

# *Congestion Control for Interactive Video-on-Demand over ATM*

An interactive video-on-demand (IVoD) system consists of a video server and a client connected together through a high speed network. Use of interactive video-on-demand in an educational environment is very interactive because students frequently pause, fast-forward, skip and view frame-by-frame. Designing a network able to support interactivity and make efficient use of bandwidth is challenging. In this paper we evaluate the suitability of a number of congestion control schemes (based on the service categories of ATM) for carrying interactive video on demand. We have developed analytical models to evaluate and compare the various schemes. Data on interactive behaviour of students obtained from IVoD trails carried out in our laboratory have been used in the models. We show that the renegotiated constant bit rate service category has considerable potential for carrying IVoD over a high speed network, such as the asynchronous transfer mode (ATM). We also show the use of an application layer, (such as Winsock 2), by the server to limit congestion within the network.

© 1999 Academic Press

**Philip Branch<sup>1</sup>, Mohammed Atiquzzaman<sup>2</sup>**

<sup>1</sup>*Network Systems Performance and Applications Group, Department of E&CS Engineering, Monash University, Clayton, Melbourne, Victoria 3168, Australia.*

*E-mail: philip.branch@eng.monash.edu.au*

<sup>2</sup>*Department of Electrical and Computer Engineering, University of Dayton, Dayton, Ohio 45469-0226, USA.*

*E-mail: atiq@engr.udayton.edu*

## **Introduction**

Interactive video-on-demand is a useful technology in education. With video-on-demand students can review lectures, watch a documentary or examine films, plays or television. However, usage is very interactive in such an environment. Students will often skip to the point of interest, review it a number of times and then exit. It is unusual for a video sequence to be viewed from beginning to end without some interaction. Interactivity in entertainment where the video is usually viewed from beginning to end without interruption, is small. Inter-

active functions are however, heavily used in education and research [1,2].

Designing interactive video on demand systems present many challenging research issues, such as networking requirements, congestion control schemes, client buffering requirements etc. Interactive behavior in interactive-video-on-demand systems is thus an important are networking application research. Ledistchke and Johnson [3] carried out a review of techniques for entertainment systems. Li *et al.* [4] investigated the performance of large scale interactive video-on-demand

servers. Dey-Sircar *et al.* [5] suggest queuing of bandwidth requests within the server. Andersen [6] described producing subsidiary encodings for fast-forward and fast-rewind.

Unless the network is able to support different bandwidths for different interactive functions, some functions, such as fast-forward and fast rewind are very difficult to provide. Difficulties are caused by the nature of MPEG encoding. With MPEG encoding, an increase in fast-forward rate of  $n$  cannot be achieved by simply sending every  $n$ th frame. MPEG gains much of its compressions through temporal encoding which creates dependencies between frames. In addition, MPEG pictures are encoded in different modes. A group of pictures (GOP) in MPEG consists of one I frame followed by a number of B and P frames. An I frame, with no temporal dependencies, contains more data than B or P frames. So sending only I frames without increasing the bandwidth will result in only a small speedup. It is possible to construct fast-forward by simply sending I frames but to be effective, this requires several times the bandwidth than for the play mode.

In simple low cost systems fast-forward or fast-rewind are not provided, rather a slider bar to provide random access to the scene of interest is used. Unfortunately, fast-forward and fast-rewind are useful in identifying where scenes of interest start and are essential for education use [1].

The solution adopted in many high end systems is to produce a separate encoded stream for fast-forward at a similar bit rate to that of normal play [6]. Unfortunately, this scheme requires around 20% additional server storage space. However, in educational environments storage is a scarcer resource than bandwidth, therefore, scheme is unsatisfactory. Furthermore, it does not lend itself to new modes of viewing such as high resolution zoom. Storing a separate encoding for every viewing mode, and possibly every variation of every mode is unacceptable.

Another scheme which does not require additional storage is to allocate a virtual circuit at  $n$  times the ordinary rate and allow the client to selectively display every  $n$ th picture. This is wasteful of bandwidth. It has the advantage of being simple to implement, and is a reasonable solution in networks with an abundance of bandwidth [3]. Unfortunately, it is not scalable to wide area networks where bandwidth is scarce and expensive.

To design a system that minimizes bandwidth and server space, both the network and server must be considered. Modern networks such as ATM offer considerable flexibility in delivery of video on demand [7–10]. We can take advantage of flexibility in both the server and the network to support more clients than we could otherwise. However, to do this, it is necessary for the application to communicate quality of service information to the network. Until recently applications had to assume best effort service or take advantage of networks able to support quality of service (such as ATM) by using proprietary programming interfaces for ATM network interface cards. Recently, the Winsock 2 programming interface was developed by a consortium of vendors to allow application developers to take advantage of ATM [11]. Winsock 2 allows an application to specify a Quality of Service to the underlying network, and then police it. Winsock 2 is likely to be popular for applications development in the near future. The interaction of application layer interfaces, such as Winsock 2, with underlying network technologies is an emerging and importance area of network research.

For video-on-demand to be affordable, storage and bandwidth must be utilized efficiently. The *objective* of this paper is to evaluate schemes that will better use these scarce resources. We begin with a discussion of user interactivity, and present statistics on the nature of interactivity obtained from wide area video-on-demand system trails carried out over the last two years in our laboratory. We model and evaluate the performance of a number of congestion control schemes based on the different ATM categories of service. Finally, we analyse the effectiveness of the policing mechanism of Winsock 2 in an interactive video-on-demand environment.

The rest of the paper is structured as follows. In “Video-on-Demand System Model” we describe the video-on-demand system used in our work. It is very similar to an actual system developed as apart of our video-on-demand trails at Monash University. “Interactive Behavior of Users” and “Source Model” describe the interactive behavior of users and a source model to capture the user interactivity along with some preliminary statistics relating to the interactivity in a video-on-demand system used in education. In “Suitability of ATM Service Categories for IVoD” we propose schemes for carrying interactive video-on-demand on various ATM categories of service. We develop analytical models of the proposed schemes based on different

categories of service and compare the performance of the schemes. In “Effect of Winsock 2 on Service Traffic”, we show techniques to control the impact of extremes of interactive behavior on the schemes using an application layer such as Winsock 2. Our concluding remarks are presented in the conclusion.

### Video-on-Demand System Model

We consider an interactive video-on-demand (IVoD) system where the client behavior is highly interactive in nature. In addition to play, clients frequently pause, fast-forward and fast-rewind. The video clients and the server are connected together using a wide area ATM network (see Figure 1). The edge switches connect to the wide area ATM network. Switched virtual circuits the video traffic from the server to the clients. Video encoded at a constant rate is used for play. Fast-forward (FFR) and fast-rewind (FRW) are implemented by sending only I frames at the maximum rate supported by the server. We make the following assumptions regarding the IVoD system discussed in this paper.

