Bandwidth Allocation and Admission Control Schemes for the Distribution of MPEG Streams in VOD Systems*

Marwan Krunz,† George Apostolopoulos,‡ and Satish K. Tripathi§

Abstract

Large-scale deployment and successful commercialization of digital video systems will strongly depend on their ability to provide cost-effective services to potential customers. Network bandwidth is one of the major factors that impact the cost of a video service. In this paper, we investigate efficient bandwidth allocation and admission control strategies for transporting MPEG-compressed video over a wide-area network. These strategies are based on exploiting the periodic nature of MPEG compression for the purpose of reducing the bandwidth requirements while maintaining a high level of quality of service. We derive a theoretical bound on the call blocking probability at the server, and show how this bound can be used in offline capacity planning and resource dimensioning. For online operation, we provide a simple CAC test that can be used in conjunction with the proposed bandwidth allocation and stream scheduling strategies. The effectiveness of our approach is evaluated through simulations of a video distribution network that consists of a server and several switches. Issues related to its practicality and implementation feasibility are discussed.

Keywords: bandwidth allocation, MPEG, VBR video, traffic envelope, statistical multiplexing.

1 Introduction

Broadband networks (e.g., B-ISDN/ATM) are expected to support a wide range of multimedia applications, such as video-on-demand (VOD), high definition TV (HDTV), and multimedia teleconferencing. These applications generate video streams that must be transported in a timely manner to ensure coherent reception and playback at the receiver. Video traffic imposes huge demand on network bandwidth, which has been a major hindrance facing the economic viability of digital video services over computer networks. Unless efficient approaches to bandwidth allocation are devised, the cost of digital video services will prevent their widespread acceptance among potential customers.

Invariably, the approaches used to transport compressed video over computer networks rely on one or more of the following techniques: (1) temporal averaging (or smoothing) on a stream-by-stream

*Part of this paper was presented at IEEE INFOCOM '97 Conference, Kobe, Japan. This work was supported in part by the National Science Foundation under Grants ANI-9733143 and CCR-9318933, and by a SUR IBM grant.

†Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ 85718. Tel. (520) 621-8731. krunz@ece.arizona.edu (Corresponding Author).

‡Department of Computer Science, University of Maryland, College Park, MD 20742.

§Bourns College of Engineering, University of California, Riverside, CA 92521.

basis (e.g., [8, 17, 21, 26, 27, 29, 11]); (2) batching (e.g., [1, 2, 3, 6, 10, 18]); and (3) spatial averaging (or aggregation) by means of statistical multiplexing (e.g., [32, 25, 20, 24]). Video smoothing is an attractive approach that has been used for both archived and real-time video. For archived video, smoothing is often used in conjunction with prefetching, whereby video frames are transported prior to their playback times. This so-called work-ahead approach, while attractively simple, has two limitations. First, it requires some knowledge of the end-to-end network delay to be used in constructing an "optimal" transmission schedule (under the constraints that buffer overflow and underflow at the receiver should not occur). In practice, the exact delay value is not known in advance, and an estimated bound on it is used instead. Unfortunately, network delay is quite variable, resulting in conservative delay estimates that reduce the effectiveness of smoothing. (Note that the estimated bound is usually deterministic, so it has to account for the worst-case delay.) The second limitation of smoothing is that its effectiveness depends on the allowable buffer buildup delay; the larger the delay the more effective is the smoothing algorithm. A large buildup delay (e.g., tens of seconds) may not be an issue for playback-only services, but is not acceptable for interactive operations. For example, the client may request a 'jump' during the playback of a movie. If the jump is to a point that is far away from the previous playback point, then the first frame from the new playback point will not be available in the client buffer, and it must be prefetched from the server.

Video batching is yet another popular approach to distributing archived video. It relies on multicasting to distribute a particular video to several clients. This approach is naturally applicable to broadcast TV (e.g., HDTV). For VOD the efficiency of batching depends on the number of simultaneous requests for a particular movie. One major drawback of batching in VOD systems is the difficulty to support interactive VCR-like operations. To remedy this problem, some researchers suggested using a combination of work-ahead and batching approaches [2]. Others suggested serving each interactive request by a dedicated (i.e., unicast) stream for the duration of the interactive request. Once the interactive operation is completed, the client is served again by one of the ongoing batches ("split-and-merge" approach [19]). Other approaches for interactivity in VOD systems are discussed in [4, 5, 20, 23, 28, 30].

In statistical multiplexing (**SM**) bandwidth gain is attained by allowing the sum of the peak rates of input streams to exceed the output rate of the multiplexer. This results in packet delay and potential buffer overflow, the amounts of which are determined by evaluating the queueing performance under a given traffic model. Thus far, SM has been investigated within a stochastic framework, where the traffic is modeled using stochastic models or, more recently, *time-invariant* deterministic enevelopes [7, 12]. Both modeling techniques give rise to statistical quality-of-service (QoS) guarantees.

The current paper is motivated by the need to provide *stringent, deterministic QoS guarantees* for video traffic without having to allocate bandwidth based on the peak rates of the individual sources. More specifically, we focus on efficient bandwidth allocation for transporting MPEG-coded variable-bit-rate (VBR) video streams. The broad acceptance of MPEG compression (in its various versions) provides strong justification for investigating MPEG-specific bandwidth allocation schemes. To provide deterministic QoS guarantees, we consider VBR-coded streams that are transported using a CBR network service (i.e., constant-bandwidth channel). While other network services can also be used (e.g., VBR, ABR), it is unlikely that such services will be able to provide stringent deterministic QoS guarantees. We argue that such guarantees can be provided through SM whose gain is determined under *time-varying* deterministic traffic envelopes[1]. Our approach is based on exploiting the periodic structure of the Group-of-Pictures (GOP) pattern in MPEG video to provide appropriate traffic envelopes. Based on such envelopes, we have formulated a framework for stream scheduling, multiplexing, and bandwidth allocation at a video server [16, 15, 31]. An important aspect of this framework is that the achievable bandwidth gain depends on the relative synchronization of video sources with respect to their GOPs, which implies the potential for call blocking of new requests (assuming a video distribution network with fixed-capacity bandwidth channels). Accordingly, in the present work we focus on the admission control and capacity planning aspects of our framework. We derive an upper bound on the nominal call blocking probability at the server. Using this bound, we show how capacity planning in a video distribution network can be performed while satisfying a given blocking probability. For online operation, we provide a simple admission control test that can be used in conjunction with our bandwidth allocation approach. Simulation of a video distribution network that comprises of a server and several switches is used to study the effectiveness of our approach. Finally, we demonstrate the feasibility of the proposed approach by implementing it over an ATM testbed. It should be noted that our approach is applicable to all versions of MPEG video (i.e., MPEG-1, MPEG-2, MPEG-4, etc.) as long as the GOP pattern is generated in a repetitive fashion. For short, we will use the generic term 'MPEG' without specifying a particular version.

The remaining sections are structured as follows. Section 2 summarizes our previous work on scheduling and bandwidth allocation for MPEG-coded video streams. In Section 3 we discuss the practicality of our allocation scheme. Call admission control is addressed in Section 4. In Section 5 we derive the worst-case blocking probability and show how it can be used in dimensioning the bandwidth capacities of the virtual connections between the server and various head-end switches. Simulation results are presented in Section 6. Implementation issues are discussed in Section 7. The paper is concluded in Section 8.

2 Bandwidth Allocation for Multiplexed MPEG Streams

In this section we summarize the main aspects of our bandwidth allocation approach for distributing MPEG-coded video streams. More details can be found in [15, 31].

[1]'Statistical multiplexing' is somehow a misleading term since it really refers to *asynchronous* multiplexing. The reason behind the terminology is that the SM gain is typically specified in a statistical manner.

2.1 Traffic-Envelope Model

A standard MPEG encoder generates three types of compressed frames: Intra-coded (I), Predictive (P), and Bidirectional (B) frames. Different combinations of compression modes are used to encode these three types. On average, I frames are larger than P frames which, in turn, are larger than B frames. MPEG frames are often compressed according to a pre-defined GOP pattern that determines the frame types. An example of a GOP pattern is `'IBBPBBPBBPBB'`, which can be applied repeatedly in compressing the frames of a video stream. The specification of a given GOP pattern prior to compression is not mandatory, but is often used to simplify the codec design. The GOP pattern is typically *regular* in that the number of *successive* B frames within an MPEG-coded sequence is fixed. Regular GOP patterns can be specified by two parameters:

N : Number of frames between two successive I frames in an MPEG stream.

M : Number of frames between an I frame and the subsequent P frame in an MPEG stream.

The regularity of the GOP pattern implies that N is a multiple of M.

