Network Algorithms and Protocol for Multimedia Servers*

Pawan Goyal and **Harrick M. Vin**

Distributed Multimedia Computing Laboratory
Department of Computer Sciences, University of Texas at Austin
Taylor Hall 2.124, Austin, Texas 78712-1188, USA
E-mail: {pawang,vin}@cs.utexas.edu, Telephone: (512) 471-9732, Fax: (512) 471-8885

Abstract

In this paper, we present a network service specifically designed for multimedia servers. It uses a histogram based traffic characterization and an overload control protocol to eliminate packet losses in the network while providing heterogeneous statistical QoS. The key contribution of our protocol lies in combining open-loop and feedback-based control to: (1) provide heterogeneous QoS to clients in networking environments consisting of switches that may not have any scheduling support; and (2) migrate the functionality of discarding packets, in the event of congestion, to the sources which understand the semantics of the data. The protocol is efficient, makes very few assumptions about the underlying network, is realizable on current switching hardware (supporting FCFS scheduling), and is completely integrated with the architecture of a multimedia server.

1 Introduction

A common feature in a wide variety of multimedia applications is their requirement for storing, retrieving, and transporting various data types (e.g., textual and numeric data, images, audio, video, animation sequences, etc.) over high-speed networks. Among these data types, motion video is the most demanding in terms of its size and real-time performance requirements. Several projects have recently investigated techniques for storage and retrieval of video streams from disk arrays [12]. However, techniques for efficient transmission of *stored* video over high-speed networks have not been adequately investigated.

Variable bit rate (VBR) video data has multiple time-scale variations, i.e., bit rate requirement varies over millisecond, second and minute time-scales [3]. Hence, to achieve efficient utilization of network resources, network capacity would have to be overbooked. However, overbooking of resources may lead to transient overloads and consequently packet losses. Although uncompressed video streams, due to the inherent spatial and temporal redundancy, are highly resilient to packet losses, compressed video streams are not. Since real-time nature of video playback precludes any retransmission of lost packets, minimizing the impact of such packet losses on video quality will require the sender and the receiver to employ additional error recovery methods. For instance, at a receiver, the decoder may mitigate the effect of lost packets by approximating the missing data through temporal or spatial interpolation [10]. On the other hand, if the network provides multiple priority levels and if the source employs a multi-layer encoder, then the reconstructed image quality at the receiver can be improved by transmitting layers of the encoded video stream at different priority levels (e.g., transmitting the essential and the enhancement layers at high and low priorities, respectively) [8]. Though such methods may be appropriate for live video sources where overload can not be predicted a priori, protocols that are optimized for stored video communication can exploit predictability of the bit rate requirement to completely avoid any packet loss in the network. Such protocols can shift the error recovery functionality to the servers which know the semantics of the data and can ensure minimum degradation in the perceived video quality during congestion. The design of such a specialized protocol constitutes the subject matter of this paper.

We present a network service specifically designed for multimedia servers. Since end-to-end delay requirements for stored video are not very stringent, our protocol shapes the traffic such that the rate changes are regular, infrequent, and predictable. A histogram-based characterization of this traffic, when coupled with an admission control algorithm, enables the network to provide heterogeneous statistical QoS guarantees to clients. To meet these guarantees, we propose an *overload control* algorithm and protocol that exploits the regularity and predictability in traffic to

*This research was supported in part by IBM Graduate Fellowship, Intel, the National Science Foundation (Research Initiation Award CCR-9409666), NASA, Mitsubishi Electric Research Laboratories (MERL), and Sun Microsystems Inc.

provide heterogeneous QoS while completely eliminating packet losses in the network. Specifically, the protocol: (1) detects congestion in the network prior to its occurrence, (2) allocates bandwidth to competing sources based on their QoS guarantees, and (3) transmits feedback packets to the sources indicating the rate allocations. We formulate the problem of allocating bandwidth which meets the QoS guarantees of sources as a linear program, and then present an optimal, linear-time algorithm which exploits the special structure of the problem. By ensuring that the feedback is received by the sources sufficiently prior to overload occurrence, the protocol enables the sources to employ application specific procedures to adjust their transmission rates so as to conform to the rate allocations, thereby eliminating any packet losses in the network.

The key contribution of our protocol lies in combining open-loop and feedback-based control to: (1) provide heterogeneous QoS to clients in networking environments consisting of switches that may not have any scheduling support; and (2) migrate the functionality of discarding packets, in the event of congestion, to the sources which understand the semantics of the data. The protocol is efficient, makes very few assumptions about the underlying network, is realizable on current switching hardware (supporting FCFS scheduling), and is completely integrated with the architecture of a multimedia server.

The rest of the paper is organized as follows. The histogram-based traffic characterization and admission control algorithm are presented in Section 2. The overload control algorithm and protocol are presented in Section 3. Salient features of our protocol are outlined in Section 4. Section 5 presents the results of our experiments, and finally, Section 6 summarizes our results.

