was obtained for a delay lower than requested, because $d_{\max,8} < 420$ ms.

If the requested delay is 180 ms, the maximum number of connections for the given link is 7 (see figure 2), and this number is obtained for $r = \rho_6$, $b = \sigma_6$, $R = r$. Here also, the maximum number of connections was obtained for a delay lower than requested, because $d_{\max,6} \le 180$ ms.

4.1.2 GSmd case

Suppose that the maximum delay that the receiver can request is 800 ms. Since GS uses only one of the possible (σ_i, ρ_i) pairs, and for each such pair the number of possible connections versus the delay bound is a growing staircase in [$d_{\min}$, $d_{\max,i}$], the pair which grants the maximum number of connections at the right extreme of the plotted range should be chosen. So we choose $r = \rho_9$, $b = \sigma_9$, and R is computed using equation (5) depending on the delay requested. If the receiver asks for a 420 ms delay bound, from figure 2 we see that we only get 6 connections, instead of 8 as in the GSkd case. For a requested delay of 180 ms, 4 connections can be set up, while 7 connections were accepted in the GSkd case.

Copyright 2001 by AEI. <URL: http://www.aei.it/>



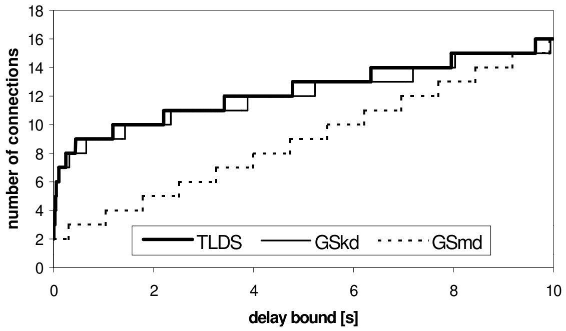


Figure 3: Performance of TLDS, GSkd and GSmd for different delay bound ranges.

Studying the GSkd and GSmd cases highlights that the choice of (σ_i, ρ_i) couple significantly affects the performance of the GS service. Figure 3 compares the theoretical limit described in section 1 with the performance that can be obtained from GS in the GSkd and the GSmd cases, for two different ranges of delay bounds. The line labeled GSkd uses all the token buckets in the Hull, while the one labeled GSmd uses a token bucket chosen with the criterion explained above in the GSmd example. As before, the capacity of the link is 7.5 Mbps. We make a number of reservation requests for identical flows, each with the characteristics of the Jurassic Park MPEG1 trace, until one is refused.

The graphs in figure 3 indicate that the channel utilization for the GSkd case is very close to the TLDS case, while GSmd may significantly underutilize the network resources.

In the next section we define a type of service with the same efficiency as the GSkd case, but which does not require a priori knowledge of the requested delay.

5 GS ENHANCED

The above results highlight that the optimal (*b, r*) parameters can only be chosen if the delay bound desired by the receiver is known. We propose a novel deterministic type of service, to be used in the framework of the RSVP signaling protocol, which we call GSn. This type of service is an extension of the GS type of service that is based on the mechanism for selecting the traffic parameters presented in section 3. Only the case of a single sender is considered which is, for example, the common case for a video-on-demand server. Multiple senders are not considered in this paper.

The main characteristics of GSn are:

- Deterministic guarantees, analogously to GS
- Better usage of the network than GS: for a given range of user requested delays, usage is nearly optimal
- *Tspec* contains information about an arbitrary number of token buckets
- *Rspec* contains an *R* rate for each token bucket
- *Adspec* is the same as for GS
- Same scheduling algorithms as for GS
- Same policing as for GS.

Like GS, the *Tspec* contains a description of the traffic characteristics in terms of token buckets that define a concave traffic constraint function. GS uses the maximum segment size *M* and peak rate *p*, plus a single token bucket (*b*, *r*). GSn extends this characterization with an array of token buckets $(\boldsymbol{b}, \boldsymbol{r})$. This extension can make the traffic constraint function close to the empirical envelope for a wide range of time intervals.