To provide deterministic guarantees, we model the ith video stream by a time-varying traffic envelope $\overline{b}_i(t)$, which is parameterized by the 5-tuple $\boldsymbol{E}_i = \left(I^{(i)}_{max},\ P^{(i)}_{max},\ B^{(i)}_{max},\ N^{(i)},\ M^{(i)}\right)$. Here, $I^{(i)}_{max}$ is the largest frame in the sequence (typically an I frame), $P^{(i)}_{max}$ is the largest frame among P and B frames (typically a P frame), and $B^{(i)}_{max}$ is the largest B frame. Accordingly, $I^{(i)}_{max} \geq P^{(i)}_{max} \geq B^{(i)}_{max}$ for all i. The parameters $N^{(i)}$ and $M^{(i)}$characterize the GOP pattern of the ith stream. Frames are transported at a constant frame rate which is equal to the playback rate at the receiver (the frame rate is the same for all streams). Thus, no buffering is required at the receiver. We assume that the bit rate over one frame period is uniform and is given in ATM cells/frame. Because of the regularity of the GOP, $\overline{b}_i(t)$ is periodic with period $N^{(i)}$. An example of $\overline{b}_i(t)$ is shown in Figure 1. It is also possible to define $\boldsymbol{E}_i$ for several segments of a video stream, which will result in tighter traffic envelopes [31]. However, this increases the computational complexity for updating the utilized bandwidth at the server.

2.2 Scheduling and Bandwidth Allocation Approach

Consider n video streams, $s_1, \ldots, s_n$, with end-to-end QoS requirements that consist of zero loss rate and small, bounded delay. Such stringent requirements are typically met by transporting each stream at its peak rate. For all i, s_i is modeled by a time-varying traffic envelope $\overline{b}_i(t)$, as described before. Suppose that the n streams are multiplexed onto a CBR bandwidth channel (e.g., an ATM virtual channel connection) that extends from the server to a head-end (HE) switch over a wide-area network. Let t_i be the starting time of s_i, i.e., the time at which the first frame of s_i arrives at the multiplexer. We let $t_1 \triangleq 0$ to be used as a reference. For $i = 1, 2, \ldots$ we define the *phase* of s_i by $u_i = t_i \bmod \widetilde{N}$,

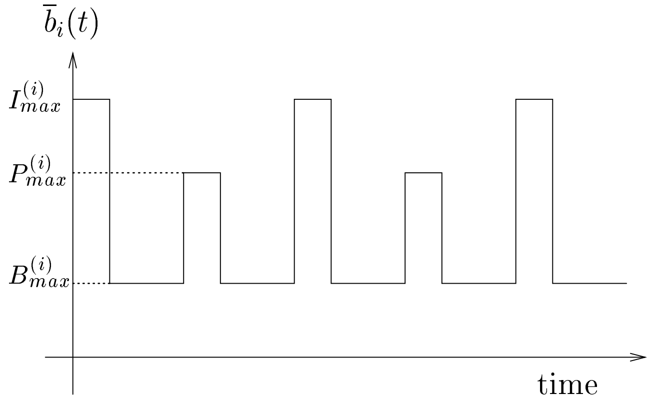

Figure 1: Traffic envelope with $N^{(i)} = 6$ and $M^{(i)} = 3$.

where

$$\widetilde{N} = \textit{least common multiple of} \{N^{(1)}, N^{(2)}, \ldots, N^{(n)}\} \tag{1}$$

When $N^{(i)} = N$ for all i (the homogeneous case), the phase of s_i represents the frame lag between a GOP of s_i and the most recent GOP of s_1. The vector $\boldsymbol{u} = (u_2, u_3, \ldots, u_n)$ is referred to as the *arrangement* of the n multiplexed streams ($u_1 \stackrel{\triangle}{=} 0$). This vector completely specifies the synchronization structure of the n streams with respect to their GOPs. Without loss of generality, we assume that the boundaries of the arriving frames from various sources are aligned in time, so that $u_2, u_3, \ldots$ take only integer values in $\{0, 1, \ldots, \widetilde{N} - 1\}$. This restriction is not crucial, but it significantly reduces the computational complexity of updating the allocated bandwidth. Let $\overline{b}_{tot}(t)$ be the *aggregate* traffic envelope resulting from the superposition of the n streams; $\overline{b}_{tot}(t) = \sum_i \overline{b}_i(t - u_i)$. Note that $\overline{b}_{tot}(t)$ is periodic with a period $\widetilde{N}$. Our bandwidth allocation strategy is based on the following quantity:

$$C(\boldsymbol{u}, n) \stackrel{\triangle}{=} \frac{1}{n} \max_{t \geq 0} \overline{b}_{tot}(t) = \frac{1}{n} \max_{t \geq 0} \left(\sum_{i=1}^{n} \overline{b}_i(t - u_i) \right), \tag{2}$$

which we refer to as the *per-stream allocated bandwidth* (**PSAB**). Since $nC(\boldsymbol{u}, n)$ is an upper bound on the peak rate of the aggregate traffic, allocating $C(\boldsymbol{u}, n)$ per stream ensures that the aggregate input rate over a frame period can never exceed the output rate. A small buffer of n packets is needed at the multiplexer to accommodate simultaneous packet arrivals from different streams. With this buffer, zero loss rate and a maximum delay of n/W frame periods are guaranteed, where W is the total capacity of the bandwidth channel; $W \geq nC(\boldsymbol{u}, n)$. Since $I^{(i)}_{max} \geq P^{(i)}_{max} \geq B^{(i)}_{max}$ for all i, it is easy to see that $nC(\boldsymbol{u}, n) \leq \sum_i I^{(i)}_{max}$, with the strict inequality holding for most values of $\boldsymbol{u}$. Only when $\boldsymbol{u}$ is the zero vector does $nC(\boldsymbol{u}, n) = \sum_i I^{(i)}_{max}$ (e.g., all streams overlap in their I frames).

Because of the periodicity of $\overline{b}_{tot}(t)$, it is sufficient to take the maximum in (2) over $\widetilde{N}$ successive frame intervals. Given that the boundaries of frames are aligned, $\overline{b}_{tot}(t)$ is completely specified by $\widetilde{N}$ values, which are updated at the server when a new stream is added or an ongoing stream is terminated. To update the bandwidth, the video server maintains a "bandwidth table" $\boldsymbol{A} = [a_{ij}]$ of n rows and $\widetilde{N}$ columns, and a vector $\boldsymbol{V} = [v_1\ v_2\ \cdots v_{\widetilde{N}}]$, where n the number of ongoing streams. For $i = 1, \ldots, n$, the ith row of $\boldsymbol{A}$ contains $\widetilde{N}/N^{(i)}$ copies of the $N^{(i)}$-tuple that describe the traffic envelope of the ith stream. For $j = 1, \ldots, \widetilde{N}$, v_j contains the sum of the jth column of the bandwidth table, i.e.,

$$v_j = \sum_{i=1}^{n} a_{ij} \tag{3}$$

It is obvious that

$$C(\boldsymbol{u}, n) = \frac{1}{n} \max_{1 \leq j \leq \widetilde{N}} v_j \tag{4}$$

The PSAB depends on the arrangement $\boldsymbol{u}$, which in turn depends on the starting times of the multiplexed streams. Thus, $\boldsymbol{u}$ can be optimized by allowing the server to control the starting times of new streams for the purpose of minimizing the PSAB. This type of stream scheduling comes at the expense of delaying the initiation of a new stream by a maximum of a GOP period (1/2 second). Let $C_{min}(n)$ be the minimum PSAB over all possible vectors $\boldsymbol{u}$ and let $\boldsymbol{u}^*$ be the optimal *arrangement* that results in $C_{min}(n)$:

$$C(\boldsymbol{u}^*, n) = C_{min}(n) \stackrel{\triangle}{=} \min_{u} C(\boldsymbol{u}, n) \tag{5}$$

For homogeneous multiplexed streams (i.e., traffic envelopes are identical except for a time shift), a closed-form expression for the optimal *arrangement* $\boldsymbol{u}^*$ was provided in [15]. However, no tractable expression is available for the optimal *arrangement* of heterogeneous streams. Instead, a computationally feasible suboptimal scheduling scheme, known as Minimal-Rate Phase (MRP), was provided for the heterogeneous streams case [15]. According to MRP scheduling, given n multiplexed streams, a new stream (the $(n+1)$th) is scheduled for multiplexing such that

$$u_{n+1} = k - 1 \text{ where } v_k = \min_{1 \leq j \leq \widetilde{N}} v_j \tag{6}$$

In other words, the new stream is scheduled in a phase for which the aggregate bit rate is minimal. The MRP policy is described in Figure 2.

It can be shown [31] that when MRP scheduling is applied *successively*, then after each scheduling step we have:

$$|v_j - v_i| \leq 2\overline{I}_{max}, \quad \text{for all } i \text{ and } j \tag{7}$$

where $\overline{I}_{max} = \max\{I_{max}^{(1)}, \ldots, I_{max}^{(n)}\}$. Let $C_{mrp}(n)$ be the PSAB resulting from MRP. Then it can be shown [31] that

$$C_{min}(n) \leq C_{mrp}(n) \leq C_{min}(n) + 2\overline{I}_{max}/n \tag{8}$$

Thus, $C_{mrp}(n)$ is no more than $2\overline{I}_{max}/n$ from the minimal possible PSAB. As $n \to \infty$, $C_{mrp}(n) \to C_{min}(n)$, i.e., $C_{mrp}(n)$ is asymptotically optimal.