Video on demand exhibits different statistical behavior depending on the time it is examined. At each time scale, resources are provided differently. At the *finest* time scale of the cell, it appears to be periodic, with large I frames followed by smaller P frames and smallest B frames. In addition, small buffers within the switch, server and client absorb the small time scale variation of traffic to allow a smooth playout of video. At the *coarsest* time scale, one can see session commencements and terminations. At this time scale, the traffic rate

variations are absorbed through limiting session commencements. Our *interest* is in behavior or a system at the *intermediate* timescale (typically seconds to minutes) which capture user interactions such as pause, play FFW, FRW, etc. At this time scale, bandwidth provisioning is the key technique used to deal with traffic variations introduced by interactive behavior of users.

It is assumed that used interactivity exhibits Markovian behavior. Statistics from our video-on-demand trails appear to back up this assumption [12]. We only model video systems using constant rate encoded video.

### Interactive Behavior of Users

The IVoD system described in the previous section has been used in an IVoD trial at Monash University. The server and clients were connected using the Experimental Broadband Network (Telstra Australia’s experimental wide area ATM network connecting the major cities of Australia). The trial involved students from the Visual Arts department who used the system in their academic courses for analysis and criticism of video films. The MPEG encoded films were stored in the server, and the students accessed the video films over the ATM network.

As part of the trial, we have collected data on the interactive behavior of the users with a view to developing statistical models of users interactivity. The models are applied in this paper to determine the suitability of the various ATM service categories in carrying IVoD. The models are also suitable for describing the performance of systems at the design phase (see “suitability of ATM Service Categories for IVoD”).

Analysis of the collected data shows that the time spent by a user in the play mode can be approximated by an exponential distribution with mean of 170 seconds. Time spent in fast-forward/rewind and pause are also exponentially distributed and have means of 20 seconds and 50 seconds respectively [13]. We use this information in the next section. To develop a three state Markov Chain model (Figure 2) to describe the interactive behaviour of users. In Figure 2, states 0,1 and 2 represent the user in pause, play and FFW/FRW modes. We denote the bandwidth required in states 0,1 and 2 by  $r_0$ ,  $r_1$  and  $r_2$  respectively.

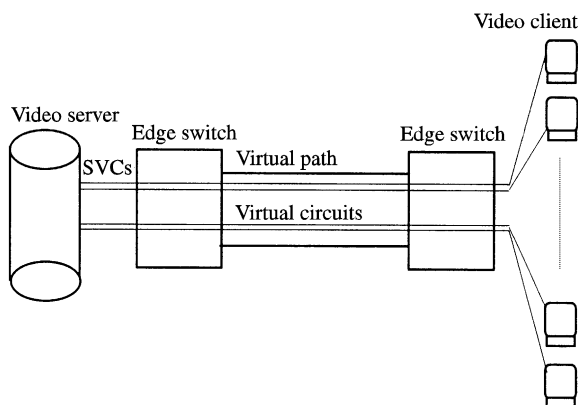
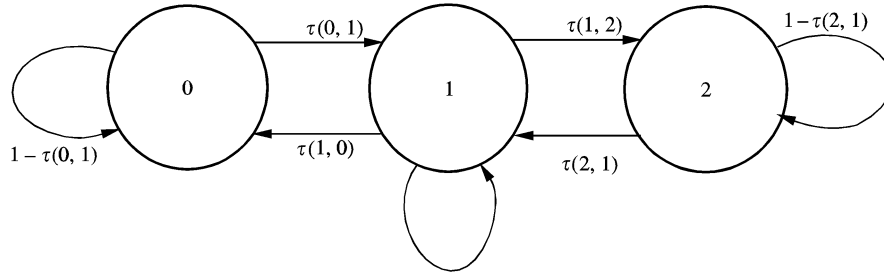


Figure 1. A typical video-on-demand system.



**Figure 2.** Transition diagram for single client.

We have found from subjective tests that for visual arts students studying video material the minimum acceptable level of quality is MPEG encoding at 2.0 Mbps. When header information is included, this encoding rate corresponds to a transmission rate of 2.5 Mbps. The maximum rate supported by the server in our testbed for an individual video stream is 5 Mbps. Consequently, we use the above two values as representative data rates in our subsequent analysis, i.e.  $r_0 = 0$ ,  $r_1 = 2.5$  Mbps and  $r_2 = 5$  Mbps.

### Source Model

In this section we develop a *source model* to represent the interactive behavior of users. The behaviour of a user at any instant of time determines the bit rates of the virtual circuit connecting the user to the server. We use the source model as the basis of our analysis to determine the suitability of the ATM service categories to transport IVoD in the following section and to investigate the effect of the Winsock 2 layer on individual VCs in “Effect of Winsock 2 on Source Traffic”. It is also used to study the effect of the ATM service categories on the aggregate traffic resulting from multiplexing of the VCs at the server.

A traffic source, representing the bit stream of a user, is modelled as a three-state process (Figure 2) representing *pause*, *play* and *fast-forward/rewind*. The transitions between the states are governed by the behavior of the client. Each state results in a different bit rate from the server to the network. Based on our measurement of user behavior (see previous section) we model the duration of each state as Markovian; i.e. the transition probabilities between the states are constant and depend only on the current state. The transition probability from state  $i$  to state  $j$  is represented by  $\tau(i, j)$ ,  $0 \leq i, j \leq 2$ .

The *source transition probability matrix*  $\mathbf{Q}$ , derived directly from the above diagram, is given by:

$$\mathbf{Q} = \begin{pmatrix} 1 - \tau(0, 1) & \tau(0, 1) & 0 \\ \tau(1, 0) & 1 - \tau(1, 0) - \tau(1, 2) & \tau(1, 2) \\ 0 & \tau(2, 1) & 1 - \tau(2, 1) \end{pmatrix}. \quad (1)$$

As part of our video-on-demand trail, we made available to undergraduate students some of the video material being studied by them as part of their courses through a video-on-demand system. Statistics as to user behavior were collected over a two week period during the busiest time of student usage of video. From the statistics, we developed the above model the derived source transition probabilities. The measured transition probabilities are given in Table 1 and the mean times spent in each state are shown in Table 2. The values in Table 2 are derived from the matrix  $\mathbf{Q}$  through  $t_i = 1/(1 - q_{ii})$ . From Table 1 and Table 2, we obtain the probabilities of a client being in the different states as shown in Table 3.

