Multirate Scheduling of VBR Video Traffic in ATM Networks

*Debanjan Saha** *Sarit Mukherjee*† *Satish K. Tripathi*‡

Abstract

One of the major attractions of Asynchronous Transfer Mode (ATM) networks for transporting bursty video traffic is its ability to exploit the multiplexing gains of packet switching, while providing quality of service guarantees. Unfortunately, most of the multiplexing mechanisms proposed in the literature fail to exploit this feature. We propose a multirate service mechanism that allows a session to be served at different rates at different of times. Applications generating bursty data, such as VBR video, can take advantage of multirate service by requesting for a peak rate of service for brief periods of bursty arrivals and a much lower average rate of service for all other times. Consequently, the applications can improve their delay performance without reserving peak bandwidth for the entire duration of the sessions. Furthermore, the scheduler can multiplex the peaks and the lulls in service rates of different sessions and there by increasing the utilization of the system. Using MPEG video traces from a number of applications, we show that our scheme outperforms Packet-by-packet Generalized Processor Sharing (PGPS) in terms of number of connections admitted, while providing the same level of service guarantees. We also investigate the performance of multirate service when service quality need not be guaranteed. We refer to this as predictive service. We propose a measurement based admission control procedure for predictive service and show that it helps increase the size of the admissible region even further.

*IBM T.J. Watson Research Center, Yorktown Heights, NY 10598. Email:debanjan@watson.ibm.com.
†Dept. of Computer Science & Engg. University of Nebraska, Lincoln, NE 68588. Email:sarit@cse.unl.edu.
‡Dept. of Computer Science, University of Maryland, College Park, MD 20742. Email:tripathi@cs.umd.edu.

1 Introduction

One of the major attractions of Asynchronous Transfer Mode (ATM) networks is its ability to exploit the multiplexing gains of packet switching, while providing quality of service guarantees. A number of service architectures have been proposed [1] to realize this goal. At the heart of all of these service architectures is a multiplexing policy that is used to allocate link capacities to competing connections at the switching nodes. The manner in which multiplexing is performed has a profound effect on what service guarantees are provided and to what extent the multiplexing gain is exploited.

Multiplexing mechanisms proposed for ATM networks can be broadly classified into two classes: (a) ones that provide guarantees on maximum delay at the switching nodes, and (b) ones that guarantee a minimum throughput. The multiplexing disciplines providing delay guarantees [18, 3] typically use static or dynamic priority based scheduling to bound the worst case delay of a connection at each switching node. The end-to-end delay is computed as the sum of the delays at the switching nodes on the path of the connection. The multiplexing disciplines providing throughput guarantees [2, 5, 12, 19, 16, 15, 14] typically use variations of fair queueing or frame based scheduling to guarantee a minimum rate of service at each switching node. Knowing the traffic arrival process, this rate guarantee can be translated into guarantees on other performance metrics, such as delay, delay jitter, worst-case buffer requirement, etc.

Rate based schemes are preferred over schemes providing delay guarantees primarily because of their simplicity. Typically, they offer a fixed rate of service to a connection for the entire duration of its life time. While a fixed rate of service is adequate for constant bit rate traffic, it is quite unsuitable for bursty traffic, such as variable bit rate (VBR) video. For example, an MPEG [10] coded video stream generates traffic at significantly different rates at different times and no single rate service is sufficient to transport it across the network.

We propose a multirate service mechanism to address this problem. In our scheme, a connection is served at different rates at different times. For example, a session can be serviced at the peak rate during the times of bursty arrivals and at a lower rate at all other times. In general, we can have more than two rates of service. The length and the service rate of each of the service periods is specified by the applications at the time of connection setup. A multirate service discipline is superior to its single rate counter-part in two ways. It allows the applications generating bursty data to request for a higher rate of service for the periods of bursty arrivals and a lower average rate of service at other times. Consequently, applications can improve their delay performance without reserving a high bandwidth for the entire duration of the session. The scheduler can exploit this feature by multiplexing the peaks and the lulls in service rates of different sessions, and thereby increasing the utilization of the system. The complexity of the scheduling mechanism underlying multirate service is comparable to that of Packet-by-packet Processor Sharing (PGPS) [12] and Self Clocked Fair Queueing (SFQ) [6]. Recently it has been shown [17] that variations of PGPS and SFQ can be implemented with $O(loglog(N))$ complexity, where N is the queue length.

We consider two different service models: (a) guaranteed service, and (b) predictive service. In the guaranteed service model each connection is guaranteed a specific end-to-end delay and jitter. It also guarantees no cell loss. Although some applications cannot do without a guaranteed service, there exists a large class of applications that are robust against occasional cell losses and delay violations. The predictive service is designed for these applications. In predictive service, each connection is promised a specific grade of service with the understanding that it may be violated at times. If the applications are robust to these violations, the lack of strict guarantees can help increase the system utilization. The scheduling mechanisms used for both guaranteed and predictive services are the same. They differ only in the admission control process. In the guaranteed case we always assume the worst case scenario. A new connection is admitted if and only if its admittance does not violate the service qualities promised to all connections, assuming worst case traffic arrivals. In the predictive service, the same admission control process is administered with the difference that instead of worst case estimate, we use a measurement based estimate of traffic arrivals [7].

Using MPEG video traces from a variety of real life applications (including news clips, basketball games, class lecture, and music videos), we have shown that multirate scheduling outperforms PGPS in terms of number of connections admitted, while providing the same level of service guarantees. We also investigate the performance of multirate scheduling in the context of predictive service. We propose a measurement based admission control procedure for predictive service and show that it helps increase the size of the admissible region even further.

Except for some very recent works, multirate service mechanisms have not undergone a very through investigation. In [4] a hop-by-hop shaping mechanism is proposed. It can be adapted to provide multirate service. However, the impact of this service mechanism on real life application traffic is not addressed. A multiple time scale characterization of traffic has been proposed in [9, 8]. In [9, 8] the improvement in network utilization due to this enhanced traffic characterization is demonstrated in the context of rate controlled static priority scheduling (RCSP). Authors show that when the characterization of the sources is sufficiently accurate, a high network utilization is achievable. However, it is difficult to characterize a source accurately using commonly used usage parameter control (UPC) mechanisms, such as leaky buckets.

The rest of the paper is organized as follows. In section 2 we discuss how an arbitrarily bursty source can be smoothed to a multirate source. Section 3 is dedicated to details of the scheduling mechanism, and the admission control algorithm. Numerical results are presented in section 4. We conclude in section 5.

2 Traffic Profiles

In order to understand the suitability of multirate service for bursty traffic let us consider an application generating MPEG coded video. From uncompressed video data, an MPEG encoder produces a sequence of encoded frames. There are three types of encoded frames: I (intracoded), P (predicted), and B (bidirectional). The sequence of frames are specified by two parameters: M, the distance between I and P frames, and N, the distance between I frames. For example, when M is 2 and N is 5, the sequence of encoded frames is IBPBIBPB ...(see figure 1). The pattern IBPB repeats indefinitely. The interarrival time t between two successive frames is fixed, and depends on the frame rate. In general, an I frame is much larger than a P frame, and a P frame is much larger than a B frame. Typically, the size of an I frame is larger than the size of a B frame by an order of magnitude.

Let us assume that $|I|$, $|P|$, and $|B|$ be the sizes of I, P, and B frames, respectively [1].

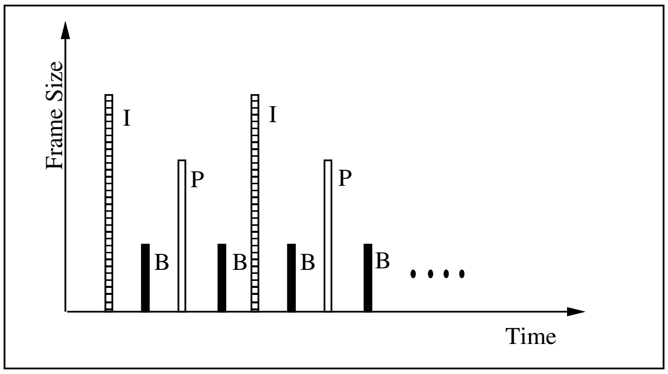

Figure 1: An example of an MPEG coded stream.

Now consider the problem of choosing a single rate of service for the video stream described above. We can either choose a long term average rate, a short term peak rate, or any rate in between. If we choose the peak rate of service, that is $|I|/t$, delay in the network is minimal. However, since the source generates traffic at the peak rate only for a small fraction of the time, a peak rate allocation leads to severe under-utilization of network resources. If we choose the average rate of service, that is $(|I| + |P| + 2|B|)/4t$, the network utilization is high, but only at the cost of increased network delay. Clearly, neither the average rate, nor the peak rate is a good choice. As a matter of fact no single rate is good choice since the source generates traffic at different rates at different times. The ideal approach is to have a service curve that mimics the traffic generation pattern of the source. Unfortunately, this is not a feasible since: (a) it is difficult to capture the traffic generation pattern of an arbitrary source accurately, and (b) it results in a complex service curve that is very difficult to realize using a simple scheduling mechanism.

A good approximation to this ideal service policy is to model a bursty source as a multirate source that generates traffic at a few different rates over different periods of time. We can then use a service curve that resembles this approximated source. For example, we can approximate the MPEG video source described above as one that generates traffic at the peak rate $|I|/t$ over a period of time t, but maintains an average rate of $(|I| + |P| + 2|B|)/4t$ over a period of length $4t$. We show in the next section that a service curve that mimics the traffic generation pattern of this approximated source is easily realizable using a simple scheduling mechanism.