2 Traffic Characterization and Admission Control Algorithm

A multimedia server organizes the storage of video streams on disk arrays in terms of fixed size media blocks [12]. Due to the periodic nature of the video playback, the server can service retrieval requests from clients by proceeding in fixed duration *rounds*. In a round, a server retrieves fixed number of media units (video frames) sufficient to sustain playback for the duration of the round for each client. The media blocks retrieved by the server are then transmitted to clients over a network.

In the simplest case, the server may associate a maximum delay with each frame in a media block, and then transmit the frame at a rate that would satisfy the delay requirement. In such a scenario, however the transmission rate may change very frequently, potentially with each frame. Since detecting congestion prior to its occurrence requires the rate changes to be conveyed to a network, such a protocol will incur significant overhead and hence may be infeasible. To develop network protocols that detect congestion a priori the rate changes must be infrequent, regular, and predictable.

Since the end-to-end delay requirements for stored video are not very stringent, a server can make the rate changes infrequent, regular, and predictable by transmitting the video information accessed during a round at a constant rate during the next round. Such a method allows a server to easily characterize the variation in the bit rate requirements of a video stream which can then be utilized by network admission control algorithm. In what follows, we first detail our method for characterizing the bandwidth requirement of stored video, and then present a statistical network admission control algorithm.

2.1 Characterization of Traffic from Multimedia Servers

A multimedia server can facilitate the development of efficient network protocols by transmitting media blocks retrieved in a round at a constant rate in the next round. To ensure continuous playback of video streams at client sites as well as to minimize the buffer space requirement at the server, the rate of transmission must be chosen such that all the blocks accessed in a round are transmitted by the end of the next round. Thus, if S and B_i^j denote the size of media block and the number of blocks of client i accessed during round j, respectively, then the transmission rate for client i during round $j+1$ is : $\gamma_i^{j+1} = \frac{B_i^j * S}{T}$ where T is the duration of the round. Observe that this transmission method does periodic averaging of a video source and generates a piecewise constant bit rate traffic[11]. Moreover, it enables a multimedia server to succinctly capture the variations in the bit rate requirement of a video sequence. Specifically, since a server can derive all the frame sizes at the time of storage of a video sequence, it can compute a histogram of the number of media blocks that would be accessed for each video stream in each round. Such a histogram characterizes the network bandwidth requirement of a stored video sequence, which, when sent by a multimedia server in a connection establishment message, can be utilized for admission control.

Observe that, due to the large data rate requirement of a video stream, a multimedia server selects large media block sizes. Consequently, the number of histogram bins required for such a characterization are likely to be very small. Thus, inclusion of the histogram does not significantly increase the size of a connection establishment message. Furthermore, while multiplexing a large number of connections, the mean and the variance of the bit rate requirement derived from such a histogram are sufficient for network admission control, thereby further reducing the size of connection establishment message. Finally, such a histogram based characterization of data rate requirement is also necessary for implementing admission control at the multimedia server [12]. Hence, the traffic characterization method does not impose any additional requirement on the server.

2.2 Admission Control Algorithm

Given a histogram-based traffic characterization, precise formulation of an admission control algorithm depends on the QoS parameters (end-to-end packet delay, delay jitter and packet loss probability) desired by the sources. For stored video, end-to-end delay requirements are not very stringent. On the other hand, due to the compressed nature of each video stream, packet losses resulting from buffer overflow at switching nodes may significantly degrade the video quality. Given that the hard timing constraints imposed by continuous video playback may preclude any retransmission of lost packets, minimizing packet loss can be considered more important. Hence, in what follows, we utilize packet loss probability for deriving a statistical admission control algorithm.

Consider a network switch that is multiplexing N video sources originating from distributed multimedia servers onto a link of capacity C (bits/sec). Let random variable $\mathcal{X}_i$ denote the bit rate requirement of the i^{th} video source, and let q_i denote the maximum acceptable loss probability. Without loss of any generality, let us assume that $\forall j \in [i..N] : \; q_i \leq q_j$. To support heterogeneous QoS requirements of sources, let a priority be assigned to each source in accordance with the packet loss probability desired by the sources. Specifically, the higher the packet loss probability, the lower is the priority. In such a scenario, to meet the QoS requirements of all the channels, the switch must ensure that, at each priority level, the probability of cumulative bandwidth of sources, belonging to priority levels equal to or higher than that level, exceeding the link capacity is within the tolerable packet loss at that level. That is, the following set of inequalities must hold:

$$\forall j \in [1, N] : \quad P\left(\sum_{i=1}^{i=j} \mathcal{X}_i > C\right) \leq q_j \qquad (1)$$

Clearly, to evaluate the above set of inequalities, the distribution function of random variable $\mathcal{Y}_j = \sum_{i=1}^{i=j} \mathcal{X}_i$ is required. Since $\mathcal{X}_i$'s are independent, discrete random variables, the distribution of $\mathcal{Y}_j$ can be derived using:

$$\forall j \in [1, N] : \quad Z(\mathcal{Y}_j) = \prod_{i=1}^{i=j} Z(\mathcal{X}_i) \qquad (2)$$

where $Z(\mathcal{X})$ denotes the z-transform of a discrete random variable $\mathcal{X}$.