The ATM Forum has defined five categories of service for ATM: Constant Bit Rate (CBR), Variable

**Table 1.** Approximate transition probabilities

$\tau(0,1)$	0.050
$\tau(2,1)$	0.020
$\tau(1,2)$	0.003
$\tau(1,0)$	0.003

**Table 2.** Mean times in each state

State	Duration (seconds)
$t_0$	50
$t_1$	170
$t_2$	20

**Table 3.** State probability

State	Probability
$p_0$	0.208
$p_1$	0.709
$p_2$	0.083

Bit Rate—real-time (VBR-rt), Variable Bit Rate—non real-time (VBR-nrt), Available Bit Rate (ABR) and Unspecified Bit Rate (UBR). In this section we evaluate the suitability of the service categories for running IVoD.

CBR is intended for low latency, low loss and high quality real-time services. A fixed amount of bandwidth is allocated during connection set up and applications can continuously transmit at that rate throughout the duration of connection. VBR-rt provides a similar high quality service, but is intended for the transfer of variable bit rate encoded video. Video can be encoded at either constant quality or constant rate. When encoded at constant quality, the rate will fluctuate widely over short time scales. VBR-rt is intended to allow these variations onto the network, within defined delay variations, such as those introduced by queueing within switches. VBR-nrt also allows fluctuations over short time scales, but has less stringent delay variation requirements than VBR-rt. Examples of traffic suitable for VBR-nrt include CAD/CAM and distributed super-computer applications. VBR connections are accepted by switches based the switch's ability to meet the requested quality of service, defined by Peak Cell Rate, Sustainable Cell Rate, Maximum Burst Size and Cell Delay and Cell Delay Variation (for VBR-rt).

ABR makes use of excess bandwidth in the network for the transport of traffic which is not time critical and can tolerate variations in the bandwidth available from the network. Examples include file transfer and web access. ABR attempts to increase the utilization of the network bandwidth using complex feedback control mechanisms. ABR provides some service guarantees such as a Minimum Cell Rate.

UBR is intended for applications which do not require any guarantees. An examples is TCP/IP traffic, where losses can be remedied by retransmissions.

In the above paragraphs, we have discussed the various ATM service categories and their suitability for various applications such as circuit emulation, data

traffic, Web traffic etc. In the subsections to follow, we will discuss the suitability of the ATM service categories for carrying IVoD. The suitability depends on a number of competing priorities. First of all, not all categories of service are supported in all networks. For example, many carriers and equipment vendors do not support ABR. The second consideration is the quality of service required by the IVoD service. Generally, higher quality services will cost more. Therefore, most clients are likely to opt for the lower quality of service that is adequate for the IVoD service.

#### *Constant bit rate*

In this subsection, we look at two bandwidth allocation schemes for running IVoD over a CBR service.

#### *Peak rate allocation*

CBR circuits are allocated a fixed amount of bandwidth. In the case of IVoD, the allocated bandwidth is the peak rate required for the different viewing modes. For usage where the peak rate is  $P$  and the average rate is  $A$  the bandwidth utilisation is  $A/P$ .

For our system where the peak rate is used when in FFW/FRW mode,  $P$  is 5 Mbps and  $A$  is 2.2 Mbps (derived from Table 3). This results in a low bandwidth utilisation of only  $2.2/5=0.44$ .

#### *Renegotiated constant bit rate*

The Renegotiated Constant Bit Rate service has been proposed as a suitable service for the carriage of variable rate encoded video [14]. In this section we investigate the potential of RCBR for interactive video-on-demand. In the Renegotiated Constant Bit Rate service, when a client requests additional bandwidth from the network, the request is queued within the server until bandwidth becomes available. Once available, a CBR connection is opened. Clients give up the bandwidth they already have when their request is queued. We believe that this scheme has great potential, since it has the advantages of CBR, but without the wasted bandwidth.

In this scheme:

- All clients share a fixed amount of bandwidth over a virtual path.
- Fast forward is provided by transmitting only I frames at a higher rate over a constant bit rate (CBR) channel.
- When a client requests either fast-forward or play, the bandwidth held by the client is released and the

request for additional bandwidth is queued within the server. The connection is kept open, but service to the client ceases until bandwidth becomes available. Other information such as location within the video is not lost.

- If bandwidth is available, a higher rate channel is opened and the I frames are transmitted.

If bandwidth is not available, the request is queued within the server.

We develop an analytical model of this scheme by using a multi-server, finite population queue. Each 2.5 Mbps unit of bandwidth is modelled as a separate server while each request for bandwidth is treated as an arrival in the queue.

The multi-server, finite population queue is in a sense self-regulating. As the number of clients waiting for bandwidth increases, the rate of arrivals decrease. Because of this, multi-server, finite population systems tend to be quite robust under heavy loads.

If  $N$  is the number of clients,  $\rho$  is the arrival rate of requests for additional bandwidth divided by the rate at which they are met for a single client, and  $m$  is the number of units of 2.5 Mbps bandwidth and  $p_k$  is the probability of  $k$  requests in the queue, then the steady state probabilities of queue occupancy for a multi-server, finite population queue are from [15]:

$$P_k = \begin{cases} p_0 \rho^k \binom{2N}{k}, & 0 \leq k \leq m-1 \\ p_0 \rho^k \binom{2N}{k} \frac{k!}{m!} m^{m-k}, & m \leq k \leq 2N \end{cases} \quad (2)$$

and

$$p_0 = \left( \sum_{k=1}^{m-1} \binom{2N}{k} \rho^k + \sum_{m=1}^{m-1} \rho^k \frac{k!}{m!} m^{m-k} \right)^{-1}. \quad (3)$$

If  $1/\lambda$  is the mean time between requests for bandwidth and  $1/\mu$  the mean time between releases of bandwidth for an individual client, then we can calculate  $\rho$  in terms of the mean time spent in each state as follows:

$$\rho = \frac{\lambda}{\mu} = \frac{t_2 p_2 + t_1 p_1}{t_0 p_0 + t_1 p_1}. \quad (4)$$

As the bandwidth requests arrive and enter the queue, the arrival rate decreases. When there are  $i$  clients waiting for bandwidth, the aggregate arrival rate  $\lambda_i$  is:

$$\lambda_i = (m - i)\lambda. \quad (5)$$

Therefore the average aggregate arrival rate of bandwidth requests is:

$$\bar{\lambda} = \sum_{i=0}^{m-1} p_i \lambda (m - i). \quad (6)$$

We now apply Little's result to find the expected delay  $E(T)$  experienced by a client while waiting for bandwidth:

$$E(T) = \frac{E(n)}{\bar{\lambda}}. \quad (7)$$

Figure 3 shows the mean delay a client experiences while waiting for a fast forward request to be serviced in a small system of 10 clients sharing a 34 Mbps path. Given the inherent robustness of the multi-server, finite population queue, it is not surprising that the scheme performs very well even at very high loads.