In the rest of the discussion we assume that the traffic generated by a source is passed through a shaper before entering the network. The shaper smoothes an arbitrarily bursty source to a multirate source that generates traffic at a finite number of different rates over different periods of time. The service curve used at the network nodes mimics the traffic envelope enforced by the shaper. Besides smoothing traffic, the shaper also plays the dual role of policing a connection so that it conforms to its advertised traffic envelope.

[1] In general, $|I|$, $|P|$, and $|B|$ are random variables. However, it is not unreasonable to assume that $|I|$ is larger than $|P|$ and $|P|$ is larger than $|B|$.

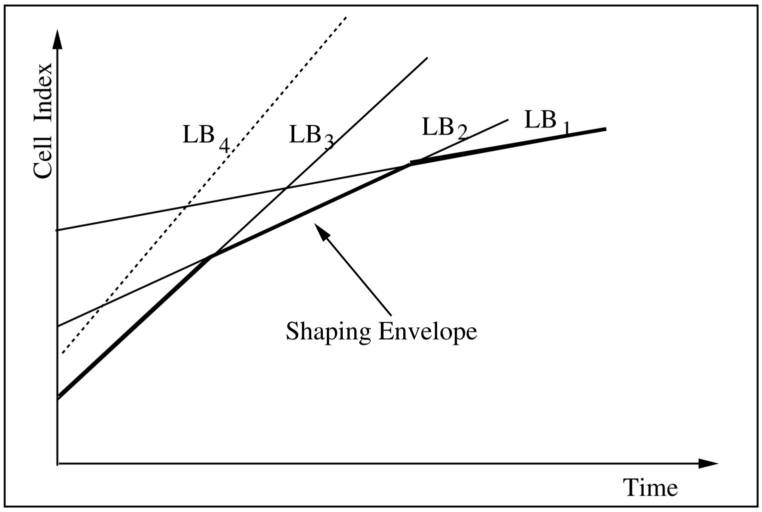

Figure 2: Shaping with multiple leaky buckets.

Several shaping mechanisms enforcing different classes of traffic envelopes have been proposed in the literature. The most popular among them are *leaky bucket*, *jumping window* and *moving window* [13]. In this paper we restrict ourselves to leaky bucket shapers only. However, the techniques developed here can be extended to moving and jumping window shapers also. A leaky bucket shaper consists of a token counter and a timer. The counter is incremented by one each t units of time and can reach a maximum value b. A cell is admitted into the system/network if and only if the counter is positive. Each time a cell is admitted, the counter is decremented by one. The traffic generated by a leaky bucket regulator consists of a burst of up to b cells followed by a steady stream of cells with a minimum inter-cell time of t. A single leaky bucket enforces a specific rate constraint on a source, typically a declared peak or average rate. To enforce multiple rate constraints, we can use multiple leaky buckets, each enforcing different rate constraints. For example, two leaky buckets arranged in series can be used to enforce a short-term peak rate and a long-term average rate on a source.

The traffic envelope enforced by a composite leaky bucket is the intersection of the traffic envelopes of the constituent leaky buckets. In figure 2 a composite leaky bucket consisting of leaky buckets LB_1, LB_2, LB_3, and LB_4 is shown. The composite traffic envelope is marked by the dark line. The exact shape of the envelope depends on the number of components and the associated parameters. Inappropriate choice of shaper parameters may give rise to redundant components which may not have any role in defining the traffic envelope. For example, LB_4 is a redundant component in the composite shaper shown in figure 2. For the ease of exposition, in the rest of the paper we assume that all traffic sources are shaped using two leaky buckets. One of the buckets monitors the short term peak rate of the source and the other controls the long term average rate.

So far we have focused on characterizing traffic coming out of shapers and entering the network. The more difficult problem is to choose appropriate shaper parameters given the characterization of the traffic source [11]. We assume that the source is orchestrated, that is we know exact form of the traffic generated by the source. We characterize a traffic source as a finite sequence of tuples of the form $\langle f_i, t_i \rangle$, where f_i is the volume of data generated at time t_i. Although all traffic sources can be mapped into this model, it is particularly useful for characterizing video sources. In the rest of the discussion we assume that f_is and t_is are all known.

First, let us consider the simpler problem of choosing the parameters for a simple leaky bucket. A single leaky bucket is characterized by (b, r), where b is the size of the bucket and $r = 1/t$ is the rate of token generation. Let us denote by $x(t_i)$ the number of tokens available at time t_i. Precisely speaking, $x(t_i)$ is equal to the number of tokens in the token bucket when it holds a non-zero number of tokens. When there are no tokens in the token bucket, $x(t_i)$ equals the number of cells in the shaper buffer with the sign reversed (negative). That is, $x(t_i)$ represents the current state of the shaper. A positive value represents a credit, and a negative value represents a debit. The following lemma expresses $x(t_i)$ in terms of b, r, and the traffic arrival pattern.

Lemma 2.1 *Given an arrival sequence $\langle f_i, t_i \rangle$, $i = 0, \ldots, N$, and a leaky bucket (b, r) the number of tokens (positive or negative) $x(t_i)$ present at time t_i can be expressed as,*

$$x(t_i) = \min_{0 \le j \le i} \{b + rt_i - rt_j - \mathcal{F}(j, i-1)\},$$

where $\mathcal{F}(i, j)$ is the sum of f_ks from $k = i$ to $k = j$.

Proof: We prove this by induction.

Base Case: For $i = 0$, we have $x(t_0) = b$, which is the number of tokens in the leaky bucket at the system initialization time.

Inductive Hypothesis: Assume that the premise holds for all $i \le k$. To prove that it holds for all i, we need to show that it holds for $i = k + 1$.

$$\begin{aligned} x(t_{k+1}) &= \min\{b, x(t_k) + rt_{k+1} - rt_k - f_k\} \\ &= \min\{b, \min_{0 \le j \le k} \{b + rt_k - rt_j - \mathcal{F}(j, k-1)\} + rt_{k+1} - rt_k - f_k\} \\ &= \min_{0 \le j \le k+1} \{b + rt_{k+1} - rt_j - \mathcal{F}(j, k)\} \end{aligned}$$

This completes the proof. □

If B is the size of shaper buffer, to guarantee loss-less shaping we have to satisfy the following set of constrains:

$$\begin{aligned} x(t_i) &\ge f_i - B, 0 \le i < N \\ \text{or, } \min_{0 \le j \le i} \{b + r(t_i - t_j) - \mathcal{F}(j, i-1)\} &\ge f_i - B, 0 \le i < N \\ \text{or, } b + r(t_i - t_j) + B &\ge \mathcal{F}(j, i), 0 \le i < N, 0 \le j \le i \end{aligned}$$

This is a linear programming formulation (the linear constraints) and and can be easily solved when the objective function is linear. For some specific non-linear objective functions also, the problem is solvable.

It is quite easy to extend this linear programming formulation to a composite leaky bucket. Assume n leaky buckets (b_i, r_i), such that $b_i > b_j$ and $r_i < r_j$ for $1 \leq i < j \leq n$. We denote by $x_k(t_i)$ the number of tokens (credit or debit) available in bucket k. Using the results from lemma 2.1 we can express $x_k(t_i)$ in terms of b_k, r_k, and the traffic arrival pattern. Let $x(t_i)$ denote the number of tokens available for the composite leaky bucket at time instant t_i. It is easy to observe that $x(t_i)$ is the minimum of $x_k(t_i)$s, where $k = 1, \ldots, n$.

Now, if B is the size of the shaper buffer, to guarantee loss-less shaping we have to satisfy the following set of conditions:

$$\begin{aligned}
x(t_i) &\geq f_i - B, 0 \leq i < N \\
\text{or, } \min_{1 \leq k \leq n} \{x_k(t_i)\} &\geq f_i - B, 0 \leq i < N \\
\text{or, } x_k(t_i) &\geq f_i - B, 1 \leq k \leq n, 0 \leq i < N \\
\text{or, } \min_{0 \leq j \leq i} \{b_k + r_k(t_i - t_j) - \mathcal{F}(j, i-1)\} &\geq f_i - B, 1 \leq k \leq n, 0 \leq i < N \\
\text{or, } b_k + r_k(t_i - t_j) + B &\geq \mathcal{F}(j, i), 1 \leq k \leq n, 0 \leq j \leq i, 0 \leq i < N
\end{aligned}$$

Given a linear objective function, we can use commonly available solvers to find b_is and r_is satisfying the constraint set. The results derived in this section are used in section 4 to obtain the leaky bucket parameters for PGPS that maximize the number of connections admitted, given a traffic trace.

3 Multirate Service

In this section we discuss the scheduling mechanism used to realize a multirate service curve. We also present the admission control algorithms for both guaranteed and predictive services. In general, the service curve for a session can be any convex function of piece-wise linear components, where each component corresponds to a different rate of service over a period of time. For simplicity, we assume that the service curve (see figure 3) consists of two components $\langle b_p, r_p \rangle$ and $\langle b_a, r_a \rangle$, where $b_a \geq b_p$ and $r_a \leq r_p$. Observe that, the service curve mimics the traffic envelope enforced by a dual leaky bucket shaper $\langle \langle b_p, r_p \rangle \langle b_p, r_p \rangle \rangle$.

