# On Guaranteed Bandwidth Channels *

Shailender Chaudhry Mohammed Raziuddin and Alok Choudhary
{ schaudhr, razi, choudhar }@cat.syr.edu
Syracuse University

## Abstract

This paper introduces a new scheme and design of a protocol for guaranteed bandwidth channels. Given traffic characteristics of media streams, it is possible to bound delays experienced through each node in the network, using appropriate discretion at channel establishment time. Once a channel with such bounded delays is established, our scheme uses feedback techniques to match the flow of media units at each node with that of the playback rate at the client site. The feedback is triggered by preset marks in the buffers at each node called water marks. The receipt of a feedback by a node changes the rate of flow at that node, thus matching the rate of playback at the client. Jitter is implicitly controlled in such a scheme, as buffering absorbs the different delays experienced by media units. The scheme also provides continuity of playback for multimedia streams, removing this responsibility from the higher software layers at the client. We present analytical proofs that our scheme provides guaranteed bandwidth and prove the protocol safe. Further, simulation results are provided to validate the protocol.

## 1 Introduction

With the advent of high-speed networks it has become possible to support multimedia applications in a distributed environment. In fact, one of the immediate utilization of ATM networks has been envisioned as support of multimedia communication. Thus far, researchers [16, 6, 18] have proposed network channels(virtual connections) that offer *hard real-time* guarantees for delivery of packets to build distributed multimedia applications.

Multimedia data is isochronous in nature [16]. Thus, to convey the information held in a media stream, the media units of the stream must be presented in time continuum. This is unlike textual data, for which spatial continuity is sufficient to convey meaning. Therefore, network channels that offer *real-time* guarantees may be used to display data in a time continuous manner if input to the channel matches the playback characteristics. Such models abstract the network from the larger multimedia application, making continuity of display the responsibility of the higher layers at the client site.

We propose to increase efficiency by bringing the knowledge of the isochronous nature of multimedia data to the lower network layers. Higher layers are suited for synchronization, but ensuring continuity of playback can be done while regulating the channel(virtual connection) traffic. The burden of this knowledge is that at each hop in the network(VC), the buffer can not be empty or over full. This buffer occupancy criterion constitutes a necessary and sufficient condition for uninterrupted and continuous playback. The criterion is necessary in that an empty (over full) buffer would result in interrupted (skipped) playback. It is sufficient since it ensures that data is always available for display. Our approach increases efficiency by avoiding scheduler saturation through the absence of *real-time* guarantees. Furthermore, it achieves greater efficiency without adding excessive complexity to the lower network layers, since the additional burden consists solely of buffer management.

Finally, a scheme for flow control via feedback has already been proposed by Kung *et. al.* in [12]. The incorporation of our proposed scheme would merely require the extension of the language of the feedback.

The paper is organized as follows: First, motivation for including the continuity in the lower layers is given. This is followed by an overview of the abstraction we call buffer managed guaranteed bandwidth channels. The main results of this paper are presented in Sections 4, 5. In Section 4 we present a buffer status update protocol and a finite state machine model of the protocol. It is proved *safe* (deadlock free and free from unspecified receptions). In Section 5 we analyze the properties of the proposed channels and prove support for continuity. Further, we show how priorities may be used to guarantee different QOS for different channels. In Subsection 5.1 we explore the

*This work is supported by a NSF Young Investigator Award CCR-9357840 and Intel Corporation.



0-8186-7216-1/95 $04.00 © 1995 IEEE

gains due to the QOS parameter percentage dropped and present simulation results to validate the increase in efficiency.

## 2 Guaranteed Bandwidth Channels

An ideal guaranteed bandwidth channel is a virtual connection such that the connection behaves as a link operating at a predetermined bandwidth, irrespective of other connections that maybe sharing links. Implementation of such channels involves using admission control to reserve and check availability of resources such as bandwidth and buffer space.

The traditional approach towards implementing such channels offers delivery deadlines based on given input traffic parameters [6, 18, 15]. For example, if input traffic parameters are given as 1 packet every 10 milliseconds of size 50K bits, and delivery deadline is guaranteed at 5 milliseconds, then the virtual connection appears as a guaranteed bandwidth channel of 10M bits/second from the input perspective. The two immediate drawbacks of this scheme are scheduling saturation and complexity of traffic reshaping. At nodes that are shared by two or more guaranteed bandwidth channels, arrival patterns may be such that deadlines of all packets can not be simultaneously met, despite the existence of sufficient bandwidth. Thus, a scheduling saturation test must be performed before a new guaranteed bandwidth channel can be established [6, 18]. Further, local deadlines (at each node in the connection) are established based upon input traffic criterion to a node. Since the only criterion known is based on global input traffic parameters, the test to check whether local deadlines can be met is also based upon it. Therefore, due to jitter, traffic needs to be reshaped at each node to conform to declared parameters. This adds complexity and overhead to the delivery of packets.