It should be pointed out that we have made the simplifying assumption that arrivals are Poisson, whereas, for each stream they will only be approximately Poisson. Each request for a transition from state 1 to state 2 generates two arrivals (requests for bandwidth), while a transition from state 2 to state 1 releases two units of bandwidth. Also, the time between arrivals depends on whether the previous state was pause or view. However, the arrivals and service times will be dominated by the mean time in ordinary viewing mode, since it is much larger than the time in pause or fast-forward. So overall, we would expect Poissonian behavior to be a reasonable model, and our simulations show that this is so.

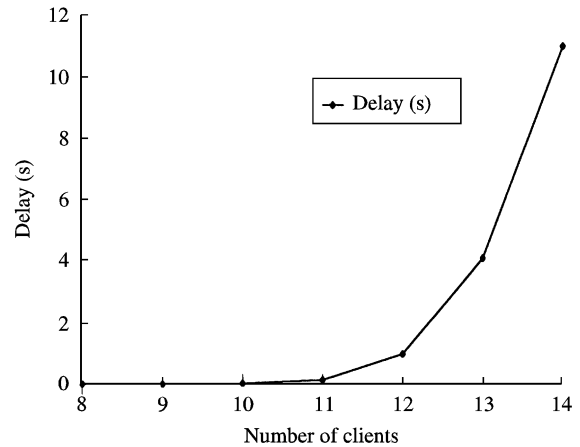


Figure 3. Delay vs. number of clients.

This scheme is appealing for a number of reasons. First, clients use CBR for carriage of video. For constant rate encoded video, CBR is the most appropriate category of service. This method has the advantages of CBR without the wastefulness of bandwidth in CBR. Second, degraded service is expressed as delay, rather than intermittent frame loss. This delay can be controlled by using the Winsock 2 mechanism, to ensure bandwidth is shared fairly amongst the clients and to limit the delay. Third, there is no queueing of video data within the server or the network. Video requires substantial space and large queues of it both add to queueing delay and consume scarce network and server buffer space. With this scheme, only requests for bandwidth are queued. Fourth, using the Winsock 2 mechanism, the system developer has great control over bandwidth allocation. Finally, the inherent robustness of the multi-server, finite population queue should provide robustness under heavy loads.

This scheme has a significant disadvantage in that it generates additional signalling traffic as connections are opened and closed. This could be ameliorated somewhat if a renegotiated constant bit rate category of service were defined. A connection would be opened for the duration of a session, but its parameters could be renegotiated with the network. Regardless of further developments with ATM categories of service, even with the existing CBR category of service, this scheme appears very promising.

#### Variable bit rate

Variable bit rate (VBR) is designed for the transmission of real-time, constant quality, variable rate video. Bandwidth fluctuations in VBR occur on a small time scale. It is assumed that several multiplexed, small time scale traffic will result in statistical multiplexing gain. Any bandwidth variations above the maximum rate will be able to be absorbed by a small amount of buffering within the switch. The buffering within the switch cannot be large since large buffers add to delay variation which causes performance degradation in real-time applications. Since buffering times within the switch are small when compared with the time scale of interactions, there is no distinction in between VBR-rt and VBR-nrt in our analysis.

For statistical multiplexing to be effective, traffic bandwidth fluctuations need to be large and rapid. With the traffic generated by user activity in an IVoD system, traffic fluctuations are comparatively small and slow.

Consequently, we might expect VBR to perform poorly for this application.

As a *measure of performance*, we will use the probability that the aggregate bit rate exceeds the bandwidth allocation to the virtual path. Since buffering within the switch is small compared to the timescale of interactions, we assume that aggregate bit rates exceeding that of the total capacity of the virtual path result in congestion.

The aggregate behavior of a number of clients each having a number of states is a multinomial combinatorial problem [16]. If we have  $N$  sources, each of which has states of 0,1 and 2, with probabilities of  $p_0, p_1$  and  $p_2$  respectively then the maximum output will be  $2N$  and the probability that the aggregate output will be  $n$  is:

$$Pr(n) = \left\{ \begin{array}{l} \sum_{i=0}^{Trunc(n/2)} M(N; i + (N - n), n - 2i, i) \\ \times p_0^{i+N-n} p_1^{n-2i} p_2^i, \quad n \leq N \\ \\ \sum_{i=0}^{Trunc(N-n/2)} M(N; i - N + n, 2N - n - 2i, i) \\ \times p_0^i p_1^{2N-n-2i} p_2^{i-N-n}, \quad n > N \end{array} \right\} \quad (8)$$

The probability of congestion occurring during any unit of time is the probability that the aggregate rate exceeds the rate of the output link. If the output link has a maximum capacity of  $c$  then the probability of congestion is:

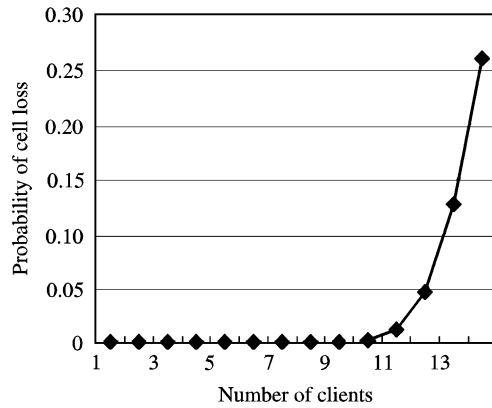
$$Pr(\text{congestion}) = \sum_{i=c}^{2N} Pr(i). \quad (9)$$

The following plot shows the performance as the number of clients ( $N$ ) increases for a system with an E3 connection of 34 Mbps. In this case,  $c$  is  $Round(34/2.5) = 14$ .

From the plot we see that the probability of cell loss is high when there are more than 13 clients. This scheme is probably only tolerable for 9 or less clients when the probability of cell loss is less than  $10^{-6}$ .

#### Unspecified bit rate

UBR is intended for carriage of data traffic and to allow unused cells to carry waiting data. It is unlikely that it could be used for video. Without any QoS guarantee, it



**Figure 4.** Probability of cell loss for VBR.

is unlikely that it could be used for the carriage of video. However, if a virtual path is allocated for UBR streams which are multiplexed across it, and cell delay and cell delay variation are absorbed by buffers within the client, the analysis and conclusion are the same as for the VBR case.

#### *Available bit rate*

In this section, we evaluate the performance of ABR (a rate based control scheme) for transport of video on demand traffic [17].

#### *Rate based control*

Rate based control schemes make use of feedback and large buffers within the network to control congestion. Traffic variation is absorbed by buffers that may in turn become congested. Once congestion is imminent, sources are informed that they must reduce their rate. The ABR category of service uses feedback to control congestion. Within the ABR specification, there are many options for implementing this. In this section, we consider the very simplest based on Backward Explicit Congestion Notification.