3.1 Scheduling Mechanism

The scheduling mechanism is very simple. As the cells arrive, they are stamped with their expected transmission deadlines. We compute the transmission deadline of i^{th} cell F_i of a session with a service curve $\langle \langle b_p, r_p \rangle \langle b_a, r_a \rangle \rangle$ as following,

$$F_i^a = max\{F_{i-1}^a, A_i - b_a/r_a\} + 1/r_a$$

$$F_i^p = max\{F_{i-1}^p, A_i - b_p/r_p\} + 1/r_p$$

$$F_i = max\{0, F_i^a, F_i^p\}$$

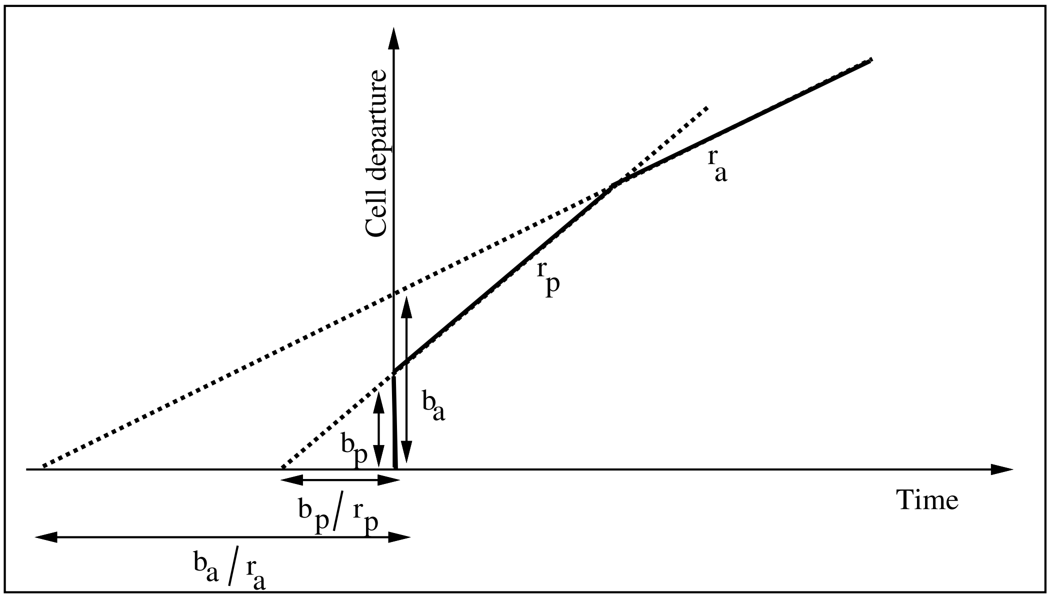

Figure 3: Service curve of a session.

In the expressions above F_i^a is the transmission deadline of the i^{th} packet of a session that follows a service curve with slope (or rate) r_a and an initial credit of b_a cells. We assume that A_i is the arrival time of the i^{th} packet and cell transmission time is the unit of time. Similarly, F_i^p is the transmission deadline of the i^{th} packet of a session that follows the service curve with rate r_p and an initial credit of b_p cells. We get F_i by taking the maximum of F_i^a and F_i^p. Note that system time is initialized to zero at system startup, and both F_0^p and F_0^a to $-\infty$.

The intuition behind the time stamping algorithm is very simple. It is a simple extension to the algorithms used in virtual clock, PGPS, and SFQ. In each one of these, a session is assigned a single rate of service, say r_a. As the cells from this session arrive, they are time stamped as follows,

$$F_i^a = max\{F_{i-1}^a, A_i\} + 1/r_a.$$

In the above expression, F_i^a is the expected finish time of the i^{th} cell of the session, and A_i is its arrival time. The virtual clock, PGPS, and SFQ differ in the way they assign the arrival time A_i. Now, if the service curve is a little complex than a simple rate curve and a has a bounded burst component b_a, like a leaky bucket, we can revise the time stamping algorithm as follows,

$$F_i^a = max\{F_{i-1}^a, A_i - b_a/r_a\} + 1/r_a.$$

The implication of having a burst component is that the session starts with an initial credit of b_a cells and can accumulate a credit of up to b_a cells during the periods of inactivity. This credit is reflected in the time stamp as the subtractive factor b_a/r_a. Note that $-b_a/r_a$ is the point where the service curve in figure 3 intersects the time axis. The service curve in our case is the composition (minimum) of two such segments, $\langle b_p, r_p\rangle$ and $\langle b_a, r_a\rangle$. Hence, the time stamp on the cell is the

Variables:

A_i: Arrival time of the ith cell of a session.

$\langle\langle b_p, r_p\rangle\langle b_a, r_a\rangle\rangle$: Service curve of a session.

F_i^p, F_i^a: Auxiliary variables used for time stamping.

Events:

Initialize:: /* Invoked at connection setup time. */

$F_0^p \leftarrow -\infty; F_0^a \leftarrow -\infty.$

Enqueue:: /* Invoked at cell arrival time. */

$F_i^p \leftarrow max(F_i^p, A_i - b_p/r_p) + 1/r_p.$

$F_i^a \leftarrow max(F_i^a, A_i - b_a/r_a) + 1/r_a.$

Add the cell to the queue with time stamp max(0,F_i^p, F_i^a).

Dequeue:: /* Invoked at the beginning of a busy period. */

while not end-of-busy-period **do**

If time stamp on the cell at the head of the dispatch queue is greater than current time, dispatch the cell.

end while

Figure 4: Multirate Scheduling.

maximum of F_i^a and F_i^p. A time stamp computed this way may be negative. We round up the negative time stamps to zero.

The cells from different sessions are placed in a single queue sorted in the increasing order of their time stamps. The cells are served from the head of the queue if and only if the time stamp on the cell at the head of the queue is less than or equal to current time. Since the packets are held in the queue until their scheduled departure time, each session maintains its original traffic envelope as it exits the switch. In figure 4 we formally describe the algorithm.

3.2 Properties of the Scheduling Algorithm

In order to analyze the properties of the scheduling algorithm, we introduce the concept of an universal utilization curve or UUC. Informally, the UUC is the superposition of normalized service curves of all sessions assuming that they start their busy periods at the same time. It is a convex and piece-wise linear function where each segment represents the aggregate rate of arrivals of all sessions over that period. We normalize the aggregate rate by the link speed. We represent the UUC

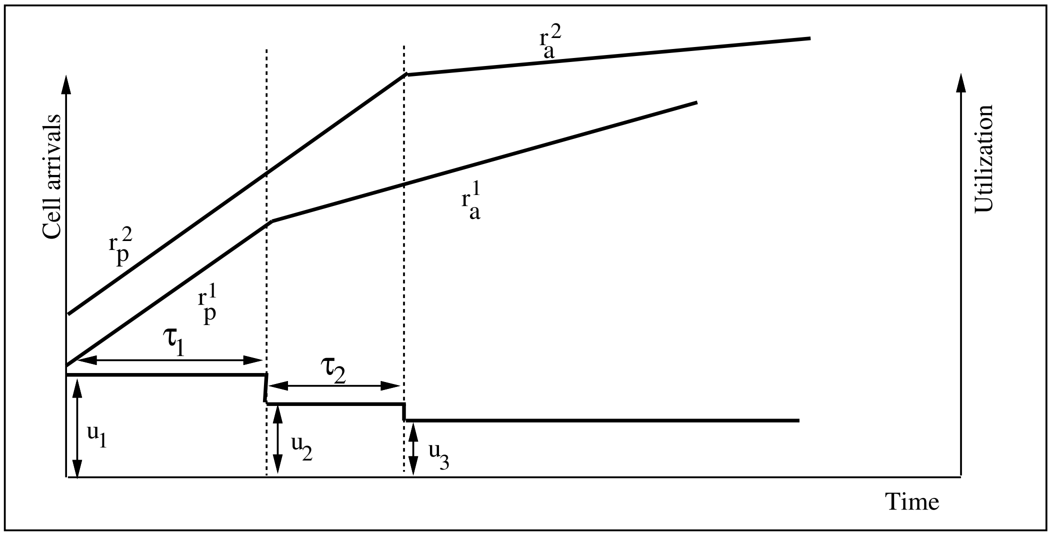

Figure 5: Universal utilization curve.

as a sequence of tuples $\langle u_i, \tau_i \rangle$, where u_i and τ_i are the utilization and length of the i^{th} segment of the UUC, respectively. The utilization of one or more segments of the UUC may exceed one. However, in order for the system to be stable, the utilization of the last segment has to be less than one. Figure 5 shows the UUC when two sessions $\langle\langle b_a^1, r_a^1\rangle\langle b_p^1, r_p^1\rangle\rangle$ and $\langle\langle b_a^2, r_a^2\rangle\langle b_p^2, r_p^2\rangle\rangle$ are active at a switching node. In this example, the UUC consists of three segments $\langle u_i, \tau_i \rangle$, where $i = 1, \ldots, 3$. We can compute τ_is from the time axis coordinates of the points of inflexions of the service curves. If L is the link speed, we can compute u_is as $u_1 = (r_p^1 + r_p^2)/L$, $u_2 = (r_a^1 + r_p^2)/L$, and $u_3 = (r_a^1 + r_a^2)/L$. In the following we present a few key results on buffer requirements and session delays.