An alternative interpretation of the bandwidth experienced by a channel is to measure it in terms of the outflow from the connection. Given output traffic characteristics (rate at which packets are needed at the client – same as input traffic parameters in the previous approach), the channel can be seen as experiencing guaranteed bandwidth if we guarantee that the client buffer will never underflow. As we will demonstrate in later sections, our scheme will avoid both scheduling saturation and the need to reshape traffic, if link-by-link per VC flow modification is used.

Guaranteed bandwidth channels, when implemented according to our scheme, offer the advantage of higher utilization and reduced overhead. Multimedia connections, which require guaranteed bandwidth channels, can be assured of continuous playback, since the client buffer is never allowed to underflow (or overflow). This offers motivation for including continuity in the lower network layers rather than the higher layers at the client site.

## 3 Buffer Managed Guaranteed Bandwidth Channels

To ensure that the client buffer never underflows (or overflows), we must match the inflow rate with that of the outflow rate. This is essentially a feedback control problem [11]. Our scheme has been adapted from the water model in control theory, where the water level in a pool is controlled by feeding back the height of water to the controller to modify the input. We maintain a buffer at each node in order to reduce response time. Adequate reaction to prevent underflow is ensured by our admission control, which in turn ensures sufficient bandwidth.

In order to minimize the number of media units buffered along the channel without increasing the number of feedbacks, we match the input rate with the output rate of media units in the buffer at each switch. We now present the implementation of our scheme.

### 3.1 Implementation

At each switch, we maintain separate queues called *regulator queues* [1] for each stream, as shown in Figure 1. The function of the regulator queue is to supply media units of the stream to the switch queue at a rate which is communicated to it by an algorithm. All media units of the stream are thus fed into the regulator to be consequently fed into the switch queue.

All media units are put onto the link via a *switch queue*. Thus the switch queue is shared by all streams using the same link. The service discipline of the switch queue is First Come First Served (FCFS). The service discipline may be changed if different QOS is required for different channels/connections.

Further, marks pertaining to the buffer level are set in the regulator queues to facilitate communication of the state of the buffer to the switch before it in the path from the server to the client. **The High Water Mark** indicates impending buffer overflow at the current outgoing and incoming rates of media units to the buffer. **The Low Water Mark** indicates impending buffer underflow at the current outgoing and incoming rates of media units to the buffer. **The Optimum Water Mark** is introduced to indicate a return to nominal buffering after touching either of the previous water marks.

[1] Buffer and regulator queue is used interchangeably in the rest of the paper

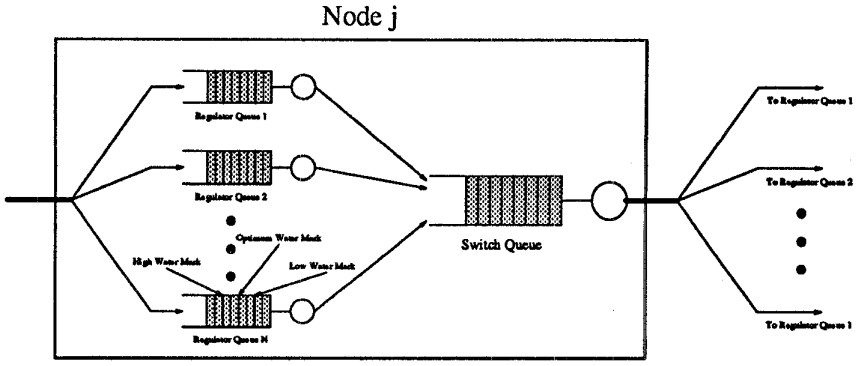


Figure 1: Node Architecture

For any given switch, the switch immediately downstream may be considered as a client for this switch, as its behavior is identical to that of the client, i.e. in terms of the consumption of media units supplied by it. Also, for any switch, the switch immediately upstream may be considered as a server for this switch, as its behavior is identical to that of the server, i.e. the supply of media units for consumption by it.

Consider a stream $i$ passing through a switch $j$. Furthermore, assume that the buffer level at the regulator queue for stream $i$ is falling, and touches the low water mark. This will trigger the client algorithm to communicate its state to switch $j-1$. The server algorithm at switch $j-1$ of stream $i$, will adapt to this change in the state of switch $j$, stream $i$, by increasing its rate of feeding media units to switch $j$. As long as this rate is larger or equal to the current playback rate (which is higher than normal), the buffer will not run dry. Similar but opposite, actions occur when the high water mark is touched in the regulator queue for stream $i$ at switch $j$.

In order to provide Guaranteed Bandwidth Channels, we require that the clients declare their traffic requirements at the time of channel establishment. There are various meaningful ways in which the offered load of a channel can be specified. We require that the client must at least declare the minimum packet inter arrival time on the channel, $X_{min}$, and the maximum packet size, $S_{max}$. In this paper, a subscript $i$ is added to symbols of parameters, *e.g.* $X_{min,i}$ to indicate a particular stream whenever necessary.