In our scheme, the server and the clients are connected through ABR connections. Wide area bandwidth is scarce. Therefore, we assume that the switch that connects the server to the wide area is the bottleneck. The switch has a large buffer that can absorb substantial amounts of video and release it onto the network when bandwidth becomes available. However, once the buffers fill to a trigger point, it informs the server that it must reduce its output. The server, on receiving the message, drops frames from the output streams until a message is received from the switch that it can resume

output. The rate of frame drop is the performance statistic in this scheme.

For this scheme to be useful, large amounts of video must be able to be buffered within the edge switch, which is usually undesirable because of the cost and because large switch buffers add to network delay. However, in ABR delay is usually not important and the cost of buffer memory is decreasing rapidly. Since ABR connections are likely to be much cheaper than CBR, it is worth considering the potential of ABR for carrying interactive video-on-demand.

The ABR control mechanism is a simple STOP/START scheme where a STOP message is sent to the source when the buffer occupancy reaches a high tide marker. The source stops transmitting data, and the buffer starts to empty. We assume a negligible propagation delay. Once the buffer reaches a low tide marker, a START message is transmitted to the server which starts video transmission.

The presence of large buffers within the switches will introduce large delay variations in the video received at the client. This implies that the client will need significant buffer space to allow continuous playout of video.

The amount of wide area bandwidth in ABR will fluctuate considerably as other applications open and close higher priority connections. Unfortunately, the nature and duration of these other connections may not be predictable. However, ABR does allow the specification of minimum cell rates and at the very least we can gain some insights into the suitability or otherwise of ABR by considering its behavior when constant wide area bandwidth is available.

#### *Analysis of rate based scheme*

We use Markov Chains to examine the performance of this scheme. We begin by defining two “super states”. Each state within the super state is buffer occupancy. Super state I is entered when the switch sends a START message to the server, allowing it to transmit video. During super state I, the buffer occupancy grows and shrinks as the client actions cause server traffic to vary. Super state II is entered when the buffer within the switch reaches a trigger point and sends a STOP message to the server. During super state II, the server drops video frames, while the buffer occupancy within the switch drops to a more acceptable level. We model the occupancy of the buffer with two Markov Chains, one for each super state.



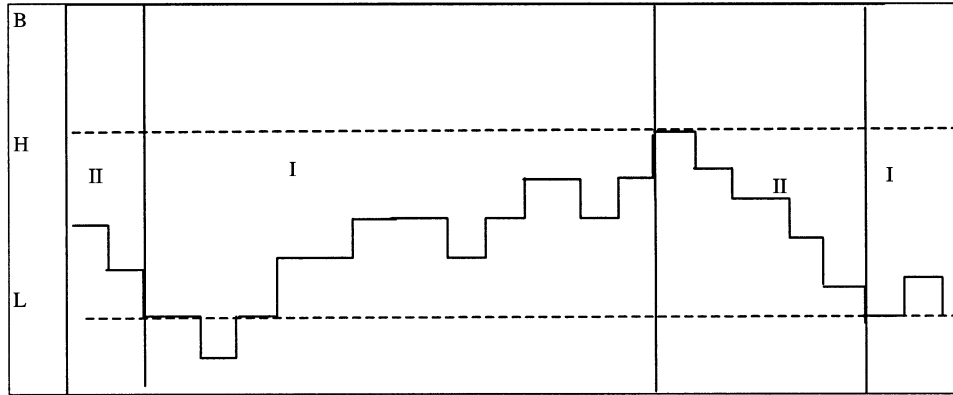


Figure 5. Buffer occupancy and super states.

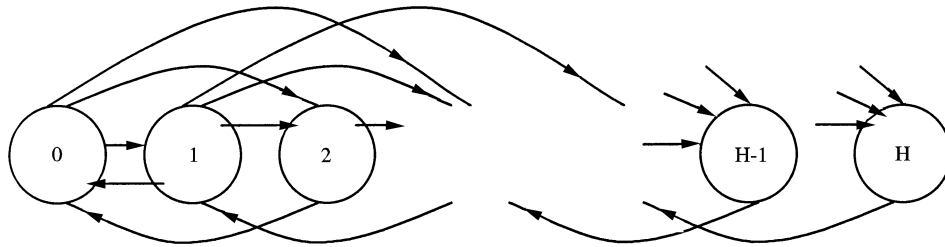


Figure 6. Markov Chain of buffer states in super state I.

We use our Markov Chain model to calculate the mean time spent in each of the superstates, and from this we can derive the mean frame loss probability within the server. The mean time spent in super state I, can be calculated by treating super state II as a single absorbing state, and all states in super state I as transient states.

We denote the transition matrix by:

$$\mathbf{P} = \begin{pmatrix} 1 & \mathbf{o} \\ \mathbf{r} & \mathbf{Q} \end{pmatrix} \quad (10)$$

where  $\mathbf{Q}$  are the transition probabilities between occupancy 0 and  $H$ ,  $\mathbf{o}$  is the zero vector and  $\mathbf{r}$  is the vendor associated with transition from transient to absorption states. The values of  $\mathbf{Q}$  are obtained by considering the probability of a transition within the buffer.

In developing a Markov Chain model for the buffer we let  $m$  be the number of cells removed from the buffer each unit of time and  $N$  the number of clients. If the buffer is serviced at the rate of  $m$  frames per second, then the probability of a transition from  $i$  to  $i+k$  is given by the probability that  $k+m$  cells arrive.

For three state traffic as we have in our model, the probability of  $n$  arrivals from  $N$  clients has a multinomial distribution given by Eqn (8), from which we can construct the transition matrix  $\mathbf{P} = \{p_{ij}, 0 \leq i \leq H, 0 \leq j \leq H\}$

where

$$P_{ij} = \begin{cases} 0, & m+j > i > j-m \\ \Pr(m+j-i), & m+j \leq i \leq j-m \\ \sum_{k=0}^{m-i} \Pr(k), j=0, & j=0, 0 \leq i \leq m \\ \sum_{k=H-1}^m \Pr(k), & j=H, i \geq H-m \end{cases} \quad (11)$$

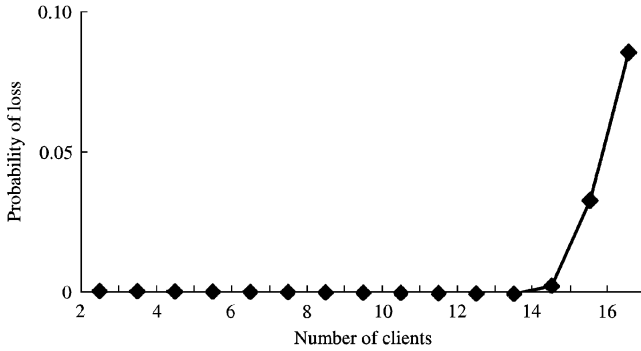
From  $\mathbf{P}$  we define the fundamental matrix [18] as

$$\mathbf{N} = (\mathbf{I} - \mathbf{Q})^{-1}. \quad (12)$$

If the initial probability vendor is  $\mathbf{p}$ , then the mean time until absorption is

$$t = \mathbf{pNe}. \quad (13)$$

Apart from initial startup time, when the buffer is empty, we know that super state I is only ever entered from super state II when the occupancy is  $L$ .