Let us follow the flow of information starting from the sender. We assume a single sender which, before starting the transmission, knows the empirical envelope of the traffic to be transmitted, and has computed an array of *n* token buckets $(\boldsymbol{b}, \boldsymbol{r})$, plus the maximum segment size *M* and peak rate *p*. The choice of the token buckets $(\boldsymbol{b}, \boldsymbol{r})$ is made by the sender, either using the whole set of token buckets describing the Hull, or a meaningful subset obtained, for example, with the procedure outlined in the appendix.

Using these parameters, the sender builds the *Tspec* of the traffic flow, and includes it in the *Path* messages that it starts to send when it is ready to transmit. During its journey downstream, the *Adspec* contained in the *Path* message is updated by each traversed router, following the same rules as with GS.

We assume that the receiver needs a deterministic delay bound on the received data, which it knows a priori. Upon reception of a *Path* message, in the case of GS, a receiver uses the information carried in the *Tspec* and in the *Adspec* to translate its desired delay bound into a request for a rate *R* — see equation (5). With GSn, the same computations are performed for each token bucket (*b*, *r*) contained in the *Tspec*. Thus, the *Rspec* of GSn consists of an array of rate values $\boldsymbol{R}$, one rate value for each token bucket of the *Tspec*. Using the sender's *Tspec* and the computed *Rspec*, the receiver then builds a *Resv*

Copyright 2001 by AEI. <URL: http://www.aei.it/>

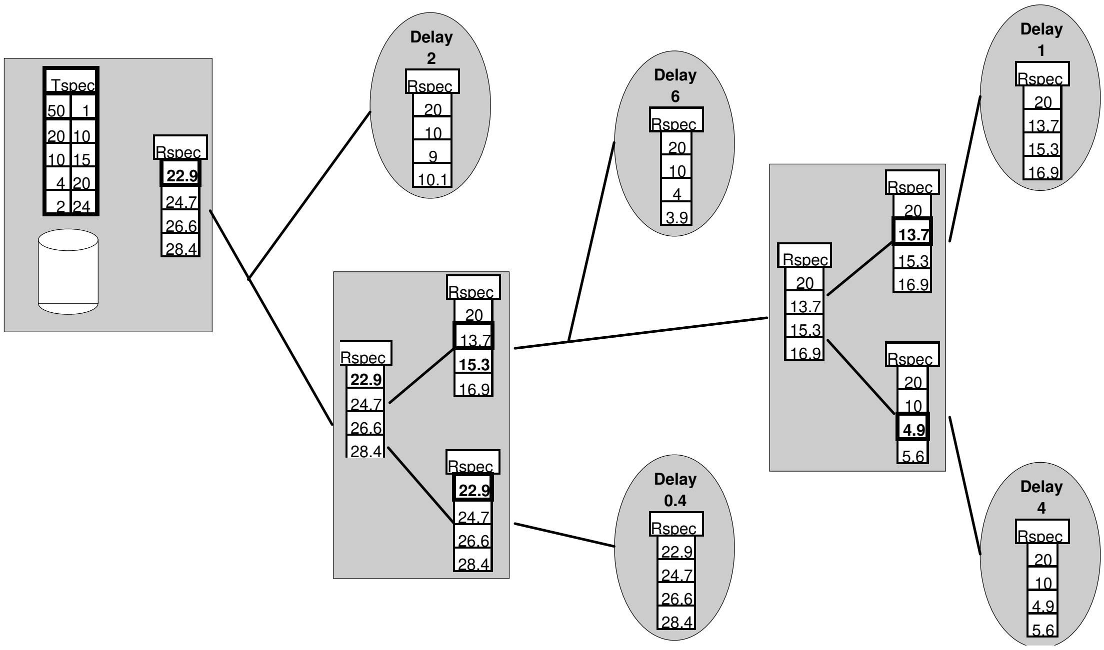


Figure 4: Example of the merging mechanism for GSn.

message, and sends it upstream, following the reverse path indicated in the *Path* message.

When a router receives the *Resv* message on a downstream interface, it merges the received *Rspec* with the effective *Rspec* already installed on that interface. The merging rules are the same as those of GS, but are applied row by row on the vector $\boldsymbol{R}$. In practice, for each R_i, the maximum of the received and installed ones is chosen. The *Tspec* contained in the received *Resv* message is equal to the sender's *Tspec*.

