# Transmission of Aggregate VoD Streams Using Playback Rate

Seok-Kyu Kweon and Kang G. Shin

Department of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, Michigan 48109–2122.
email: {skkweon, kgshin}@eecs.umich.edu

## Abstract

*In Video-on-Demand (VoD) systems, using the entire frame-size sequence for each video stream, the VoD service provider can derive a video transport schedule which will cause neither buffer overflow nor buffer starvation without requiring excessive amounts of network bandwidth and buffer space. Specifically, such a transport schedule uses temporal averaging to reduce the rate variability of a video stream. In this paper, we propose a simple and efficient VBR VoD-stream transport mechanism called the* **Play-Back Rate Transport with Minimum Rate Guaranteed** *(PBRT-MRG), which exploits statistical multiplexing as well as temporal averaging. The proposed scheme provides* **statistical** *loss guarantees — instead of absolute loss-freedom guarantees — to maximize network utilization. Temporal averaging is done through buffering both at the server and at receivers. To exploit the advantages of statistical multiplexing, a set of VoD streams are multiplexed onto a common transport channel. For the aggregated video streams which are averaged temporally and spatially, we present a simple transport mechanism and a call-admission scheme both of which together provide statistical loss guarantees. Our empirical study shows that one can provide a VoD service without requiring any complex control while achieving high bandwidth/buffer utilization at a reasonable build-up delay. In one case, the proposed scheme is shown to save buffer requirement by as much as three orders of magnitude for the same bandwidth usage.*

*Index Terms* — Video-on-Demand (VoD), playback rate, statistical traffic envelope, MPEG.

## 1 Introduction

The advent of high-speed packet-switched networks has made it possible to provide a wide range of distributed multimedia services. In particular, Video-on-Demand (VoD) services have been drawing considerable interest because of their potential use for entertainment and education. VoD services prefer a variable bit-rate (VBR) video such as MPEG to a constant-bit-rate (CBR) video, mainly because with VBR videos, the service provider can offer viewers constant-quality while reducing consumption of storage and network resources. Thus, finding an efficient but effective transport and playback mechanism for VBR videos is a key to the success of any VoD service. By "efficient but effective," we mean that the transport and playback mechanism must reduce consumption of resources while keeping the probability of buffer overflow or underflow (starvation) at the receiver below a pre-determined threshold. Buffer overflow occurs when newly-arrived video data cannot be stored in the receiver buffer because the buffer is already full of video data waiting for playback, and buffer starvation occurs when the video data to be played back has not yet arrived.

Many researchers have been looking into the problem of transporting real-time or live video with the guaranteed packet-delivery service [1–4], where a certain amount of bandwidth is reserved for each virtual connection. Although the transport of VoD data resembles that of real-time video data in that both require bounded delivery delays, a VoD service has distinct characteristics. In particular, in a VoD service, pre-recorded video frames are transported from a server to multiple receivers, and thus, the entire video frame size sequence for each video stream is known *a priori* to the server. Using this information, one can derive a transport schedule which does not cause buffer overflow nor starvation while reducing resource consumption. Mcmanus and Ross [5], for example, developed a Constant Rate Transmission and Transport (CRTT) schedule. CRTT establishes a CBR virtual channel between the server and the receiver, then transmits video data from the server through this channel at a constant rate. Assuming that the frame size sequence is completely known *a priori*, they derived the relation among transmission rate, receiver buffer size, and build-up delay, which guarantees neither buffer starvation nor buffer overflow. Since a constant transmission rate is used during the entire lifetime of a stream, CRTT may cause a large build-up delay and thus require a very large buffer at the receiver, depending on the char-

The work reported in this paper was supported in part by the Office of Naval Research under Grant N00014-94-1-0229. The opinions, findings and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the view of the ONR.
Seok-Kyu Kweon is currently with Cisco Systems, San Jose, CA. E-mail: skkweon@cisco.com



0-7695-1134-1/01 $10.00 © 2001 IEEE

acteristics of the transmitted VBR video. To resolve this problem, several researchers proposed Piecewise-Constant-Rate Transmission (PCRT) schedules [5–8]. This approach varies the transmission rate over a number of intervals during each of which the transmission rate is kept constant to reduce the build-up delay and buffer requirement. However, it requires the calculation of an optimal transmission schedule, which is computation-intensive, and the obtained schedule must be stored in the server's memory.

Video smoothing approaches like CRTT and PCRT attempt to reduce the rate variability of a *single* video stream using temporal averaging. Therefore, a virtual channel must be assigned for each individual video stream and these virtual channels must be managed individually when multiple video streams are transported concurrently. In an environment where a large number of video streams can be transported concurrently, this per-stream channel assignment and management may incur excessive overhead. In [9–11], aggregation of multiple video streams has been proposed as a solution to this problem. Like CRTT and PCRT, this approach reduces the rate variability by exploiting statistical multiplexing (spatial averaging). In this approach, a virtual channel is assigned to aggregated video streams, and thus, channel management is simpler. Krunz and Tripathi [9] solved the bandwidth-allocation problem for multiplexed video streams using a time-varying traffic envelope. To reduce the bandwidth requirement, they arranged the start times of component video streams in such a way that $I, P$, and $B$ frames evenly coincide. Such an arrangement may be possible to implement in a single-hop network but is extremely difficult, if not impossible, to implement in a multi-hop network. Reisslein and Ross [10] proposed two call-admission schemes for a set of video streams, subject to a packet-loss constraint: one using the Central Limit Theorem (CLT) and the other using the large deviation approximation. Unlike the previous approaches, their approach provides statistical packet loss/delay guarantees. However, they used a bufferless system in their call-admission schemes, and thus, network utilization is low as compared to a buffered system which is the case in most VoD systems. Liew and Chan [11] proposed a method for transmitting multiple stored VBR video streams over a common CBR channel. Although their approach uses a single CBR channel for multiple video streams, it requires dynamic control of the transport rates of individual component video streams in such a way that the sum of the component video streams' rates is equal to the transport rate of the CBR channel while each individual stream does not suffer buffer starvation. This means that the server must keep track of individual stream transmissions and change their transport rates a number of times during their lifetime, and thus, their approach doesn't scale well.