Lemma 3.1 *Given the UUC $\equiv \langle u_i, \tau_i \rangle$, $i = 1, \ldots, M$, and a set of session $\langle\langle b_p^i, r_p^i\rangle\langle b_a^i, r_a^i\rangle\rangle$, $i = 1, \ldots, N$, the maximum backlog at a switching node is bounded by,*

$$B_{system} \leq \sum_{i=1}^{N} b_p^i + L \sum_{i=1}^{n} (u_i - 1)\tau_i,$$

where L is the link speed, and $u_n > 1$ and $u_{n+1} \leq 1$.

Proof: The worst case backlog occurs when all sessions start their busy periods at the same time. The term $\sum_{i=1}^{N} b_p^i$ on the right hand side of the expression represents the buffer required to absorb the initial bursts from all the sessions. The second term corresponds to the buildup due to the mismatch between the arrival rate and the service rate. Observe that the rate of arrival into the switch is greater than rate of departure from the switch when u_i is greater than one. Also notice (see figure 6) that the rate of accumulation at any point of time is $L(u_i - 1)$, where u_i is the utilization of the system at that point. Clearly, the summation on the right hand side enumerates to the total accumulation over the duration for which u_i remains above one. □

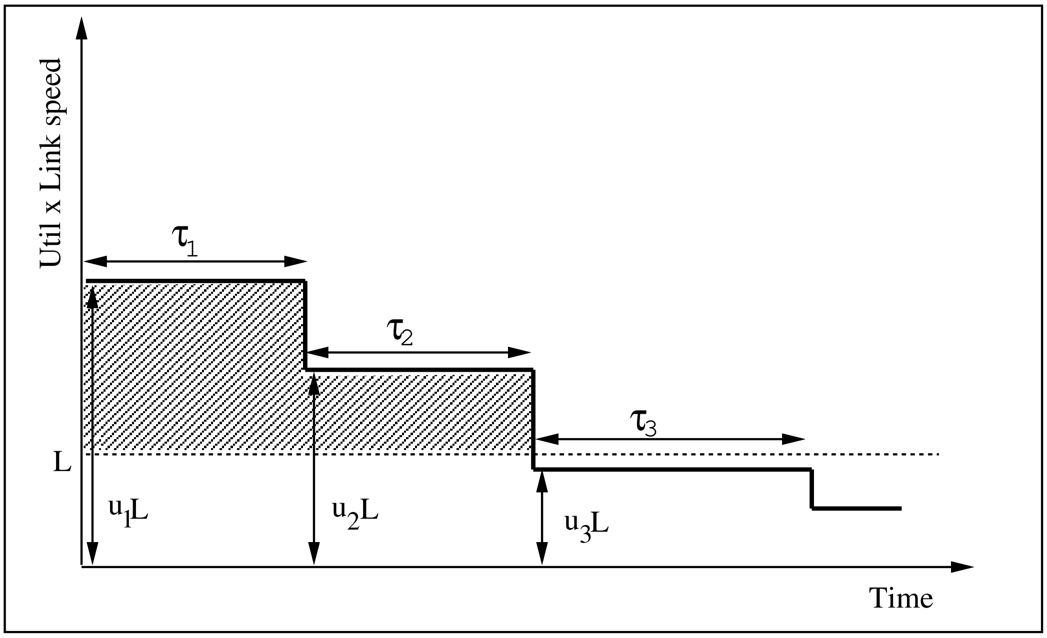

Figure 6: Buffer build up at the switch.

Lemma 3.2 *Given the UUC* $\equiv \langle u_i, \tau_i \rangle$, $i = 1, \ldots, M$, *the maximum backlog of a session characterized by* $\langle \langle b_p, r_p \rangle \langle b_a, r_a \rangle \rangle$ *cannot exceed,*

$$B_{session} \leq min\left(r_p \sum_{i=1}^{n} (1 - 1/u_i)\tau_i + b_p, r_a \sum_{i=1}^{n} (1 - 1/u_i)\tau_i + b_a \right)$$

where $u_n > 1$ *and* $u_{n+1} \leq 1$.

Proof: The backlog of a session increases as long as u_i stays over one. The session backlog reaches its highest point when the UUC changes its slope from $u_n > 1$ to $u_{n+1} \leq 1$. If the service curve of the session consisted only of the segment $\langle b_p, r_p \rangle$, the maximum session backlog would have been $r_p \sum_{i=1}^{n} (1 - 1/u_i)\tau_i + b_p$. If the service curve consisted only of the $\langle b_a, r_a \rangle$, the maximum session backlog would have been $r_a \sum_{i=1}^{n} (1 - 1/u_i)\tau_i + b_a$. Since the actual service curve is the minimum of these two segments, the maximum session backlog cannot exceed the minimum of the backlogs computed using each these segments in isolation. □

Lemma 3.3 *Given the UUC* $\equiv \langle u_i, \tau_i \rangle$, $i = 1, \ldots, M$, *the maximum delay suffered by any cell belonging to a session characterized by* $\langle \langle b_p, r_p \rangle \langle b_a, r_a \rangle \rangle$ *cannot exceed,*

$$D_{session} = min\left(\sum_{i=n+1}^{n+k} \tau_i + \delta_1, \sum_{i=n+1}^{n+l} \tau_i + \delta_2 \right)$$

where B *is the maximum session backlog,* $u_n > 1$ *and* $u_{n+1} \leq 1$, *and*

$$(b_p + B)/r_p = \sum_{i=n+1}^{n+k} (1 - 1/u_i)\tau_i + (1 - 1/u_{n+k+1})\delta_1, \qquad \delta_1 \leq \tau_{n+k+1}$$

$$(b_a + B)/r_a = \sum_{i=n+1}^{n+l} (1 - 1/u_i)\tau_i + (1 - 1/u_{n+l+1})\delta_2, \qquad \delta_2 \leq \tau_{n+l+1}$$

Proof: We compute the worst case bound on delay by enumerating the time taken to clear the worst possible backlog. We compute the clearing times in two cases: (a) using service segment $\langle b_p, r_p \rangle$, and (b) using service segment $\langle b_a, r_a \rangle$. The worst case delay is the minimum of the two. □

One of the interesting points to note here is that the aggregate rate of arrivals can exceed the link capacity for a finite length of time. The buffers in the switch are used to absorb this mismatch between the arrival and the service rates. This is an important difference between the multirate service discipline proposed and the ones that offer a single rate guarantee, such as virtual clock, PGPS, and SFQ. In PGPS for example, input traffic is shaped using a single leaky bucket and is characterized by a tuple $\langle b, r \rangle$, where b is the burst size and r is the arrival rate. In order to achieve stability, PGPS keeps the aggregate arrival rate from all sessions at all times below the link speed. Another point to note here is that the scheduler does not introduce any delay if the UUC is always less than one. Also note that the scheduler maintains the original traffic envelope of a session. Hence, we can compute the end-to-end delay of a session using lemma 3.3 if we know the UUCs of all the switching nodes on the path of the connection.

3.3 Admission Control

We discuss admission control policies for two different classes of service: (a) guaranteed service, and (b) predictive service. In the guaranteed service each session is guaranteed a certain target delay and lossless delivery of data. In the predictive service, designed for adaptive applications prepared to tolerate occasional cell losses and delay violations, a session is promised a delay target with the understanding that it may be violated at times and there may be cell losses once in a while. In the following we discuss the admission control algorithms for each of these classes of service.

Guaranteed Service

We assume that each session is characterized by $\langle \langle b_p, r_p, \rangle \langle b_a, r_a \rangle$. The process of admitting a new connection includes three steps: (a) computing the new UUC, (b) checking that the worst case system backlog does not exceed switch buffer limit , and (c) checking that the delay bound for each connection is satisfied. In the following we discuss each of these steps in detail.

Recomputing the UUC is rather simple. It can be done incrementally. Note that the new UUC has at most one more point of discontinuity which coincides with the point of inflexion of the service curve of the new connection. Also, the value of u_i for each segment of the curve goes up by r_p/L

or r_a/L depending on whether the segment is on the left or on the right of the point of inflexion introduced by the new connection.

Using the result from lemma 3.1 it is quite simple to check if the admission of the new connection can lead to buffer overflow. As we update the UUC, we can add up $(1 - 1/u_i)\tau_i$ for each segment until u_i changes from values higher than one to less than one. Once this sum is known, checking for the buffer overflow is trivial. For the purpose of quick and easy computation, it may be worthwhile to keep the cumulative sum along with $\langle u_i, \tau_i \rangle$ for each segment of the UUC. We can also compute the maximum backlog of each session in the same pass. These results are prerequisite to delay computation for individual sessions.

We can compute the session delays using the results from lemma 3.3. For quick computation, we can store the cumulative sum of $(1/u_i - 1)\tau_i$ starting from the point where u_i changes from values greater than one to less than one. Note that delay bounds of all the sessions can be computed in one pass, and its of $O(N)$ complexity, where N is the number of connections.

Predictive Service

In the predictive service, a session is given a loose guarantee on delay and loss. We exploit this laxity to improve network utilization and expedite admission control checks. In predictive service the utilization of the system is measured rather than computed. When a new flow is to be admitted, delay and buffer occupancy are estimated based on the measured utilization. In the following we explain the procedure in detail.