Thus, if a new source requires a maximum loss probability of q_k (corresponding to priority level k), and if the total number of sources prior to admitting the channel is $N-1$, the switch must ensure that:

$$\forall j \in [k, N] : \quad P\left(\mathcal{Y}_j > C\right) \leq q_j \qquad (3)$$

Observe that since the distribution of $\mathcal{Y}_j$ can be evaluated using distribution of $\mathcal{Y}_{j-1}$, the computational complexity of admission control algorithm is not significant. The complexity of computing the distribution function of Y_j can be further reduced by using central limit theorem to approximate the distribution of $\mathcal{Y}_j$ by a normal distribution.

Since the admission control algorithm provides statistical QoS, the cumulative bandwidth requirement of the sources may sometimes exceed the link capacity, i.e., an *overload* may occur. To ensure that the QoS guaranteed by the admission control algorithm is satisfied in the event of overload, a network switch may employ a scheduling algorithm to serve packets from sources in the priority order assigned by the admission control algorithm. Such a technique would cause the packets of lower priority sources to be discarded in preference to packets from higher priority sources. Since this was the only assumption made in the admission control algorithm, the QoS of the sources would be satisfied. However, in addition to increasing the complexity, such a technique may cause arbitrary packets to be discarded and may lead to significant degradation in the video quality during overload. On the other hand, an overload control mechanism which exploits the regularity and predictability of video traffic retrieved from multimedia servers can completely avoid packet losses in the network and provide better video quality during overload. Furthermore, it can also eliminate the requirement of employing priority service discipline and thereby simplify the switch design. In the next section, we present such an overload control technique.

3 Overload Control Algorithm and Protocol

Observe that due to the sequential playback of video, the bandwidth requirement of video transmitted from a server is known a priori and hence overload at a switch can be detected prior to its occurrence. When an overload is detected, a switch can determine the rate allocations for each of the source that would avoid overload while satisfying the QoS of the sources. The rate thus determined can be sent in a feedback packet to each of the servers from which video is being transmitted. A server can then exploit the semantic knowledge of a video sequence to reduce the spatial, temporal, or chroma resolution, and thereby completely eliminate packet loss in the network [1]. The key requirements for an efficient and feasible implementation of such an overload control mechanism are:

- The rate changes at a server must be required at most at a round boundary. Since a server proceeds in rounds, this would simplify the design of the server and facilitate an efficient implementation.
- The available rate for a round should be known at the server at least Δ_o prior to its initiation. This would allow a server to reduce the bit rate by exploiting semantic information of the data.

In order to develop an overload control mechanism that meets the above requirements, let us assume that the round time of each of the source (multimedia server) is T, and that there is no phase relationship between the sources (i.e., the rounds are not synchronized). Since rate can be changed at a source only at round boundaries, let the switches periodically detect and control overload at an output link with period T.

Let r_i be the round of source i that is in progress at the switch when an overload control procedure is invoked. As there is no phase relationship between the time a round of a source begins at a switch and the invocation of the overload control algorithm, the earliest round, for which the overload control algorithm can change the rate and still ensure that feedback is received by a source at least Δ_o prior to initiation of that round, is $(r_i + W + 1)$, where W is defined as:

$$W = \left\lceil \frac{\Delta_d + \Delta_c + \Delta_f + \Delta_o}{T} \right\rceil \tag{4}$$

where Δ_c is the computation time required by a switch to detect and control overload, and Δ_d and Δ_f denote the maximum delay experienced by data and feedback packets, respectively. Thus, the overload control algorithm has two main functions: (1) determine if overload would occur in round (r_i+W+1) of sources, and (2) in the event of a possible overload, determine rate assignments for appropriate rounds of sources such that packet loss in the network is avoided. These two functions are described in detail in the following sections.

3.1 Overload Detection

Let t be the time at which the overload control algorithm is invoked, b_i be the time at which round r_i of source i begins at the switch, and S be the set of sources. Hence, the times at which round $r_i + W + 1$ begins and ends at the switch are $b_i + (W+1)T$ and $b_i + (W+2)T$, respectively. Thus, if t_1 and t_2 denote the earliest and the latest times at which round $(r_i + W + 1)$ of sources begin and end, respectively, then:

$$t_1 = \min_{i \in S}\{b_i + (W+1)T\}; \qquad t_2 = \max_{i \in S}\{b_i + (W+2)T\} \tag{5}$$