**Figure 7.** Probability of cell loss against number of clients for rate based control.

Consequently, the initial probability vector  $\mathbf{p}$  is a vector of  $H - L$  elements defined by

$$p_i = \left\{ \begin{array}{l} 1, i = L \\ 0, i \neq L \end{array} \right\}. \quad (14)$$

All frames generated by the server are lost when the system is in super state II. Frames are not lost at any other time. If we assume that the client requests are unaffected by whether the switch is in super state I or II, then the probability of any frame being lost is the probability of being in super state II.

$$\Pr(\text{loss}) = \frac{H - L}{\mathbf{p}\mathbf{N}\mathbf{e} + H - L}. \quad (15)$$

#### *Effect of buffer size on cell loss rate*

In this section we apply the analysis in the previous section to our system with 34 Mbps wide area connection to see how cell loss probability varies as the number of clients is increased.

The above chart shows the probability of cell loss for a large buffer of 3 Mbytes. The lower limit of  $L$  is 0. From the graph we can see that the rate based scheme can perform very well in keeping cell loss under control, provided a large buffer is available. However, once the number of clients reaches a threshold, performance declines rapidly. Also, a large buffer in the switch will cause large variations in cell delay experienced by the client, and so will require large buffers in the client to absorb this variation. We can summarize our analysis of ABR by saying that if buffer memory for both the client and the switch were to become very cheap, and if the cost of an ABR service were considerably less than that of a CBR service, it might be suitable for carriage of interactive video-on-demand traffic.

#### *Comparison of schemes*

CBR can be used to provide a very robust service. Its performance is measured in delay within the server until bandwidth becomes available to service a fast-forward or ordinary view request. It can be used with the Winsock 2 scheme to limit delays experienced by clients.

VBR is less well suited to the traffic generated by interactive video-on-demand. VBR is designed for traffic where the bandwidth fluctuates rapidly and widely, whereas interactive video-on-demand generates traffic which fluctuates over comparatively long time frames. The use of large buffers within VBR is not possible because it is intended for real time services. Consequently, VBR based scheme experiences considerable congestion.

UBR is a data service intended for non-time dependent traffic. The analysis and the conclusions are the same as for VBR.

ABR has potential, but requires large buffering within the network, which is currently very expensive.

Video-on-demand is unusual in that any of the categories of service can be used to carry it. Use of any particular category of service depends on the application, the relative cost of bandwidth in each category of service, buffer space available within the client, network and server, and the extent of user interactivity.

#### **Effect of Winsock 2 of Source Traffic**

The previous section has assumed that all clients behave in a consistent and fair manner. No client attempts to use more than the fair share of bandwidth. However, malicious users or equipment malfunctioning may make the above assumption impractical. In this section, we show how a policing mechanism such as that defined for Winsock 2 can be used to enforce long term behavior of video-on-demand clients [11].

Until recently, applications had to assume best effort service, or could take advantage of only those networks which are able to support quality of service (such as ATM) by using proprietary programming interfaces for ATM network interface cards. Recently, the Winsock 2 programming interface has been developed

by a consortium of vendors to allow application developers to take advantage of ATM. Winsock 2 allows an application to specify a Quality of Service to an underlying network, and then police it. Winsock 2 uses a token bucket mechanism to both define and enforce the traffic characteristics that the user transmits onto the network. In this section, we investigate the effect of the token bucket mechanism on user traffic.

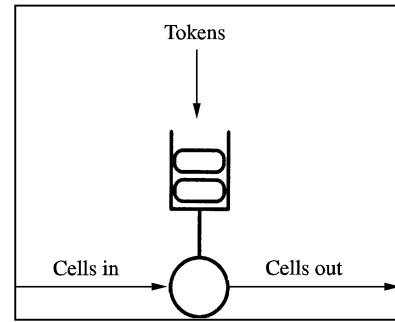
#### *Winsock 2 control overview*

In the previous section we saw how different categories of service can be used to support interactive video-on-demand. We now turn our attention to the server, and describe how Winsock 2, can be used to limit the effects of aberrant user behavior on the network.

Winsock 2 provides an application programming interface that has a mechanism for programming the underlying network. In particular it can communicate to the network quality of service information needed for the application. Winsock 2 uses the token bucket mechanism to specify and control the traffic streams that the application will generate. The Winsock 2 mechanism can be used to discard video frames within the server to reduce congestion within the network. Discarding at this point is preferable to discarding within the network. To minimize the effects on the client, the server can choose which video frames are dropped, and which client will be least affected by losing frames. Also, it is preferable to discard frames at the earliest point in the network, than to waste resources in transporting video only part of the way to its destination.

The token bucket is a simple mechanism to implement, and if implemented as we describe, is fair in its selection of frames to drop, since it will only drop frames that are in high rate modes of fast-forward or fast rewind.

The token bucket forces a video stream to conform to a traffic pattern. It limits the maximum burst size to the token bucket size and the sustainable rate to the token rate. Generally, the larger the bucket and the greater the token rate, the lower the lose rate. However, we are confronted with a trade-off between the server and network. A low frame loss rate in the server may result in degraded network performance, while infrequent congestion within the network may be at the cost of high rates of loss within the server.



**Figure 8.** Winsock 2 layer model.

#### *Analytical model of Winsock 2 and interactive behavior*

In this section we develop an analytical model of the effect of the Winsock 2 application interface on interactive video-on-demand traffic. Winsock 2 uses the Token Bucket scheme described in [11]. Tokens build up in the token bucket until they reach a maximum, beyond which tokens no longer accumulate. A token is removed when a unit of data (a frame, cell or packet) is transmitted onto the network. When the bucket is empty, and data waiting to be transmitted must wait for a token. The rate at which tokens build up depends on the long term maximum rate at which the source can transmit. The size of the token bucket is the maximum burst size that can be allowed onto the network in one unit of time. Previous analyses of token buckets have considered very bursty traffic on a small time-scale [19].

We model the effect of the token bucket on the video traffic by assigning a state to each rate/token pair as follows. First, we make a number of assumptions as to the order in which events task place during a slot (Figure 9).