We measure the utilization of the system over different time scales. Let us assume that the measurement is taken over three time scales T_1, T_2, and T_3, where $T_3 > T_2 > T_1$. The measurement process is very simple. We count the number of arrivals C_1, C_2, and C_3 over a period of T_1, T_2, and T_3, respectively. We compute the utilization of the system in T_1, T_2, and T_3 as $u_1 = C_1/T_1$, $u_2 = C_2/T_2$, and $u_3 = C_3/T_3$, respectively. From these measurements we can estimate the UUC as $\langle u_1, \tau_1 \rangle$, $\langle u_2, \tau_2 \rangle$, $\langle u_3, \infty \rangle$, where $\tau_1 = T_1$, and $\tau_2 = T_2 - T_1$. The measured values of u_1, u_2, and u_3 are updated every T unit of time, where $T \geq T_3 > T_2 > T_1$. To be on the conservative side, we update u_is with the highest u_i recorded in the last measurement period T.

The admission control test for a new connection, characterized by $\langle \langle b_p, r_p \rangle \langle b_a, r_a \rangle \rangle$, consists of the following steps:

- Estimate the new UUC of the system after the admittance of the new flow from the current UUC and the service curve of the new connection. If the utilization of the last segment of the updated UUC is greater than one, the connection is rejected right away.

- If any segment of the UUC is greater than one, we estimate system buffer occupancy B_e using lemma 3.1. If $B_e > \alpha B$, where α is a multiplicative factor less than one, and B is the switch buffer size, the connection is rejected.

- Delay estimates for each connection, including the new one, are recomputed using the modified UUC and the original traffic specification of each connection. If the delay estimate of any connection exceeds the promised/requested maximum bound, the connection is rejected.

Estimation of system utilization is a very important component of the admission control algorithm. The number and the lengths of the measurement periods determine the accuracy of the measurement, and consequently impacts the size of the admissible region and discrepancy between the estimated and actual delay and loss characteristics. The value of α also has a significant impact on the system performance. The higher is the value (less than one) more optimistic is the admission control process. However, a higher value of α also increases the risk of buffer overflows and cell losses. On the other hand, a conservative choice of α reduces the chance of cells losses, but only at the cost of lower utilization.

4 Numerical Results

In this section we compare the performance of multirate scheduling and PGPS in providing guaranteed service. We investigate multiplexing gains of predictive service over that of guaranteed service. We also present numerical results comparing multirate and single rate predictive services.

We used four 320×240 video clips 1, each approximately 10 minutes long in our study. In order to understand the effects of traffic variability on the performance of multirate service, we selected videos with different degrees of scene changes. The first video is an excerpt from a very fast scene changing basketball game. The second clip is a music video (MTV) of the rock group REM. It is composed of rapidly changing scenes in tune with the song. The third sequence is a clip from CNN Headline news where the scene alternates between the anchor reading news and different news clips. The last one is a lecture video with scenes alternating between the speaker talking and the viewgraphs. The only moving objects here are the speaker's head and hands. Figure 7 plots frame sizes against frame number (equivalently time) for all four sequences for an appreciation of the burstiness in different sequences. In all traces, frames are sequenced as IBBPBB and frame rate is 30 frames/sec. Observe that, in terms of the size of GoP [2] and that of an average frame, basket ball and lecture video are at the two extremes (the largest and the smallest, respectively), with the other two videos in between.

Guaranteed Service: For this study, we consider a network consisting of nodes in tandem, connected by OC-1 (51 Mbps) links and transporting fixed-size ATM cells. Data from the source is passed through a shaper and then fed to the network. All connections traverse from the source to the sink through five switches. We assume that the end-to-end delay bound is 300 ms. The following set of graphs (figures 8, 9, 10, 11) compare the number of connections admitted when PGPS is used in conjunction with a leaky bucket shaper and multirate service is used in conjunction with a dual leaky bucket shaper. We use results from section 2 to compute the leaky bucket parameters that maximizes the number of connections admitted by PGPS. The choice of shaper parameters for multirate service is bit more ad hoc [3]. We pick $b_p = 0$ and use the results from section 2 to find the minimum r_p that guarantees loss-less shaping. We use two different values for b_a. The first plot for each set (for a particular sequence) uses $b_a = 1000$ cells and the second plot uses $b_a = 3000$ cells. In

[2] The repeating sequence (IBBPBB in this case) is called a GoP or Group of Pictures.

[3] This is because we do not have a closed from delay bound for multirate service and hence we cannot use the linear programming formulation to find the optimal parameters.

Traces	Type of Frame	Maximum Frame Size	Minimum Frame Size	Average Frame Size	Variation (Std. Dev)
Basketball	I	41912	8640	26369.14	4672.12
	P	40128	6400	15570.31	1846.77
	B	35648	4288	11137.18	2856.51
	Avg. Frame			14414.69	
	GoP	215232	59008	86488.12	6284.27
MTV Video	I	34496	8512	21770.88	4808.31
	P	28544	9152	15833.20	1437.66
	B	32640	4608	12211.39	3123.87
	Avg. Frame			14408.27	
	GoP	146304	65472	86443.91	4017.07
News Clip	I	43200	17144	27728.71	3537.86
	P	20160	11392	16025.23	547.62
	B	26304	6208	10643.04	2691.66
	Avg. Frame			14387.68	
	GoP	118464	81664	86326.10	839.50
Lecture	I	13504	5312	10956.67	1498.87
	P	13312	2048	4673.82	642.53
	B	6592	768	3292.95	617.45
	Avg. Frame			4800.38	
	GoP	33984	23424	28802.33	555.95

Table 1: Characteristics of the MPEG traces. Size is in bytes and frame sequence is IBBPBB.

each case, we find the corresponding lowest r_a that guarantees loss-less shaping. We have plotted the percentage improvement in the number of connections admitted using multirate service over that of PGPS for different switch and shaper buffer sizes.

For all sequences, the smaller is the shaper buffer the larger is the improvement. A smaller buffer results in a burstier stream coming out of the shaper and entering the network. The more bursty is a stream the more beneficial it is to use multirate service. Also observe that for all sequences, as the size of the switch buffer increases, the improvement due to using multirate service also increases. For the basketball, MTV, and CNN clips the curve reaches a peak and then declines as the switch buffer size increases further. For the lecture video, it keeps increasing with the size of the switch buffer. However, we have examined that it also reaches its peak at a certain buffer size (larger than the range shown in the plot) and then goes down. This behavior is also due to the fact that multirate service is more effective for bursty traffic. Initially with increase in switch buffer size the benefits of using multirate service increases since it effectively uses this buffer to multiplex the peaks and the lulls of different sessions. During the same time, the best leaky bucket parameters for PGPS consists of a small burst and a large token rate. The number of connections admitted by PGPS is computed as the ratio of the link speed and the token rate, and a high token rate results in small number of connections being admitted. Therefore, we observe a very sharp increase in the percentage gain.

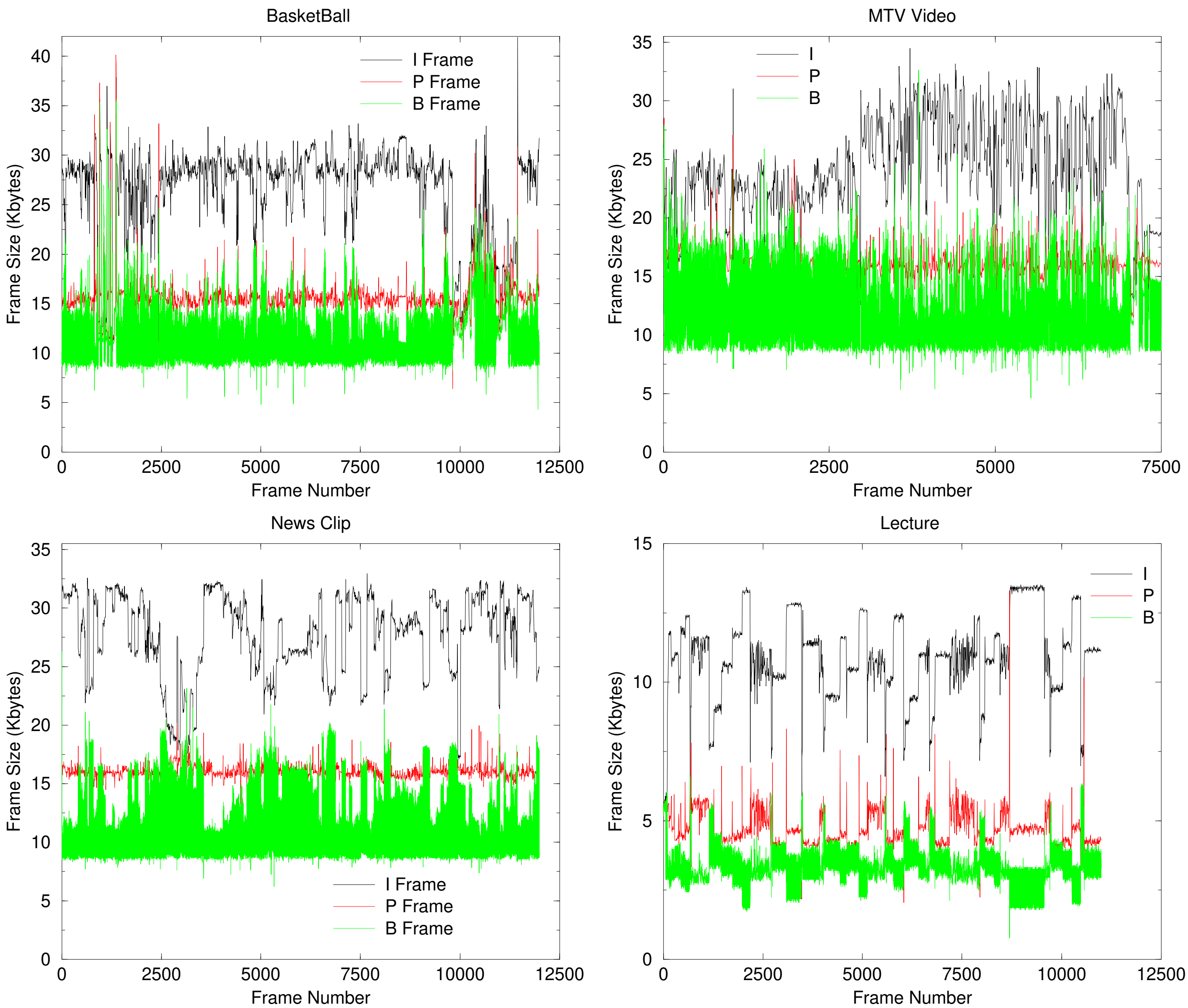

Figure 7: MPEG compressed video traces. Frame sequence is IBBPBB.