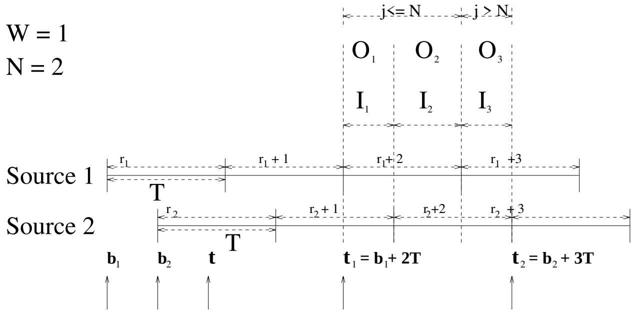

Figure 1 : Overload control: example scenario

To detect an overload, we divide the interval $[t_1, t_2]$ into subintervals $I_1, I_2, ..., I_n$ such that no subinterval intersects with more than one round of a source. Thus, starting from t_1, a new subinterval is created whenever a round of any source begins. Since $t_2 - t_1 > T$, for ease of exposition, if we assume that no two rounds begin at the same time, then the interval $[t_1, t_2]$ is partitioned into $2N-1$ subintervals, where $N = |S|$. For example, in Figure 1, with two sources, the interval $[t_1, t_2]$ has been partitioned into three subintervals. For each of the subintervals, an overload exists if the bandwidth requirement exceeds the link capacity. Specifically, if $BW(I_j)$ and C denote the aggregate bandwidth requirement in subinterval I_j and the link capacity, respectively, then, overload in subinterval I_j, denoted by O_j, is defined as:

$$O_j = \max\{0, BW(I_j) - C\} \tag{6}$$

Formally, an overload exists if $O_j > 0$ for some j. Since in interval $[t_1, t_2]$, the rate of exactly one source changes between two adjacent subintervals, overload in all the $(2N-1)$ subintervals can be computed in $O(N)$ time.

Observe that to detect overload, a switch must know the bandwidth requirement in the interval $[t_1, t_2]$. Since $\forall i \in S$, $b_i \leq t$, we conclude $t_2 \leq t + (W+2)T$ and $t_1 > t$. Hence, a switch can detect overload if the bandwidth requirement is known in the interval $[t, t+(W+2)T]$. The overload control protocol followed by the sources meets this requirement (see section 3.3).

3.2 Rate Assignment

When an overload is detected, the objective is to determine permissible rates for each of the sources so that overload is avoided. However, the rate assignments should be such that the QoS of the sources is satisfied. In order to determine rates that would satisfy the QoS of the sources, we define *loss affordability* for a source. Loss affordability of a source, intuitively, is the amount by which its rate can be reduced by a switch without violating the desired QoS. Let γ_i^j and $\hat{\gamma}_i^j$ be the rate desired by and assigned to source i in j^{th} round, respectively. Then, loss affordability for source i, denoted by $\mathcal{A}_i$, is defined as:

$$\mathcal{A}_i = \max \left\{ 0, q_i \sum_{j=1}^{j=r_i+W+1} \gamma_i^j - \sum_{j=1}^{j=r_i+W+1} (\gamma_i^j - \hat{\gamma}_i^j) \right\} \tag{7}$$

From the protocol requirements, we know that only rates for rounds greater than round $r_i + W$ should be changed. Consequently, some sources may not be eligible for rate reduction in a particular subinterval. Hence, we define eligible set for subinterval I_j, denoted by $E(I_j)$, to consist of a set of sources such that $\forall i \in E(I_j)$, I_j intersects with intervals corresponding to rounds greater than $r_i + W$. For example, in Figure 1, $W = 1$, $E(I_1) = \{1\}$ and $E(I_2) = E(I_3) = \{1, 2\}$.

Given the loss affordability for all the sources and the eligible set for all the subintervals, a simple rate assignment algorithm may consider each of the subintervals in isolation and avoid overload in each of them. However, since rate reduction in one subinterval effects the rate reduction required in other subinterval, a naive algorithm may result in higher aggregate rate reduction than necessary. In order to minimize rate reduction, we formulate the rate assignment problem as a linear programming problem and then present an optimal linear time algorithm.

To formulate the problem as a linear program, consider subinterval I_j (i.e., the j^{th} subinterval in $[t_1, t_2]$). From Figure 1, it is evident that if $j \le N$, rates for only rounds $r_i + W + 1$ can be reduced. However, if $j > N$, rate may be reduced for either round $r_i + W + 1$ or round $r_i + W + 2$ of a source. Let the rate reduction for round $r_i + W + 1$ and round $r_i + W + 2$ be denoted by δ^i and δ^{N+i}, respectively [1]. Let the sources be ordered in ascending order of begin time of rounds, i.e., $\forall k \in [i..N]: \quad b_i < b_k$. Consider the two cases $j \le N$ and $j > N$:

- $1 \le j \le N$: In subinterval I_j, only sources $1, ..., j$ are eligible for rate reduction. Moreover, rate can be reduced for only round $r_i + W + 1$ of sources. Hence, for overload avoidance we have:

$$\sum_{i=1}^{i=j} \delta^i \ge O_j \tag{8}$$

- $N < j \le 2N - 1$: In subinterval I_j, all the sources are eligible for rate reduction. From Figure 1, we conclude that for source i such that $i + N \le j$, rate can be reduced for round $r_i + W + 2$. For all other sources, rate can be reduced for round $r_i + W + 1$. Hence, for overload avoidance we have:

$$\left(\sum_{i=1}^{i=j-N} \delta^{N+i} + \sum_{i=j-N+1}^{i=N} \delta^i \right) \ge O_j \tag{9}$$