In this paper, we propose a simple and efficient VBR video stream transport mechanism called the *Play-Back Rate Transport with Minimum Rate Guaranteed* (PBRT-MRG), by exploiting *both* temporal averaging and statistical multiplexing. Rather than providing deterministic Quality-of-Service (QoS) guarantees, the proposed scheme provides statistical QoS guarantees to maximize network utilization. Temporal averaging is exploited through buffering both at the server and at receivers. To exploit the advantages of statistical multiplexing, we multiplex a set of video streams onto a common transport channel. Since a single transport channel is maintained for the multiplexed VoD channels, the proposed scheme is simpler than those in [5–8]. For the aggregated video streams which are averaged temporally and spatially, we present a simple transport mechanism and a call-admission scheme both of which together provide statistical QoS guarantees. For this, we introduce the concept of *statistical traffic envelope* and derive it using a histogram of rates. Using the proposed scheme, one can provide a VoD service without requiring any complex control while achieving both reasonable build-up delay and high bandwidth/buffer utilization.

The rest of this paper is organized as follows. Section 2 describes our system model. Sections 3 and 4 introduce some basic concepts necessary for the development of our approach, and describe the proposed approach to VoD services. In Section 5, our simulation results are shown to demonstrate the effectiveness of our approach in providing VoD services. The paper concludes with Section 6.

## 2 A Generic VoD System

As shown in Fig. 1, a generic VoD system consists of a video server, a network, and a set of receivers. A video server is equipped with a single storage or multiple storages. In either case, we assume that the server controls all of its storages, and thus, virtually there is only a single storage for the server. Compressed videos are stored in this storage. Since MPEG is one of the most popular compression standards for a VoD service, we assume MPEG-coded videos. The network can be very general as long as it provides the video server a guaranteed bandwidth. Some of candidate networks are ATM networks, FDDI, and Cable TV networks. For per-stream channel management like CRTT, each VoD stream must be guaranteed to have its own required bandwidth. In our approach, however, a bandwidth guarantee needs to be made only to the server. A receiver consists of a set-top box and a monitor (TV). The set-top box is equipped with a (receiver) buffer. When a receiver requests a certain video from the server, the server starts, and continues, transport of the video according to its transport schedule until the entire video data is transported. The server fragments each video frame into packets, and video packets are then transmitted through the network. Upon arrival of the first video frame, the set-top box stores the incoming video frames in its receiver buffer. After a certain delay called the *build-up delay*, the set-top box starts to play back the video on the monitor.

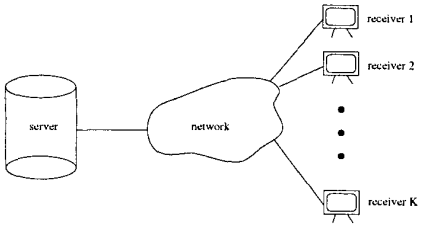


**Figure 1. A generic VoD system**

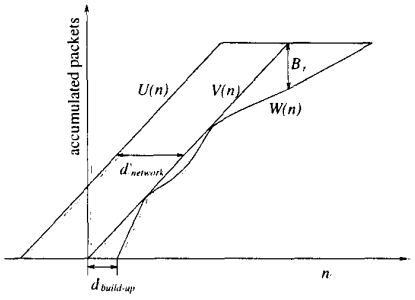


**Figure 2. Constant rate transmission and transport**

## 3 Transport of a Single VoD Stream

Before addressing our main problem, we discuss the basic concepts used by existing approaches to transporting and playing back a *single* VoD stream, beginning with CRTT. CRTT derives the transport and playback schedules of a single video using the knowledge of its frame size sequence. Here we assume that each video frame consists of a number of fixed-size packets. Let $N$ denote the number of frames of the video and $x(n)$ denote the number of packets in the $n^{th}$ video frame. Let $R$ (packets/frame) be the reserved bandwidth and $d_{build-up}$ be the build-up delay for this video. We can then derive the curves of transmission, reception, and playback, as shown in Fig. 2 in which the horizontal axis represents the time measured in number of frame intervals. We will henceforth use a discrete-time representation for both existing approaches and ours, and denote time 0 (origin) as the earliest time the first video frame arrives at the receiver. Then, the transmission curve $U(n)$ is defined as the cumulative sum of sizes of frames which had been transmitted during the period from $-d_{network}$ to time $n$. The reception curve $V(n)$ is defined as the cumulative sum of sizes of frames which have been received by the receiver until time $n$. Finally, the playback curve $W(n)$ is defined as the cumulative sum of sizes of frames which had already been played back at the receiver by time $n$.

