To appear in Computer Communications

Providing End-to-End Performance Guarantees Using Non-Work-Conserving Disciplines

Hui Zhang
School of Computer Science
Carnegie Mellon University
hzhang@cs.cmu.edu

Abstract

A non-work-conserving server is one that may be idle even when there are packets available to be sent. Since they do not provide the optimal average performance, non-work-conserving disciplines were seldom studied in the past. For the guaranteed performance service class in integrated services networks, the main performance index is the end-to-end delay bound, instead of the average delay. Providing end-to-end delay bounds in a networking environment is difficult. While bounding delay in any server requires a bound on the input traffic, complex interactions among traffic streams usually distort the traffic pattern so that traffic inside the network is different from the source traffic. Previous techniques of bounding end-to-end delay in a networking environment usually start from the source traffic characterizations and iteratively "push" the traffic characterizations through the network. This does not only require non trivial analysis, but also has several important limitations. In this paper, we show that non-work-conserving disciplines greatly simplify the analysis in a networking environment and overcome most of the limitations of previous work by controlling traffic distortion inside the network and thus allowing a single node analysis to be extended to arbitrary topology networks.

1 Introduction

Future packet-switching integrated-services networks will have to support real-time communication services that allow clients to transport information with performance guarantees expressed in terms of delay, delay jitter, throughput and loss rate [5]. In a packet-switching network, packets from different connections interact with each other at each switch. Without proper control, these interactions may adversely affect the network performance experienced by clients. The service disciplines at the switching nodes, which control the order in which packets are serviced, determine how packets from different connections interact with each other.

Service disciplines and associated performance issues have been widely studied in the contexts of hard real-time systems and queueing systems. However, results from these studies are not directly applicable in the context of integrated-services networks. Analyses of hard real-time systems usually assume a single server environment, periodic jobs, and the job delay that is bounded by its period [24]. However, network traffic is bursty, and the delay constraint for each individual connection is independent of its bandwidth requirement. In addition, bounds on *end-to-end* performance need to be guaranteed in a *networking* environment, where traffic dynamics are more complex than in a single server environment. Queueing analysis is often intractable for realistic traffic models. Also, classical queueing analyses usually study *average* performance for *aggregate* traffic [13], while in integrated-services networks performance bounds need to be derived on a *per-connection* basis [5, 17].

A service discipline can be classified as either work-conserving or non-work-conserving. With a work-conserving discipline, a server is never idle when there is a packet to send. Non-work-conserving disciplines were seldom studied in the past. This is mainly due to two reasons. First, in most of previous performance analyses, the major performance indices are the *average* delay of all packets and the *average* throughput of the server. With a non-work-conserving discipline, a packet may be held in the server even when the server is idle. This may increase the average delay of packets and decrease the average throughput of the server. Secondly, most previous queueing analyses assumed a single server environment. The potential advantages of non-work-conserving disciplines in a networking environment were therefore not realized. In integrated-services networks, the more important performance index is the end-to-end delay *bound* rather than the average delay. In addition, delay needs to be bounded in a *networking* environment rather than just in a single node. Therefore, the above reasons for not using non-work-conserving disciplines do not hold any more.

In this paper, we study a general class of non-work-conserving disciplines. We show that non-work-conserving service disciplines greatly simplify the analysis in a networking environment by allowing a single node analysis to be extended to arbitrary topology networks. Non-work-conserving service disciplines, when used together with the associated admission control algorithms, can provide end-to-end delay and delay jitter bounds on a per-connection basis in a network of arbitrary topology. Unlike existing bounding techniques for work-conserving disciplines which apply only to feed-forward networks and a restricted class of feedback networks (defined in Section 2), non-work-conserving disciplines can provide guarantees in arbitrary topology networks, both feed-forward and feedback. Unlike existing bounding techniques which only apply to simple network environments where switches are connected by physical links, the performance guarantees provided by non-work-conserving disciplines also apply to internetworking environments where switches may be connected by subnetworks. In addition, non-work-conserving disciplines achieve more efficient usage of buffer space when comparing to work-conserving disciplines.

The rest of the paper is organized as follows. In Section 2, we discuss difficulties and limitations of bounding end-to-end delays in a networking environment when work-conserving disciplines are used. In Section 3, we describe a general class of non-work-conserving disciplines that use rate-control to maintain certain traffic characteristics inside the network. Two general classes of rate-control policies are presented. We show that end-to-end performance guarantees can be obtained by using non-work-conserving disciplines without the limitations described in Section 2. We discuss the tradeoffs of using the two classes of rate-control policies in Section 4. Section 5 reviews related work. We summarize the paper in Section 6.

2 Background and Motivation

Most analyses for bounding end-to-end delay in a networking environment use a two step process:

1. analyze a single server case and derive the local delay bound;
2. combine the local delay bounds at each server on the path traversed by a connection and obtain the end-to-end delay bound.

Much research has been done in providing local delay bounds in a single server. It is the second step that we will emphasize in this paper. In this section, we will illustrate the difficulty in extending one node analysis to a network environment.

A server can provide local performance guarantees to a connection only when the traffic on that connection satisfies certain traffic specifications. However, network load fluctuations may distort the traffic pattern of a connection and result in a burstier traffic at some server even when the connection satisfies the client-specified traffic constraint at the entrance to the network.

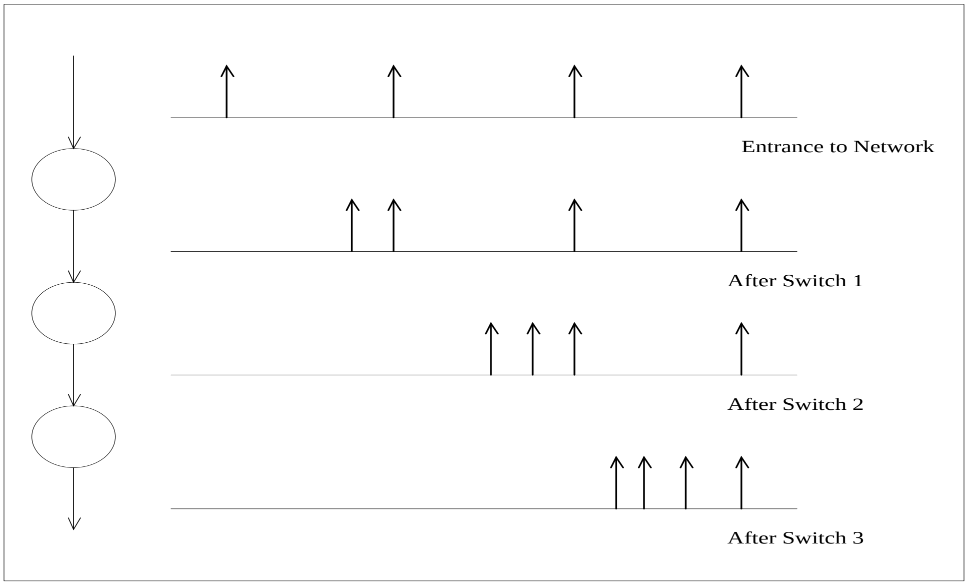


Figure 1: Traffic pattern distortions due to load fluctuations

Consider the example shown in Figure 1. Four packets from one connection are sent with a certain inter-packet spacing from the source into a network where links have constant delay. At the first server, the first packet is delayed by a certain amount of time (less than the local delay bound) due to instantaneous cross traffic load, but the other three packets pass through instantly. Because the first packet was delayed longer than the second packet, the spacing between first and second packets becomes smaller when they arrive at the second server. At the second server, the first and the second packet are delayed some more time, but packets 3 and 4 pass through instantly. At the third server, the first three packets are delayed but packet 4 passes through with no delay. The figure shows traffic patterns at the entrance to each of the servers. Two things can be observed: (a) the traffic pattern of a connection can be distorted due to network load fluctuations, (b) the distortion may make the traffic burstier and cause instantaneously higher rates. In the worst case, the distortion can be accumulated, and downstream servers potentially face burstier traffic than upstream servers. Therefore, the source traffic characterization may not be applicable inside the network.

There are two solutions to address this problem:

1. controlling the traffic distortion within the network, or

2. characterizing the traffic distortion.

To control the traffic distortion within the network, some packets need to be held even when the server has the extra capacity. This requires non-work-conserving disciplines, which we will discuss in more detail in Section 3.

If traffic distortion is not explicitly controlled by non-work-conserving disciplines, it must be characterized throughout the network. The problem can be formulated as the following: given the traffic characterization of all the connections at the entrance to the network and all the service disciplines at the switches, can the traffic be characterized on a per connection basis on all the links inside the network? Several solutions have been proposed to address this problem with different traffic models and service disciplines [1, 3, 16, 23]. They all employ a similar technique that consists of two steps. In the first step, a single node analysis technique is developed to characterize the output traffic of a server given the characterizations of all its input traffic. In the second step, starting from the characterizations of all the source traffic, an iterative process push the traffic characterizations from the links at the edge of the network to those inside the network. This approach, taken in [1, 3, 16, 23], has several limitations.