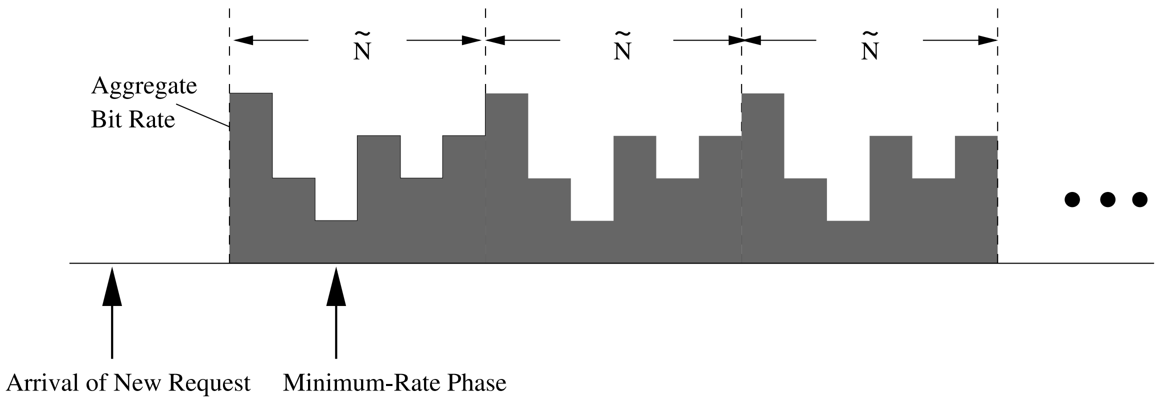

Figure 2: Minimal-Rate Phase scheduling for heterogeneous envelopes.

Upon the arrival of the new stream, a row is added to $\boldsymbol{A}$ based on the envelope and phase of this stream (with the phase being determined using MRP scheduling). The number of columns of $\boldsymbol{A}$ can be fixed a priori by choosing $\widetilde{N}$ based on all anticipated values of $N^{(i)}$ (which are few in practice). The PSAB is recomputed by updating $\boldsymbol{V}$ using $v_j := v_j + a_{n+1,j}$, for $j = 1, \ldots, \widetilde{N}$, and then applying (4) with $n+1$ replacing n. An analogous procedure is used to update the PSAB when an ongoing connection is terminated.

It should be noted that even if no scheduling is performed, some bandwidth gain can still be realized by multiplexing MPEG streams under time-varying traffic envelopes. In this case, $\boldsymbol{u}$ has an arbitrary structure, which is determined only by the times of video requests and are not controlled by the server. The achievable bandwidth gain in this case is discussed in [16].

3 Practical Considerations

The ability to achieve bandwidth gain using the above approach depends on the following factors: (1) the ability to determine the parameters of the traffic envelope, (2) the ability to determine the boundaries between MPEG frames, and (3) the ability to synchronize frames boundaries at the multiplexer. Depending on where multiplexing is implemented (e.g., proprietary server, private ATM switch, public ATM switch), these factors are satisfied by some multiplexing nodes more than others.

For archived video, the traffic envelope parameters can be easily computed offline at the video server. For real-time video (e.g., HDTV), these parameters can, in principle, be conservatively estimated based on the quantization levels and frame attributes. They can also can be enforced by the encoder using a rate-control mechanism, which is part of the MPEG-2 standard.

Determining frame boundaries at the server is facilitated by an appropriate storage design. One can also rely on the information included in the header of an MPEG frame, which contains a special bit sequence for identifying the start of a frame.

Synchronization of frame intervals (i.e., *frame alignment*) can be done at a video server with sufficient support from the operating system (OS). Many "real-time" OSs use scheduling algorithms that ensure deterministic delivery of packets down to the device driver or the network interface card (where multiplexing typically takes place). If the OS cannot ensure the deterministic delivery of frames, then a "synchronization buffer" is needed to align the boundaries of frames. The use of this buffer results in a fixed delay of no more than one frame period.

Given the above considerations, one potential application of our allocation approach is in a video distribution network that consists of a central (remote) server, several HE switches, and several fixed-capacity channels (e.g., ATM VCs) that extend from the server to the HE switches over a wide-area network. Video streams that are destined to the same head-end (HE) switch are multiplexed together at the server, separately from streams going to other HE switches (Figure 3). It is predicted that each HE switch will provide access to as many as 1000 VOD clients [22]. Note that the MPEG-specific multiplexing and bandwidth allocation are performed at the server only, transparent to the public network. We assume, as is typically the case, that the HE switches are proprietary so that a VP can be terminated at these switches rather than at the end systems.

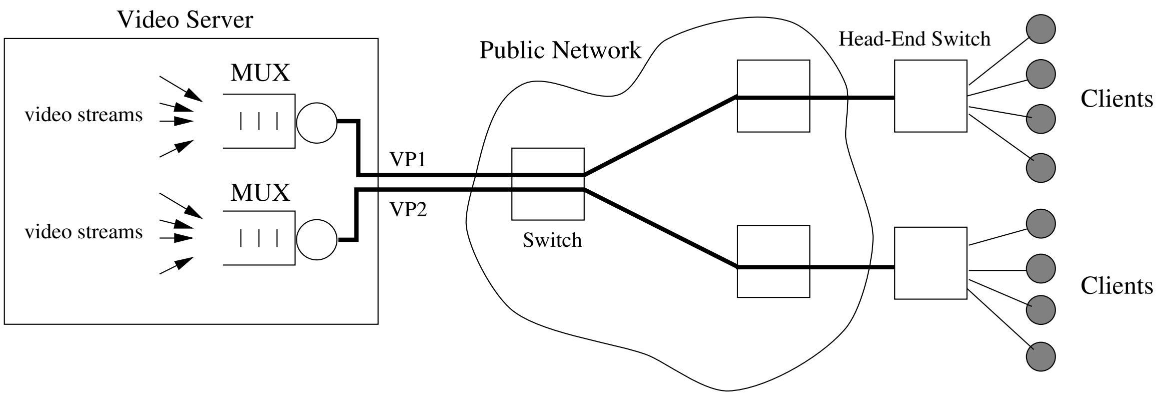

Figure 3: Example of a video distribution network.

The effectiveness of our bandwidth allocation strategy when used in such a distribution network can be quantified as follows. Let k be the number of HE switches and let $n_1,\ n_2, \ldots, n_k$ be the numbers of streams that are destined to these HE switches, respectively. The total number of ongoing streams is given by $n = n_1 + n_2 + \ldots n_k$. Let d_i be the number of links between the server and the ith HE switch, $i = 1, \ldots, k$. Assuming the cost of bandwidth is the same over all links, the total used bandwidth is

$$\text{Used bandwidth } = \sum_{i=1}^{k} d_i n_i C(\boldsymbol{u}, n_i) \tag{9}$$

where $\boldsymbol{u} = \boldsymbol{u}(n_i)$ is the *arrangement* of n_i streams. Note that the server maintains k multiplexers (one per HE switch). Given that packets in a video frame are evenly distributed over the duration of a frame, a buffer of n_i packets is required at the ith multiplexer, for a total buffer space of n packets. This buffer is needed to accommodate the arrival of simultaneous packets at the multiplexer.

4 Call Admission Control

In our framework the distribution network consists of pre-established fixed-capacity bandwidth channels onto which streams are multiplexed and transported. Consequently, whenever a client issues a request for a video stream, this request must be subjected to admission control. The server accepts the request only if sufficient bandwidth is available to transport the new stream from the server to the corresponding HE switch. We now describe a simple online admission control test to be used in conjunction with our bandwidth allocation approach.

Consider a distribution network that consists of one central server and k HE switches. Video streams that are destined to the same HE switch are multiplexed together, separately from other streams. Accordingly, k multiplexers are needed at the server, each of which is associated with its own bandwidth table. Consider one of these multiplexers. Let W be the capacity of a given VP (given in fixed-length packets/frame period). Suppose that n *heterogeneous* video streams are being multiplexed onto that VP. Since the server implements MRP scheduling, the used bandwidth is given by $nC_{mrp}(n)$. When a new video request arrives at the server, the server checks its admissibility by executing the MRP scheduling algorithm, computing $(n+1)C_{mrp}(n+1)$, and contrasting this value against W. If $W \geq (n+1)C_{mrp}(n+1)$ the new connection is admitted. Otherwise, it is rejected. Very few computations are needed to execute this CAC test.

5 Nominal Blocking Probability and VP Dimensioning

Dimensioning the bandwidth capacities of the the distribution network is an important *offline* design problem, which is done infrequently over long time periods, typically in response to major changes in client demand (e.g., increasing demand in the summer season). Similar to the situation in telephone networks, the bandwidth capacity of a video distribution network can be dimensioned at a given *nominal* blocking probability. This nominal blocking probability is *not* used for admission control, but rather serves as a basis for offline computation of the capacities of the various VPs. In contrast, admission control is performed online as was described in the previous section. Naturally, the nominal blocking probability represents an upper bound on the actual blocking probability of a request.