We define $\tau_{i,j}$ (the maximum service time of a packet of stream $i$ at node $j$) to be $\tau_{i,j} = S_{max,i} \ / \ LinkSpeed_j$. The maximum offered load of a stream $i$ is given by $\frac{\tau_{i,j}}{X_{min,i}}$. The same maximum load over an interval $\mathcal{I}$ ($\mathcal{I} > X_{min,i}$) will be offered by another stream $k$ if maximum number of arrivals $n_k = \lfloor \frac{\mathcal{I}}{X_{min,i}} \rfloor$, in an interval of length $\mathcal{I}$; $X_{min,k} = \frac{\mathcal{I}}{n_k}$, and $\tau_{k,j} = \frac{\tau_{i,j}}{X_{min,i}} \times \frac{\mathcal{I}}{n_k}$. Also, the same maximum load over an interval $\mathcal{I}$ ($\mathcal{I} < X_{min,i}$) will be offered by a stream $k$ if maximum number of arrivals $n_k = \lceil \frac{\mathcal{I}}{X_{min,i}} \rceil = 1$; $X_{min,k} = \frac{\mathcal{I}}{n_k}$, and $\tau_{k,j} = \frac{\tau_{i,j}}{X_{min,i}} \times \frac{\mathcal{I}}{n_k}$. Therefore, the offered load of any stream $i$ can be interpreted over any arbitrary interval $\mathcal{I}$ by having a software layer that begins with one packet in reserve, and then, on each interval dictated by the calculated equivalent $X_{min,i}$ send a packet of maximum size as dictated by the calculated $\tau_{i,j}$ at node $j$. Thus, when we say $X_{min,i}$, $n_i$, $\tau_{i,j}$ or $S_{max,i}$, we mean the new calculated values of these parameters interpreted over a preset interval $\mathcal{I}$.

The tests to be done at each node are concerned with the availability of sufficient bandwidth in the links, as well as buffer space in the node to hold incoming packets. Thus these tests are designed to determine whether the new stream can go through the node without jeopardizing the continuity of streams already passing through the same node.

If any test fails at a node, the channel cannot be established along that route. The message will be sent back, either to the sender (which may then decide to wait or try another link) or an intermediate node, that can try sending the message towards the destination along another path.

The following describes in some detail the algorithms/tests for channel establishment to be executed in each node along the path from the server to the client.

### 3.2 Tests for Channel Establishment

- **The bandwidth test**
  The bandwidth test consists of verifying that enough capacity is available in the link following

the node, to accommodate the additional stream without impairing the continuity of the other streams even in the worst case. The maximum utilization of a node $j$ by a stream $i$, whose packets have maximum service time equal to $\tau_{i,j}$ is $\tau_{i,j}/X_{min,i}$. The condition to be tested is

$$\sum_{i=1}^{n} \frac{1}{X_{min,i}} \times \tau_{i,j} < 1 \qquad (1)$$

- **Buffer availability test**
  This test insures that enough buffer capacity is available at each node in the worst case. To ensure enough buffer space is reserved (and thus available) so that continuity can be guaranteed, the following local conditions must be met at each switch $j$ along the path from the server to the client.

$$R_{bu,j} + 4n_i S_{max,i} \leq \mathcal{B}_j \qquad (2)$$

  where,

  $R_{bu,j}$ is the buffer space already reserved at node $j$,

  $\mathcal{B}_j$ is the total buffer capacity at node $j$,

  $n_i$ is the maximum number of arrivals in a given interval $\mathcal{I}$, and

  $4n_i S_{max,i}$ is the space to be reserved for stream $i$ at node $j$.

  The sufficiency of the reserved buffer space for media continuity without overflow or underflow of the buffer, is proved in Section 5.

If the channels were to be established in a network based on ATM switches, then maximum rate ($S_{max}$ every $X_{min,i}$) may be calculated at 48 bytes every new $X_{min,i}$ to make the rates the same. Then this can again be interpreted over any system interval $\mathcal{I}$ as explained before. $\mathcal{I}$ can be different for each link. Similar calculations can be done for average and other rates. The client will extract data according to the original specifications and the client buffer can be maintained in the segmentation and reassembly sublayer (SAR). The higher layers at the client will request each packet and the sublayer will then forward the packet.

Since ATM networks are high speed networks, we must keep the overheads low and thus expect the protocol to be executed in hardware. Kung et al [12, 9] have proposed FCVC channels that require feedback and link-by-link control of each VC. These features are also required by our protocol and thus a switch capable of implementing the scheme proposed in [12] could easily implement our protocol with minimal changes. Such an experimental switch is available at BNR/Harvard. Our memory requirements at each switch are proportional to the Propagation delay and bandwidth of the link and can not be considered excessive for guaranteed bandwidth channels. It is noteworthy that the requirements are independent of the channel length or network architecture as opposed to the channels proposed in [6].