First, characterizing the traffic inside the network is difficult and may not always be possible. In [4, 16, 23], it is shown that this is equivalent to solving a set of multi-variable equations. In a feedback network, where traffic from different connections forms traffic loops, the resulting set of equations may be unsolvable. To illustrate this, consider the following example given in [3] and also discussed in [21].

In the 4-nodes network shown in Figure 2, there are four 3-hop connections and the aggregate traffic of the four connections forms a loop. In order to characterize the traffic on link 1, the characterization of the input traffic to server 1 has to be known. Assuming links only introduce fixed delay, the input traffic to server 1 is identical to the output traffic of server 0, or the traffic on link 0. There are three traffic streams on link 0, which are from connections 0, 2, and 3. While connection 0 traffic on link 0 is the same as its source traffic, connection 2 and connection 3 traffic on link 0 needs to be characterized. The characterizations of connection 2 and 3 traffic depend on their characterizations on link 3, which in turn depend on their characterizations on link 2. This dependency finally comes back to traffic characterizations on link 0. Because of this inter-dependency of traffic, characterizing all the traffic inside the network is equivalent to

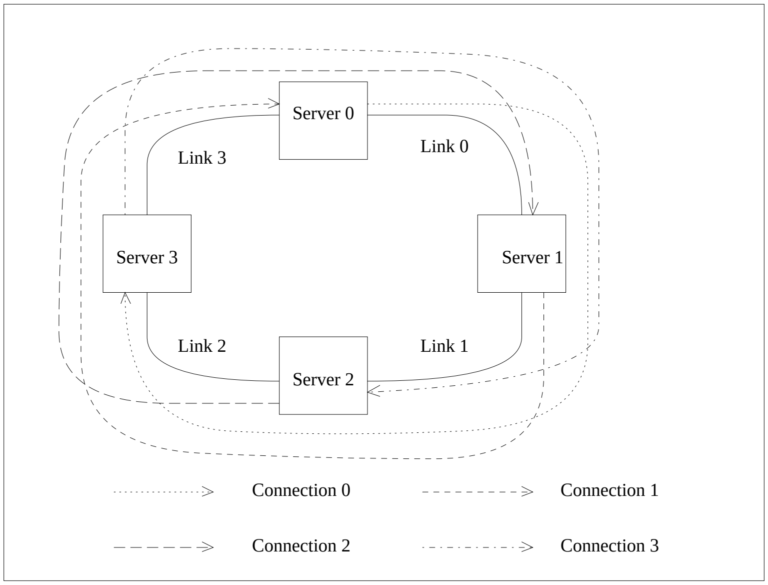


Figure 2: Example of a Feedback Network

solving a set of multi-variable equations, where each variable corresponds to one parameter in the traffic characterization of one connection on one link. The equations are solvable only under certain conditions. In this particular example, it is shown in [4] that if each server has a policy such that the traffic originating from the server has a lower priority than the through traffic, the condition for solving the equations is that the aggregate throughput of the connections must be less than 75% of the link bandwidth on each of the four links. This condition is not merely a technical restriction. The network *can* actually be unstable, i.e., have unbounded queue lengths, when the condition is violated [21]. How to derive the stability condition in a general networking environment is still an open problem.

One notable exception to such a restriction is the case when the service discipline used is a special class of Generalized Processor Sharing (GPS) called Rate Proportional Processor Sharing [21]. A GPS sever is characterized by N positive real numbers, $\phi_1, \phi_2, \cdots, \phi_N$. The server operates at a fixed r and is work conserving. Let $S_j(t_1, t_2)$ be the amount of connection j traffic served in an interval $[t_1, t_2]$, then a GPS server is defined as one for which

$$\frac{S_j(t_1, t_2)}{S_{j'}(t_1, t_2)} \geq \frac{\phi_j}{\phi_{j'}} \quad j' = 1, 2, \cdots, N \tag{1}$$

holds for any connection j that is backlogged throughout the interval $[t_1, t_2]$. The assignment of ϕ's in a GPS server affects allocations of both bandwidth and delay bound. In general, even when sophisticated service disciplines like GPS are used, it may still be impossible to characterize the traffic in a feedback network under general resource (ϕ) assignment. If ϕ's are allocated proportional to the bandwidth required by connections, the resulted policy is called Rate-Proportional Processor Sharing. With RPPS, end-to-end delay bounds can be obtained in a network of *arbitrary* topology. However, in this case, there is a coupling between bandwidth and delay bound allocation: if a connection's traffic is constrained by (σ, ρ) characterization[1], the end-to-end delay bound of the connection will be $\frac{\sigma+(n-1)Smax}{\rho} + \sum_{i=1}^{n} \frac{Smax}{r_i}$, where $Smax$ is maximum packet size, n is the number of hops traversed by the connection, and r_i is the link speed of the i^{th} server. Notice that the delay bound is inverse proportional to the allocated long term average rate. Thus, in order for a connection to get a low delay bound, a high bandwidth channel need to be allocated. This will result in a waste of resources when the low delay connection also has a low throughput.

The second limitation of characterizing traffic inside the network is that it only applies to networks with *constant* delay links. Constant delay links have the desirable property that the traffic pattern at the receiving end of the link is the same as that at the transmitting end. This property is important for these solutions because central to the analysis is the technique of characterizing the output traffic from a server and using it as the characterization of the input traffic to the next-hop server. However, in an internetworking environment, where the link between two switches may be a subnetwork such as an ATM network or a FDDI network [6], load fluctuations within subnetworks may also introduce traffic pattern distortions. Though it is possible to bound delay over these subnetworks, the delays for different packets will be *variable*. Thus, these solutions do not apply to an internetworking environment.

Finally, in networks with *work-conserving* service disciplines, even in the situations when traffic inside the network can be characterized, the characterization usually represents a burstier traffic inside the network than that at the entrance. This is independent of the traffic model being used. In [3], a deterministic fluid model (σ, ρ) is used to characterize traffic source. A source is said to satisfy (σ, ρ) if during any time interval of length u, the amount of its output traffic is less than $\sigma + \rho u$. In such a model, σ is the maximum burst size, and ρ is the average rate. If the traffic of connection j is characterized by (σ_j, ρ_j) at the entrance to the network, its characterization will be $(\sigma_j + \sum_{h=1}^{i-1} \rho_j \overline{d}_{h,j}, \rho_j)$ at the entrance to the i^{th} server along the path, where $\overline{d}_{h,j}$ is the local delay bound for the connection at the h^{th} switch. Compared to the characterization of the source traffic, the maximum burst size increases by $\sum_{h=1}^{i-1} \rho_j \overline{d}_{h,j}$. This increase of burst size grows monotonically along the path.

In [16], a family of stochastic random variables are used to characterize the source. Connection j is said to satisfy a characterization of $\{(\mathbf{R}_{t_1,j}, t_1), (\mathbf{R}_{t_2,j}, t_2), (\mathbf{R}_{t_3,j}, t_3) ...\}$, where $\mathbf{R}_{t_i,j}$ are random variables and $t_1 < t_2 < \cdots$ are time intervals, if $\mathbf{R}_{t_i,j}$ is *stochastically larger* than the number of packets generated over any interval of length t_i by source j. If the traffic connection j is characterized by $\{(\mathbf{R}_{t_1,j}, t_1), (\mathbf{R}_{t_2,j}, t_2), (\mathbf{R}_{t_3,j}, t_3) ...\}$ at the entrance to the network, its characterization will be
$\{(\mathbf{R}_{t_1+\sum_{h=1}^{i-1} b_h,j}, t_1), (\mathbf{R}_{t_2+\sum_{h=1}^{i-1} b_h,j}, t_2), (\mathbf{R}_{t_3+\sum_{h=1}^{i-1} b_h,j}, t_3), ...\}$ at the h^{th} switch, where b_h is the length of the maximum busy period at switch h. The same random variable $\mathbf{R}_{t_m+\sum_{h=1}^{i-1} b_h,j}$ that bounds the maximum number of packets over an interval of length $t_m + \sum_{h=1}^{i-1} b_h$ at the entrance to the network, now bounds the maximum number of packets over a much *smaller* interval of length t_m at server i. I.e., the traffic is burstier at server i than at the entrance.

[1] A source is said to satisfy (σ, ρ) if during any time interval of length u, the amount of its output traffic is less than $\sigma + \rho u$. In such a model, σ is the maximum burst size, and ρ is the average rate [3]

In both the (σ_j, ρ_j) and $\{(\mathbf{R}_{t_1,j}, t_1), (\mathbf{R}_{t_2,j}, t_2), (\mathbf{R}_{t_3,j}, t_3)...\}$ analysis, the burstiness of a connection's traffic accumulates at each hop along the path from source to destination. More resources such as buffer space and schedulability need to be reserved for a burstier traffic.