Consider n streams that are being multiplexed onto a VP that extends from the server to a HE switch. At an arbitrary time instant t, the *arrangement* $\boldsymbol{u}$ of these streams has an arbitrary structure. To obtain tractable results, we assume that video sources are characterized by a common traffic envelope $\bar{b}(t)$ with parameters $\boldsymbol{E} = (I_{max},\ P_{max},\ B_{max},\ N,\ M)$. As before, $I_{max} \geq P_{max} \geq B_{max}$. Given a set of n heterogeneous streams, the above common envelope is obtained by taking I_{max}, P_{max}, and B_{max} to be the largest $I^{(i)}_{max}$, $P^{(i)}_{max}$, and $B^{(i)}_{max}$ over all $i \in \{1, \ldots, n\}$, respectively. This will result in a conservative estimate of the blocking probability.

Given that n streams are already being multiplexed onto a VP of maximum capacity W, the unused bandwidth of that VP is $W - NC(\boldsymbol{u}, n)$. Clearly, the blocking probability depends on W, n, $\overline{b}(t)$, and $C(\boldsymbol{u}, n)$. Fixing the first three variables, we let $\boldsymbol{u}$ be an n-dimensional random vector with a discrete uniform distribution, which captures the equal likelihood of all different *arrangements*. Since the server cannot anticipate the arrival time of a video request, the blocking probability must be computed assuming the worst-case scenario for the transmission of video frames. This happens when I frames of the new stream overlap with the phase in which the total arrival rate of the n ongoing streams is largest (i.e., the phase for which the aggregate envelope has a value of $nC(\boldsymbol{u}, n)$). Accordingly, the worst-case blocking probability is defined by

$$\Pr\{I_{max} > W - nC(u,n)\} = \Pr\left\{C(\boldsymbol{u},n) > \frac{W - I_{max}}{n}\right\} \triangleq G_n\left(\frac{W - I_{max}}{n}\right) \tag{10}$$

where $G_n(x) \triangleq$ the complementary distribution function for the rv $C(\boldsymbol{u}, n)$ (with randomness due to $\boldsymbol{u}$). Next, we determine the tail of $G_n(x)$.

5.1 Tail Distribution of $C(\boldsymbol{u}, n)$

For homogeneous streams with aligned boundaries, $C(\boldsymbol{u}, n)$ can be written in the following form:

$$C(\boldsymbol{u},n) = \frac{n_I I_{max} + n_P P_{max} + (n - n_I - n_P) B_{max}}{n} \tag{11}$$

where n_I and n_P are two rvs with probability space $\{0, 1, \ldots, n\}$. Therefore, $G_n(x)$ can be obtained from the joint distribution of (n_I, n_P). We first introduce some elementary results. Notice that since frames boundaries are assumed to be aligned, only the first N values of $\overline{b}_{tot}(t)$ (which are given by $\boldsymbol{V} = [v_1 \; v_2 \; \cdots v_N]$) are needed to compute $C(\boldsymbol{u}, n)$. Recall that a stream s_i belongs to phase k, where $k = 0, \ldots, N-1$, if $u_i = k$, i.e., s_i sends its I frames during phase k. Define

$r_k \triangleq$ number of streams that belong to phase k.

$z_k \triangleq$ number of streams that belong to phases that differ from k by a nonzero multiple of M.

r_k and z_k give the numbers of streams sending I and P frames during phase k, respectively. The following proposition follows directly from the periodicity and regularity of the GOP pattern.

Proposition 1 *Consider any two streams s_i and s_j with phases u_i and u_j, $u_i \neq u_j$. If during phase u_i s_j sends a B frame, then during phase u_j s_i sends a B frame. Also, if during phase u_i s_j sends a P frame, then during phase u_j s_i sends a P frame.* ■

From Proposition 1, it is easy to see that for any two phases, i and j, with $\mid i - j \mid=$ a multiple of M, we have $r_i + z_i = r_j + z_j$. Based on this result, we introduce the following proposition.

Proposition 2 *Let phase k be such that $r_k = \max_i r_i$. If $r_k > n/2$, then $n_I = r_k$. Moreover, phase k is the only phase for which $C(\boldsymbol{u}, n) = v_k/n$.*

Proof: Let phase k be such that $r_k = \max_i r_i$. First, consider an arbitrary phase j for which $|k-j| =$ a multiple of M. Then, $r_k + z_k = r_j + z_j$. In addition, both phases will have the same number of sources that send B frames. Since $r_k > r_j$ (strictly, since only one phase can exist with $r_k > n/2$), $v_k > v_j$.

Next, suppose that j is a phase for which $|k-j| \neq$ a multiple of M. Then all r_k streams that are in phase k will send B frames during phase j (Proposition 1). Thus, $n - r_j - z_j \geq r_k$, which leads to $r_j + z_j \leq n - r_k < n/2 < r_k$. Consequently, $v_k > v_j$. By definition of $C(\boldsymbol{u}, n)$, it must be true that $C(\boldsymbol{u}, n) = v_k/n$, and thus $n_I = r_k$. ■

The implication of Proposition 2 is that when $r_k > n/2$, the peak bit rate of the aggregate envelope during phase k is greater than the peak bit rate during all other phases, regardless of the values of I_{max}, P_{max}, and B_{max}. Therefore, we can compute the joint distribution of (n_I, n_P) by simply computing the joint distribution of (r_k, z_k) for $r_k > n/2$. For $i > n/2$ we have

$$p_{ij} \stackrel{\triangle}{=} \Pr\{n_I = i, n_P = j\} = \sum_{l=0}^{n-1} \Pr\{r_l = i, z_l = j\}, \quad \text{for all } j \in \{0, \ldots, n-i\} \tag{12}$$

Note that when $i > n/2$, the events $[r_l = i]$ and $[r_m = i]$, with $l \neq m$, are mutually exclusive. Since $u_1 \stackrel{\triangle}{=} 0$, the first term in the above sum is given by:

$$\Pr\{r_0 = i, z_0 = j\} = \binom{n-1}{i-1} \left(\frac{1}{N}\right)^{i-1} \binom{n-i}{j} \left(\frac{N/M-1}{N}\right)^{j} \left(\frac{N-N/M}{N}\right)^{n-i-j} \tag{13}$$

Observe that there are $\binom{n-1}{i-1}$ possibilities for the $n-1$ streams (excluding the first stream) to send $i-1$ I frames. Each possibility has a probability of $(1/N)^{i-1}$. Among the remaining $n-i$ streams, there are $\binom{n-i}{j}$ possibilities to send j P frames, each possibility with probability $((N/M-1)/N)^j$ (since the number of P frames in a GOP period is $N/M - 1$). A similar argument justifies the last term in (13), which is related to the probability of sending B frames.

In a similar manner, it is easy to show that for $l \in \{M, 2M, 3M, \ldots, (N/M-1)M\}$

$$\Pr\{r_l = i, z_l = j\} = \binom{n-1}{i} \left(\frac{1}{N}\right)^{i} \binom{n-1-i}{j-1} \left(\frac{N/M-1}{N}\right)^{j-1} \left(\frac{N-N/M}{N}\right)^{n-i-j} \tag{14}$$

Finally, for $l \in \{0, 1, 2, \ldots, M-1, M+1, M+2, \ldots, 2M-1, 2M+1, \ldots, N-1\}$ we have

$$\Pr\{r_l = i, z_l = j\} = \begin{pmatrix} n-1 \\ i \end{pmatrix} \left(\frac{1}{N}\right)^i \begin{pmatrix} n-1-i \\ j \end{pmatrix} \left(\frac{N/M-1}{N}\right)^j \left(\frac{N-N/M}{N}\right)^{n-1-i-j} \tag{15}$$

From (13), (14), and (15), and after some manipulations, (12) can be written as

$$p_{ij} = \frac{\begin{pmatrix} n \\ i \end{pmatrix} \begin{pmatrix} n-i \\ j \end{pmatrix} (N/M-1)^j \, (N-N/M)^{n-i-j}}{N^{n-1}} \tag{16}$$

Since (16) is valid only for $i > n/2$, x must be chosen sufficiently large to ensure that the event $[C(\boldsymbol{u}, n) > x]$ implies the event $[n_I > n/2]$. Let $x^* \triangleq \inf\{x : [C(\boldsymbol{u}, n) > x] \Rightarrow [n_I > n/2]\}$. Then, for $x > x^*$

$$G_n(x) = \sum_{\substack{i,j \text{ s.t.} \\ f_{ij} > x}} p_{ij} \tag{17}$$

where

$$f_{ij} \triangleq \frac{iI_{max} + jP_{max} + (n-i-j)B_{max}}{n} \tag{18}$$

It is easy to see that

$$x^* = \frac{(n/2)I_{max} + (n/2)P_{max}}{n} = \frac{I_{max} + P_{max}}{2} \tag{19}$$

since any value of $C(\boldsymbol{u}, N)$ that is greater than the RHS of (19) implies necessarily that $n_I > n/2$. For $x > x^*$, $G_n(x)$ is obtained from (16), (17), and (18).