- Token arrives at the beginning of a slot. If the token buckets is full, the token is lost.
- Packet arrives, removes a token (if one is available), and then departs. If there are no tokens, the packet is lost.
- If a rate change occurs, it happens at the end of the slot.

For the three state interactive traffic, we have observed on our system, the token bucket reduces the time spent in the higher rate and increased the time spent in the intermediate rate. We look at two statistics: the probability that during any unit of time the token bucket will force the rate to drop from rate 2 to rate 1, and the mean duration spent in rate 2.

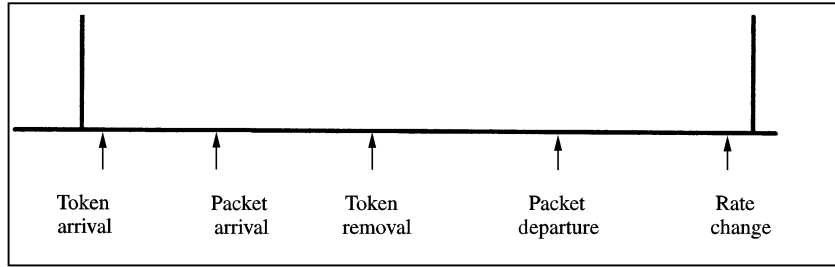


Figure 9. Event ordering.

We construct a Markov Chain to describe this system as follows. We assume a token arrives each unit of time. The frame time (one-thirtieth of a second) is an appropriate unit of time. When paused, tokens are unused so the token bucket fills at the rate of one token every thirtieth of a second. When viewing, the token bucket depletion rate matches the token arrival rate, so the token bucket is unchanged. When fast-forward or fast-rewinding, the token bucket starts to empty.

The Markov Chain describing the token bucket scheme consists of a separate state for each pair of possible token and rate values. Figure 10 shows a Markov Chain for a token bucket system with a maximum of three tokens, and a token rate of 2.5 Mbps.

We denote the state with rate  $i$  and token bucket occupancy  $j$  by  $s_{ij}$ . From the Markov Chain we construct a transition matrix  $P_1$  which consists of the transition probabilities between the different states as shown in Figure 10. We define rate  $i$  as the bit rate in state  $s_{ij}$  for all  $j$ .

The state  $s_{2,0}$  represents a request for rate 2, but with insufficient tokens to sustain that rate. When a source is in  $s_{2,0}$ , its rate will be 1, and not 2 as requested. Consequently the steady state probability of  $s_{2,0}$  gives us the probability that at any particular time, a source's rate will be reduced from 2 to 1.

To obtain the probability that the policing mechanism will drop cells, we solve the Eigensystem

$$\pi = P_1 \pi \tag{16}$$

where  $\pi$  is the steady state probability vector consisting of the states  $S_{ij}$ , and the probability of the policing mechanism dropping cells is given by the probability  $s_{2,0}$ .

We can also obtain the mean duration of a source remaining in rate 2, until it changes to rate 1, either of its own accord, or is forced to by the token bucket mechanism. We treat those states representing rate 2 as transient states and all other state (including  $s_{2,0}$ ) as a single absorbing state. We then find the mean time until absorption.

We modify  $P_1$  to obtain another transition matrix,  $P_2$ . A source eventually enters rate 1 which we treat as an absorbing state as in Eqn (10).

For this case,  $\mathbf{0}$  is the zero vector,  $Q$  represents the token rate pairs where the rate is 2 and  $\mathbf{r}$  is the vendor which takes the Markov Chain from transient states to the absorbing state. The fundamental matrix  $N$  is given by Eqn (12).

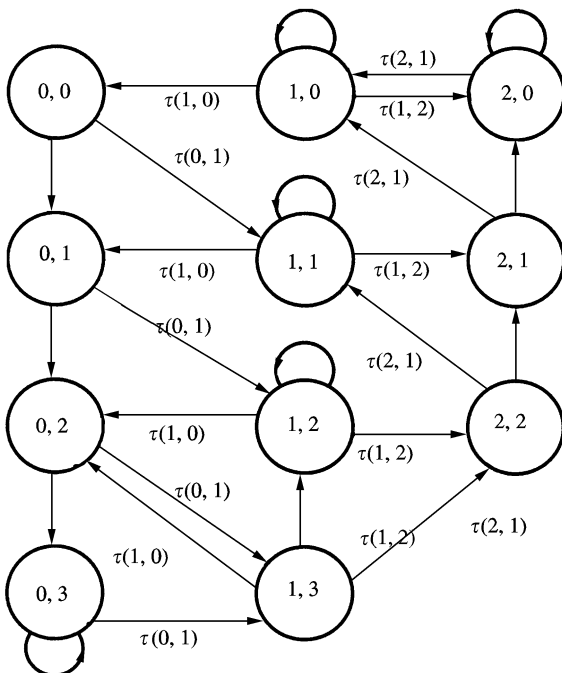


Figure 10. Markov Chain for token bucket and interactive traffic.

If  $\pi'$  is the initial probability of the transient states, and  $\mathbf{e}$  the unit vector, then the mean time spend in state 2 is:

$$t = \pi' N \mathbf{e}. \quad (17)$$

We obtain  $\pi'$  by observing that a source begins its entry into rate 2 from rate 1 (see Figure 10). Consequently, the initial probability vector consists of the state probabilities of the states where the rate is 1 is:

$$\pi' = \frac{\{s_{1,j}\}}{\sum S_{1,j}}. \quad (18)$$

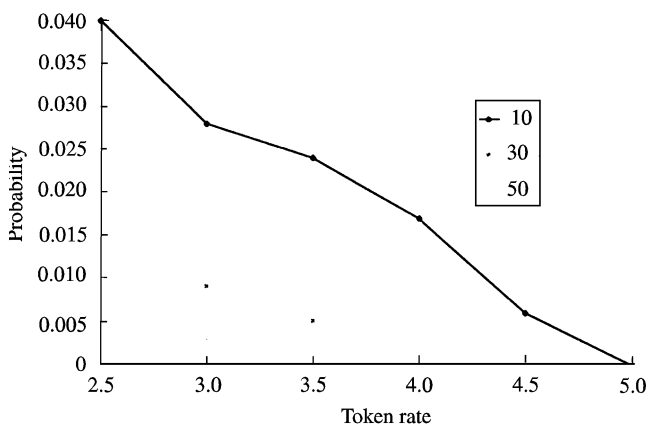
The next section uses this analysis to investigate the performance of the token bucket when it is used to control interactive traffic.