To ensure that the QoS of the sources are satisfied, we further require that rate reduction for each source is no more than the loss affordability, i.e,

$$\delta^i + \delta^{N+i} \le \mathcal{A}_i \quad 1 \le i \le N \tag{10}$$

Furthermore, since the rate reduction for a source in a round can be at most the rate desired by the source in that round, we have:

$$0 \le \delta^i \le \gamma_i^{r_i+W+1} \quad 1 \le i \le N \tag{11}$$

$$0 \le \delta^{N+i} \le \gamma_i^{r_i+W+2} \quad 1 \le i \le N \tag{12}$$

Thus, the rate assignment problem is to determine values for δ^i that satisfy inequalities (8), (9), (10), (11), and (12), which minimizes the aggregate rate reduction $\sum_{i=1}^{i=N} (\delta^i + \delta^{N+i})$. The above problem can be solved by employing conventional linear programming techniques. However, the computational complexity may make them infeasible. To address this limitation, in what follows, we present an optimal linear time algorithm.

The algorithm initializes the variables δ^i to 0 and determines the optimal values by proceeding in steps. In step j, it determines the values of the variables that would satisfy inequality corresponding to subinterval I_j in the following way:

1. If the current values of the variables satisfy the inequality, the values are left unchanged and step j is complete.

2. Otherwise, new values are determined by increasing the variables in the descending order of the superscript, i.e., the variables are increased in the order $\delta^j, \delta^{j-1}, ..., \delta^1$. The value of variable δ^k ($k \le j$) is increased only if the variables δ^j through δ^{k+1} have been assigned maximum values consistent with (10), (11), and (12), and the inequality corresponding to I_j is still not satisfied. Furthermore, δ^k is increased only as much as is necessary to satisfy the inequality corresponding to interval I_j while not violating (10), (11), and (12).

3. If δ^1 has been increased to the maximum possible value and the inequalities are still not satisfied, then overload is avoided by iteratively reducing the rate

[1] Since there are only N sources, δ^i are well defined.

of the sources in the eligible set of the subinterval in decreasing order of loss probability, until there is no overload. Rate is reduced equally from all sources that have the same loss probability.

Though a naive implementation of the above algorithm would have $O(N^2)$ complexity, an implementation that exploits the special structure of the inequalities has $O(N)$ complexity and is presented in [4]. As Theorem 1 shows, the above algorithm finds an optimal solution (proof is presented in [4]).

Theorem 1 *If* $\forall i \in [1..N] : \mathcal{A}_i \leq \min\{\gamma_i^{r_i+W+1}, \gamma_i^{r_i+W+2}\}$ *and a feasible solution to the inequalities exists, then the algorithm always finds a solution which minimizes* $\sum_{i=1}^{i=N}(\delta^i + \delta^{i+N})$.

Observe that $\mathcal{A}_i < \min\{\gamma_i^{r_i+W+1}, \gamma_i^{r_i+W+2}\}$ deos not hold only when the QoS desired by the sources is very low (i.e., the loss probability is high). In such scenarios, our algorithm may not be able to find an optimal solution, and hence, if optimal solution is always desired, one of the numerous techniques for linear programming problems may have to be used. The principal advantage of our algorithm is that it can be implemented with $O(N)$ complexity with simple computations at each step.

3.3 Overload Control Protocol and Implementation

The overload control algorithm requires that at all time t, a switch should know the bit rate requirement of all the sources during the interval $[t, t + (W + 2)T]$. To meet this requirement, a source sends the bit rate required in the first $W+2$ rounds in the connection establishment message. The connection establishment procedure, in our protocol, is assumed to be the same as in [2] with the admission control criteria being replaced by our algorithm. Since overload can not be controlled during the first $W + 1$ rounds of a source, we require that a connection establishment request returns with the available rates for the first $W + 1$ rounds. Once the connection is established, a source identifies the first packet of each round. In first packet of each round n it sends the desired bit rate for round $n + W + 2$.

A switch, on receipt of a first packet of a round from a source, updates the rate requirement and the round count for the source. It periodically controls overload at an output link and when the overload control algorithm reduces rate of source i for round $r_i + W + 1$, it sends a feedback packet to the source containing the assigned rate and the round number.

Since switches independently detect overload and send a feedback packet, a source may receive feedback from multiple switches for a particular round. In such a scenario, the source is required to send at the minimum feedback rate received for that round. If a source does not receive any feedback Δ_o prior to the initiation of a round, then it sends at the desired rate for that round.