5.2 Blocking Probability

We now derive the nominal blocking probability under $C(\boldsymbol{u}, n)$-based bandwidth allocation. To facilitate a comparison with source peak-rate based allocation, we set the total capacity to $W = (n-K)I_{max}$, where K is a nonnegative integer and n is the number of ongoing streams. The probability that j simultaneous requests are rejected ($j \geq 1$) given n ongoing connections and a total capacity of $W = (n-K)I_{max}$ is given by:

$$\begin{aligned} P_K^{(n,j)} &\triangleq \Pr\{jI_{max} > W - nC(\boldsymbol{u}, n) \; / \; W \geq nC(\boldsymbol{u}, n)\} \\ &= \frac{\Pr\left\{C(\boldsymbol{u}, n) > \frac{n-K-j}{n} I_{max}, \; C(\boldsymbol{u}, n) \leq \frac{n-K}{n} I_{max}\right\}}{\Pr\left\{C(\boldsymbol{u}, n) \leq \frac{n-K}{n} I_{max}\right\}} \\ &= \frac{G_n\left(\frac{n-K-j}{n} I_{max}\right) - G_n\left(\frac{n-K}{n} I_{max}\right)}{1 - G_n\left(\frac{n-K}{n} I_{max}\right)} \end{aligned} \tag{20}$$

$P_K^{(n,j)}$ is referred to as the K-order blocking probability for j simultaneous requests given that n streams are already multiplexed onto a VP with capacity $W = (n-K)I_{max}$. Note that the fact that n connections are already admitted implies necessarily that $nC(\boldsymbol{u}, n) \leq W$, which must be conditioned on when computing the blocking probability. Since $G_n(x)$ is known only for $x > x^*$, n must be sufficiently large so that $G_n\left(\frac{n-K-j}{n}I_{max}\right)$ can be computed. Let $P_{max} = \alpha I_{max}$ for some $0 < \alpha \leq 1$. Define the following quantity:

$$n^*(j) \stackrel{\triangle}{=} \min\{n : (n - K - j)I_{max}/n > x^*\}. \tag{21}$$

It is easy to show that $n^*(j) = \left\lceil \frac{2(K+j)}{1-\alpha} \right\rceil$. Thus, $P_K^{(n,j)}$ is given by (20) for $n \geq n^*(j)$.

A special case of interest is when $K = 0$, i.e., W is equal to the sum of source peak rates of ongoing streams. Under source-peak-rate allocation, no additional connections can be admitted. Under $C(\boldsymbol{u}, n)$-based allocation, the blocking probability for j new connection requests reduces to:

$$P_0^{(n,j)} = G_n\left(\frac{n-j}{n}I_{max}\right) \tag{22}$$

In practice the blocking probability is needed for $j = 1$, since requests seldom arrive in batches.

Some numerical examples are given in Figures 4 and 5. Using the traffic envelope of the *Wizard of Oz* trace (parameters are given in Table 2), $G_n(x)$ is plotted in Figure 4 as a function of x for three values of n. For this trace, $G_n(x)$ is obtainable for $x \geq x^* = 818$ cells. When $n = 15$ the zero-order blocking probability for a new stream is $P_0^{(15,1)} = G_n(14I_{max}/15) \approx 1.866 \times 10^{-10}$. The zero, first, and second-order blocking probabilities of a request are plotted in Figure 5 as a function of n using the traffic envelope of *Lecture* trace [13].

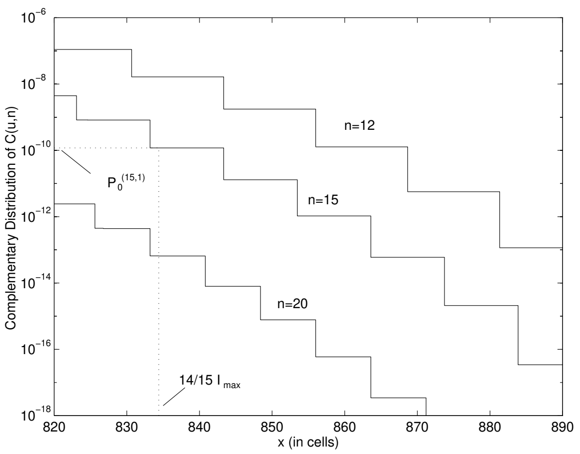

Figure 4: $G_n(x)$ versus x based on *Wizard of Oz* envelope.

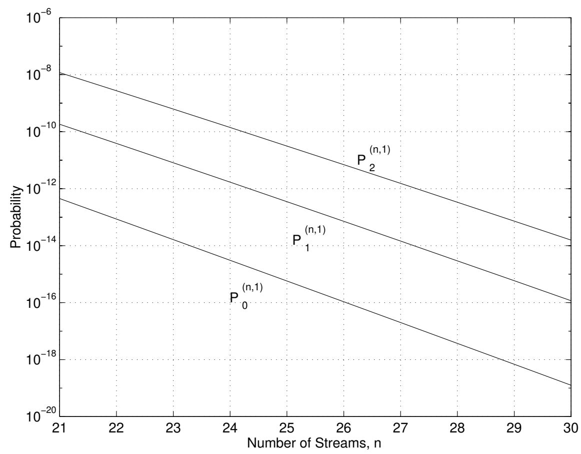

Figure 5: $P_K^{(n,1)}$ versus n for $K = 0, 1, 2$ using *Lecture* traffic envelope.

5.3 Dimensioning the VP Capacity

The VP-dimensioning problem can be formulated as follows: Given $G_n(x)$ for a fixed n and given a target blocking probability ξ, determine the minimum value of W that satisfies

$$G_n\left(\frac{W - I_{max}}{n}\right) \leq \xi \tag{23}$$

This can be done numerically by computing the complementary distribution $G_n(x)$ for a fixed n and for $x \geq x^* = (I_{max}+P_{max})/2$, and then determining the minimum x that satisfies (23). As an example, consider the *Wizard of Oz* envelope. In this case, $x^* = (I_{max} + P_{max})/2 = (894 + 742)/2 = 818$ cells. Table 1 depicts the minimum value of W (normalized to I_{max}) such that $P_0^{(n,1)} \leq \xi = 10^{-10}$. Had dimensioning been performed according to the peak rates of the individual sources, then the required capacity would have been set to $(n+1)I_{max}$ for a zero blocking probability. In other words, the difference $(n+1)I_{max} - W$ provides an indication of the total gain from $C(\boldsymbol{u}, n)$-based dimensioning. For $n < 10$, the value of W is similar to that obtained under source-peak-rate dimensioning. However, as n increases, W starts to lag behind $(n+1)I_{max}$, with a nonzero call blocking probability.

n	W/I_{max}	Blocking probability
2	3	0
4	5	0
6	7	0
8	9	0
10	10.84	2.60E–11
12	12.66	5.66E–12
14	14.33	1.27E–11
16	15.81	8.69E–11

Table 1: Dimensioning of the capacity of a VP at a given nominal blocking probability.

6 Simulation Results

In this section, simulations are conducted to study the effectiveness of $C(\boldsymbol{u}, n)$-based allocation. We consider a distribution network that consists of a video server and seven switches: an access switch, two intermediate switches, and four HE switches (Figure 6). A virtual connection extends from the server to each HE switch. Video streams are multiplexed *at the server* onto these connections.

We examine the performance under uniform and skewed loads. In the case of a uniform load, the rate at which video requests are generated is the same for all HE switches. Under skewed load this rate differs from one HE switch to another. To skew the load, we assume that a video request is sent via HE switches 0, 1, 2, and 3, with probabilities 0.4, 0.3, 0.2, and 0.1, respectively.

We first study the performance without consideration to call blocking by assuming that the capacities of the VPs are sufficiently large. Then we study the performance with call blocking, in which