# 4 Buffer Update Protocol

## 4.1 Reference Model

A Buffer Managed Guaranteed Bandwidth Channels (BMGBC) is composed of one or more flow controlled links. Each of these links connect a pair of nodes. Data cells transport data for the VC and feedback cells transport buffer levels for management of the VC.

Similar to the model proposed in [9], the reference model of BUP is given in figure 2. Node D is downstream to node U with respect to the data flowing along the three virtual channels shown in the figure. The source of the data flow is either at or upstream of node U and the sink is either at or downstream of node D. The dark arrows indicate this data flow. The dotted arrows represent the flow of control cells which indicate the buffer level to the upstream node. The rectangle between the two nodes represents the packets in transit between nodes D and U. The circle S represents the service discipline that may be used. The blocks labeled FSMUi and FSMDi represent the finite state machines that implement the rate control policy of VCi over this link at the upstream and downstream nodes respectively. These finite state machines are shown in greater detail in Figure 3.

All feedback cells are self-contained. The information contained consists of the watermark (low, high or optimum) that has been touched at node D and the VC number. They are sent out-of-band at highest priority in a separate VC. If the underlying cell transport is not robust, an acknowledgment of receipt of feedback cells will be required. The receipt can consist of the rate of the regulator queue at node U for the VC.

Initialization of the BUP begins with the admission control requesting a node for a particular amount of bandwidth and buffer space. If this is available, initialization phase is entered and a BMGBC is set up by reserving buffer space. At this point, the BUP is in a "wait" state in which all packets of the VC are buffered as prefetch. Once optimal buffering is reached, a message indicating the same is sent back to the admission control, awaiting a message to start playback (transmission from regulator queue). This

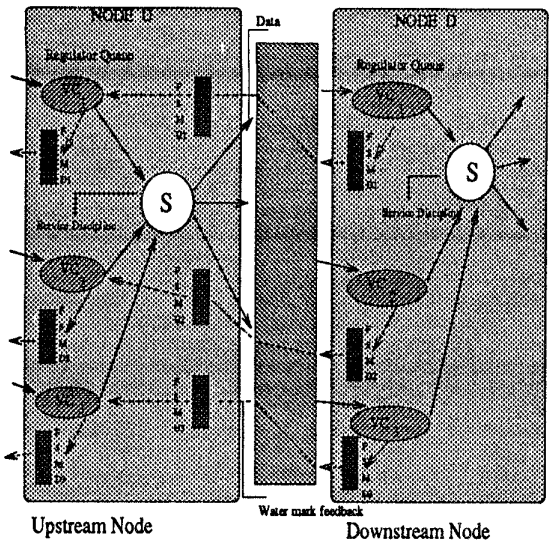


Figure 2: Reference Model for Buffer Update Protocol

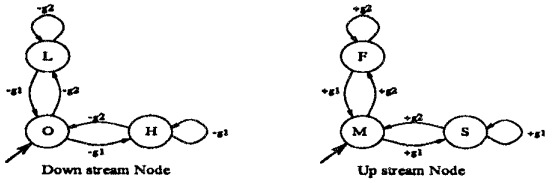


Figure 3: Finite State Machines for the Buffer Update Protocol

leaves the protocol in the initialized state indicated in figure 3

Figure 3 describes the BUP using two communicating state machines. All states in the upstream node have a self loop to send data at the prescribed rate, which is not shown in the figure. Similarly, all states in the down stream node have an implicit loop to receive data when it arrives at the node and this self loop is also not shown in the figure. Our model employs a simple and commonly used representation [2] of the communicating processes. Each pair of communicating processes are connected via a channel with FIFO property. The channel is assumed error-free for simplicity (In case the channel is not error-free, timeouts and acknowledgments will be required.) A minus sign identifies the transmission of a message,and a plus sign its reception.

The figure 3 indicates two types of feedback messages, even though we indicate three types of buffer levels. By interpreting $g1$ as speedup and $g2$ as slowdown, it can easily be seen as equivalent to sending explicit buffer levels. Further, an advantage of just two messages is that the protocol can accommodate many watermarks.