The slope of $U(n)$ is $R$,[1] and once $U(n)$ reaches $\sum_{n=1}^{N} x(n)$, $U(n)$ remains constant. Video data transmitted by the server arrive at the receiver after traveling through the network and thus experiencing a delay. We assume that the network delay is constant throughout this paper.[2] Then, $V(n)$ is obtained by right-shifting $U(n)$ by the network delay $d_{network}$. Since the receiver starts to play back the video after a delay of $d_{build-up}$, the playback curve, $W(n)$, begins to increase at $d_{build-up}$. $W(n)$ can be expressed as:

$$W(n) = S(n - d_{build-up}), \tag{1}$$

where $S(n)$ represents the cumulative sum of sizes of frames which have been played back until $n$, assuming that the playback has started at time 0. Thus, we call $S(n)$ the *zero-delay receiver playback curve*.

Fig. 2 shows the key properties of transport and playback schedules. For example, in order to avoid starvation, the reception curve must always stay above the playback curve. In addition, the maximum vertical distance between the reception and playback curves indicates the minimum buffer requirement at the receiver, $B_r$, to avoid buffer overflow. Keeping in mind these properties, one can derive the relation among $B_r$, $R$, and $d_{build-up}$, as discussed in [5].

Although CRTT is simple in terms of implementation complexity, it may require a large receiver buffer or a large build-up delay because of its reliance on a single transport rate. To resolve this problem, several PCRTs have been proposed [5–8]. Although PCRTs are inefficient in terms of implementation complexity as argued in Section 1, it can reduce the buffer requirement significantly, depending on the number of transport-rate changes. In an extreme case, the server can change the transport rate every frame period in such a way that each video frame is transported within a single frame period. That is, a video stream is transported at the rate it would be played back if the server had played it back. We call this transport schedule the *Play-Back Rate Transport Schedule* (PBRT). Except that the bandwidth reserved for a video must be larger than its peak traffic generation rate — if we don't use bandwidth renegotiation [12] — PBRT has the following desirable features. First, since the frame size sequence of a video is already known, its transport schedule need not be calculated or stored. So, PBRT has lower computation complexity and memory requirement. Second, since a frame arrives at a receiver every frame period, the receiver need not build up its buffer before starting to play back a video. As soon as the first frame arrives, the receiver can start to play back a video. As a result, PBRT does not require any build-up delay, hence unneeding the receiver buffer for building up frames. Fig. 3 illustrates the transport and playback schedules of PBRT. Since video frames are transported by the server at the receiver's playback rate, the transmission curve, $U(n)$, has the same shape as the zero-delay receiver playback curve, $S(n)$, except that

[1] since $R$ packets are transmitted in every frame interval.

[2] When the network delay is variable, the build-up delay and the receiver-buffer size must be increased to accommodate delay jitter. Since this requires only minor changes to our approach, we use the assumption for the convenience of our discussion.

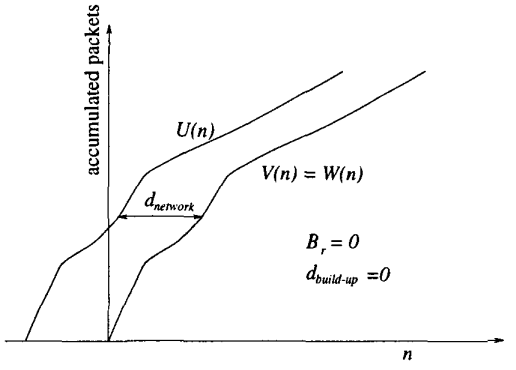


**Figure 3. Playback rate transport schedule**

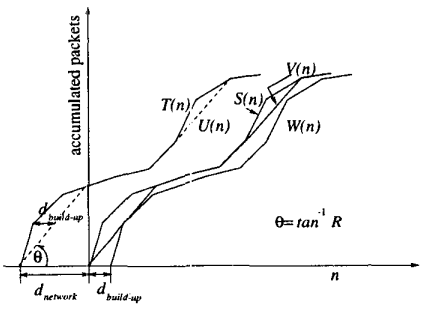


**Figure 5. Playback rate transport schedule with minimum rate guaranteed**

it starts to increase at time $-d_{network}$. The reception curve, $V(n)$, is equal to $S(n)$ because video frames experience a constant delay, $d_{network}$. Lastly, since the received frames are played back immediately without delay, the playback curve, $W(n)$, is identical to the reception curve, $V(n)$, and thus, both the build-up delay, $d_{build-up}$, and the minimum receiver-buffer requirement, $B_r$, are equal to zero.

Now, we present a modified version of PBRT called the *Play-Back Rate Transport Schedule with Minimum Rate Guaranteed* (PBRT-MRG). Fig. 4 depicts the operation of PBRT-MRG for a single VoD stream. PBRT-MRG has a buffer named the *server-buffer* at the server. Let $B_s$ denote the size of the server-buffer. As the server "dumps" an entire frame into the network every frame period in PBRT, the server dumps an entire frame into the server-buffer every frame period in PBRT-MRG (we call this operation *server playback*). The video frames stored in the server-buffer are transmitted at a constant rate of $R$ (packets/frame). $R$ can be less than the peak-traffic generation rate of the video. $B_s$ must be large enough to avoid buffer overflow. The receiver builds up the first several video frames at its receiver-buffer and starts to play back the video as in CRTT.