The router then chooses the smallest rate R_i from the array $\boldsymbol{R}$ contained in the *Rspec*, and passes it to traffic control, which tries to allocate this rate on the downstream interface. The traffic control can implement the same scheduling policy it uses for GS, using the token bucket (b_i, r_i) related to the rate R_i. If the reservation can be installed, the router then merges the effective *Rspec* of all the downstream (outgoing) interfaces, using the same rules outlined above, and forwards the merged *Rspec* upstream.

If the reservation is successfully installed all the way along the path to the sender, data will begin to flow from the sender to the receiver. At each router along the path, the scheduling applied to the data will be of the same type as that used for GS, but different token buckets may be used at each downstream interface, chosen from those specified in the sender's *Tspec*. Given the procedure outlined above, the receiver is guaranteed that its target delay will not be exceeded.

The overall result of the merging process is that, on each downstream interface, the optimum token bucket for a GS type of service is chosen, that is, the token bucket associated with the minimum rate that satisfies the delay constraints of the downstream receivers.

Figure 4 shows an example with one server (the sender) providing a traffic characterization described by a (p, M) pair (on the first line of the *Tspec*) and four (r, b) pairs. Two routers, and five receivers with different delay requirements are considered. For the sake of simplicity, a null *Adspec* is assumed, i.e. one with $C_{tot} = 0$ and $D_{tot} = 0$. In the figure, inside each receiver a table with the *Rspec* is shown, together with the desired delay bound used to

Table I: GS and GSn parameters.

	GS	GSn
Tspec	r, b, p, M, m	$n, r_1, \ldots, r_n, b_1, \ldots, b_n, p, M, m$
Adspec	C_{tot}, D_{tot}	C_{tot}, D_{tot}
$R(d)$ using formula (5)	$r \le R(d) \le p$	$r_n \le R_i(d) \le p$
Rspec	R, S	$N, R_1, \ldots, R_n, S$
merged *Rspec*'s	max(*Rin*), min(*Sin*)	max(Rin_1), …, max(Rin_n), min(*Sin*)
scheduling token buckets	$(p, M), (r, b)$	$(p, M), (r_i, b_i) \| R_i = \min(R_1, \ldots, R_n)$

Copyright 2001 by AEI. <URL: http://www.aei.it/>

build the table. At the downstream interfaces of the routers and the sender, a table with the installed *Rspec* is shown, while at the upstream interface of the routers, a table with the merged *Rspec* is shown. An installed *Rspec* contains the four R values obtained by merging the *Rspec*'s that have arrived at that interface. The smallest R is highlighted because it corresponds to the single token bucket used on that downstream interface to police and schedule the traffic.

Table I summarizes the differences between GS and GSn. The terms and parameters used in the table reflect those of [2]. The two types of service are very similar, because GSn can be viewed as an extension of GS where the *Tspec* parameter is vectorialized: setting n equal to 1 makes GSn identical to GS.

Below are some properties of the GSn algorithm:

1. On each downstream interface, the minimum R is chosen which is compatible with the desired delay bounds of the receivers which are downstream of that interface.

2. Traveling downstream, from the sender towards a given receiver, the sequence of the R values used at the router interfaces is non-increasing.

3. Given a downstream interface, there is at least one of its downstream receivers that has a guaranteed delay equal to its desired delay bound, computed with formula (3). All the downstream receivers have a guarantee at least as stringent as their desired delay bound.

As a consequence of the above properties, each receiver is connected to the sender by a virtual channel with a bandwidth R at least as big as is needed to guarantee its desired delay bound. Moreover, on every link, no more bandwidth R than needed by the downstream receivers is allocated. This informally proves that the algorithm provides the required guarantees, and allocates no more than the necessary resources.

6 CONCLUSIONS

When creating a deterministic characterization of traffic, in the form of a traffic constraint function $A^*(t)$, we are faced with two different sets of choices. The first relates to the accuracy of the *admission control* test, which is used to decide whether a new reservation request can be accepted. The second regards the complexity of *policing* the traffic which is injected into the network.

A choice for the admission control test that promotes effective use of the available link capacity depends on a good characterization of the traffic, that is one that allows nothing more than the necessary resources to be allocated. The admission control test is done only once at the beginning of a transmission, when the reservation request is received, so its complexity is not an issue as far as the performance of the network is concerned.