3 Non-work-conserving disciplines

In the previous section, we discussed some of the difficulties and limitations in obtaining end-to-end delay bounds in a network with work-conserving servers. In this section, we show that most of the limitations can be overcome by using non-work-conserving disciplines that control the traffic distortion inside the network. We consider a general class of non-work-conserving disciplines called rate-controlled service disciplines. Most of the non-work-conserving disciplines proposed in high speed networks such as Stop-and-Go [8], Jitter-Earliest-Due-Date [26], Hierarchical Round Robin [11], and Rate-Controlled Static Priority Queueing [28] either belong to this class or can be implemented by a rate-controlled server [30]. A rate-controlled server has two components: a rate-controller and a scheduler. The rate-controller is responsible for controlling traffic distortion introduced by multiplexing effects and load fluctuations in previous servers. The scheduler is responsible for multiplexing the regulated traffic and servicing the packets according to certain scheduling policy. Various rate-control and scheduling policies can be used. In this section, we define two general classes of rate-control policies. We show that by combining the proposed rate-control policies with *any* schedulers that can provide local delay bounds, per connection end-to-end delay bounds can be guaranteed in a network of *arbitrary* topology. The result applies to both simple networks with constant link delays and internetworks with *bounded* but possible *variable* link delays. In addition, the problem that bursts may accumulate along the path is eliminated.

The rest of the section is organized as follows. In Section 3.1 and Section 3.2 we present two classes of rate-control policies: delay-jitter controlling policies and rate-jitter controlling policies. In Section 3.3, we derive the end-to-end delay characteristics and buffer space requirement in a network of non-work-conserving disciplines with delay-jitter and rate-jitter controlling policies.

3.1 Delay-jitter-controlling Policies

The function of the rate-controller is to control traffic distortion introduced in previous servers and links. Controlling traffic distortion is equivalent to controlling jitter. In the literature, there are different definitions of *jitter*. In [11], the term is used to capture the *burstiness* of the traffic, and is defined to be the maximum number of packets in a *jitter averaging interval*. In [5, 26], the term is used to capture the magnitude of the distortion to the traffic pattern caused by the network, and is defined to be the maximum difference between the delays experienced by any two packets on the same connection. In this paper, we call these two quantities *rate jitter* and *delay jitter* respectively. As will be discussed in Section 3.2, we use a more general definition of rate-jitter than that used in [11]. Corresponding to the two definitions of jitter, we study two general classes of rate-control policies: delay-jitter controlling policies, which control delay jitter

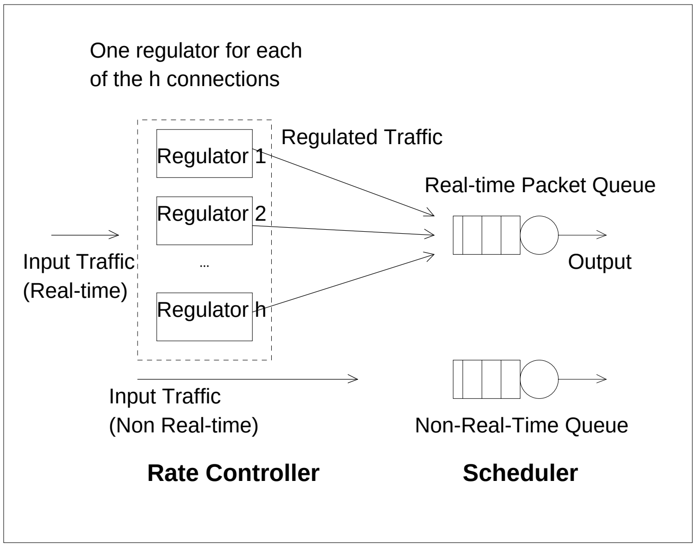


Figure 3: A Rate-Controlled Server

by fully reconstructing the traffic pattern; and rate-jitter controlling policies, which control rate jitter by partially reconstructing the traffic pattern. We will examine delay-jitter controlling policies in this section, and discuss rate-jitter controlling policies in Section 3.2.

Conceptually, a rate controller consists of a set of regulators corresponding to each of the connections traversing the server. Each regulator is responsible for shaping the traffic of the corresponding connection into the desired traffic pattern. Upon arrival of each packet on a connection, the corresponding regulator assigns an eligibility time to the packet, holds it till its eligibility time, then passes it to the scheduler. Different ways of calculating the eligibility time of a packet will result in different regulators.

The eligibility time of a packet for a delay-jitter controlling regulator is defined with reference to the eligibility time of the same packet at the immediately upstream server. The definition assumes that the queueing delays of packets on the connection at the immediately upstream server and the link delay from the upstream server to the current server are bounded. Let $\overline{d}_{i-1,j}$ be the local delay bound for connection j in the scheduler at $i-1^{th}$ server along the path traversed by connection j, and $\overline{\pi}_{i,j}$ be the maximum link delay from the $i-1^{th}$ server to the i^{th} server for any packets on connection j. For a delay-jitter controlling regulator, $ET^k_{i,j}$, the eligibility time of the k^{th} packet at the i^{th} server is defined as:

$$ET^k_{1,j} = AT^k_{1,j} \tag{2}$$

$$ET^k_{i,j} = ET^k_{i-1,j} + \overline{d}_{i-1,j} + \overline{\pi}_{i,j} + \theta_{i,j}, \quad i > 1 \tag{3}$$

where $AT^k_{1,j}$ is the arrival time of the k^{th} packet at the entrance to the network, and $\theta_{i,j}$ is a constant delay.

There are two noteworthy points. First, the maximum link delay is a function of not only the link indexed by i, but also the connection indexed by j. This allows the modeling of an internetworking environment, in which a link connecting two servers may be a subnetwork and different connections may have different delay bounds in the subnetwork. Secondly, notice that the definition of the delay-jitter controlling regulator is not associated with a particular traffic model. This allows the resource management algorithm to use *any* traffic model to characterize the traffic on a connection. Some of the proposed models are (σ, ρ) [3], $(Xmin, Xave, I, Smax)$ [2] [7], and the Bounding Interval Dependent or the BIND model [15].

Service Discipline	$ET^k_{i,j}$
RCSP	$ET^k_{i-1,j} + \overline{d}_{i-1,j} + \overline{\pi}_{i,j}$
Jitter-EDD	$ET^k_{i-1,j} + \overline{d}_{i-1,j} + \pi_i$
Stop-and-Go	$ET^k_{i-1,j} + T_m + \pi_i + \theta_{i,j}$

Table 1: Special Cases of the Delay-Jitter Controlling Regulator

The definition of the delay-jitter controlling policy is general. As shown in Table 1, regulators used in previously proposed non-work-conserving such as RCSP [28], Jitter-EDD [26], and Stop-and-Go [8] are its special cases. In RCSP, $\theta_{i,j}$ in (3) is 0. In Jitter-EDD, $ET^k_{i,j}$ is defined to be $AT^k_{i,j} + Ahead^k_{i-1,j}$, where $AT^k_{i,j}$ is the arrival time of the k^{th} packet at the i^{th} server, and $Ahead^k_{i-1,j}$ is the amount of time the k^{th} packet was ahead of schedule at the $i - 1^{th}$ server. It is easy to show that this definition is equivalent to the definition in Table 1 when the link delay is constant, or $\pi_i = \overline{\pi}_{i,j}$. In Stop-and-Go, the frame time T_m is the local delay bound for the connection at all servers along the path. Also, constant link delay is assumed.

For a delay-jitter regulator, it is easy to show that the following holds:

$$ET^{k+1}_{i,j} - ET^k_{i,j} = AT^{k+1}_{1,j} - AT^k_{1,j} \quad \forall k, i > 0 \tag{4}$$

This leads to the following proposition:

Proposition 1 *Consider a connection that traverses a cascade of rate-controlled servers with delay-jitter controlling regulators. If deterministic delay bounds can be provided at the scheduler of each server and link delay can be bounded at each hop, the traffic pattern of the connection at output of each rate-controller is exactly the same as the traffic pattern of the connection at the entrance to the network. Formally, the following holds:*

$$ET^p_{i,j} - ET^q_{i,j} = AT^p_{1,j} - AT^q_{1,j} \quad \forall p, q, i > 0 \tag{5}$$