The global state of the protocol is a tuple $< u, v, x, y >$ where $u(v)$ is the state of the down stream node (up stream node) and $x(y)$ represents the string of messages in transit to the down stream node (up stream node). The initial global state is $< O, M, \lambda, \lambda >$, where $\lambda$ denotes an empty sequence. The global state may change due to a change of state of either machine. If the global state changes from $S_1$ to $S_2$ due to a transition in either finite state machine then $S_2$ is a *successor* of $S_1$. State $S_2$ is *reachable* from a state $S_1$ if there exists a sequence of successor global states starting from $S_1$ and ending at $S_2$. As in [13], if the following hold: $< u, v, \lambda, \lambda >$ is a reachable state and if $< u, v, x, y >$ is a reachable state then $x = y = \lambda$, then the state pair $(u, v)$ is called a *stable state.*

A protocol is deadlock free if there does not exist a stable state pair $(u, v)$ such that all transitions from $u$ and $v$ are receiving transitions. The protocol is free from *unexecutable receptions* if there does not exist a reachable state $< u, v, x, y >$ such that there is no transition defined for $x$ from state $u$ and for $y$ from $v$. If the protocol is free from unexecutable receptions and deadlock free then it is deemed as *safe.*

For the moment, let us assume that only one message can be outstanding; i.e. before a second feedback can be generated, the transmission of and action resulting from the first feedback will be completed. In such a case, the only stable states are $(O, M)$, $(L, F)$ and $(H, S)$. Thus, it can be seen that the protocol is safe. Further, it can be inferred that the protocol ensures that when the buffer level is at a certain mark (if in between two marks, then the mark referred to is the last water mark touched), then for each water mark there is a unique sending rate at the up stream node.

Due to the fact that feedback is sent at the highest priority, its time for receipt and action is bounded by the interval $\mathcal{I}$. Further, if the watermarks are placed far enough such that two watermarks will be touched a minimum of $\mathcal{I}$ apart, then the preceding assumption will hold true. Specifically, the distance between watermarks should be $2 * n_i$.

If the watermarks are not far enough apart, so that the assumption is no longer true, then glitches caused by the immediate arrival and departure of a packet may result in undesirable behavior. This is because at some time in the future, the FSM at node U will have state transitions which can lead to packets being sent

at a faulty rate.

## 5 Validation of the Scheme

The validation of this scheme is demontrated by the following Lemmas, theorems and Corollaries. Only the statements are provided here and the details are provided in [3].

**Lemma 1** *Let there be n streams numbered 1 through n active at a switch j with traffic characteristics given by $X_{min,i}$, $\tau_i$ for each stream i. A packet of stream i through the switch will not wait for $n_i$ or more packets of stream i.*

**Corollary 2** *The maximum delay experienced at a switch queue j by a packet of a stream i passing through the switch is $\mathcal{I}$, provided we have a work conserving scheduling discipline at the switch queue.*

**Theorem 3** *If the low water mark for a stream i at a node j is set for $n_i$, then we can prevent underflow of the regulator queue i at node j.*

**Theorem 4** *If the high water mark for a stream i at a node j is set for $n_i$ less than $\mathcal{B}_j$, then we can prevent overflow of the regulator queue i at node j.*

### 5.1 Multiple Priority Switch Queues

If users of the channels define a quality-of-service parameter by declaring a maximum amount of packets that maybe dropped (or proportion of packet under flow compared to number of packets released at a regulator queue) $Q_{max}$, then the Bandwidth Test of the Admission Control may be relaxed to let the worst case utilization of a link exceed 100%. Specifically, consider a link used by $n$ guaranteed bandwidth channels. Let streams 1 through $j$ ($n > j > 1$) have $Q_{max} = 0$ and let streams $j+1$ through $n$ have $Q_{max} = a$, $a > 0$. Further, let $\sum_{i=1}^{n} \tau_i / X_{min,i} > 1$. Then, all streams can be admitted and served within guaranteed QOS if the following conditions hold:

1. All streams are independent of each other.
2. $\sum_{i=1}^{j} \tau_i / X_{min,i} < 1$.
3. $\sum_{i=1}^{n} S_{ave} / (LinkSpeed.X_{ave,i}) < 1$.
4. Prob(Delay of a packet in any switch queue $> t$) $< a/(n-j)$.
5. Buffer space at down stream node of the link is sufficient *i.e.* streams 1 through $j$ have $4n_iS_{max}$ each reserved for them and streams $j+1$ through $n$ have $\max(1, \lfloor \frac{j}{X_{min,i}} \rfloor) 4S_{max}$ each reserved for them.
6. Maintain two switch queues such that streams 1 through $j$ send packets to one switch queue and streams $j+1$ through $n$ send to the other, where priority is given to the first switch queue. This is equivalent of modifying the cell sequencer shown in figure 2.

Clearly, Condition 2 is the same as the Bandwidth Test and in conjunction with Condition 6, the service seen by streams 1 through $j$ is the same as guaranteed/desired. Further, if Conditions 4 and 6 hold, then (as explained before) the switch queue acts like a real-time pipe. However, delay of some packets may be larger than $t$, which at worst will cause an under flow. Moreover, let us assume all packets (in any switch queue) belong to streams $j+1$ through $n$. This is true if $t > \mathcal{I}$ by Condition 6 or a worst case assumption otherwise. Thus this bounds the probability of under flow in streams $j+1$ through $n$, ensuring that these streams experience the guaranteed QOS.