The second set of choices must be made regarding the policing and scheduling algorithm. These modules have to process every single packet in the flow, so their complexity should be kept as low as possible in order to obtain an acceptable performance. In particular, the policing algorithm has to check the traffic profile against the available characterization, which has been provided during the reservation phase, thus a high number of traffic description parameters is unacceptable.

The best description of the traffic flow should thus be a compromise between the accuracy desired by the admission control and the simplicity desired by the policing algorithms.

In this paper we have shown that using only two token buckets for policing does not necessarily degrade the utilization of the transmission channel. If an accurate characterization of the traffic source is available, and the delay bound requested by the receiver is known, then it is possible to achieve a channel utilization that is close to the theoretical limits. Using GSn, the sender provides a description of the traffic flow using an arbitrary number of token buckets, and the optimal token bucket choice is performed by each traversed router.

Since GSn uses two token buckets for policing, it is no more complex than GS as far the data transmission is concerned, which is the most critical issue for routers performing policing. On the other hand, as far as reservation state management is concerned, GSn is more complex than GS, thus possibly exacerbating scalability problems, which may be the critical issue for routers at the core of the network. Several approaches have been envisaged for tackling this problem [15, 16], all of which propose different criteria for aggregating the state of RSVP flows in the routers at the core of the network, thus dramatically reducing the importance of scalability issues.

The added complexity of GSn with respect to GS is dependent on the number of token buckets used for characterizing the source. An approach to choosing a set of token buckets for GSn is discussed in the appendix.

APPENDIX: SELECTING THE TOKEN BUCKETS

Characterizing the traffic using the Hull as described in section 1 may be unnecessarily complex for practical purposes because of the great number of token buckets that are usually necessary. We need a method to choose a small number of significant token buckets, in order to build a traffic constraint function which is an approximation of the Hull. The problem of reducing the number of token buckets is expressed as follows:

Given an n-pieces linear function B^*_n, *find an m-piecewise linear function* B^*_m *with* $m < n$ *such that* $B^*_m(t) \geq B^*_n(t)$ *for all* $t > 0$.

Copyright 2001 by AEI. <URL: http://www.aei.it/>

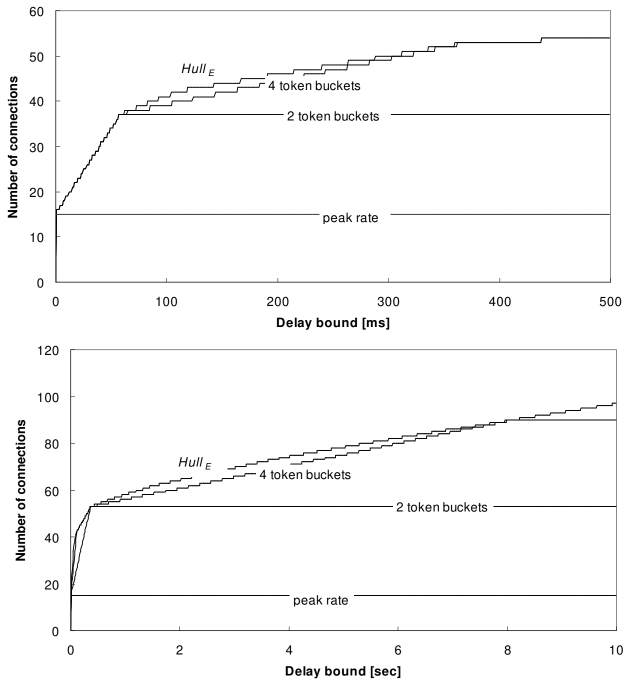


Figure 5: Effect of the number of token buckets on link utilization for different delay ranges.

In order to select the m token buckets we maximized the number of connections that an EDF scheduler could allow on a link of a given capacity, for a given range of delays [17]. This is not the only possible criterion for the selection, and other more general choices have been proposed. For example, [6] approaches the problem from a geometrical point of view, by trying to approximate the empirical envelope as much as possible, with the constraint of a given number of token buckets.