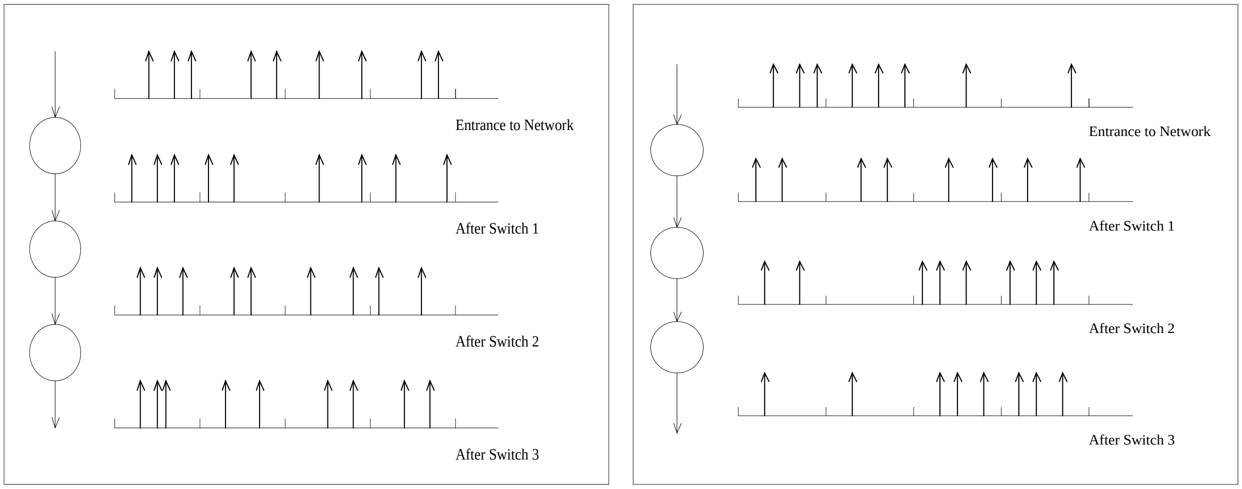
Proposition 1 can be easily proven by using (4) and applying induction on $|p - q|$.

If applications that require guaranteed performance service specify their source traffic characteristics to the network during the connection establishment, the above proposition ensures that per connection traffic characteristics is known at the input to each scheduler over the *entire* path traversed by the connection. This is independent of the traffic model used, and allows the resource reservation algorithm to use *any* traffic model.

[2] In the $(Xmin, Xave, I, Smax)$ characterization, $Xmin$ is the minimum spacing between two packets, $Xave$ is the worst-case average inter-packet spacing over any interval of length I, and $Smax$ is the maximum packet size.

3.2 Rate-jitter-controlling policies

In a network with delay-jitter controlling regulators, traffic pattern on a connection is completely reconstructed at the entrance to each scheduler. Since resource allocation algorithms only use the information on specific traffic characteristics of a connection, it is sufficient to maintain only those characteristics inside the network. In this section, we define a family of rate-jitter controlling policies that only maintain certain traffic characteristics inside the network.



Delay-jitter control in SG

Rate-jitter control in HRR

Figure 4: Comparison of delay-jitter control and rate-jitter control

The example shown in Figure 4 illustrates the difference between rate-jitter control and delay-jitter control by comparing Stop-and-Go [8] and Hierarchical Round Robin (HRR) [11], which use delay-jitter control and rate-jitter control respectively [3]. In both Stop-and-Go and HRR, resource allocation is based on the traffic characterization of (r, T). A connection is said to satisfy (r, T) constraint if no more than $r \cdot T$ bits are transmitted on any frame size of T. In a network with fixed size packets, such as the ATM network, this is equivalent to a characterization of (n, T), where n is the maximum number of packets transmitted in a frame size of T. In the example, it is assumed that 3 packet transmission times are allocated to the connection in each frame. In Stop-and-Go, packets that are transmitted in the same frame at the entrance to the network will be transmitted in the same frame on all the links traversed by the connection. The difference between delays experienced by any two packets from the source to any server is bounded by T, where T is the frame size. In HRR, packets that are transmitted in the same frame at the entrance to the network do not necessarily stay in the same frame inside the network; however, the property that *no more than three packets from the connection are transmitted during one frame time* holds throughout the network.

Rather than defining the eligibility time of a packet with respect to the eligibility time of the same packet

[3] Actually, (5) does not hold in Stop-and-Go. Only $|(ET^p_{i,j} - ET^q_{i,j}) - (AT^p_{0,j} - AT^q_{0,j})| < T$ holds. Thus, Stop-and-Go can be viewed as using an approximate form of delay-jitter-control with a granularity of a frame size T.

in the previous server as done by delay-jitter controlling regulators, rate-jitter controlling regulators define the eligibility time of a packet with reference to the eligibility times of packets arriving earlier at the server on the same connection. By properly defining the eligibility time, a rate-jitter controlling regulator ensures that the output of the regulator satisfies certain traffic characteristics. Since the output of all the regulators become the input to the scheduler, the input traffic to the scheduler is thus characterized. The local delay bound for the scheduler can then be obtained based on the input traffic characterizations and the scheduling policy.

In the literature, different traffic models have been used for different schedulers. For example, Stop-and-Go and HRR use the (r, T) traffic model. Delay-EDD, Jitter-EDD, and RCSP use the $(Xmin, Xave, I, Smax)$ model [7], in which $Xmin$ is the minimum packet inter-arrival time, $Xave$ is the worst-case average packet inter-arrival time over any interval of length I, and $Smax$ is the maximum packet size. In [3], the traffic model of (σ, ρ) is used.

Each of the above models defines a family of parameterized deterministic traffic constraint functions. A monotonic increasing function $b_j(.)$ is called a deterministic traffic constraint function of connection j if during *any* interval of length u, the number of bits arriving on j during the interval is no greater than $b_j(u)$. Or formally, let $A_j(t_1, t_2)$ be the total number of bits that arrive on connection j in the interval of (t_1, t_2), $b_j(.)$ is a traffic constraint function of connection j if $A_j(t, t+u) \le b_j(u)$, $\forall t, u >$ 0. Notice that $b_j(.)$ is a time invariant deterministic bound since it constrains the traffic stream over every interval of length u. For a given traffic stream, there are an infinite number of traffic constraint functions, out of which, a deterministic traffic model defines a parameterized family.

For the discussion of this paper, we assume that a packet arrives at a server right after the time instant its last bit is stored in the buffer, i.e., if a packet of size S arrives at time t, $A_j(t, t+\epsilon) = S$ holds, where ϵ is an arbitrary small time interval. Thus, in order for $b_j(.)$ to be a traffic constraint function for connection j, the following must hold:

$$b_j(0^+) \ge Smax_j \tag{6}$$

where $Smax_j$ is the maximum packet size on the connection.

For each deterministic traffic constraint function, we can construct a rate-jitter controlling regulator with the following definition of $ET_{i,j}^k$:

$$ET_{i,j}^k = min\{v : v \ge max(ET_{i,j}^{k-1}, AT_{i,j}^k),\ \ a_{i,j}(u,v) \le b_j(v-u)\ \ \forall u \le v\} \tag{7}$$

where $a_{i,j}(.,.)$, defined below, is the number of bits on connection j that become eligible in interval (u, v) at the i^{th} server:

$$a_{i,j}(u,v) = \sum_k (S_j^k | u \le ET_{i,j}^k < v) \tag{8}$$

and S_j^k is the number of bits in the k^{th} packet on connection j. Obviously, the following holds:

$$a_{i,j}(u_1, v_1) \ge a_{i,j}(u_2, v_2)\ \ \forall\ u_1 \le u_2 \le v_2 \le v_1 \tag{9}$$

Intuitively, $ET^k_{i,j}$ defined in (7) is the earliest time the k^{th} packet can become eligible without violating the traffic constraint function. From the definition, Proposition 2 immediately follows.

Proposition 2 *The output traffic from a $b_j(.)$ rate-jitter controlling regulator satisfies the traffic constraint function $b_j(.)$.*

(7) is very general and defines a class of rate-jitter controlling policies. Any deterministic traffic model that can be defined with a traffic constraint function has a corresponding rate-jitter controlling regulator. The closed form of (7) can be derived for most of the proposed deterministic traffic models. For example, for the traffic model of $(Xmin_j, Xave_j, I_j, Smax_j)$, the traffic constraint function is:

$$b_j(u) = (min(\lceil \frac{u \ mod \ I_j}{Xmin_j} \rceil, \lceil \frac{I_j}{Xave_j} \rceil) + \lfloor \frac{u}{I_j} \rfloor \lceil \frac{I_j}{Xave_j} \rceil) Smax_j \qquad (10)$$

and the definition of eligibility time of the k^{th} packet on connection j is:

$$ET^k_{i,j} = max(ET^{k-1}_{i,j} + Xmin_j, \ ET_{i,j}^{k - \lfloor \frac{I_j}{Xave_j} \rfloor + 1} + I_j, \ AT^k_{i,j}), \quad k > 1 \qquad (11)$$

where $ET^1_{i,j} = AT^1_i$ and $ET^k_{i,j} = -I_j$ for $k < 0$. Intuitively, the $(Xmin_j, Xave_j, I_j, Smax_j)$ model imposes two constraints: the minimum packet inter-arrival time, which is $Xmin_j$, and the maximum number of packets over any interval of length I_j, which is $\lceil \frac{I_j}{Xave_j} \rceil$. The first two terms on the right hand side of (11) enforce these two constraints. In addition, (11) also requires that a packet's eligibility time be greater than its arrival time, i.e., a packet will not be eligible before has arrived.