As with PBRT, PBRT-MRG does not require calculation of a transport schedule, nor is required to change the transport rate according to a pre-recorded sequence. Therefore, PBRT-MRG has an implementation complexity similar to that of PBRT although it requires additional buffer space (server-buffer).

To find the relations among minimum buffer requirements of both the server and the receiver, build-up delay, and the necessary bandwidth, let's consider Fig. 5 which shows transport and playback schedules. First, let's examine the server playback curve, $T(n)$, which is defined as the cumulative sum of sizes of frames which were dumped into the server-buffer by the server at the rate of playback until $n$. So, $T(n)$ is obtained by left-shifting the zero-delay receiver playback curve, $S(n)$, by $d_{network}$. Next, let's examine the transmission curve, $U(n)$. When the increasing rate of $T(n)$ is larger than $R$, a backlog is formed at the server-buffer. In such a case, $U(n)$ increases at the rate of $R$ (packets/sec) until the backlog is cleared. When the backlog is cleared, $U(n)$ starts to keep up with $T(n)$. The dotted line in Fig. 5 indicates $U(n)$. The reception curve, $V(n)$, is obtained simply by right-shifting $U(n)$ by $d_{network}$. Finally, the receiver playback curve, $W(n)$, is obtained by right-shifting $S(n)$ by $d_{build-up}$ as shown in Eq. (1).

To avoid starvation, as in CRTT, the build-up delay, $d_{build-up}$, must be chosen such that

$$W(n) \leq V(n), \quad \text{for all } n. \tag{2}$$

To derive the build-up delay which satisfies this condition, we note that $W(n)$ is a right-shifted version of $S(n)$ whose shift is equal to $d_{build-up}$. To satisfy Eq. (2), we must right-shift $S(n)$ at least by the maximum horizontal distance between $S(n)$ and $V(n)$. Therefore, the build-up delay must be larger than, or equal to, the maximum horizontal distance between $S(n)$ and $V(n)$. Formally,

$$d_{build-up} \geq \max_{0 \leq n \leq N} \min\{m : 0 \leq m \leq N - n \text{ and } S(n) \leq V(n+m)\}. \tag{3}$$

Now, since $T(n)$ and $U(n)$ are obtained by left-shifting $S(n)$ and $V(n)$ by $d_{network}$, respectively, the maximum horizontal distance between $T(n)$ and $U(n)$ also indicates the minimum build-up delay. Then, since the maximum horizontal distance between $T(n)$ and $U(n)$ indicates the maximum backlog clearing time of the server-buffer, the minimum build-up delay is equal to the maximum backlog clearing time of the server-buffer, that is, the worst-case queuing delay that a packet experiences in the server-buffer. This implies that all the packets arrive at the receiver-buffer within the minimum build-up delay, and thus, they are available for playback at the receiver without causing starvation. Thus, the minimum build-up delay can be derived by analyzing the worst-case traffic arrival scenario at the server-buffer.

In order to analyze the worst-case delay and backlog characteristics of the server-buffer, we use the concept of *traffic envelope* which has been widely used for analyzing the delay and backlog characteristics of a buffered multiplexer based on its worst-case traffic arrival scenario [13,

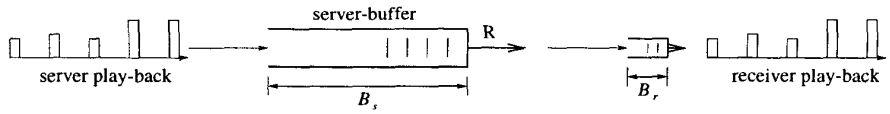


**Figure 4. Operational overview of PBRTS-MRG for a single VoD stream**

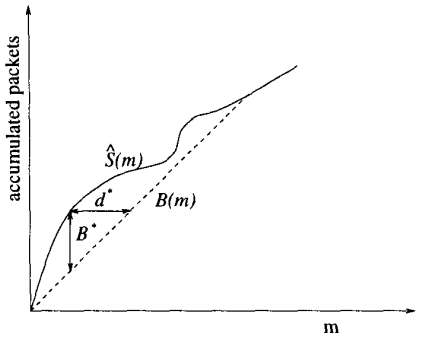


**Figure 6. Deterministic traffic envelope**

14]. It is commonly called a *deterministic* traffic envelope in that it describes a deterministic upper bound on the amount of one or more streams' traffic. Specifically, we define a traffic envelope of a single video, $\hat{S}(\eta)$, such that

$$\hat{S}(\eta) = \max_{n \geq 0}\{S(n+\eta) - S(n)\}, \quad \eta \geq 0. \tag{4}$$