However, if the buffer size is increased even further, the advantage becomes less effective since PGPS uses this extra buffer to choose leaky bucket parameters consisting of larger bursts but smaller token rates. Also note that with a higher value of b_a, the peak shifts towards the right. A higher b_a signifies a longer peak rate segment (for multirate service).

If we analyze the results for different sequences, we observe that the improvement due to multirate service is the lowest for the lecture video. This can be explained from the fact that it is the least bursty of all the clips. Here, even with small amount of shaper buffer the difference between the peak

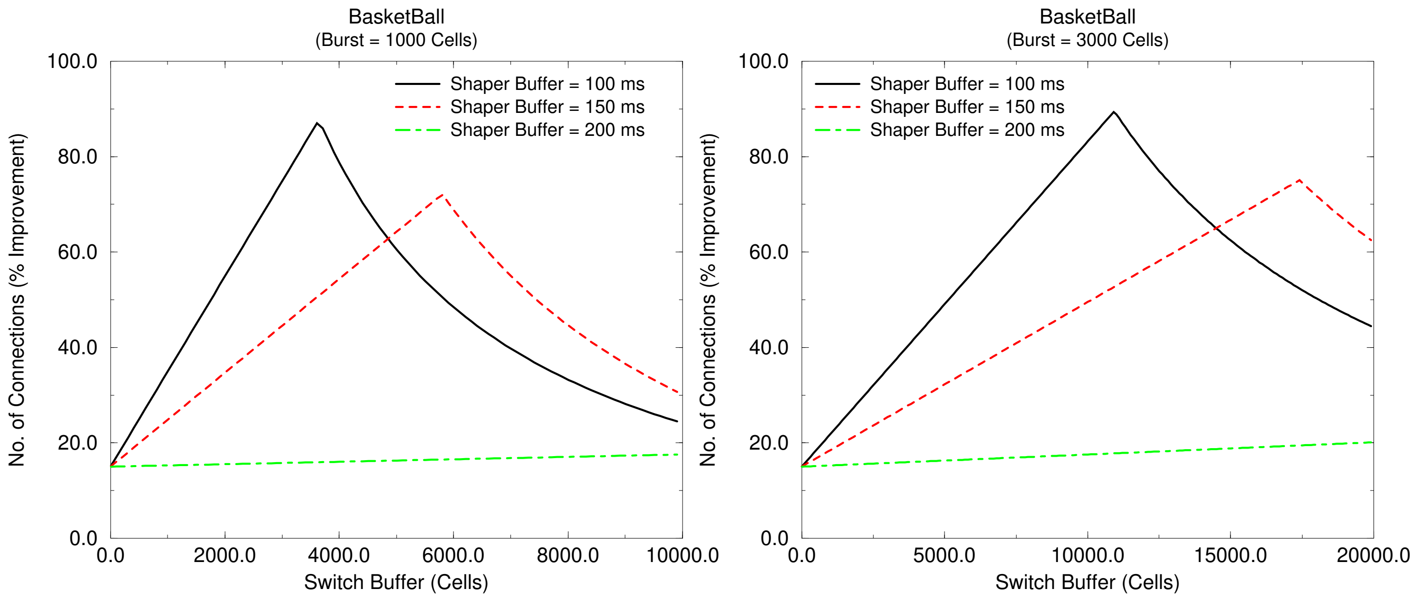

Figure 8: Percentage improvement in number of connections admitted for the Basketball trace.

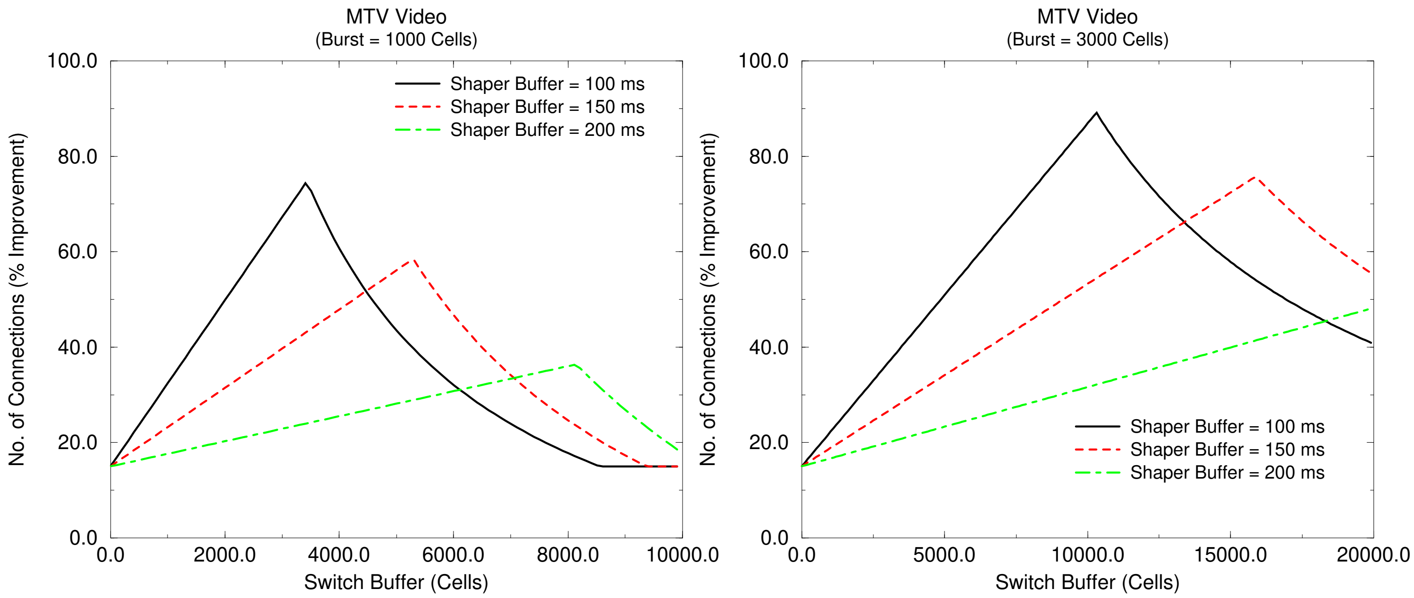

Figure 9: Percentage improvement in number of connections admitted for the MTV trace.

and the average rate is not as much as the other more bursty clips. Consequently, the multiplexing gain is also less.

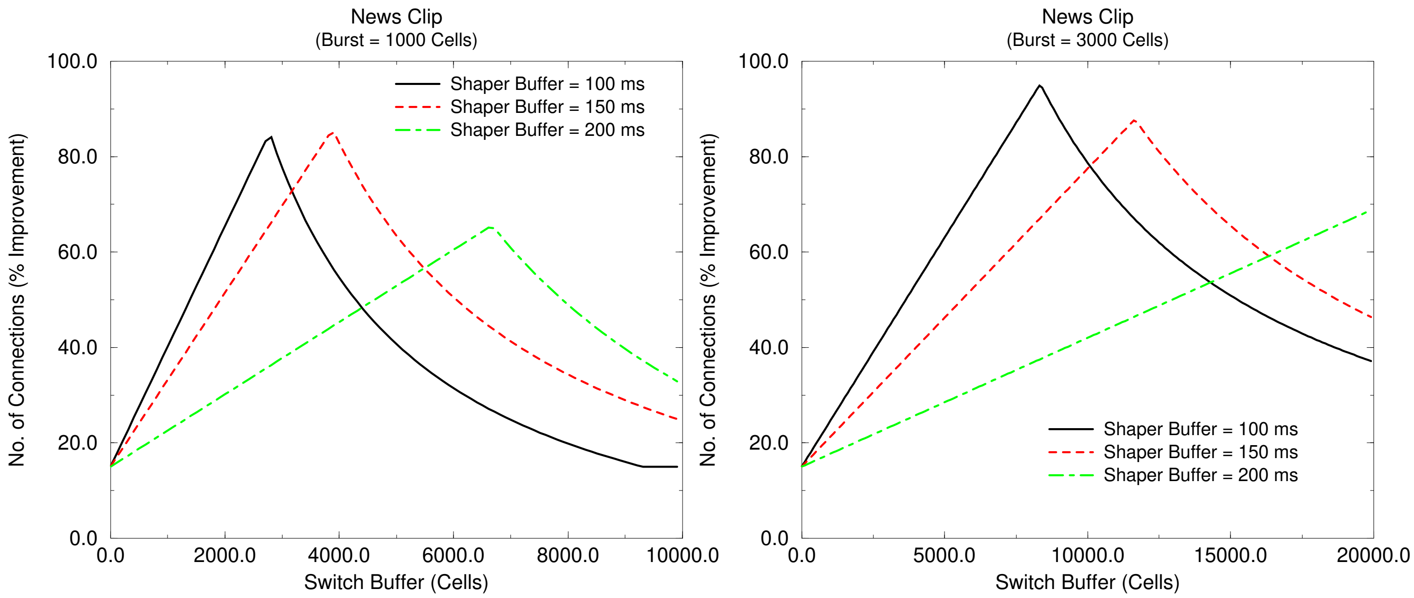

Figure 10: Percentage improvement in number of connections admitted for the News clip.