For the traffic model of (σ, ρ) , the traffic constraint function is: $\sigma + \rho u$, and the regulator is a leaky bucket [25]. Other traffic models such as the BIND model [15] and the multiple leaky buckets [14] also have corresponding traffic constraint functions and rate-control regulators.

3.3 End-to-end Delay Characteristics

In the previous sections, we have defined two general classes of rate-control policies. The rate-control policies ensure that the input traffic to the scheduler satisfies the *same* traffic characteristics as the source traffic. Since the source traffic characteristics are specified by the communication clients, the characteristics of input traffic to *all* schedulers inside the network are thus known. Various techniques have been developed to bound delay in a single scheduler when the input traffic is characterized by deterministic constraint functions. In [3], the delay characteristics were analyzed for general working conserving disciplines, the First-Come-First-Served discipline, and Locally-First-Come-First-Served disciplines for input traffic characterized with (σ, ρ) model. In [7], the Earliest-Due-Date-First (EDD) policy was studied for input traffic with $(Xmin, Xave, Smax, I)$ model. The bound for EDD was further tightened in [18, 33]. In [28, 29], the Static Priority scheduler was studied for traffic with $(Xmin, Xave, Smax, I)$ model. In [22], the Generalized Processor Sharing (GPS) policy was analyzed for traffic with (σ, ρ) model. In this paper, we don't study any particular scheduler. *Any* scheduler can be used as long as local delay bounds are provided by the scheduler when the input traffic is constrained.

It should be noticed that guaranteeing deterministic delay bounds does not necessarily mean peak-rate allocation [29]. In [15], it has been shown that by using better traffic models and tighter delay analysis techniques, reasonable high network utilizations can be achieved for real MPEG video traces even with deterministic guarantees.

The residence time of a packet in a rate-controlled server consists of two components: the holding time in the rate-controller and the waiting time in the scheduler. In order to provide end-to-end delay bounds, the holding time in the rate-controller has to be taken into account. In [30], end-to-end delay characteristics were studied for rate-controlled service disciplines with a special case of delay-jitter controlling policy and a special case of rate-jitter controlling policy based on the $(Xmin, Xave, I, Smax)$ model. It was shown that end-to-end delays of all the packets on a connection can be bounded, as long as the delays on links and waiting time at each of the schedulers can be bounded. Holding packets in rate controllers increase the end-to-end average delay, but does *not* increase the *end-to-end delay bound* of the connection.

In this section, we show that the same conclusion holds for rate-controlled service disciplines with general rate-control policies as defined by (3) and (7). Formally, the following theorem holds:

Theorem 1 *Consider connection j passing through n servers connected in cascade, with $\overline{\pi}_{i,j}$ and $\hat{\pi}_{i,j}$ being the upper and lower bounds on the link delay from the $i-1^{th}$ server to the i^{th} server. The scheduler of server i guarantees that delays of all packets on the connection be bounded by $\overline{d}_{i,j}$ as long as the connection's input traffic to the scheduler satisfies a deterministic traffic constraint function $b_j(.)$. If the traffic on the connection is constrained by $b_j(.)$ at the entrance to the first server,*

1. *the end-to-end delay for any packet on the connection is bounded by $\sum_{i=1}^{n} \overline{d}_{i,j} + \sum_{i=2}^{n} \overline{\pi}_{i,j}$ if rate-jitter controlling regulators with traffic constraint function $b_j(.)$ are used at each server;*

2. *the end-to-end delay and the delay jitter for any packet are bounded by $\sum_{i=1}^{n} \overline{d}_{i,j} + \sum_{i=2}^{n}(\overline{\pi}_{i,j} + \theta_{i,j})$ and $\overline{d}_{n,j}$, respectively, if delay-jitter controlling regulators are used at each server;*

3. *reservation of $b(\overline{d}_{i,j} + \overline{d}_{i-1,j} + \overline{\pi}_{i,j} - \hat{\pi}_{i,j})$ buffer space for connection j will prevent packet loss at server i.*

To prove the theorem, we first establish the delay characteristics of two servers connected in cascade by the following lemma.

Lemma 1 *Consider connection j traversing two rate-controlled servers which are labeled by $i-1$ and i respectively. For the k^{th} packet on the connection, let $d_{i-1,j}^{k}$ be its delay in the scheduler of server $i-1$, $\pi_{i,j}^{k}$ its link delay between from server $i-1$ to server i, and $h_{i,j}^{k}$ its holding time in the regulator of server i. Assume $\pi_{i,j}^{k} \leq \overline{\pi}_{i,j}$ and $d_{i-1,j}^{k} \leq \overline{d}_{i-1,j}$ holds for all k's if the connection satisfies the traffic constraint function $b_j(.)$ at the input to scheduler $i-1$. We have,*

1. *if a delay-jitter controlling regulator is used at server i, and the connection satisfies the traffic constraint function at the input to scheduler $i-1$,*

$$d^k_{i-1,j} + h^k_{i,j} + \pi^k_{i,j} = \overline{d}_{i-1,j} + \overline{\pi}_{i,j} + \theta_{i,j} \tag{12}$$

2. *if rate-jitter controlling regulators with traffic constraint function $b_j(.)$ are used at both server $i-1$ and i,*

$$d^k_{i-1,j} + h^k_{i,j} + \pi^k_{i,j} \leq \overline{d}_{i-1,j} + \overline{\pi}_{i,j} \tag{13}$$

The proof of the lemma is given in the appendix. Notice that in both (12) and (13), $h^k_{i,j}$ disappears on the right hand side of the inequality, i.e., holding time does not contribute to the delay bound. Intuitively, a packet is held in a regulator only when the packet was transmitted ahead of schedule by the previous server, or, when the packet experienced less delay over the link than the maximum link delay. The amount of holding time in the regulator is never greater than the amount of time the packet is ahead of schedule plus the difference between the maximum link delay and the actual link delay. Thus, holding does not increase the *accumulative delay bound*.

In addition, it can be easily seen from the proof in the appendix that the regulating process is *robust* in the sense that even when some packets are dropped, either due to bit error, or due to buffer overflow when resources are reserved less than that are required in a worst-case senario, the above lemma is still true, i.e., packets who arrive at the regulator on time will also leave the regulator on time. This is particularly important for rate-jitter controlling regulators, because the eligibility time of a packet depends on the eligibility times of previous packets, and in the case when packets are dropped, the sequences of previous arrival packets will vary from one server to the other.

Given Lemma 1, we are now ready to prove Theorem 1.

Proof of Theorem 1. For the first two parts of the theorem, consider the end-to-end delay of the k^{th} packet on the connection, D^k_j, which can be expressed as:

$$D^k_j = \sum_{i=1}^{n}(h^k_{i,j} + d^k_{i,j}) + \sum_{i=2}^{n}\pi_{i,j} \tag{14}$$

where $h^k_{i,j}$ and $d^k_{i,j}$ are the holding and the waiting times at server i, respectively. If we rearrange the terms, (14) becomes:

$$D^k_j = h^k_{1,j} + \sum_{i=2}^{n}(d^k_{i-1,j} + h^k_{i,j} + \pi_{i,j}) + d^k_{n,j} \tag{15}$$

If the traffic obeys the traffic constraint function of $b_j(.)$ at the entrance to the first server, there is no holding time in the first regulator, or $h^k_{1,j} = 0$. D^k_j can then be further simplified as:

$$D^k_j = \sum_{i=2}^{n}(d^k_{i-1,j} + h^k_{i,j} + \pi_{i,j}) + d^k_{n,j} \tag{16}$$

(1) If delay-jitter controlling regulators are used, according to Proposition 1, the traffic satisfies the traffic constraint function $b_j(.)$ at the entrance to each of the schedulers. From Lemma 1, we have

$$d^k_{i-1,j} + h^k_{i,j} + \pi^k_{i,j} = \overline{d}_{i-1,j} + \overline{\pi}_{i,j} + \theta_{i,j} \tag{17}$$

Combining (16) and (17), we have,

$$D^k_j = \sum_{i=2}^{n} (\overline{d}_{i-1,j} + \overline{\pi}_{i,j} + \theta_{i,j}) + d^k_{n,j} \tag{18}$$

Since $0 < d^k_{n,j} \leq \overline{d}_{n,j}$, we have

$$\sum_{i=2}^{n} (\overline{d}_{i-1,j} + \overline{\pi}_{i,j} + \theta_{i,j}) < D^k_j \leq \sum_{i=2}^{n} (\overline{d}_{i-1,j} + \overline{\pi}_{i,j} + \theta_{i,j}) + \overline{d}_{n,j} \tag{19}$$

thus the end-to-end delay bound is $\sum_{i=1}^{n} \overline{d}_{i,j} + \sum_{i=2}^{n} (\overline{\pi}_{i,j} + \theta_{i,j})$, and the end-to-end delay jitter bound is $\overline{d}_{n,j}$.