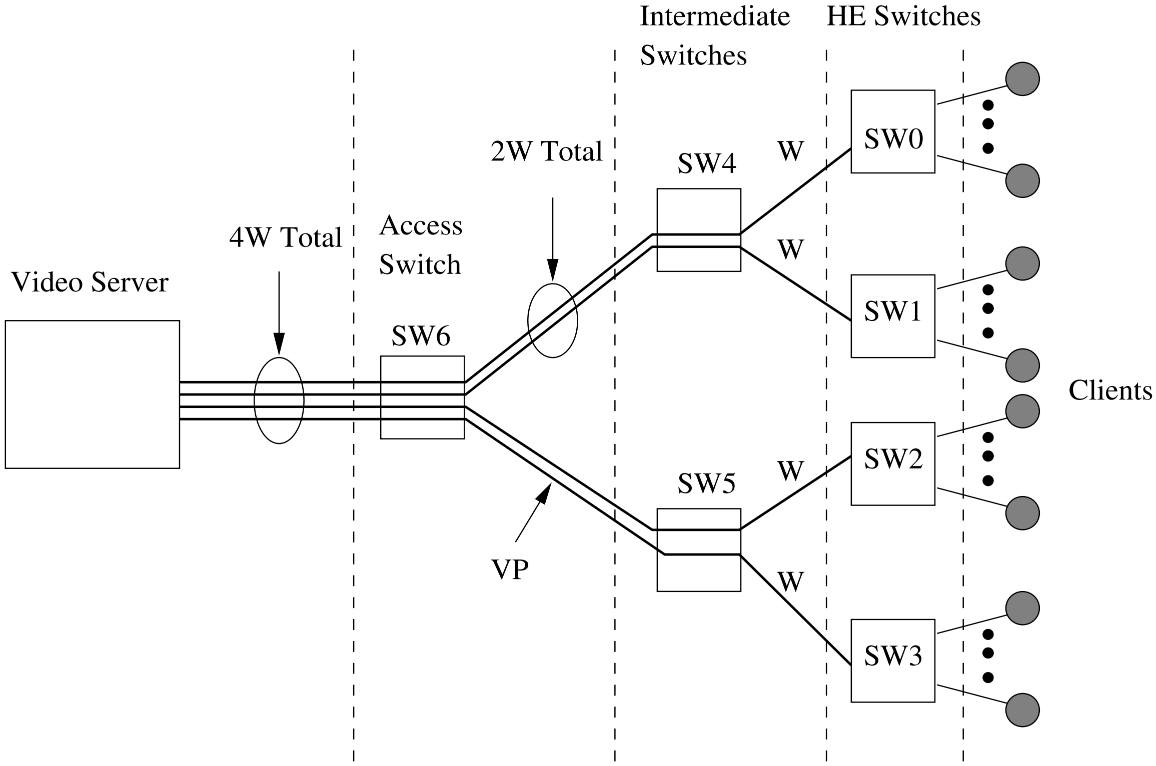

Figure 6: Distribution network used in the simulations

case the capacity of each VP is set to W (the four VPs have the same capacity). The purpose of considering both blocking and no blocking is to investigate the impact of call blocking on the efficiency of the bandwidth allocation strategy.

When a client requests a movie, the request is sent to the server, which performs admission control and bandwidth computations (in the non-blocking case, only bandwidth computations are performed). If the stream is accepted, the server schedules it to be multiplexed with other streams. Scheduling at the server is based on the MRP scheme. The server maintains a bandwidth table for each VP, and it performs admission control and bandwidth computations based on the unused capacity of the VP over which the new stream is to be transported. Thus, the number of bandwidth tables is equal to the number of HE switches.

In our simulations, requests for movies are generated according to a Poisson process with rate λ. Once a request is accepted, the corresponding stream stays active for a random duration that is uniformly distributed in the interval $[\sigma - \epsilon, \sigma + \epsilon]$, where σ represents the average duration of a movie. We fix σ and ϵ at $\sigma = 100$ and $\epsilon = 10$ (in minutes). Let $n(t)$ be the total number of transported video streams in the distribution network at time t. Under no blocking and for a fixed t, $n(t)$ is a Poisson rv with parameter $E[n(t)] \stackrel{\triangle}{=} \rho_{tot} = \lambda\sigma$. The quantity ρ_{tot} represents the total load on the system. In our experiments, ρ_{tot} is varied by varying λ.

Our simulations are based on two MPEG video traces: *Star Wars* [9] and *The Wizard of Oz* [14]

(*Wizard*, for short). Focusing only on two traces makes it easier to contrast the performance under homogeneous streams (i.e., identical envelopes) with the performance under heterogeneous streams. The traffic envelopes for the two traces are described in Table 2. In the heterogeneous case, we assume that either movie can be requested with probability 0.5. Let $\rho^{(i)}$ indicate the load (i.e., average number of active streams) destined to HE Switch i, for $i = 0, 1, 2, 3$. Notice that $\rho_{tot} = \sum_i \rho^{(i)}$.

Trace	I_{max}	P_{max}	B_{max}	N	M
Star Wars [9]	483	454	169	12	3
Wizard of Oz [14]	894	742	157	15	3

Table 2: Traffic envelopes for two MPEG-coded movies (frame sizes in ATM cells).

6.1 Performance Without Call Blocking

When blocking is not taken into consideration, it is sufficient to focus on one VP only, since no interactions take place between different VPs. Figure 7 depicts a representative sample path for the number of ongoing connections over a particular VP. The corresponding PSAB (normalized to the source peak rate) is shown in Figure 8. Notice that the PSAB is also a function of time (since the number of active connections continuously varies with time).

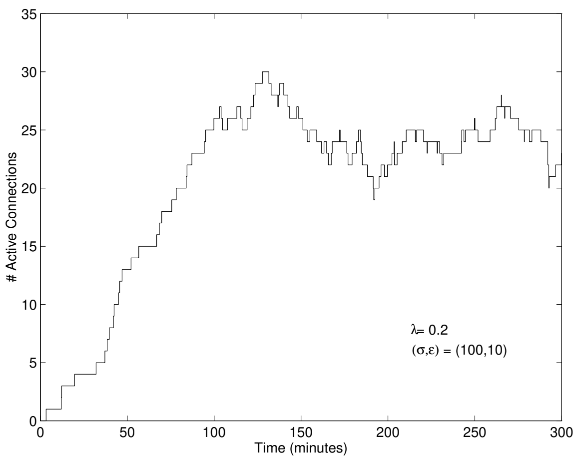

Figure 7: Number of connections in progress as a function of time.

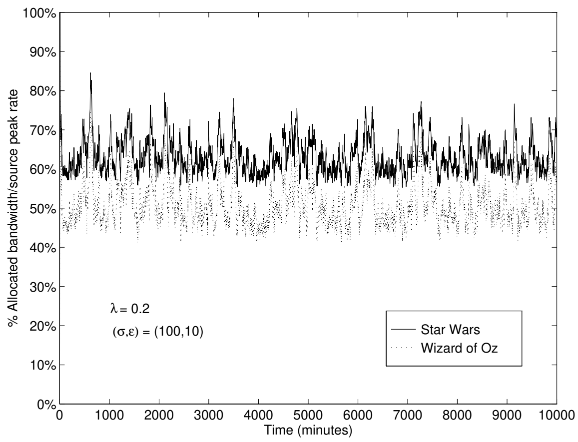

Figure 8: Percentage of $C(\boldsymbol{u}, n^{(i)}(t))/I_{max}$ as a function of time.

Since $\{C(\boldsymbol{u}, n, t) : t > 0\}$ constitutes a random process, it is constructive to compute the mean and variance of this process (i.e., $E[C(\boldsymbol{u}, n)]$ and $\text{var}[C(\boldsymbol{u}, n)]$), which would give an indication of the effectiveness of $C(\boldsymbol{u}, n)$-based allocation. Unfortunately, both moments cannot be determined in closed form since the marginal distribution of $C(\boldsymbol{u}, n)$ is generally intractable when n is finite. Instead, for finite n the *temporal* moments can be computed from synthetic realizations of the stochastic process

$\{C(\boldsymbol{u},n,t); t \geq 0\}$. More formally,

$$E\left[C(\boldsymbol{u},n)\right] = C_{avg} \triangleq \lim_{t\to\infty} \frac{1}{t}\int_0^t C(\boldsymbol{u},n,\tau)d\tau \approx \frac{1}{T}\sum_i C\left(\boldsymbol{u}(t_i),n(t_i)\right)\Delta t_i \tag{24}$$

$$\mathrm{var}\left[C(\boldsymbol{u},n)\right] = v_{avg}^2 \triangleq \lim_{t\to\infty} \frac{1}{t}\int_0^t [C(\boldsymbol{u},n,\tau) - C_{avg}]^2 d\tau \approx \frac{1}{T}\sum_i \left(C\left(\boldsymbol{u}(t_i),n(t_i)\right) - C_{avg}\right)^2 \Delta t_i \tag{25}$$

where t_i is the time of the ith change in $n(t)$. In the results below, each temporal moment is computed based on five independent runs, and the sample average of these runs is reported. The confidence intervals for both C_{avg} and v_{avg} over the five runs were found extremely tight, and are not reported for brevity.

Table 3 depicts the results for different values of $\rho^{(i)}$. The values in the table are given as a percentage of the source peak rate. In the heterogeneous case, the source peak rate is taken as $\overline{I}_{max} = (1/M)\sum_{i=1}^M I_{max}^{(i)}$. Clearly, as $\rho^{(i)}$ increases, C_{avg} (also v_{avg}) decreases to a limiting value, which can be simply obtained by taking the limit of $C_{mrp}(n)$ as $n \to \infty$:

$$\lim_{\rho^{(i)}\to\infty} C_{avg} = \lim_{n\to\infty} C_{mrp}(n) = \lim_{n\to\infty} C_{min}(n) \tag{26}$$

$$= \frac{1}{M}\sum_{i=1}^{M} \frac{1}{\widetilde{N}} \int_0^{\widetilde{N}} \overline{b}_i(t)dt \tag{27}$$

$$= \frac{1}{M}\sum_{i=1}^{M} \frac{I_{max}^{(i)} + (N^{(i)}/M^{(i)} - 1)P_{max}^{(i)} + (N^{(i)} - N^{(i)}/M^{(i)})B_{max}^{(i)}}{N^{(i)}} \tag{28}$$

As an example, if the two examined traces can be requested with equal probability (the heterogeneous case), then $\lim_{n\to\infty} C_{mrp}(n) = 45.6\%$ of $\overline{I}_{max}$.