If burst lengths are bounded and its characteristics known, then the delay at switch queues is also bounded [19, 4, 5]. This information may be used to calculate the probability specified in Condition 4. Alternatively, if burst characteristics are not known, exactly (as may be the case in user-interactive applications), but an estimate of the playback variation in terms of averages and variance is known then the Heavy Traffic Approximation [10] may be used to to evaluate Condition 4. Playback variation can be given by $X_{min}, X_{ave}, \sigma_X^2, S_{max}, S_{ave}, \sigma_S^2$ and it may be argued that at any upstream node the average $X_{ave}$ and $S_{ave}$ is the same and the variances of these parameters ($\sigma_X^2$ and $\sigma_S^2$) are upper bounds. Then, P(delay $> t$) $< e^{-\frac{2(1-\rho)}{X_{ave}(\sigma_X^2+\sigma_S^2)}t}$, where $X_{ave}$ is the weighted average of inter playback time of all streams, $\sigma_X^2$ is the sum of variances of inter playback time of all streams, $\sigma_S^2$ is the sum of variances of $S_{ave}$ of all streams, and $\rho$ is the average utilization calculated in Condition 3. This probability holds if all the streams were fed into one switch queue, but our assumption that all packets delayed beyond $t$ belong to streams $j+1$ through $n$ makes this an upper bound on the proportion of packets of these streams delayed beyond $t$. The upper bound is a tight upper bound if $t > \mathcal{I}$ and average utilization approaches 1. The result is quite robust [14], is easy to calculate, and requires very little information about the nature of the bursts.

In general, if multiple QOS are desired, then multiple switch queues (or priorities) have to be maintained. We feel that our scheme will be able to utilize $Q_{max}$ effectively because the protocol ensures reduc-

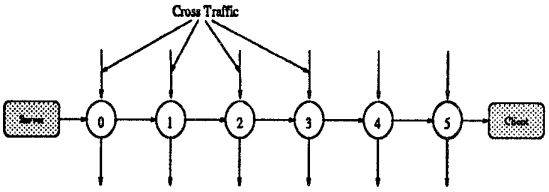


Figure 4: Network Configuration 1

tion of bandwidth consumed by a channel if it is not needed. If streams are independent, then the total bandwidth being consumed at any time will be close to that specified by the sum of average utilization of each stream with high probability. This is opposed to schemes that focus solely on delivery deadlines without regard to which channel is experiencing a burst and which channel is experiencing a slowdown.

## 6 Simulation

In the previous section, we have given proofs of our scheme's capability to guarantee quality of service to various streams. In this section, we provide simulation results for our scheme. The simulation results provide an evaluation of the performance of our scheme in terms of network availability, buffering requirements, and correctness.

Simulations were performed using various network topologies and traffic characteristics. We present the results for the network shown in Figure 4. This particular network configuration was chosen for presentation to highlight the number of streams that may travel along a common path. Further, jitter along long paths can be substantial, thus testing the adequacy of our buffering thoroughly. Moreover, such a network removes the effects of routing decisions affecting overall availability and is close to the fairness configurations of GFC1 and GFC2 given in [17].

Future multimedia traffic is expected to consist of several diverse types, e.g. full motion video, audio, animation, etc. Taking this into consideration, we have chosen our workload to consist of two types that are expected to tax the network the most. The traffic types chosen have data rates of 2.0 Mbits/sec and 8.0 Mbits/sec , roughly corresponding to MPEG-1 [8] and MPEG-2 [7] compressed full-motion video, respectively. Each link was assumed to have a transmission capacity of 100 Mbits/s. Cross-traffic was introduced at each link from the server to the client, such that the Bandwidth Test just failed, whereas buffering was assumed to be adequate.

The traffic characteristics of the various streams in the simulation are given in the following table.

| *Stream* | $X_{min}$ | $\mathcal{I}$ | $n_i$ | $S_{max}$ | $X_{ave}$ | $S_{ave}$ |
|---|---|---|---|---|---|---|
| MPEG-1 | 41 | 41 | 1 | 80 | 41.667 | 27.7 |
| MPEG-2 | 10.25 | 41 | 4 | 80 | 10.75 | 27.7 |

Each frame length in the MPEG-1 type of traffic is stochastically generated in accordance with the data observed and modeled in [1]. The client extracts data from its buffer according to these criteria. The media units of such a stream correspond to typical frames of digitized video. As mentioned earlier, MPEG-2 type of traffic requires approximately four times the data rate of a similar MPEG-1 encoded stream. This load has been modeled by shortening the interarrival times of the media units by a factor of four in our simulation.