(2) If rate-jitter controlling regulators are used, according to Proposition 2, the traffic satisfies $b_j(.)$ at the entrance to each of the schedulers. From Lemma 1, we have

$$d^k_{i-1,j} + h^k_{i,j} + \pi^k_{i,j} \leq \overline{d}_{i-1,j} + \overline{\pi}_{i,j} \tag{20}$$

Combining (16) and (20), we have,

$$\begin{aligned} D^k &= \sum_{i=2}^{n} (d^k_{i-1,j} + h^k_{i,j} + \pi_{i,j}) + d^k_{n,j} && (21) \\ &\leq \sum_{i=2}^{n} (\overline{d}_{i-1,j} + \overline{\pi}_{i,j}) + \overline{d}_{n,j} && (22) \\ &= \sum_{i=1}^{n} \overline{d}_{i,j} + \sum_{i=2}^{n} \overline{\pi}_{i,j} && (23) \end{aligned}$$

(3) To verify the third part of the theorem, notice that the longest times a packet can stay in the regulator and the scheduler of the i^{th} server are $\overline{d}_{i-1,j} + \overline{\pi}_{i,j} - \hat{\pi}_{i,j}$ and $\overline{d}_{i,j}$, respectively. From the definition of the traffic constraint function, it follows that $b(\overline{d}_{i-1,j} + \overline{\pi}_{i,j} - \hat{\pi}_{i,j} + \overline{d}_{i,j})$ is the amount of buffer space needed to prevent packet loss. **Q.E.D.**

Notice that the result holds in arbitrary topology networks — the difficulty of bounding end-to-end delay in feedback networks with work-conserving disciplines does not exist any more because of the traffic regulation at each server. Also, the theorem assumes a network model with bounded but possibly *variable* link delays, which make the results applicable to both simple networks and internetworks. In addition, the buffer space requirement for a connection depends only on the local delay bounds at the current and the immediately upstream server. In contrast, for work-conserving policies, more buffer space is needed at downstream servers due to the potential accumulated distortion to the traffic inside the network. For example, if a Delay-EDD scheduler is used, and the $(Xmin, Xave, I, Smax)$ traffic model is adopted and the amount of buffer space required at the i^{th} server along the path traversed by connection j is $\frac{\sum_{h=1}^{i} \overline{d}_{h,j}}{Xmin_j} Smax_j$, where $\overline{d}_{h,j}$ is the local delay bound at the h^{th} server [31].

4 Regulator Tradeoffs

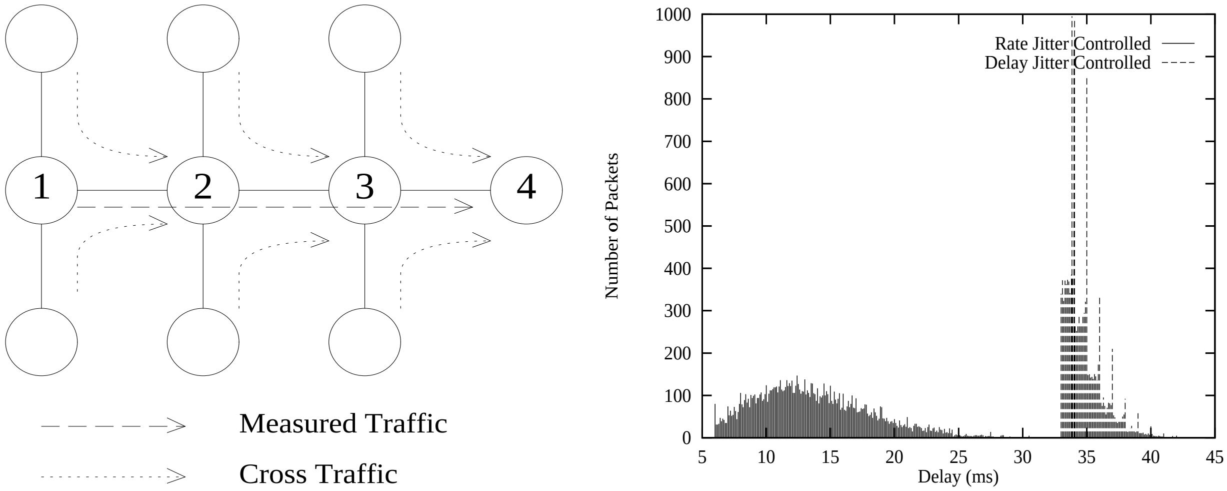
In the previous section, we proposed two general classes of rate-control policies, and showed that, by combining either one with a scheduler that can provide local delay bounds, end-to-end performance guarantees can be obtained in a network of arbitrary topology. In this section, we discuss the tradeoffs of using these two classes of policies. We consider the following three aspects: relative implementation complexity, services offered to the clients, and effects on the design of the network.

The first consideration is the relative complexity of implementation. There are two types of cost associated with implementing a regulator: computing the eligibility time for each packet, and holding packets if necessary. Holding packets is equivalent to managing a set of timers. One mechanism for managing timers is the calendar queue [2]. An implementation of RCSP [28] which is based on the calendar queue and requires constant number of processing steps per packet is proposed in [28]. Another implementation based on a two-dimensional array of shifters is also proposed [19].

Thus, the only difference between a delay-jitter controlling regulator and a rate-jitter controlling regulator is in the computation of the eligibility time. To implement delay-jitter controlling regulators, there must be some mechanism for the synchronization between consecutive switches. In Stop-and-Go, the physical links are framed and the framing structures at the two ends of a link are same. In Jitter-EDD, one more quantity is needed to compute the eligibility time of a packet than in a rate-jitter controlling regulator, i.e., the amount of time the packet was ahead of schedule in the previous switch. This quantity can be calculated in the previous switch and stamped into the packet header as proposed in [26]. If links can have variable delays, synchronized clocks are needed to implement delay-jitter controlling regulators. In an ATM network where each packet is only 53 bytes, the timestamping of each packet is too expensive. Thus, synchronization, either at the link level or at the switch level, is needed to implement delay-jitter control. In a fast packet network, or an internetwork, where the packet size is in the order of kilobytes, timestamping causes relatively small overhead. Also synchronization among hosts and gateways in the current Internet is becoming more and more common due to the widespread use of synchronization protocols such as the Network Time Protocol (NTP) [20].

In both rate-jitter control and delay-jitter control, eligibility time needs to be computed on a per packet basis. This is feasible even for very high speed networks. For example, a 1 Gbps link sending out 53 byte ATM cells must process cells at the rate of approximately 2.4 million cells per second. A 50 MIPS processor is thus allowed 20 instructions per cell, which is more than enough to compute the eligibility time of a cell. For a fast packet network with larger packet size, the number of packets to be processed per second is smaller, thus the number of instructions available to process a packet is larger. Experimental systems that implement non-work-conserving policies are already available. For example, the experimental XUNET switch implements the Hierarchical Round Robin discipline of 16 priority levels at 1 Gbps speed [12].

The second consideration relates to the services that can be provided to the clients. Delay-jitter control within the network provides bounded-delay-jitter service to communication clients for little additional cost. Rate-jitter control is not sufficient to provide bounded-delay-jitter service. This is illustrated by the following simulation experiment. The network being simulated is shown in Figure 5 (a). All links are 10 Mbps. The connection being measured has an end-to-end delay bound of 45 ms. Additional connections are established to introduce cross traffic. Traffic for each connection is generated according to the $(Xmin, Xave, I, Smax)$ model. The packet inter-arrival time is generated from a uniform distribution between $Xmin$ and $Xave$. For the measured connection, the parameters are (20 ms, 35 ms, 1 second, 1Kbits). For the cross traffic, all parameters are generated randomly from uniform distributions, with $Xave$ between 20 ms and 60 ms, $Xmin$ between 10 ms and $Xave$, I between 0.5 second and 2 seconds, and $Smax$ between 500 bits and 2000 bits. The average utilization over each link is 84%.



(a) (b)

Figure 5: Effects of rate-jitter and delay jitter controls

Figure 5 (b) gives the results of two experiments that show the delay distributions of a rate-jitter controlled connection and a delay-jitter controlled connection. As can been seen, packets on both the rate-jitter controlled connection and the delay-jitter controlled connection meet the end-to-end delay bound. The delay jitter on the rate-jitter controlled connection is about 32 ms, whereas the delay jitter on the delay-jitter controlled connection is only about 10 ms. That is, the delay jitter is about three times larger on the rate-jitter controlled connection than on the delay-jitter controlled connection. This is due to the accumulation of traffic pattern distortions at each node in the case of rate-jitter control. If the measured connection were to traverse more hops, the delay jitter for the rate-jitter controlled connection would be even larger, while the delay jitter for the delay-jitter controlled connection would be little affected.