The overload control algorithm and protocol described above are simple and can be implemented in current hardware. Some of the implementation considerations are:

- The protocol requires a switch to distinguish the first packet of a round from other packets. However, this is not an additional requirement on the switches. To implement Available Bit Rate (ABR) services defined for ATM networks, existing switches distinguish a Resource Management (RM) packet from the data packets. The first packet of a round can be considered to be a RM packet and hence, the protocol is realizable in current hardware.

- Since the overload detection and rate assignment algorithms involve very simple steps in each iteration of the algorithm, the computational requirement of overload control is insignificant. Similarly, on receipt of an RM packet, a switch is only required to update the round number and the rate requirement of the source which, also, is computationally inexpensive.

- One of the assumptions of the protocol is that the round length of all the sources is the same. This assumption does not require each server to select the same round time for retrieving data from the disks but is a network service abstraction provided to the servers which guarantees that rate changes would be separated by at least T. If the round length for retrieval from disk is not the same as T, a server can easily map the disk round to the network round abstraction and achieve the benefits of the protocol. Hence, this assumption is not restrictive from implementation perspective.

- We have assumed that if the overload control algorithm ensures that the cumulative bandwidth requirements of all the sources is less than the link bandwidth, then overload would not occur. As the sources are smooth and piecewise constant bit rate, we expect this assumption to be valid. Our experiments and some of the results reported in [5] demonstrate that this assumption holds for a simple scheduling discipline like FCFS.

The protocol employs traffic shaping and, hence, the initiation latency may increase, an upper bound for which has been derived in [4] but omitted here due to space constraints.

4 Salient Features of Our Protocol

The protocol that we have presented is not only simple and realizable in current hardware, but also has several desirable features:

- The overload control algorithm assigns rates as per desired QoS of sources and provides heterogeneous QoS. It does not require any scheduling support at the switches; in fact, simple FCFS scheduling algorithm at the switches is sufficient. This is in contrast to most of the existing approaches which rely on more sophisticated scheduling support at the switch to provide heterogeneous QoS to clients.

- It avoids any packet losses in the network by requiring a source to reduce its rate during overload. Since a source understands the semantics of the data it transmits, the video quality perceived by clients during overload period would be better than the video quality perceived when packet losses occur in the network.

- The parameter Δ_o provides considerable flexibility to a server. Since feedback is received sufficiently prior to the beginning of a round, a server can retrieve only as much data as it can transmit and thereby optimize the utilization of disk bandwidth.

- An important aspect of the protocol is that it requires rate changes at a multimedia server at most at every round boundary. Hence, the schedule at the server network interface card has to be changed at the most at a round boundary. Since a round typically consists of several frame intervals and is of the order of a second, this significantly reduces the computational complexity of scheduling at network interface card as compared to other traffic shaping mechanisms in which the network interface card schedule may have to be changed every frame interval [7].

5 Experimental Evaluation

To experimentally evaluate the various aspects of the algorithms and protocol, we have carried out extensive trace driven simulations using an enhanced version of the REAL network simulator (available from the University of California at Berkeley). To simulate an ATM network, the packet size was selected to be 53 bytes with 5 bytes of header. The link scheduling algorithm was assumed to be FCFS. The complete protocol was implemented in the simulator and was evaluated using 6 MPEG video sequences, each consisting of about 10000 frames, derived from a variety of video sources including television serials, newscasts, movies and cartoons [4]. For the purpose of the simulation, we chose $W = 1$ and the round length T to be 1 second [12].

QoS		
10^{-1}	10^{-2}	10^{-3}
7.3 x 10^{-3}	2.3 x 10^{-3}	5.7 x 10^{-4}

Table 1 : QoS provided by the overload control algorithm

5.1 Admission Control

Figure 2(a) demonstrates the effect of the desired loss probability on the achievable link utilization when varying number of sources are multiplexed. It demonstrates that whereas the achievable utilization is 56.1% when all sources require deterministic service, the utilization increases to as high as 88.6% when 72 sources requiring a loss probability of 10^{-4} are multiplexed. This demonstrates that a network can increase its utilization by 57.9% while providing a very good QoS. The figure also demonstrates that multiplexing higher number of sources leads to a higher utilization.

Figure 2(b) shows the error in approximating probability distribution function of aggregate bandwidth requirement of sources by the distribution of a normal random variable as the number of multiplexed sources vary. It demonstrates that the error decreases significantly as the number of sources increase. Moreover, the error is extremely small in the region of interest for admission control, i.e., at large values of the normalized random variable. Hence, normal approximation can be employed for admission control and a source can specify its traffic by just the mean and variance of the bit rate requirement.

5.2 Overload Control Algorithm

We employed numerous topologies for evaluating the overload control algorithm. One of the topology employed for experimentation is shown in Figure 3. In this topology, three sources were chosen from each of the sequence, with each of the source accessing a randomly selected part of a sequence. The bit rate for the links was chosen to be 33.25 Mb/s which corresponds to 85% utilization of the links between switches 2, 3, and 4. Fourteen sources were assigned packet loss probability of 10^{-2} at switches 3 and 4, three sources were assigned 10^{-3}, and one source was assigned 10^{-1}. The network was simulated for 400 seconds and overload was detected 37 times during the simulation interval. Our algorithm was always able to find an optimal solution in each of these occurrences. We found that QoS of all the sources was satisified; Table 1 summarizes the QoS received by the sources.