The traffic characteristics given in previous table were interpreted for an ATM network, yielding the following characteristics.

| *Stream* | $X_{min}$ | $\mathcal{I}$ | $n_i$ | $S$ | $X_{ave}$ | $X_{max}$ |
|---|---|---|---|---|---|---|
| MPEG-1 | 196.8 | 196.8 | 1 | 53 | 575.4 | 2016 |
| MPEG-2 | 49.1 | 196.8 | 4 | 53 | 143.85 | 504 |

Buffering was assumed adequate and $6n_i53$ bytes were reserved for each stream. This assumes propagation delay to be negligible. However this does not change our results but only the buffering requirements. Once adequate buffering is available, the network availability is solely determined by the bandwidth test, in particular Equation 1.

An MPEG-1 type stream running from the server to the client in Figure 4 was monitored. The monitoring interval was only a few seconds due to the huge data size created, but the playback rate was varied artificially to present the protocol operation over many scenarios. The buffer occupancy (and rate at upstream node) at the client and Node 3 are shown in Figures 5 and 6. The buffer occupancy (and rate) at other nodes is very similar but shifted slightly in time and thus omitted. The dark portions represent changes in buffer level in a much smaller time interval than the gradation shown on the Time axis. No overflow or underflow occurred as can be observed from the figures, hence proving our scheme to be correct. The buffer level reaching zero is not an underflow, because the media unit arrived at the client before its playback was scheduled, *i.e.* it arrived during the playback of the previous media unit.

Further, the rate of overflow at the upstream regulator queue (or server) matches the rate dictated by the waterlevel at the downstream node (or client). This validates the correctness of behavior of our protocol. It can also be seen that the average rate from each regulator queue is the same as playback rate but the variance is smaller, which validates our previous assumption in calculation of probabilities of delay in

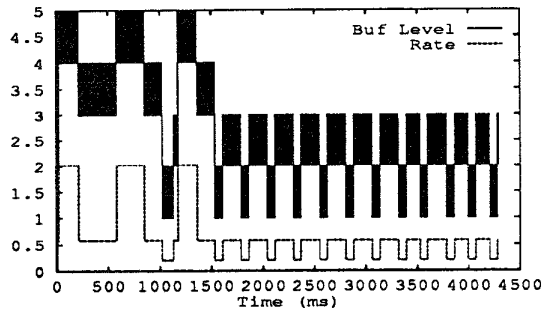


Figure 5: Comparison of Buffer Level at Node 3 with Rate at Upstream Node

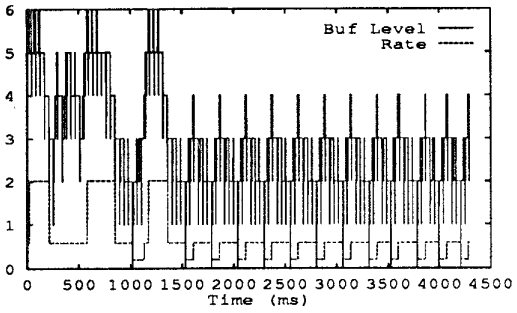


Figure 6: Comparison of Client Buffer Level with Rate at Upstream Node

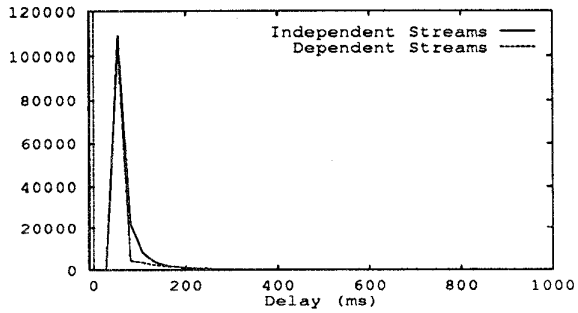


Figure 7: Delays of packets in all Switch Queues

switch queues.

Finally, to investigate our claims of behavior when multiple priority switch queues are used, we investigated switch queues delays. Four switch queues (priority 1 to 4 with 1 highest) serving three streams were investigated. For each stream we used the frame size statistics from an actual MPEG encoded scene from the movie Terminator 2. The traffic parameters are shown in the table below.

| $X_{min}$ | $\mathcal{I}$ | $n_i$ | $S$ | $X_{ave}$ | $X_{max}$ |
|---|---|---|---|---|---|
| 110.0 | 110.0 | 1 | 53 | 373.7 | 3125 |

The network bandwidth was set to 2Mbytes/sec. This allowed streams entering switch queue 1 to be served with no underflow or overflow. However, the other streams had a QOS parameter allowing for a proportion of underflow (or overflow) at the regulator queue and the client buffer, depending on the switch queue priority they were using. The parameters $X_{ave}$ and $\sigma_X^2$ can be easily calculated from the average frame size and the variance in frame size. No overflow or underflow occurred for any stream thus proving our assumptions correct.