$\rho^{(i)}$	Requested Movies		
	Star Wars	*Wizard*	Both
4	64.6%	54.3%	62.5%
8	59.9%	46.6%	54.2%
12	58.3%	44.3%	51.1%
16	57.5%	43.6%	49.5%
24	56.8%	42.5%	48.1%
32	56.3%	42.1%	47.6%

(a) % of $C_{avg}/\overline{I}_{max}$

$\rho^{(i)}$	Requested Movies		
	Star Wars	*Wizard*	Both
4	15.4%	17.4%	19.9%
8	4.9%	6.1%	7.7%
12	2.9%	3.3%	4.7%
16	2.2%	2.6%	3.6%
24	1.5%	1.9%	2.8%
32	1.2%	1.5%	2.3%

(b) % of $v_{avg}/\overline{I}_{max}$

Table 3: Mean and standard deviation of the PSAB under no call blocking.

6.2 Performance With Call Blocking

To account for call blocking, we set W to $0.7\rho_{tot}\overline{I}_{max}/4$, so that when the load is uniform the capacity of a VP is equal, on average, to 70% of the source peak rate of the aggregate traffic (i.e., $W = 0.7\rho^{(i)}\overline{I}_{max}$

for $i = 0, 1, 2, 3$). The PSAB with call blocking is given in Table 4 for $\rho_{tot} = 40$ and 80. The values of $\rho^{(i)}$ ($i = 0, 1, 2, 3$) are obtained by multiplying ρ_{tot} by the fractions 0.4, 0.3, 0.2, 0.1, which are used to skew the load. Comparing the PSAB for each $\rho^{(i)}$ in Table 4 to the corresponding value in Part (a) of Table 3, it can be observed that blocking has a minor impact on the PSAB. In general, blocking slightly reduces the *effective* value of $\rho^{(i)}$, which subsequently increases $E[C(\boldsymbol{u}, n^{(i)}]$. Note, however, that the decrease in the PSAB with an increase in $\rho^{(i)}$ is *not* always monotone, due to the fact that $C_{mrp}(n)$ does not decrease monotonically with n.

The corresponding blocking probabilities for the four VPs are shown in Table 5. As expected, an increase in $\rho^{(i)}$ with ρ_{tot} being fixed (therefore, W is fixed), results in an increase in the blocking probability over the corresponding VP. It is interesting to notice that for a fixed $\rho^{(i)}/\rho_{tot}$, the blocking probability decreases as ρ_{tot} increases (this can be seen by comparing each row in Part (a) of Table 5 to the adjacent row in Part (b) of the same figure). This can be justified by the fact that W was taken as a function of ρ_{tot}. On average, the available bandwidth over the VP that extends to HE Switch i ($i = 0, 1, 2, 3$) is given by

$$W - n^{(i)} E[C_{mrp}(n^{(i)})] = W - \rho^{(i)} C_{avg} = \rho_{tot} \overline{I}_{max} \left(0.7/4 - p^{(i)} C_{norm}\right) \tag{29}$$

where $p^{(i)}$ is the fixed ratio $\rho^{(i)}/\rho_{tot}$ and C_{norm} is the ratio $C_{avg}/\overline{I}_{max}$ given by the percentages in Table 4. For a given HE switch and a fixed $\rho^{(i)}/\rho_{tot}$, when ρ_{tot} increases, the only other term in (29) that will also change in C_{norm}, but the C_{norm} decreases a much slower rate than the rate of increase in ρ_{tot}. Thus, an increase in ρ_{tot} while fixing $\rho^{(i)}/\rho_{tot}$ results in an increase in the idle capacity, and therefore a reduction in the blocking probability.

HE Switch i	$\rho^{(i)}$	Requested Movies		
		Star Wars	*Wizard*	Both
0	16	57.7%	43.3%	50.8%
1	12	58.4%	44.3%	51.6%
2	8	59.9%	46.6%	53.9%
3	4	62.1%	50.2%	58.0%

(a) $\rho_{tot} = 40$

HE Switch i	$\rho^{(i)}$	Requested Movies		
		Star Wars	*Wizard*	Both
0	32	56.4%	42.2%	48.1%
1	24	56.7%	42.6%	48.3%
2	16	57.5%	43.6%	49.4%
3	8	59.8%	46.5%	53.9%

(b) $\rho_{tot} = 80$

Table 4: Percentage of $C_{avg}/\overline{I}_{max}$ under call blocking.

7 Implementation Issues

7.1 System Setup

Since the proposed allocation scheme relies on time-varying envelopes, timing considerations are crucial to its operation. This section addresses some of the issues related to the implementation of the

HE Switch i	$\rho^{(i)}$	Requested Movies		
		Star Wars	*Wizard*	Both
0	16	3.4E−1	2.2E−1	2.4E−1
1	12	2.0E−1	8.6E−2	1.1E−1
2	8	5.6E−2	1.0E−2	1.9E−2
3	4	3.8E−3	1.0E−4	4.0E−4

(a) $\rho_{tot} = 40$

HE Switch i	$\rho^{(i)}$	Requested Movies		
		Star Wars	*Wizard*	Both
0	32	3.1E−1	1.1E−1	1.8E−1
1	24	1.4E−1	1.7E−2	4.6E−2
2	16	1.5E−2	1.0E−4	1.1E−3
3	8	0	0	0

(b) $\rho_{tot} = 80$

Table 5: Call blocking probability.

allocation scheme. Our ideas are demonstrated using a specific setup that is composed of an RS/6000 570 IBM machine connected to an ATM network via an ATM TurboWays 100 Mbps adaptor. The RS/6000 is a Micro-Channel-based workstation that runs AIX 3.2 operating system.

In its simplest form, the video server is an application process that waits for requests to arrive. Once a request arrives, the server computes the phase of the prospective stream using the MRP scheduling scheme and spawns a new process (for example, using the *fork()* system call in UNIX) to handle the request. Each request is associated with one child process which is responsible for checking the admissibility of the request, fetching video data from disk, and copying them to the underlying kernel space. Multiplexing of video streams can be implemented in software, either in the kernel or in the network device driver. The kernel approach would be more appropriate once the operating system has been extended to handle multimedia traffic between different devices within the system (e.g., network controllers, disk, and playback hardware). With current operating systems, multiplexing is more naturally implemented in the network device driver, so as to maximize the accessibility of the allocation scheme to all application processes and make its operation transparent to them.

7.2 Admission Control

To implement the admission control scheme within the driver, a mechanism is needed to signal the traffic envelope parameters to the driver. This can be done by extending the OS interface to the device driver. In fact, UNIX systems provide the system call (*ioctl()*) for passing user specified parameters to the device drivers. Other required parameters are a flag for specifying MPEG connections and a pointer to a traffic-envelope vector. When the server process invokes the driver the request is subjected to the admission control test described in Section 4. Admission control is quite simple, requiring only few operations (additions and comparisons) on the traffic parameters and the values in the bandwidth table. If admission is granted, a file handle is returned to the application process that is handling the request. This handle is subsequently used by the application process to send video frames to the driver. A schematic diagram of how the allocation scheme can be implemented in a device driver is shown in Figure 9.

The structure of the bandwidth table is consistent with the description in Section 2.2. Each ongoing

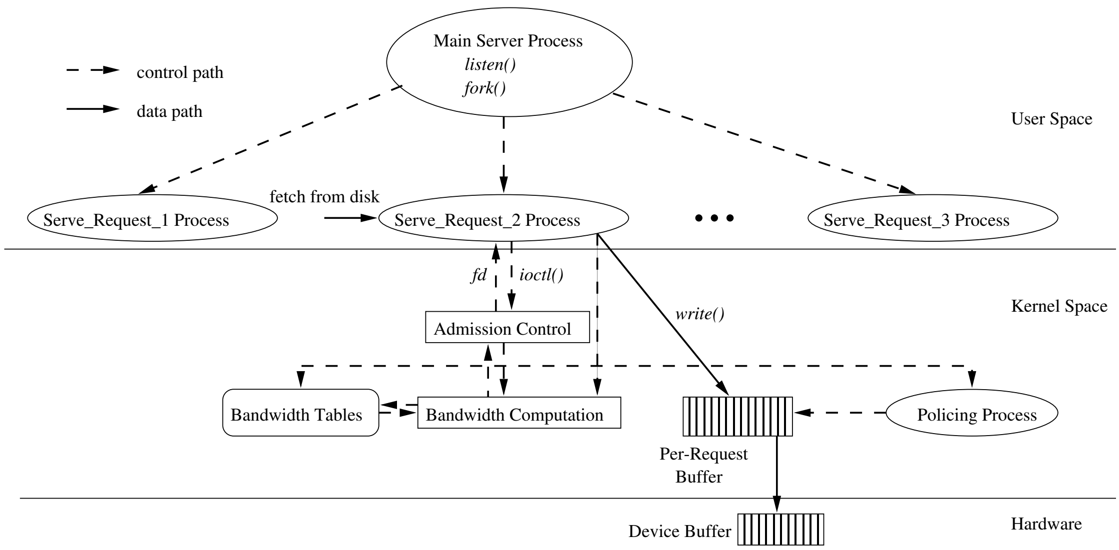

Figure 9: Schematic diagram showing how the bandwidth allocation scheme can be implemented in a network device driver.

connection is associated with a "frame-positioning" array of size $\widetilde{N}$. This array is initialized according to the traffic-envelope parameters, as specified at connection establishment, as well as the phase of the connection. An additional array is used to maintain the sum of bandwidth in each phase. This information is used for admitting new streams into the system.