Then, $\hat{S}(\eta)$ indicates the maximum number of packets that can arrive at the server-buffer for any $\eta$ frame intervals. Now, in order to calculate the worst-case packet delay at the server-buffer, let's assume that a busy period of the server-buffer starts at time 0, without loss of generality, that is, the video stream has started to be sent at time 0. Let's assume that the busy period ends at time $k$, then the number of packets that can arrive in an interval, $[0,m)$, $0 \leq m \leq k$, is bounded by $\hat{S}(m)$ according to the definition of the traffic envelope. Therefore, the maximum number of packets that can arrive at the buffer by time $m$ (starting from time 0) is given by $\hat{S}(m)$. During this period, the buffer is drained at the rate of $R$ packets/frame. So, the number of packets served by time $m$ starting from time 0, $B(m)$, is given by

$$B(m) = Rm. \tag{5}$$

$B(m)$ is called the *service curve*. Both $\hat{S}(m)$ and $B(m)$ are shown in Fig. 6. Then, the maximum horizontal distance between $\hat{S}(m)$ and $B(m)$ indicates the worst-case packet delay, and the maximum vertical distance between $\hat{S}(m)$ and $B(m)$ indicates the worst-case backlog size, which represents the minimum buffer requirement for no packet loss (or no buffer overflow). Formally, the worst-case delay, $d^*$, and the worst-case (or maximum) backlog, $B^*$, are, respectively, given by

$$d^* = \max_{0 \leq m \leq m^*} \min\{\eta : \eta \geq 0 \text{ and } \hat{S}(m) \leq B(m+\eta)\}, \tag{6}$$

and

$$B^* = \max_{0 \leq m \leq m^*}\{\hat{S}(m) - B(m)\}, \tag{7}$$

where $m^*$ represents the length of the longest busy period. One need not calculate $d^*$ and $B^*$ independently using Eqs. (6) and (7). A packet's queuing delay at the server-buffer is equal to the clearing time of the backlog that has existed at the instant the packet has arrived at the server-buffer. Therefore, the worst-case packet delay is equal to the worst-case backlog clearing time. Since the backlog of the server-buffer is drained at a constant rate, $R$ packets/frame, the worst-case backlog clearing time is equal to the clearing time of the worst-case backlog whose size is $B^*$. Hence, we have the following relation:

$$d^* = \lceil \frac{B^*}{R} \rceil \text{ (frame intervals)}, \tag{8}$$

where $\lceil \cdot \rceil$ denotes the ceiling function. As discussed earlier, $d^*$ represents the minimum build-up delay, i.e..

$$d_{build-up} \geq d^*. \tag{9}$$

In addition, $B^*$ indicates the minimum server-buffer requirement to avoid buffer overflow, and hence we set $B^*$ to $B_s$.

Now, let's consider the minimum buffer requirement $B_r$ at the receiver to avoid buffer overflow. It is given by the maximum vertical distance between $V(n)$ and $W(n)$ in Fig. 5. Formally,

$$B_r = \max_{n \geq d_{build-up}}\{V(n) - W(n)\}. \tag{10}$$

In order to obtain $B_r$ using Eq. (10), one must derive $V(m)$, which is non-trivial. Instead of using the exact minimum receiver-buffer requirement, one may use an upper bound of the minimum buffer requirement. To derive such a bound, we note the following relation between $S(n)$ and $V(n)$, as shown in Fig. 5:

$$V(n) \leq S(n). \tag{11}$$

This relation implies that

$$V(n) - W(n) \leq S(n) - W(n). \tag{12}$$

Using Eq. (1), we obtain

$$V(n) - W(n) \leq S(n) - S(n - d_{build-up}). \tag{13}$$

Then,

$$\max_{n \geq d_{build-up}}\{V(n) - W(n)\} \leq \max_{n \geq d_{build-up}}\{S(n) - S(n - d_{build-up})\}.$$

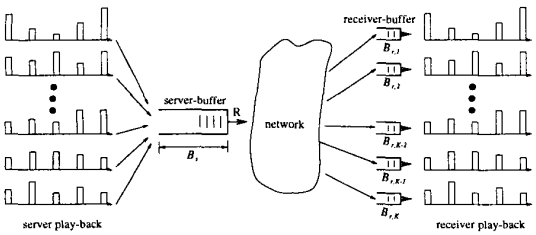


**Figure 7. Operational overview of PBRTS-MRG for aggregated VoD streams**

Using Eq. (4), an upper bound of $B_r$ is obtained by

$$B_r \leq \hat{S}(d_{build-up}). \tag{14}$$

Since we have already calculated $\hat{S}(m)$ in order to derive $B^*$ in Eq. (7), this bound can be obtained without any extra computation. $\hat{S}(d_{build-up})$ represents the maximum number of packets that can arrive during any period of length $d_{build-up}$.

## 4 Transport of Aggregated VoD Streams

In the previous section, we discussed the case of transporting a single VoD stream as a precursor to our main problem of transporting aggregated VoD streams, which is the subject of this section.

### 4.1 Deterministic Traffic Envelope

Fig. 7 depicts the operation of PBRT-MRG for aggregated VoD streams, $\{v_j\}_{j=1}^{K}$. In PBRT-MRG, transporting aggregated VoD streams is basically the same as transporting a single VoD stream except that packets generated from aggregated VoD streams are multiplexed at the server-buffer, as shown in Fig. 7. Packets from all the component VoD streams are "dumped" into the server-buffer at their own playback rates. Each video frame can be dumped at an arbitrary time as long as all the packets of a frame are dumped within a frame period.[3] The VoD packets stored at the server-buffer are transmitted at the rate of $R$ (packets/frame) and delivered to their receivers. Thus, PBRT-MRG still does not require any complex server operation. We assume that at least $R$ (packets/frame) is reserved for the VoD service in the network.