Then the independence of streams was violated to check its effects. All streams had identical starting points and were started simultaneously. Thus all bursts and slow downs were not independent. Figure 7 shows the the delays for all streams when they are independent and otherwise to ensure exponential delays as claimed. However, now there was 1 underflow per stream entering switch queue 3 out of 150 frames played back at the client. Also, there were a maximum of 10% overflow/underflow for streams entering entering switch queue 4. The overflow occurred due to the fact that the switch queue did not receive service for a long time during a burst and in a slow down many packets arrived at the client despite the feedback. However, this is not anomalous behavior and within bounds guaranteed. These observations show that our upper bound is tight if simultaneous bursts of all streams can not be ruled out, which is likely when user interactive real-time streams are serviced. However, there is room for improvement by requiring more information about the bursts and slow downs when it is available.

## 7 Conclusions

In this paper, we have proposed a scheme in which, given bounded delays (or a bound on probability of delay) at each node, guaranteed bandwidth (with parameter $Q_{max}$) can be ensured. No other conditions pertaining to arrival patterns and schedulability need be checked for admission of a new stream to the network.

Our scheme keeps the buffering reasonably small, and independent of the network load and the path length. Furthermore, our scheme eliminates unnecessary feedbacks by communicating state changes only on impending buffer overflows or underflows in the absence of corrective action. This is achieved by maintaining water marks and modifying the push of media units as needed, thereby combining aspects of the pull model.

The protocol can easily be incorporated into ATM-based networks that use link-by-link per VC flow control with minimal changes to the switch architecture and very low overhead. We would like to note that our scheme will work in any situation where delay at switches can be bounded, with minimal changes. This will prove very helpful in cases where input traffic is bursty but burst lengths are limited and delay at switches is still bounded. Also, different end-to-end delay guarantees can be offered to different media streams by using multiple switch queues served with different priorities. Additionally, if a QOS parameter providing the proportion of packets that maybe dropped, then the Bandwidth Test and the Buffer Space Test can be relaxed. These trade-offs need to be investigated further.

## References


[1] Giovanni Pacifici Aurel A. Lazar and Dimitrios E. Pendarakis. Modeling video sources for real-time scheduling. *ACM Multimedia Systems*, 1994.

[2] Daniel Brand and Pitro Zafiropulo. On communicating finite-state machines. *Journal of the ACM*, 30(2):323–342, April 1983.

[3] S. Chaudhry and A. Choudhary. Guaranteed continuity of playback of multimedia streams. *CASE Technical Report*, July 1995.

[4] Rene L. Cruz. A calculus for network delay, part I: Network elements in isolation. *IEEE Transations on Information Theory*, 37(1), January 1991.

[5] Rene L. Cruz. A calculus for network delay, part II: Network elements in isolation. *IEEE Transations on Information Theory*, 37(1):132–142, January 1991.

[6] Domenico Ferrari and Dinesh 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.

[7] Borko Furht. Multimedia systems: An overview. *IEEE Multimedia*, pages 47–59, Spring 1994.

[8] Didier Le Gall. Mpeg: A video compression standard for multimedia applications. *Communications of the ACM*, 34(4):46–58, April 1991.

[9] H. T. Kung Jon C. R. Bennett, Koling Chang and Dong Lin. A comparison of eprca september 1994 version and fcvc control schemes: Simulation results. *ATM_Forum/94-0929*, 1994.

[10] J. F. C. Kingman. On queues in heavy traffic. *J.R.S.S.*, pages 383–392, 1962.

[11] H. T. Kung. Gigabit local areas networks: A system perspective. *IEEE Communications Magazine*, pages 79–89, April 1992.

[12] H. T. Kung and Koling Chang. Receiver-oriented adaptive buffer allocation in credit-based flow control for atm networks. In *Proc. INFOCOM '95*. INFOCOM, April 1991.

[13] H. Lin and C. Tarng. An improved method for constructing multiphase communications protocols. *IEEE Transations on Computers*, 42(1), 1993.

[14] J. Medhi. *Stochastic Models in Queueing Theory*. Academic Press, Inc., 1991.

[15] Srinivas Ramanathan and P. Venkat Rangan. Adaptive feedback techniques for synchronized multimedia retrieval over integrated networks. *IEEE/ACM Transactions on Networking*, 1993.

[16] Srinivas Ramanathan and P. Venkat Rangan. Feedback techniques for intra-media continuity and inter-media synchronization in distributed multimedia systems. *The Computer Journal*, 1993. Special Issue on Distributed Multimedia Systems.

[17] Robert J. Simcoe. Configurations for fairness and other test. *ATM_Forum/94-0557*, July 1994.

[18] Dinesh C. Verma, Hui Zhang, and Domenico Ferrari. Delay jitter for real-time communication in a packet switching network. In *Proceedings of the Triangle Conference on Communication Software*, pages 35–43. TRICOMM, 1991.

[19] Hui Zhang and Domenico Ferrari. Improving utilization for deterministic service in multimedia communication. *IEEE Computers*, pages 295–304, 1994.