Another observation is that the average delay is much smaller for the rate-jitter controlled connection than for the delay-jitter controlled connection. For some clients, which need both low average delay and bounded delay services, a rate-jitter controlled connection is a better choice. For other clients, especially playback applications like video and audio applications, where the performance indices are the delay bound and the delay-jitter bound instead of the average delay, a delay-jitter controlled connection is a better choice. Notice that the simulation scenario in Figure 5 (a) is very simple. More realistic scenarios may include more complex topologies. The bounded-delay-jitter property is *proven* to hold for networks with delay-jitter controlling regulators even in these more complex settings. For networks with rate-jitter controlling regulators, a more complex topology only introduces more interactions between traffic streams, thereby resulting in a larger delay jitter.

The third consideration in deciding whether to use a rate-jitter controlling regulator or a delay-jitter controlling regulator is the effect on network design. Using delay-jitter controlling regulators will *completely* reconstruct traffic patterns at each regulator; the traffic pattern at the entrance to each scheduler will be *exactly* the same as that at the entrance to the network. Thus, if we can characterize the statistical properties of a source, the same statistical properties will hold throughout the network. By using this observation, we can provide end-to-end statistical guarantees as described in [32]. Using rate-jitter controlling regulators only *partially* reconstructs traffic patterns at each regulator; some statistical properties of the traffic may be lost, and therefore it may be difficult to provide statistical guarantees.

In summary, there are advantages and disadvantages to both delay-jitter control and rate-jitter control. In an internetworking environment, while per packet timestamping and synchronization among routers and hosts incur little overhead, delay-jitter control seems to be a better solution. This is especially true considering the accumulative property of delay-jitter and the fact that the average number of hops traversed by a connection in Internet is about 20 [10]. In an ATM environment, since it is impossible to timestamp every packet, rate-jitter control may be a better alternative.

5 Related Work

Several non-work-conserving disciplines have been proposed in the context of high-speed networks. They are Jitter-Earliest-Due-Date (Jitter-EDD) [26], Hierarchical Round Robin [11], Stop-and-Go [8], and Rate-Controlled Static Priority (RCSP) [28]. In each case, it has been shown that end-to-end deterministic delay bounds can be provided. For Jitter-EDD, Stop-and-Go, and RCSP, it has also been shown that non trivial end-to-end delay jitter bound can be provided. In [9] and [32], solutions for providing end-to-end statistical guarantees are presented for Stop-and-Go and RCSP respectively.

In [30], Zhang and Ferrari showed that all the above disciplines belong to the general class of non-work-conserving policies called rate-controlled service disciplines. They studied rate-controlled service disciplines with a delay-jitter controlling policy more narrowly defined than the one used in this paper, and a rate-jitter controlling policy for the $(Xmin, Xave, I, Smax)$ traffic model.

Cruz's work [3, 4] pioneered bounding techniques in both a single server environment and in a general networking environment. In [4], he discussed the difficulty of bounding end-to-end delay in a feedback network of work-conserving disciplines, and proposed the use of regulators to increase the throughput region. His analysis focused mainly on the (σ, ρ) regulator.

Parekh's work [22, 23] on Generalized Processor Sharing (GPS) was the first to show that end-to-end delay bounds can be obtained in arbitrary topology networks with work-conserving servers. Also, the end-to-end delay bound obtained is tighter than the simple addition of all the worst-case local delay bounds at each server, as has been done in this paper. However, the result only holds for a restricted class of resource assignments. It is still an open problem whether end-to-end delay can be bounded in an arbitrary topology network with GPS servers under general resource assignment.

In [16], Kurose developed techniques of providing end-to-end stochastic bounds in a networking environment. He also assumed work-conserving servers and discovered the difficulties in deriving bounds in feedback networks. Further, his analysis showed the problem of accumulation of traffic distortion in a network of work-conserving servers.

In [27], Yates et al. conducted a simulation study and showed that end-to-end delays observed in simulation experiments are significantly smaller than delay bounds obtained by *any* of the above bounding techniques. The simulation was based on a feed-forward network, with homogeneous sources. It is unclear whether the conclusion can be extended to general networks with heterogeneous sources.

6 Summary

Providing end-to-end delay bounds in a networking environment is difficult because bounding delay in any server requires a bound on the input traffic, but complex interactions among traffic streams distort the traffic pattern so that the traffic inside the network is different from the source traffic. Previous techniques of bounding end-to-end delay in a networking environment start from the source traffic characterization and iteratively “push” the traffic characterization inside the network. Such an approach has a number of limitations. First, they usually apply to only feed-forward networks, but not general feedback networks. Secondly, they assume a network model with constant link delays, which make them difficult to apply to an internetworking environment where links between two switches may be subnetworks. In addition, the characterization of the traffic inside the network usually represents a burstier traffic than that at the source.

In this paper, we studied a general class of non-work-conserving disciplines, which control the traffic pattern distortion and maintain certain traffic characteristics inside the network. We presented two general classes of rate-control policies: delay-jitter controlling policies, which maintain the exact traffic pattern inside the network as that at the entrance to the network, and rate-jitter controlling policies, which only maintain the traffic characteristics that are used by the resource allocation algorithm.

Rate-control introduces additional delay inside the network. However, it only increases the *average* delay, but not the end-to-end delay *bound*. We showed that end-to-end delay bound and delay-jitter bound can be guaranteed in a network of non-work-conserving servers. The result overcomes limitations in previous work on bounding end-to-end delays in a network of work-conserving servers. It applies to arbitrary topology networks, instead of just feed-forward networks. Also, it applies to an internetworking environment since it only needs a network model with bounded link delay. Furthermore, the characterization of the traffic inside the network is the same as that at the entrance rather than burstier.

The result is quite general. In the analysis, we do not require any particular traffic model, but only assume the source traffic be bounded by a general deterministic constraint function. Also, we do not require any particular multiplexing or scheduling policy, but only assume that the scheduler provide a local delay bound to a connection when the traffic on that connection satisfies a deterministic constraint function.

While work-conserving service disciplines are dominant in conventional networks, non-work-conserving service disciplines exhibit unique advantages that are suitable for supporting the guaranteed performance service class in integrated services networks.

References

[1] A. Banerjea and S. Keshav. Queueing delays in rate controlled networks. In *Proceedings of IEEE INFOCOM'93*, pages 547–556, San Francisco, CA, April 1993.

[2] R. Brown. Calendar queues: A fast O(1) priority queue implementation for the simulation event set problem. *Communications of the ACM*, 31(10):1220–1227, October 1988.

[3] R. Cruz. A calculus for network delay, part I : Network elements in isolation. *IEEE Transaction of Information Theory*, 37(1):114–121, 1991.

[4] R. Cruz. A calculus for network delay, part II : Network analysis. *IEEE Transaction of Information Theory*, 37(1):121–141, 1991.

[5] D. Ferrari. Client requirements for real-time communication services. *IEEE Communications Magazine*, 28(11):65–72, November 1990.

[6] D. Ferrari. Real-time communication in an internetwork. *Journal of High Speed Networks*, 1(1):79–103, 1992.

[7] D. Ferrari and D. Verma. A scheme for real-time channel establishment in wide-area networks. *IEEE Journal on Selected Areas in Communications*, 8(3):368–379, April 1990.

[8] S. Golestani. A stop-and-go queueing framework for congestion management. In *Proceedings of ACM SIGCOMM'90*, pages 8–18, Philadelphia Pennsylvania, September 1990.

[9] S. Golestani. Duration-limited statistical multiplexing of delay-sensitive traffic in packet networks. In *Proceedings of IEEE INFOCOM'91*, April 1991.

[10] V. Jacobson. Keynote speech. In *USENIX Symposium on High-Speed Networking*, Oakland, CA, August 1994.

[11] C. Kalmanek, H. Kanakia, and S. Keshav. Rate controlled servers for very high-speed networks. In *IEEE Global Telecommunications Conference*, pages 300.3.1 – 300.3.9, San Diego, California, December 1990.

[12] C. Kalmanek, S. Morgan, and R. C. Restrick. A high performance queueing engine for ATM networks. In *Proceedings of 14th International Switching Symposium*, Yokahama, Japan, October 1992.

[13] L. Kleinrock. *Queueing Systems*. John Wiley and Sons, 1975.

[14] E. Knightly, D. Wrege, J. Liebeherr, and H. Zhang. Fundamental limits and tradeoffs for providing deterministic guarantees to VBR video traffic. In *Proceedings of ACM Sigmetrics'95*, Ottawa, CA, May 1995.

[15] E. Knightly and H. Zhang. Traffic characterization and switch utilization using deterministic bounding interval dependent traffic models. In *Proceedings of IEEE INFOCOM'95*, Boston, MA, April 1995.