#### *Effect of token bucket size and token rate*

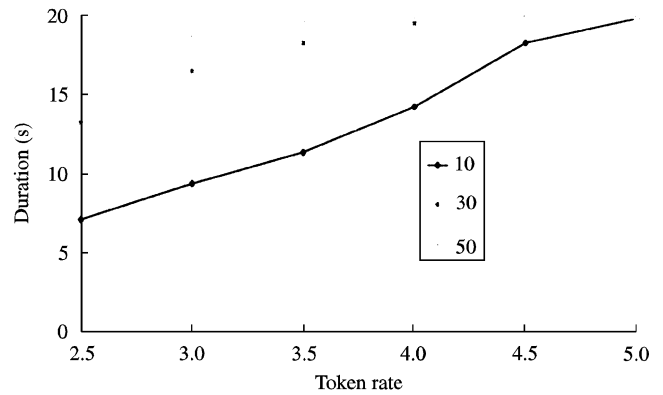
In this section we apply our analysis to a system where the play mode causes a bit rate request of 2.5 Mbps and fast-forward causes a bit rate request of 5.0 Mbps.

Figure 11 shows the probability of the Winsock 2 mechanism dropping an I frame during fast-forward/rewind. This is derived from the equations given in the previous section. The three graphs correspond to token bucket sizes of 10, 30 and 50 seconds of video at 2.5 Mbps. We find that large token buckets are needed to provide any significant effect on the source while increasing the token rate is effective in reducing the frame loss rate.

Another performance measure is the mean time until I frames start to be dropped while in fast-forward/rewind in rate 2.



**Figure 11.** Probability of frame drop vs. token rate.



**Figure 12.** Mean time in rate 2 vs. token rate.

Figure 12 shows the mean duration in rate 2 with token bucket sizes varying from 10 to 50 seconds.

Winsock 2 has two important functions that are of interest to the developer of an interactive video on demand system. The first is that it allows the application to specify (to the network) the characteristics of the traffic that is to be presented to the network. The second is that it forces the user to conform to those characteristics. This section has shown how the two token bucket parameters can be used to limit congestion within the network and allocate bandwidth fairly to clients.

## Conclusion

In this paper we have described how interactive traffic can be carried across an ATM network using the different categories of service. The suitability of a number of congestion control schemes for the transport of highly interactive video on demand over ATM have been evaluated. In order to carry out the above evaluation, we have developed a framework for representing the interactive nature of users in a video-on-demand systems (which is used in education) and a model to capture the interactivity of the users. The renegotiated CBR based scheme uses bandwidth most efficiently. If the cost of buffer space is small, ABR might also be used to carry interactive video-on-demand. Our evaluations indicate that VBR and UBR perform poorly in an interactive video-on-demand environment.

Finally, we have shown how the Winsock 2 application layer affects the underlying traffic. Winsock 2 can be used to provide limits to excessive behavior of clients

and to communicate to the network the nature of the traffic to be carried.

### Acknowledgements

The support of Telstra Australia, Siemens Ltd, Monash University and the Australian Government's Cooperative Research Centre Scheme in the preparation of this paper is gratefully acknowledged.

### References

1. Branch, P. & Durran, J. (1996) *PC Based Video on Demand Trials. Learning Technologies, Prospects and Pathways, Edtech'96*, Melbourne, July 1996.
2. Branch, P., Newstead, S. & Kaushik, R. (1996) *Design of a Wide Area, Video-on-Demand User Interface. Proceedings of Australian Telecommunication Networks and Applications Conference*, Melbourne, 3–6 December 1996, pp. 55–58.
3. Leditschke, M. & Johnson, A. (1995) *Implementation of MPEG-2 Trick Modes. Proceedings of Australian Telecommunications Networks and Applications Conference*, Sydney, Australia, 11–12 December 1995, pp. 39–44.
4. Li, V. O. K., Liao, W., Qiu, X. & Wong, E. W. M. (1996) Performance Model of Interactive Video-on-Demand Systems. *IEEE Journal on Selected Areas in Communications* **14**: 1099–1109.
5. Dey-Sircar, J. K., Salehi, J. D., Kurose, J. F. & Towsley, D. (1994) Provisioning VCR Capabilities in Large-Scale Video Servers. *Proceedings of ACM Multimedia 94*, San Francisco, California, USA, October 1994, pp. 25–32.
6. Andersen, D. B. (1996) A Proposed Method for Creating VCR Functions using MPEG Streams. *Proceedings of IEEE 12<sup>th</sup> International Conference on Data Engineering*, New Orleans, LA, USA, pp. 380–382.
7. The ATM Forum, *ATM User-Network Interface Specification, Version 3.1*, The ATM Forum, September 1994.
8. Zheng, B. & Atiquzzaman, M. (1998) Multimedia Over High Speed Networks: Reducing Network Requirements With Fast Buffer Fillup. *IEEE GLOBECOM'98*, November 8–12, Sydney, Australia.
9. Zheng, B. & Atiquzzaman, M. (1998) Multimedia over ATM: Progress, Status and Future, *IC3N: Seventh International Conference on Computer Communications and Networks*, October 12–15, 1998, Lafayette Louisiana, USA, pp. 114–121.
10. Zheng, B. & Atiquzzaman, M. (1998) Video on Demand over ATM: System Design and Networking Requirements. *ENCOM-98: Enterprise Networking and Computing'98*, June 7, 1998, Atlanta, Georgia.
11. Quinn, B & Shute, D. (1996) *Windows Sockets Network Programming*. Reading, Massachusetts, Addison-Wesley Publishing Company.
12. Branch, P. (1998) Bandwidth Allocation Schemes for Interactive Video-on-Demand. *IEEE International Symposium on Computers and Communications*, Athens, Greece, 1998.
13. Branch, P. & Atiquzzaman, M. (1997) ATM Category of Service for Interactive Video on Demand, *IEEE/IEE International Conference on Telecommunications*, Melbourne, April 1997.
14. Grossglauser, M., Keshav, S. & Tse, D., RCBP: A Simple and Efficient Service for Multiple Time-Scale Traffic. *IEEE/ACM Transactions on Networking*, **5**: 741–755.
15. Kleinrock, L. (1975) *Queueing Systems*, Vol 1, John Wiley and Sons Inc.
16. Brualdi, R. (1977) *Introductory Combinatorics*, New York: North Holland.
17. Branch, P., Atiquzzaman, M. (1996) Modelling Rate Based Congestion Control in an ATM Network Designed for Video Retrieval, *Proceedings of Australian Telecommunications Networks and Applications Conference*, Melbourne, 3–6 December 1996, pp. 55–58.
18. Kemeny, J. & Snell, J. (1960) *Finite Markov Chains*, Princeton, New Jersey: Van Nostrand.
19. Sen, P., Maglaris, B., Rikli, N. & Anasstassiou, D. (1989) Models for Packet Switching of Variable Bit Rate Video Sources. *IEEE Journal on Selected Areas of Communications*, **7**: 865–869.