Now, we want to find the server-buffer size, $B_s$, a VoD stream $v_i$'s build-up delay, $d_{build-up}(i)$, and $v_i$'s receiver-buffer size, $B_{r,i}$, for a given $R$, under the assumption that the VoD streams' frame size sequences are known *a priori*.

[3]Since a discrete-time representation is used, the frame boundaries of all the component video streams are implicitly assumed to be synchronized. However, our approach can be described with, and used for, a continuous-time representation.

First, we derive $v_i$'s receiver-buffer size, $B_{r,i}$. For this, we assume knowledge of $d_{build-up}(i)$, and let $v_i$'s server playback curve, transmission curve, zero-delay receiver playback curve, reception curve, and receiver playback curve be denoted by $T_i(n)$, $U_i(n)$, $S_i(n)$, $V_i(n)$, and $W_i(n)$, respectively. Then, they have the same relation as shown in Fig. 5. In particular,

$$W_i(n) \leq V_i(n) \leq S_i(n), \quad \text{for all } n. \tag{15}$$

One can then obtain $B_{r,i}$ by taking the maximum vertical distance between $V_i(n)$ and $W_i(n)$ as we argued in the case of single stream PBRT-MRG transport if $V_i(n)$ is known. However, in the aggregate stream PBRT-MRG transport, $V_i(n)$ changes depending on the arrival patterns of the other VoD streams multiplexed together. Hence, for $B_{r,i}$, we employ an upper bound of the receiver-buffer requirement which depends only on $v_i$'s frame size sequence. In the previous section, we had already derived such a bound for the single stream PBRT-MRG transport in Eq. (14). For the aggregate stream PBRT-MRG transport, we can use the same bound:

$$B_{r,i} = \hat{S}_i(d_{build-up}(i)), \tag{16}$$

where $\hat{S}_i(\eta)$ denotes the traffic envelope of $v_i$. Conceptually, this buffer requirement represents the worst-case backlog that can be formed in the receiver-buffer in any aggregate traffic arrival scenario for any time interval equal to $d_{build-up}(i)$. This is because packets arrived during the past $d_{build-up}(i)$ frame intervals can co-exist in the receiver-buffer and $\hat{S}_i(d_{build-up}(i))$ indicates the maximum number of packets that can arrive during that period. The fact that the receiver-buffer requirement does not depend on the number of aggregated VoD streams is important for the scalability of a VoD system. In a typical VoD system, most set-top boxes at viewers' locations are likely to have limited buffers which may not be expandable. Therefore, it is important to make the receiver-buffer requirement independent of the number of aggregated VoD streams.

Next, let's find $B_s$ and $d_{build-up}(i)$ that will ensure no buffer starvation when $K$ VoD streams are aggregated. In the aggregate stream PBRT-MRG transport, the component VoD streams can have different start times. In this environment, the server playback curve depends on the component VoD streams' start times since it is equal to the sum of the component VoD streams' server playback curves which depend on their start times. Then, if the start times of all the component VoD streams are known in advance, we can obtain the aggregate VoD streams' server playback curve from which its transmission curve can be obtained. Finally, $d^*$ and $B^*$ which correspond to the minimum build-up delay and the server-buffer requirement, respectively, can be obtained from the server playback curve and the transmission curve. However, it is unrealistic to assume that the start times are known in advance. Under the more realistic assumption that VoD streams may start at arbitrary times, we must use the worst-case traffic arrival scenario in order

to obtain the build-up delay and the buffer requirement. In this case, we naturally resort to the traffic envelope.

Let $\hat{S}_j(\eta)$ denote the traffic envelope of $v_j$. Then, the aggregate VoD streams' traffic envelope is given by $\sum_{j=1}^{K} \hat{S}_j(\eta)$, i.e., the sum of component streams' traffic envelopes. This is because the aggregate VoD streams' traffic envelope represents the number of packets that can arrive during any $\eta$ frame intervals in the worst case and the worst case happens when all the component VoD streams are in their worst cases. Therefore, we obtain $B^*$, and thus, $B_s$ and $v_i$'s minimum build-up delay by replacing $\hat{S}(\cdot)$ with $\sum_{j=1}^{K} \hat{S}_j(\cdot)$ in Eqs. (7) and (8). Thus,

$$B_s = \max_{0 \le m \le m^*} \{\sum_{i=j}^{K} \hat{S}_j(m) - Rm\}, \tag{17}$$

and

$$d_{build-up}(i) = \lceil \frac{1}{R} \max_{0 \le m \le m^*} \{\sum_{j=1}^{K} \hat{S}_j(m) - Rm\} \rceil, \tag{18}$$

where $m^*$ denotes the length of the largest busy period in all multiplexing scenarios.

## 4.2 Statistical Traffic Envelope

A desirable VoD system must have as small build-up delay and buffer requirements (both at the server and the receiver) as possible while efficiently utilizing bandwidth. So, we need to find as tight a traffic envelope as possible. However, since a traffic envelope is, by definition, a deterministic function, it is impossible to derive a tighter bounding function. The only way to evade this problem is to lower the level of QoS guaranteed to the VoD streams. By providing statistical (instead of absolute) QoS guarantees, one can lower the resource requirement for a VoD service.

In this approach, QoS requirements are described in probabilistic (instead of deterministic) terms and a certain percentage of packet losses and/or deadline misses are allowed. In particular, in a VoD system, the starvation probability can be used to describe QoS requirements.