[16] J. Kurose. On computing per-session performance bounds in high-speed multi-hop computer networks. In *ACM Sigmetrics'92*, 1992.

[17] J. Kurose. Open issues and challenges in providing quality of service guarantees in high-speed networks. *ACM Computer Communication Review*, 23(1):6–15, January 1993.

[18] J. Liebeherr, D. Wrege, and D.Ferrari. Exact admission control for networks with bounded delay services. Technical Report CS-94-29, University of Virginia, Department of Computer Science, July 1994.

[19] M. Maresca, June 1993. Personal communication.

[20] D. Mills. Internet time synchronization: the Network Time Protocol. *IEEE Transactions on Communications*, 39(10):1482–1493, October 1991.

[21] A. Parekh. *A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks*. PhD dissertation, Massachusetts Institute of Technology, February 1992.

[22] A. Parekh and R. Gallager. A generalized processor sharing approach to flow control - the single node case. In *Proceedings of the INFOCOM'92*, 1992.

[23] A. Parekh and R. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The multiple node case. In *Proceedings of the INFOCOM'93*, pages 521–530, San Francisco, CA, March 1993.

[24] J. Stankovic and K. Ramamritham. *Hard Real-Time Systems*. IEEE Computer Society Press, 1988.

[25] J. Turner. New directions in communications(or which way to the information age?). *IEEE Communication Magazine*, 24(10), October 1986.

[26] D. Verma, H. Zhang, and D. Ferrari. Guaranteeing delay jitter bounds in packet switching networks. In *Proceedings of Tricomm'91*, pages 35–46, Chapel Hill, North Carolina, April 1991.

[27] D. Yates, J. Kurose, D. Towsley, and M. Hluchyi. On per-session end-to-end delay distributions and the call admission problem for real-time applications with qos requirements. In *Proceeedings of ACM SIGCOMM'93*, pages 2–12, San Francisco, CA, September 1993.

[28] H. Zhang and D. Ferrari. Rate-controlled static priority queueing. In *Proceedings of IEEE INFOCOM'93*, pages 227–236, San Francisco, California, April 1993.

[29] H. Zhang and D. Ferrari. Improving utilization for deterministic service in multimedia communication. In *1994 International Conference on Multimedia Computing and Systems*, pages 295–304, Boston, MA, May 1994.

[30] H. Zhang and D. Ferrari. Rate-controlled service disciplines. *Journal of High Speed Networks*, 3(4):389–412, 1994.

[31] H. Zhang and S. Keshav. Comparison of rate-based service disciplines. In *Proceedings of ACM SIGCOMM'91*, pages 113–122, Zurich, Switzerland, September 1991.

[32] H. Zhang and E. Knightly. Providing end-to-end statistical performance guarantees with interval dependent stochastic models. In *ACM Sigmetrics'94*, pages 211–220, Nashville, TN, May 1994.

[33] Q. Zheng and K. Shin. On the ability of establishing real-time channels in point-to-point packet-switching networks. *IEEE Transactions on Communications*, pages 1096–1105, March 1994.

A Proof of Lemma 1

In the following, we omit the subscript of j for simplicity.

Let ET_{i-1}^k and ET_i^k be the eligibility times for the k^{th} packet at server $i-1$ and i, respectively. DT_{i-1}^k is the departure time of the k^{th} packet from server $i-1$, and AT_i^k is the arrival time of the k^{th} packet at server i. We have

$$d_{i-1}^k = DT_{i-1}^k - ET_{i-1}^k \tag{24}$$

$$h_i^k = ET_i^k - AT_i^k \tag{25}$$

$$\pi_i^k = AT_i^k - DT_{i-1}^k \tag{26}$$

Combining (24), (25), and (26), we have

$$d_{i-1}^k + h_i^k + \pi_i^k = ET_i^k - ET_{i-1}^k \tag{27}$$

1. For the case of a delay-jitter controlling regulator, from (27) and (3), we immediately have

$$d_{i-1}^k + h_i^k + \pi_i^k = \overline{d}_{i-1} + \overline{\pi}_i + \theta_i \tag{28}$$

2. For the case of a rate-jitter-controlling regulator, we will prove the lemma by applying induction with respect to k.

Step 1. With $k = 1$, from (6) and (7), we have $ET_i^1 = AT_i^1$. It follows that $h_i^1 = ET_i^1 - AT_i^1 = 0$. Also we have $d_{i-1}^1 \leq \overline{d}_{i-1}$ and $\pi_i^1 \leq \overline{\pi}_i$, it follows

$$d_{i-1}^1 + h_i^1 + \pi_i^1 \leq \overline{d}_{i-1} + \overline{\pi}_i \tag{29}$$

So (13) holds for k=1.

Step 2. Assume that (13) holds for the first $k-1$ packets. We now consider the k^{th} packet, i.e., we want to show

$$ET_i^k - ET_{i-1}^k \leq \overline{d}_{i-1} + \overline{\pi}_i \tag{30}$$

We will prove (30) by contradiction. Suppose (30) does not hold, or

$$ET_i^k - ET_{i-1}^k > \overline{d}_{i-1} + \overline{\pi}_i \tag{31}$$

we have the following by re-arranging the terms in (31)

$$ET_i^k > ET_{i-1}^k + \overline{d}_{i-1} + \overline{\pi}_i \tag{32}$$

Also, from $AT_i^k = ET_{i-1}^k + d_{i-1}^k + \pi_i^k$, $d_{i-1}^k \leq \overline{d}_{i-1}$ and $\pi_i^k < \overline{\pi}_i$, we have

$$AT_i^k \leq ET_{i-1}^k + \overline{d}_{i-1} + \overline{\pi}_i \tag{33}$$

According to (7), ET_i^k is the smallest v greater than AT_i^k such that $a(u,v) < b(v-u)$ always holds. Since $ET_{i-1}^k + \overline{d}_{i-1} + \overline{\pi}_i$ is greater than AT_i^k and less than ET_i^k, there exists u such that

$$b(ET_{i-1}^k + \overline{d}_{i-1} + \overline{\pi}_i - u) < a_i(u, ET_{i-1}^k + \overline{d}_{i-1} + \overline{\pi}_i) \tag{34}$$

Also, there is the p^{th} packet such that:

$$ET_i^{p-1} < u \leq ET_i^p \tag{35}$$

From (6), it is easy to show that $p \neq k$. Also since $p \leq k$, it follows that $p \leq k-1$.
According to the definition of $a(.,.)$, we have

$$a_i(u, ET_{i-1}^k + \overline{d}_{i-1} + \overline{\pi}_i) = a_i(ET_i^p, ET_{i-1}^k + \overline{d}_{i-1} + \overline{\pi}_i) \tag{36}$$

Combining (34) and (36), we have

$$\begin{align}
b(ET_{i-1}^k + \overline{d}_{i-1} + \overline{\pi}_i - u) &< a_i(ET_i^p, ET_{i-1}^k + \overline{d}_{i-1} + \overline{\pi}_i) \tag{37}\\
&< a_i(ET_i^p, ET_i^k) \tag{38}\\
&\leq a_{i-1}(ET_{i-1}^p, ET_{i-1}^k) \tag{39}\\
&\leq b(ET_{i-1}^k - ET_{i-1}^p) \tag{40}\\
&\leq b(ET_{i-1}^k - (ET_i^p - \overline{d}_{i-1} - \overline{\pi}_i)) \tag{41}\\
&= b(ET_{i-1}^k + \overline{d}_{i-1} + \overline{\pi}_i - ET_i^p) \tag{42}
\end{align}$$

(38) holds due to (32) and (9). (39) is from packet conservation law within the network: all the packets arriving at server i are coming from server $i-1$, and no packets are generated within the network. (40) holds because server $i-1$ uses a rate-jitter controlling regulator with the same traffic constraint function $b(.)$. (41) is because $b(.)$ is a monotonously increasing function and $ET_i^p \leq ET_{i-1}^p + \overline{d}_{i-1} + \overline{\pi}_i$ (from the assumption that (13) holds for the first to the $(k-1)^{th}$ packet) or $ET_{i-1}^p \geq ET_i^p - \overline{d}_{i-1} - \overline{\pi}_i$.

Combining (37) to (42), we have

$$b(ET_{i-1}^k + \overline{d}_{i-1} + \overline{\pi}_i - u) < b(ET_{i-1}^k + \overline{d}_{i-1} + \overline{\pi}_i - ET_i^p) \tag{43}$$

However, from (35) we have

$$ET_{i-1}^k + \overline{d}_{i-1} + \overline{\pi}_i - u \geq ET_{i-1}^k + \overline{d}_{i-1} + \overline{\pi}_i - ET_i^p \tag{44}$$

(43) and (44) contradict with the assumption that $b(.)$ is a monotonously increasing function.

Therefore, (30) holds, i.e., the lemma holds for the k^{th} packet.

From Step 1 and Step 2, it follows that the lemma holds for *any* packet on the connection.