Selecting m token buckets from a set of n is generally a combinatorial problem whose complexity is proportional to

$$\binom{n}{m} = \frac{n!}{m!(n-m)!}.$$

We used a heuristic algorithm [6] to reduce the complexity to more manageable proportions, at the cost of maybe only finding an approximation of the best solution. Figure 5 compares the link utilization when the traffic characterization function is

$$\mathrm{Hull}_{E^*}(t),$$

with the utilization obtained using 1 (peak rate only), 2 (GS), and 4 token buckets. We assumed the parameters used in [4], i.e. a link with 45 Mbps capacity, and identical flows with the characteristics of the Jurassic Park MPEG1 trace [11]. Good performances are obtained using as few as 4 token buckets, that is, the peak rate plus 3 other ones chosen using the above criterion. In the case depicted, the case of 8 token buckets is indistinguishable from using the whole Hull.

ACKNOWLEDGMENTS

We wish to thank the reviewers for their comments, which helped us to significantly improve the quality of this paper, and Michele Volpe of Telecom Italia for supporting R. Marcantonio's thesis.

Manuscript received on 21, July, 1999

REFERENCES

[1] G. Karlsson. Asynchronous Transfer of Video. *IEEE Communications magazine*, Vol. 34, No. 8, pages 118-126, 1996.

[2] S. Shenker, C. Partridge, and R. Guerin. Specification of guaranteed quality of service. IETF RFC 2212, September 1997.

[3] R. Braden, D. Clark, S. Shenker. Integrated services in the Internet Architecture: an Overview. RFC 1633, June 1994.

[4] D.E. Wrege, E.W. Knigthly, H. Zhang and J. Liebeherr. Deterministic Delay Bounds for VBR Video in Packet-Switching Networks: Fundamental Limits and Practical Tradeoffs. *IEEE/ACM Transactions on Networking*, Vol. 4, No. 3, pages 352-362, June 1996.

[5] R. L. Cruz. A Calculus for Network Delay, Part I: Network Elements in Isolation. *IEEE Transactions on Information Theory*, Vol. 37, pages 114-131, January 1991.

[6] J.Liebeherr and D.E.Wrege. An efficient Solution to Traffic Characterization of VBR Video in Quality-of-Service Networks. *ACM/Springer Multimedia Systems Journal*, Vol 6, No. 4, pages 271-284, July 1998.

[7] J.S. Turner. New directions in communications (or which way in the information age?). *IEEE Comm. Magazine*, Vol. 24, pages. 8-15, October 1986.

[8] J.W. Roclawski. The use of RSVP with IETF integrated services. IETF RFC 2210, September 1997.

[9] ATM forum. Traffic management specificationî, version 4.0, 1996.

[10] ATM forum. ATM service categories. EMAC white papers, 1996.

[11] FTP site of Werzburg University Institute of Computer Science (ftp://wi3x18.informatik.uni-wuerzburg.de/pub/MPEG).

[12] H. Zhang. Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks. In *Proc. of the IEEE*, Vol. 83, No. 10, pages 1374-1396, October 1995.

Copyright 2001 by AEI. <URL: http://www.aei.it/>

[13] J.Liebeher, D.E.Wrege and D.Ferrari. Exact Admission Control for Networks with Bounded Delay Services. *IEEE/ACM Transactions on Networking*, Vol. 4, No. 6, pages 885-901, December 1996.

[14] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: a New Resource ReSerVation Protocol. *IEEE Network*, pages. 8-18, September 1993.

[15] Y. Bernet, R. Yavatkar, P. Ford, F. Baker, L. Zhang, M. Speer, R. Braden and B. Davie. Integrated Services Operation Over Diffserv Networks. IETF internet draft (work in progress), June 1999.

[16] S. Floyd and V. Jacobson. Link-sharing and Resource Management Models for Packet Networks. *IEEE/ACM Transactions on Networking*,Vol. 3, No. 4, pages 365-386, August 1995.

[17] R. Marcantonio. Introduzione di servizi a qualità garantita in reti IP: caratterizzazione del traffico e valutazione di tecnologie per la valutazione delle risorse. Thesis, University of Pisa, 1998.

Copyright 2001 by AEI. <URL: http://www.aei.it/>