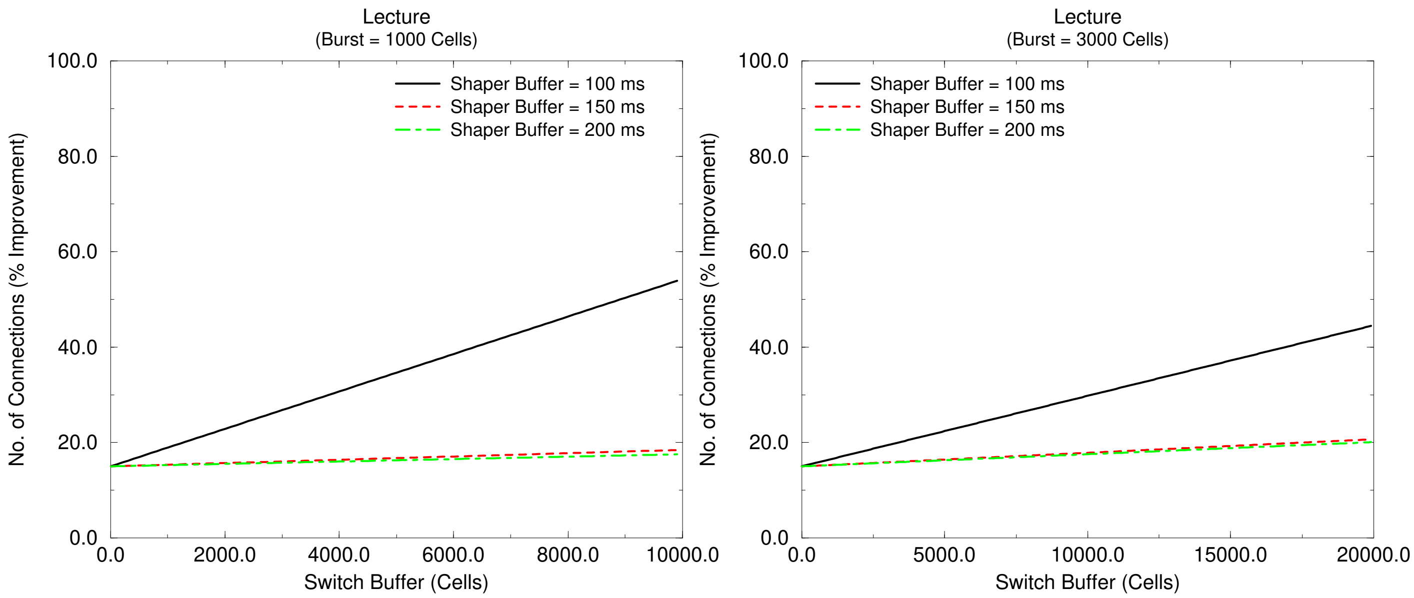

Figure 11: Percentage improvement in number of connections admitted for the Lecture Video.

Predictive Service: For this study, we consider the same network configuration used for the guaranteed case. We assume that desired end-to-end delay is 300ms. The following graphs compare the number of connections admitted for guaranteed and predictive services, both using multirate

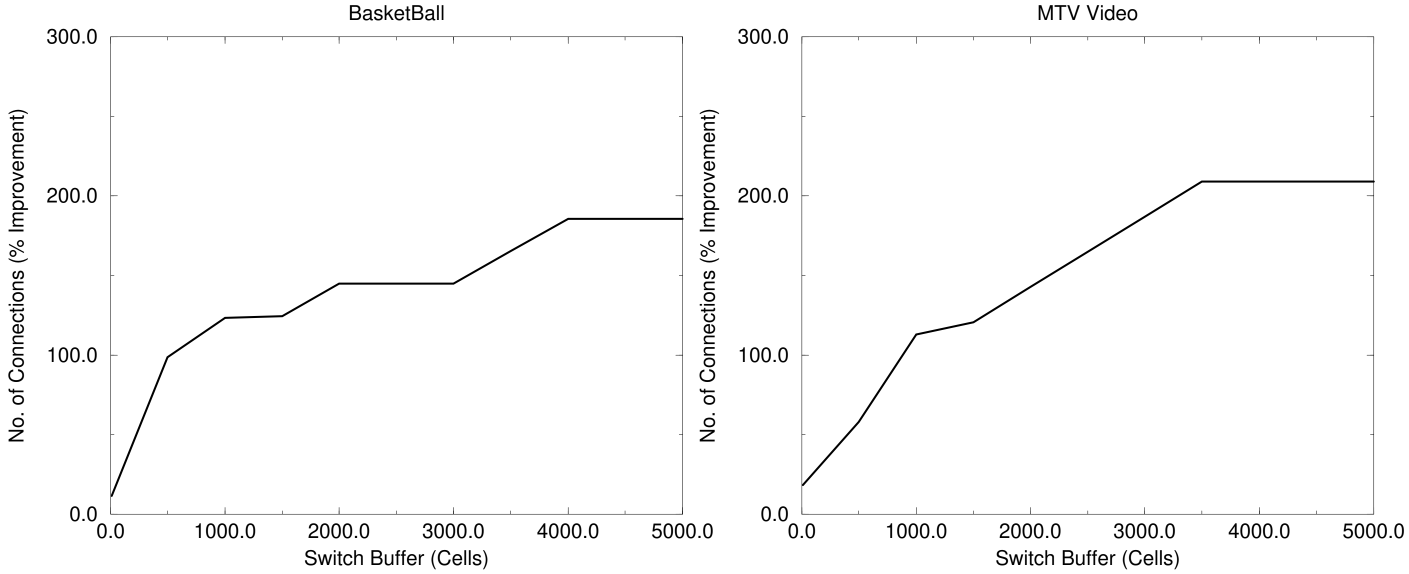

Figure 12: Percentage improvement in number of connections admitted, predictive vs. guaranteed service.

scheduling. Note that while we compute the number of connections admitted in the guaranteed case, number of connections admitted in the predictive case is obtained through simulation. To make the comparison fair, we use results from only those simulation runs that result in zero cell loss. At each switching node we estimate the UUC by measuring utilization over three time scales, $T_1 = 50$ cell time, $T_2 = 250$ cell time, and $T_3 = 500$ cell time. The measurement period is $T = 1000$ cell time.

We vary the buffer utilization factor α to achieve loss-less delivery.

In figure 12 we have plotted the percentage improvement in number of connections admitted in the predictive service over that of guaranteed service. We observe that for all video sequences the improvement in number of connections admitted declines initially with the increase in the switch buffer size. For basketball, MTV, and the News clips the increase in improvement ceases at a certain switch buffer size. For the lecture sequence it actually decreases with increasing buffer size. The initial increase in improvement with increase in the switch buffer size is because of the fact that predictive service effectively uses this buffer to multiplex traffic. The rate of increase is higher for lower switch buffer due to the fact that during this time, the multiplexing gain is hardly exploited by the deterministic scheme. However, as buffer size keeps increasing so does the number of connections and ultimately the multiplexing gain tapers off. At this point, the number of connections admitted in both guaranteed and predictive services become (more or less) constant (at different values of course). The decrease in the percentage improvement for the lecture sequence comes as a bit of surprise. This is because of the fact the lecture video is quite smooth to start with. Therefore, the as the buffer in the switch increases, the deterministic policy can do much better in exploiting multiplexing gain than the other bursty clips. With the increase in the buffer size, the multiplexing gain achieved by both the schemes becomes comparable and the gain starts to fall (note that the absolute number of connections admitted by the predictive scheme is more than a factor of two higher than that of the the deterministic scheme at all times).

The next set of graphs (see figure 13) compare the number of connections admitted using multirate and single rate predictive service. We used the same experimental setup and the same set of traffic parameters as used in the comparative study of multirate scheduling and PGPS for guaranteed service. The only difference is in the admission control algorithm. We used the measurement based admission control algorithm with $T_1 = 50$ cell time, $T_2 = 250$ cell time, and $T_3 = 500$ cell time. The measurement period is $T = 1000$ cell time. We vary the buffer utilization factor α to achieve loss-less delivery. We plot the percentage improvement in number of connections admitted using multirate service over that of single rate service. When switch buffer size is in the range of $100 - 500$ cells we get 100% — 200% improvement for all sequences. However with increasing buffer size the improvement due to multirate service goes down and stabilizes at around 50%.