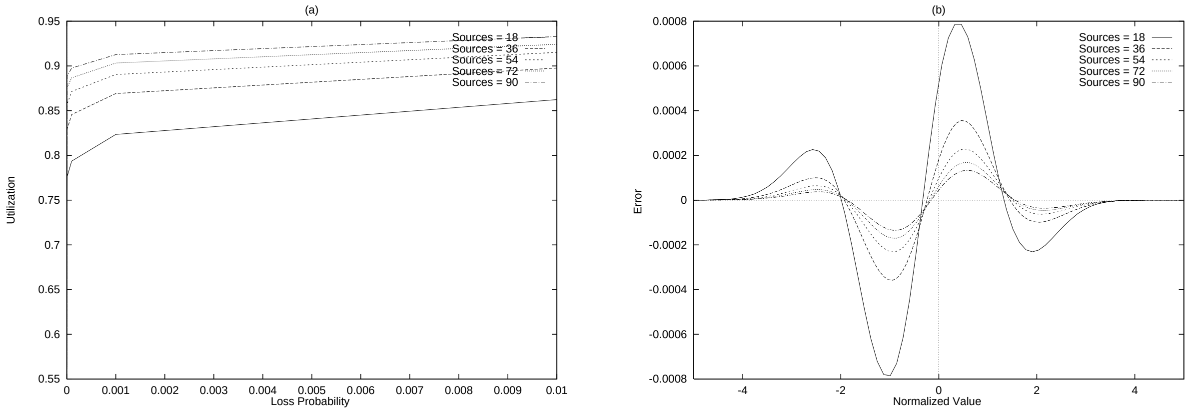

Figure 2 : (a) Effect of QoS on utilization (b) Error in normal approximation

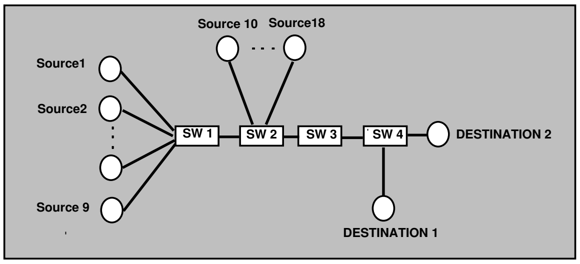

Figure 3 : A topology used for a simulation

To determine whether overload control protocol is really necessary, we simulated the previous configuration with the protocol disabled. In this scenario, the maximum queue length at switch 3 grew to 6280 packets. Furthermore, during many periods longer than 4 seconds, the queue occupancy remained more than 2000 packets. If buffer space availability was limited, this would have led to significant packet loss *in the network* and consequently significant degradation of video quality.

We had conjectured that FCFS scheduling algorithm combined with the overload control protocol is sufficient to avoid any overload at a link. This assumption would hold if the maximum queue length remains very small. Our experiments demonstrated that the maximum queue length at the link between switch 2 and switch 3 is always less then 9 packets. Similar results from many other experiments that we had conducted validate our conjecture.

5.3 Comparison with other schemes

It has been suggested in [9] that a histogram of frame sizes of stored video can be used to model the video traffic. Such a characterization of the traffic would not require any traffic shaping and have smaller end-to-end delay. However, when such a characterization is used by an admission control algorithm, the achievable utilization is lower. This is demonstrated in Figure 4(a). Moreover, when traffic shaping is not employed, the buffer requirement in the network increases significantly. To evaluate the increase in the buffer requirement, we utilized the network configuration depicted in Figure 3 under two scenarios: (1) sources are synchronized so that I frames of the sources are transmitted at the same time, and (2) no phase relationship exists among sources. Figure 4(b) plots the buffer requirement at switch 2 for both the scenarios at varying utilization. As it demonstrates, the maximum average buffer (average computed over 100ms) as well as the maximum buffer requirement is much higher than 9 packets that were required when traffic shaping was employed.

Recently, Renegotiated Constant Bit Rate (RCBR) has been independently proposed as a service for video traffic [6]. In RCBR, a source transmits at a constant rate and explicitly requests the network to assign a new rate when the data rate of the source deviates significantly from the current assigned rate. The network assigns the minimum of the desired and available rate. Though the concept of allocating variable rate are the same in RCBR and our protocol, a key difference is that RCBR does not have any mechanism to provide heterogeneous QoS (see [4] for a detailed comparison). In RCBR, the network just allows a source to renegotiate a rate. It is the responsibility of a source to demand higher bandwidth than it may actually require to decrease the probability of renegotiation failure, and thereby receive better QoS. If a source requires deterministic QoS, it has to reserve the peak bandwidth. The network can not allocate the unutilized portion of the reserved bandwidth and hence a link may be underutilized. This is in contrast to our approach where the network allocates bandwidth as per the desired QoS and ensures that a link is maximally utilized in presence of sources requiring heterogeneous QoS.