7.3 Multiplexing and Policing

Multiplexing can be implemented by means of a kernel process, which is also responsible for policing the amounts of data to be moved from the per-stream buffers to the device buffers. This policing process is executed periodically at the start of each time slot (e.g., 1/30 sec). After copying data to device buffers, it sleeps until the next time slot. Given that many operating systems do not guarantee the timely execution of tasks, it is more appropriate to implement this policing process as a kernel process so that it would be subjected to a more favorable OS scheduling (kernel processes have priority over application processes, and the variation in their execution times is much smaller).

The policing process is created when the device driver is configured. At the start of each time slot, this process resumes execution and updates a global slot counter that counts modulo $\widetilde{N}$ (in units of frame periods). It then goes through the list of active streams. If the stream is an MPEG stream, the policing process checks the frame-positioning array associated with that stream for the maximum amount of data that can be sent during that slot, and sends these data to the card. The amount of data copied from a given per-stream buffer during a time slot need not correspond to one frame or even an integer number of frames, provided that this amount does not exceed the value of the corresponding traffic envelope. If a stream has more data to send than what is allowed, the excess data are left in the per-stream buffer until the next frame slot.

8 Summary

Bandwidth allocation for VBR video with stringent, deterministic QoS requirements is typically performed based on the peak rates of the individual sources. Such an allocation strategy underutilizes the network capacity. For MPEG-coded video, a more efficient allocation strategy can be devised by exploiting the periodic manner in which the different types of MPEG-video frames are generated. By characterizing MPEG streams using *time-varying* traffic envelopes, we showed that it is possible to achieve some multiplexing gain while simultaneously supporting stringent, deterministic QoS. The achieved gain can be maximized by appropriate stream scheduling at a video server. The practicality of our proposed approach for the distribution of archived video was discussed. Its feasibility was demonstrated for a video distribution network that consists of a remote server and several HE switches that connect to the server via several channels. We analytically obtained a worst-case estimate of the blocking probability for a video request, and showed how this estimate can be used in dimensioning the bandwidth capacities of the video distribution network. Using real MPEG traces, we studied via simulations the effectiveness of our scheduling and bandwidth allocation strategies, using a distribution network of a server and seven switches. Finally, issues related to the implementation of the proposed bandwidth allocation scheme were discussed with reference to a particular experimental setup.

Acknowledgement

The authors would like to thank Mark Garrett and Ed Knightly for providing their video traces.

References

[1] C. Aggarwal, J. Wolf, and P. Yu. On optimal batching policies for video-on-demand servers. In *IEEE Multimedia Computing and Systems Conference*, pages 253–258, 1996.

[2] K. C. Almeroth and M. H. Ammar. The use of multicast delivery to provide a scalable and interactive video-on-demand service. *IEEE Journal on Selected Areas in Communications*, 14(6):1110–1122, Aug. 1996.

[3] D. Anderson. Metascheduling for continuous media. *ACM Transactions on Computer Systems*, 11(3):226–252, 1993.

[4] H. J. Chen, A. Krishnamurthy, T. Little, and D. Venkatesh. A scalable video-on-demand service for the provision of VCR-like functions. In *Proceedings of the IEEE Multimedia Conference*, 1995.

[5] M.-S. Chen, D. D. Kandlur, and P. S. Yu. Storage and retrieval methods to support fully interactive playout in a disk-array based video server. *Multimedia Systems Journal*, 3:126–135, 1995.

[6] A. Dan, D. Sitaram, and P. Shahabuddin. Scheduling policies for an on-demand video server with batching. In *ACM Multimedia Conference*, pages 15–24, 1994.

[7] A. Elwalid, D. Mitra, and R. Wentworth. A new approach for allocating buffers and bandwidth to heterogeneous regulated traffic in an ATM node. *IEEE Journal on Selected Areas in Communications*, 13(6):1115–1127, 1995.

[8] W.-C. Feng and J. Rexford. A comparison of bandwidth smoothing techniques for the transmission of prerecorded compressed video. In *Proceedings of INFOCOM '97*, Apr. 1997.

[9] M. W. Garrett and W. Willinger. Analysis, modeling, and generation of self-similar VBR video traffic. In *Proc. of the SIGCOMM '94 Conference*, pages 269–280, Sept. 1994.

[10] L. Golubchik, J. Lui, and R. Muntz. Adaptive piggybacking: A novel technique for data sharing in video-on-demand storage servers. *Multimedia Systems Journal*, 4(3):140–155, 1996.

[11] M. Graf. VBR video over ATM: Reducing network resource requirements through endsystem traffic shaping. In *Proc. of IEEE INFOCOM '97*, pages 48–57, 1997.

[12] E. W. Knightly. H-BIND: A new approach to providing statistical performance guarantees to VBR traffic. In *Proc. of the INFOCOM '96 Conference*, Mar. 1996.

[13] E. W. Knightly and H. Zhang. Traffic characterization and switch utilization using a deterministic bounding interval dependent traffic model. In *Proc. of IEEE INFOCOM '95*, pages 1137–1145, 1995.

[14] M. Krunz and H. Hughes. A traffic model for MPEG-coded VBR streams. In *Proc. of the ACM SIGMETRICS/PERFORMANCE '95 Conference*, pages 47–55, May 1995.

[15] M. Krunz and S. K. Tripathi. Exploiting the temporal structure of MPEG video for the reduction of bandwidth requirements. In *Proceedings of the IEEE INFOCOM '97 Conference*, pages 67–74, Kobe, Japan, Apr. 1997.

[16] M. Krunz and S. K. Tripathi. Impact of video scheduling on bandwidth allocation for multiplexed MPEG streams. *ACM Multimedia Systems Journal*, 5(6):347–357, Dec. 1997.

[17] S. S. Lam, S. Chow, and D. K. Y. Yau. An algorithm for lossless smoothing for MPEG video. In *Proceedings of the ACM SIGCOMM '94 Conference*, pages 281–293, Aug. 1994.

[18] V. Li, W. Liao, X. Qiu, and E. Wong. Performance model of interactive video-on-demand systems. *IEEE Journal on Selected Areas in Communications*, 14(6):1099–1109, Aug. 1996.

[19] W. Liao and V. Li. The split and merge (SAM) protocol for interactive video-on-demand systems. In *Proceedings of INFOCOM'97*, Apr. 1997.

[20] S. C. Liew and C. yin Tse. Video aggregation: Adapting video traffic for transport over broadband networks by integrating data compression and statistical multiplexing. *IEEE Journal on Selected Areas in Communications*, 14(6):1123–1137, Aug. 1996.

[21] J. M. McManus and K. W. Ross. Video on demand over ATM: Constant-rate transmission and transport. In *Proc. of IEEE INFOCOM '96*, pages 1357–1362, 1996.

[22] J.-P. Nussbaumer, B. V. Patel, F. Schaffa, and J. P. G. Sterbenz. Networking requirements for interactive video on demand. *IEEE Journal on Selected Areas in Communications*, 13(5):779–787, 1995.

[23] B. Ozden, A. Biliris, and R. R. A. Silberschatz. A low-cost storage server for movie on demand databases. In *Proc. of the 20th VLDB Conference*, 1994.

[24] D. Reininger, B. Melamed, and D. Raychaudhuri. Variable bit rate MPEG video: Characteristics, modeling and multiplexing. In *Proc. of the 14th Int. Teletraffic Congress*, pages 295–306, Antibes Juan-les-Pins, France, June 1994.

[25] M. Reisslein and K. W. Ross. Join-the-Shortest-Queue prefetching for VBR video on demand. In *Proc. of International Conference on Networking Protocols*, 1997.

[26] J. Rexford and D. Towsley. Smoothing variable-bit-rate video in an internetwork. *IEEE/ACM Transactions on Networking*, 1999. To appear.

[27] J. D. Salehi, Z.-L. Zhang, J. F. Kurose, and D. Towsley. Supporting stored video: Reducing rate variability and end-to-end resource requirements through optimal smoothing. *IEEE/ACM Transactions on Networking*, 6(4):397–410, Aug. 1998.

[28] P. J. Shenoy and H. M. Vin. Efficient support for scan operations in video servers. In *Electronic Proceedings of ACM Multimedia '95*, Nov. 1995.

[29] N. Shroff and M. Schwartz. Video modeling within networks using deterministic smoothing at the source. In *Proceedings of IEEE INFOCOM '94*, volume 1, pages 342–349, 1994.

[30] M. Vernick, C. Venkatramani, and T. cker Chiueh. Adventures in building the Stony Brook video server. In *Proceedings of ACM Multimedia*, Nov. 1996.

[31] Z. Wei, M. Krunz, and S. Tripathi. Efficient transport of stored video using stream scheduling and window-based traffic envelopes. In *Proceedings of the IEEE ICC '97*, 1997.

[32] Z. Wei and S. Tripathi. Bandwidth-efficient continuous media streaming through optimal multiplexing. In *Proceedings of the ACM Sigmetrics '99 Conference*, 1999.