From the results presented in this section we make the following conclusions:

- For guaranteed service multirate scheduling outperforms PGPS. Multirate scheduling is more effective for bursty traffic.

- The benefits of using multirate scheduling goes down as the buffering in the switch increases. Note however that most of the currently available switches use no more than couple of hundred cells of buffer per port. Due to space and power limitations on the switch wafer this number is not expected to increase dramatically in the near future. In this operating range, the benefits of multirate scheduling is very significant.

- If hard guarantees on service quality is not a requirement, predictive service can be used to improve system utilization significantly. For typical switches currently available, multirate scheduling far outperforms single rate scheduling.

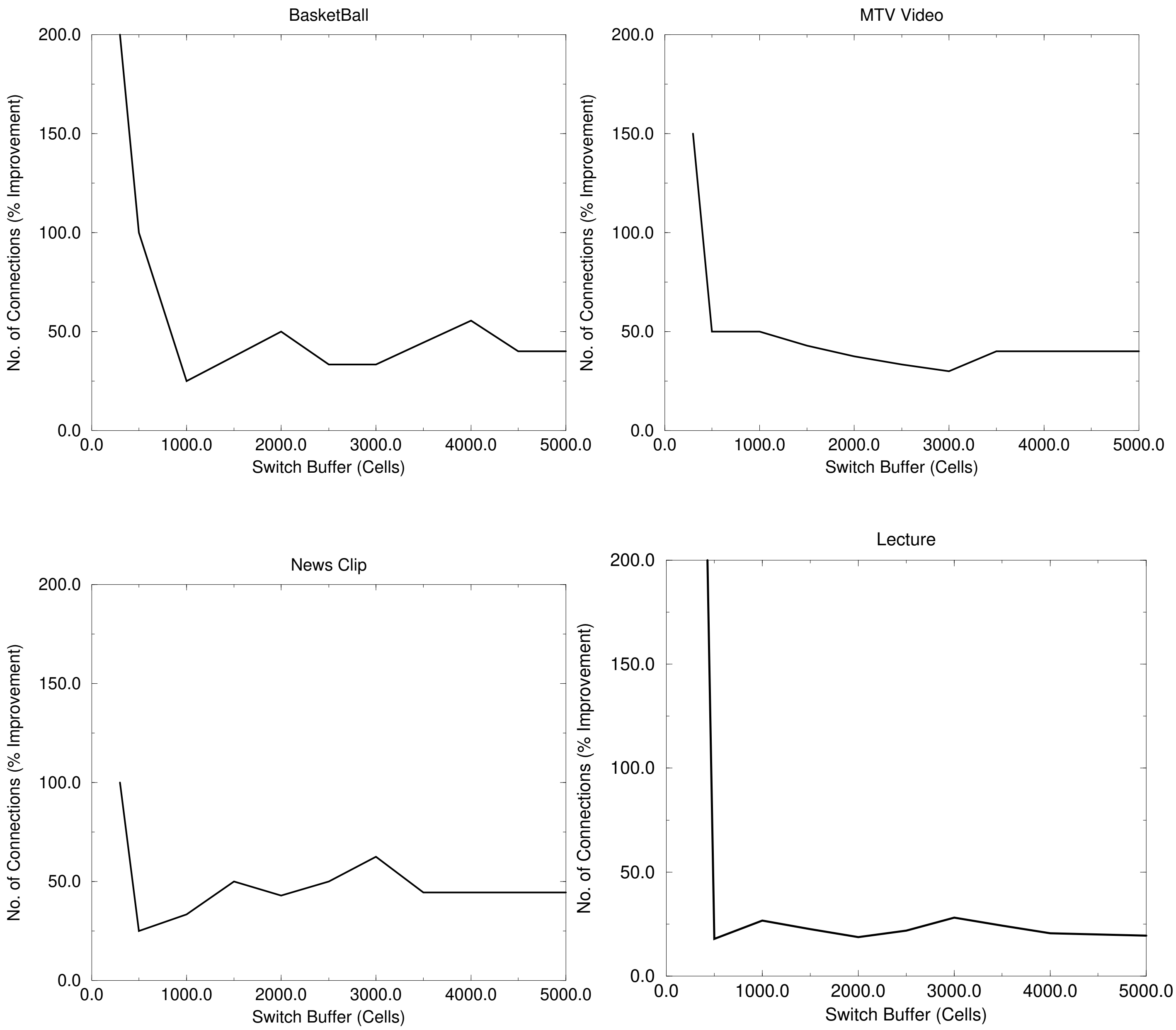

Figure 13: Percentage improvement in number of connections admitted using multirate scheduling for predictive service.

5 Conclusion

We have proposed a multirate service mechanism for ATM networks that is particularly suitable for transporting VBR video traffic. We have shown how this multiplexing mechanism can be very effective in providing both deterministic and empirical service guarantees. The major contributions of this paper are:

- Propose a scheduling mechanism that can provide different rates of service at different times. It is particularly suitable for bursty traffic, such as VBR video.
- We have shown that multirate scheduling can outperform PGPS in terms of number connections admitted while maintaining the same level of service guarantees. Our experiments with video traces demonstrate that multirate scheduling can be very effective for VBR video traffic.
- We have shown that by relaxing the service guarantees and using a measurement based admission control process we can improve system utilization significantly. Our experiments using real life video traces demonstrate that we can achieve a factor of two improvement in utilization with almost no degradation in service quality.

This study can be extended in many ways. In the predictive mode of service, the number and the lengths of measurement intervals have a profound impact on the stability and the utilization of the system. We are currently in the process of analyzing these interactions. Another interesting direction for future work is to apply multirate scheduling for data traffic. For example, it can be used to provide short response times to telnet sessions by reserving a high peak rate for a short duration and low average rate. The ftp sessions on the other hand can be optimized for throughput by reserving a relatively higher average rate of service, and a peak rate that is same as or close to the average rate.

References

[1] C. M. Aras, J. F. Kurose, D. S. Reeves, and H. Schulzrinne. Real-Time Communication in Packet-Switched Networks. *IEEE Transsactions on Information Theory*, 82(1), January 1994.

[2] S. Keshav C. R. Kalmanek, H. Kanakia. Rate Controlled Servers for Very High Speed Networks. In *Proceedings of GLOBECOM*, December 1990.

[3] L. Georgiadis, R. Guerin, and A. Parekh. Optimal Multiplexing on Single Link: Delay and Buffer Requirements. In *Proceedings of INFOCOM*, June 1994.

[4] L. Georgiadis, R. Guerin, V. Peris, and K.N. Sivarajan. Efficient Network QoS Provisioning Based on per Node Traffic Shaping. In *Proceedings of INFOCOM*, March 1996.

[5] S. J. Golestani. A Framing Strategy for Congestion Management. *IEEE Journal on Selected Areas of Comminication*, 9(7), September 1991.

[6] S.J. Golestani. A Self-Clocked Fair Queuing Scheme for Broadband Applications. In *Proceedings of INFOCOM*, June 1994.

[7] S. Jamin, P. Danzig, S. Shenker, and L. Zhang. A Measurement-based Admission Control Algorithm for Integrated Services Packet Networks. In *Proceedings of SIGCOMM*, October 1995.

[8] E. W. Knightly. H-BIND: A New Approach to Providing Statistical Performance Guarantees to VBR Traffic. In *Proceedings of INFOCOM*, March 1996.

[9] E. W. Knightly and H. Zhang. Traffic Charcterization and Switch Utilization using a Deterministic Bounding Interval Dependent Traffic Model. In *Proceedings of INFOCOM*, April 1995.

[10] D. Legall. MPEG - A Video Compression Standard for Multimedia Applications. *Communications of the ACM*, 34(4), 1991.

[11] B. L. Mark and G. Ramamurthy. Real-time Estimation of UPC Parameters for Arbitrary Traffic Sources in ATM Networks. In *Proceedings of INFOCOMM*, March 1996.

[12] A. K. Parekh and R. G. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Network: The Single Node Case. *IEEE/ACM Transactions on Networking*, 1(3), June 1993.

[13] E. Rathgeb. Modeling and Performance Comparison of Policing Mechanisms for ATM Networks. *IEEE Journal on Selected Areas of Communication*, 9(3), 1991.

[14] D. Reininger, D. Raychaudhuri, B. Melamed, B. Sengupta, and J. Hill. Statistical Multiplexing of VBR MPEG Compressed Video on ATM Networks. In *Proceedings of INFOCOM*, 1993.

[15] J.L. Rexford, A.G. Greenberg, and F.G. Bonomi. Hardware-Efficient Fair Queueing Architecture for High-Speed Networks. In *Proceedings of INFOCOM*, March 1996.

[16] D. Saha, S. Mukherjee, and S.K. Tripathi. Carry-Over Round Robin: A Simple Cell Scheduling Mechanism for ATM Networks. In *Proceedings of INFOCOM*, March 1996.

[17] S. Suri, G. Varghese, and G. P. Chandranmenon. Leap Forward Virtual Clock: An O(log log N) Fair Queuing Scheme with Guaranteed Delays and Throughput Fairness. Technical report, Department of Computer Science, Washington University at St. Louis, 1996.

[18] H. Zhang and D. Ferrari. Rate Controlled Static Priority Queuing. In *Proceedings of INFOCOM*, 1993.

[19] L. Zhang. Virtual Clock: A New Traffic Control Algorith for Packet Switching Networks. In *Proceedings of SIGCOMM*, 1990.