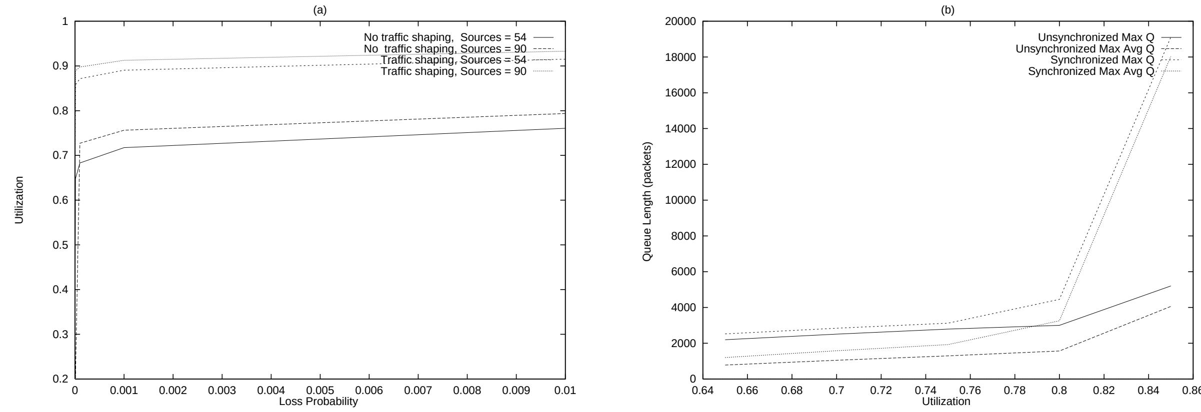

Figure 4 : Comparison of achievable utilization and buffer requirement in the network

6 Concluding Remarks

In this paper, we have presented a network service specifically designed for multimedia servers. It combines a histogram-based traffic characterization with an admission control algorithm, and provides heterogeneous statistical QoS guarantees to clients. We presented an overload control algorithm and protocol that exploits the regularity and predictability in traffic to meet these guarantees while completely eliminating packet losses in the network. The key contribution of our protocol lies in combining open-loop and feedback-based control to: (1) provide heterogeneous QoS to clients in networking environments consisting of switches that may not have any scheduling support; and (2) migrate the functionality of discarding packets, in the event of congestion, to the sources which understand the semantics of the data. The protocol is efficient, makes very few assumptions about the underlying network, is realizable in current hardware (supporting FCFS scheduling), and is completely integrated with the architecture of a multimedia server. This protocol is currently being implemented in the multimedia file server being developed at Distributed Multimedia Computing Laboratory at UT Austin.

REFERENCES

[1] T. Chiang and D. Anastassiou. Heirarchichal Coding of Digital Television. *IEEE Communications*, 32(4):38–45, May 1994.

[2] D. Ferrari and D. C. 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.

[3] M. Garrett and W. Willinger. Analysis, Modelling and Generation of Self-Similar VBR Video Traffic. In *ACM SIGCOMM 94*, 1994.

[4] P. Goyal and H. M. Vin. Network Algorithms and Protocol for Multimedia servers. Technical Report TR-95-19, The University of Texas at Austin, July 1995. Available via URL http://www.cs.utexas.edu/users/dmcl.

[5] M. Grossglauser and S. Keshav. Performance of Constant Bit Rate Traffic in Wide Area Networks. Technical report, ATT Bell Laboratories, January 1995.

[6] M. Grossglauser, S. Keshav, and D. Tse. RCBR: A Simple and Efficient Service for Multiple Time-Scale Traffic. In *Proceedings of ACM SIGCOMM'95*, 1995.

[7] S.S. Lam, S. Chow, and D.K.Y. Yau. An Algorithm for Lossless Smoothing of MPEG Video. In *Proceedings of ACM SIGCOMM'94, London*, 1994.

[8] P. Pancha and M. E. Zarki. MPEG Coding for Variable Bit Rate Video Transmission. *IEEE Communications Magazine*, pages 54–66, May 1994.

[9] P. Skelly, S. Dixit, and M. Schwartz. A Histogram-Based Model for Video Traffic Behaviour in an ATM Network Node with an Application to Congestion Control. In *Proceedings of IEEE INFOCOM*, pages 95–104, May 1992.

[10] C.J. Turner and L.L. Peterson. Image Transfer: An End to End Design. In *Proceedings of ACM SIGCOMM'92, Baltimore*, pages 258–268, August 1992.

[11] G.A Veciana. *Design Issues in ATM Networks: Traffic Shaping and Congestion Control*. PhD thesis, University of California at Berkeley, 1993.

[12] H. M. Vin, P. Goyal, A. Goyal, and A. Goyal. A Statistical Admission Control Algorithm for Multimedia Servers. In *Proceedings of the ACM Multimedia'94, San Francisco*, pages 33–40, October 1994.