Since a certain percentage of losses and starvations are allowed in the statistical approach, we can develop a smaller (and hence tighter) traffic envelope. Especially, for the same loss tolerance, the more VoD streams we aggregate, the tighter traffic envelope we get, because individual VBR VoD streams' long tail probability distributions are smoothed out as the number of aggregated VoD streams increases.

While the deterministic traffic envelope of aggregated VoD streams is obtained simply by summing the deterministic traffic envelopes of the component VoD streams, the statistical traffic envelope must be derived based on the statistical characteristics of the aggregated VoD streams. The statistical traffic envelope of aggregated VoD streams is defined as follows:

**Definition 1** *When the number of packets arrived from an aggregate of VoD streams during an interval, $[n, n+\eta)$, is given by $A[n, n+\eta]$, their statistical traffic envelope, $\tilde{A}(\eta)$, for loss tolerance $Z$ is defined as:*

$$\tilde{A}(\eta) \stackrel{def}{=} \min[x : Pr\{A[n, n+\eta] > x\} < Z, \forall n, \eta \ge 0].$$

The physical meaning of Definition 1 is that the number of packets generated by the aggregated VoD streams during any time interval of length $\eta$ is smaller than, or equal to, $\tilde{A}(\eta)$ with probability no less than $1 - Z$.

Now, we derive an upper-bound of the starvation probability for the aggregated VoD streams using the statistical traffic envelope.

**Theorem 1** *Using $\tilde{A}(\eta)$, we define:*

$$B^*_{stat} \stackrel{def}{=} \max_{\eta \ge 0} \{\tilde{A}(\eta) - R\eta\}. \tag{19}$$

*Then, the probability that the backlog at the server-buffer is smaller than, or equal to, $B^*_{stat}$ is no less than $1 - Z$.*

*Proof:* Without loss of generality, we assume that a busy period of the server-buffer starts at time 0 and ends at time $n$. For any $\eta$, $0 \le \eta \le n$, the number of packet arrivals at the server-buffer during $[0, \eta)$ is at most $\tilde{A}(\eta)$ with probability $1 - Z$ or larger according to Definition 1. Since $R\eta$ packets are guaranteed to be transmitted from the server-buffer during $[0, \eta)$, the backlog at the server-buffer is at most $\tilde{A}(\eta) - R\eta$ with probability $1 - Z$ or larger. Since, by definition,

$$B^*_{stat} \ge \tilde{A}(\eta) - R\eta, \tag{20}$$

the backlog at the server-buffer is upper bounded by $B^*_{stat}$ with probability $1 - Z$ or larger. □

Theorem 1 says that if we set the server-buffer size to $B^*_{stat}$, the server-buffer overflow probability is bounded by $Z$. When a server-buffer overflow happens, or is expected to happen because of the full server-buffer, the server can handle it using one of the following two options. The first option is to drop the incoming video frames when the server-buffer is full. Then, these frames will not be played back at the receivers. The second option is to pause the dumping process until enough room becomes available in the server-buffer, and then resume it. This will cause the receiver to temporarily stop and then resume playback. Rather than losing video frames, temporarily stopping seems more desirable, and hence, making the second option preferable in handling server-buffer overflow. However, irrespective of the option choice, the lost or delayed frames contribute to buffer starvation since they did not arrive at the receiver before their playback times. Therefore, the server-buffer overflow probability represents the starvation probability if there were no other factors that cause service disruption.

Now, let's assume that $B_s$, $d_{build-up}(i)$, and $B_{r,i}$ satisfy

$$B_s \geq B^*_{stat}, \tag{21}$$

$$d_{build-up}(i) \geq \frac{B^*_{stat}}{R}, \tag{22}$$

and

$$B_{r,i} \geq \hat{S}_i(d_{build-up}(i)). \tag{23}$$

Then, $v_i$'s starvation probability is guaranteed to be less than $Z$ according to Theorem 1 and the above argument.

Note that we used the deterministic traffic envelope in deriving the minimum receiver-buffer size, because, as discussed in the previous subsection, the receiver-buffer requirement depends only on $v_i$'s traffic characteristics, and thus, it is meaningless to derive a statistical traffic envelope for a single stream. Moreover, for the same reason, the receiver-buffer requirement does not grow with the number of aggregated VoD streams and can be controlled to remain reasonably small if the build-up delay remains small. Lastly, note that we don't need to derive $\hat{S}_i(\cdot)$ for all the intervals. We need $\hat{S}_i(\cdot)$ for $d_{build-up}(i)$ which is relatively small as will be shown in Section 5.

Before deriving a statistical traffic envelope, we briefly discuss connection admission control. Let's consider a VoD system which is already serving streams, $\{v_j\}_{j=1}^{k-1}$, whose build-up delays are $\{d_{build-up}(j)\}_{j=1}^{k-1}$, respectively. When a new VoD stream, $v_k$, is requested by a user $u_i$ whose receiver-buffer size is $B_{r,i}$, the service provider obtains $v_k$'s build-up delay $d_{build-up}(k)$ from the following condition:

$$B_{r,i} \geq \hat{S}_k(d_{build-up}(k)), \tag{24}$$

where $\hat{S}_k(\cdot)$ is $v_k$'s deterministic traffic envelope. This condition is necessary to avoid overflow of $u_i$'s receiver-buffer. The service provider selects as large $d_{build-up}(k)$ as possible among those that satisfy Eq. (24) unless it exceeds the maximum build-up delay requested by the end user. By choosing a larger build-up delay, the service provider increases future requests' acceptance probability as can be seen in the next criteria. Next, the service provider checks the following condition:

$$\min_{j=1,2,\ldots,k} d_{build-up}(j) \geq \frac{B^*_{stat}}{R}, \tag{25}$$

where $R$ is the bandwidth reserved for servicing the server-buffer and $B^*_{stat}$ is the statistical traffic envelope of the aggregated VoD stream consisting of $v_1$, $v_2$, ..., $v_k$. $\min d_{build-up}(j)$ must be stored for fast admission control of future requests. If this condition is not satisfied, the service provider increases $R$ by reserving more bandwidth. If this is not possible, $u_i$'s request must be rejected. Finally, the service provider verifies whether Eq. (21) is met. If it is not, the service provider increases $B_s$. If it is impossible to meet Eq. (21), $u_i$'s request must be rejected; else, $u_i$'s request is granted.

When an existing VoD stream is disconnected for some reason, the service provider can reduce $B_s$ and $R$ using new $B^*_{stat}$ as long as new $B_s$ and $R$ satisfy Eqs. (21) and (25), where the disconnected stream must be excluded from the calculation of the minimum build-up delay and $B^*_{stat}$ in Eq. (25).

### 4.3 Deriving a Statistical Traffic Envelope

Now, the remaining problem is to derive a statistical traffic envelope that satisfies Definition 1. One way to derive a statistical traffic envelope for an aggregate of traffic sources is to use a stochastic-bounding approach [15, 16]. The idea of this bounding approach is to find a random process which stochastically bounds the amount of traffic generated by the sources, and use it for analysis. Selection of such a random process is critical in estimating the amount of resources necessary to achieve a given level of QoS, but no general solution to it has been reported thus far.

In [17], on/off periodic sources with uniformly-distributed independent phases were used to find a worst-case network behavior. Each on/off periodic source is supposed to bound one of the original traffic sources in terms of traffic generation rate. One can adopt this on/off periodic process as a bounding process and find a statistical traffic envelope by using appropriate stochastic bounds, such as the Chebychev or Chernoff bound. However, our evaluation has shown this approach to be too pessimistic, due mainly to the conservative nature of the bounding approach in estimating the amount of traffic, as pointed out in [4]. Specifically, even when the length of an observed interval was increased slightly, the derived statistical traffic envelope got much larger than the deterministic traffic envelope of the original source. This is because an on/off bounding random process shows extremity in traffic burstiness irrespective of the length of an interval during which the traffic is observed, unlike the deterministic traffic envelope that exhibits clear smoothing effects with a longer interval. As a result, the derived statistical traffic envelope was of little use in saving resources. The stochastic-bounding approach using an on/off periodic random process may serve as a guideline for estimating the required network resources when the source traffic is random or unknown, but does not give any efficient solution when the source traffic is somewhat deterministic as in VoD applications.

In [18], Knightly derived the maximum variance of random processes subject to a given deterministic traffic envelope. Then, he utilized these variances in the CLT-based admission control. One can employ a similar approach in deriving a statistical traffic envelope of aggregated VoD streams. Instead of deriving a variance-maximizing frame size sequence as used in [18], in a VoD system, we can use the frame size sequence of each individual VoD stream directly in calculating the empirical variance of each stream's packet arrival rate over a certain period. Then, by applying the CLT, we can derive the aggregated streams' statis-

tical traffic envelope from the individual streams' variances and means. However, as pointed out in [10], the CLT-based approach proved to yield optimistic packet-loss estimates when the packet-loss tolerance is very small, according to our evaluation.

In order to obtain pessimistic packet-loss estimates, one may use the theory of large deviations as in [10]. This approach involves solving an exponential equation which is derived from a logarithmic moment generating function. Since this equation cannot be solved analytically in most cases, its usefulness is limited when the packet-arrival rate has a complex probability distribution function (PDF), which is the case in most practical problems. Especially in a VoD system, the PDF of the packet arrival rate is derived from the actual frame size sequence using a histogram with a number of bins. Therefore, the PDF takes a complex form. For this reason, using the large deviation approximation approach is impractical to use for derivation of a statistical traffic envelope for a VoD system.

Instead of using an approximation, we derive the exact PDF of the packet-arrival rate of the aggregated VoD streams by taking the convolution of the probability mass functions (pmfs) of component VoD streams in order to derive a statistical traffic envelope. One may argue that this convolution-based approach is too complex to be practical. But if each component stream's histogram has a small number of bins, the convolution can be done within a reasonable amount of time. Through a simulation study, we show that an effective statistical traffic envelope can be obtained using such a histogram.

In order to derive a statistical traffic envelope, let's assume that the number of aggregated VoD streams is $K$ and let the frame size sequence of a VoD stream $v_i$ be $\{x_i(n)\}_{n=1}^{N_i}$, where $N_i$ is the number of frames of $v_i$. Then, we can derive the pmf of the number of packets which arrived from $v_i$ during a single frame period, from the histogram of $(x_i(1),\ldots,x_i(N_i))$. We denote this pmf by $f_i(1)$.

Next, we derive the pmf of the number of packets which arrived from $v_i$ during $m$ consecutive frame intervals. For this, we form a new frame size sequence by summing $m$ consecutive frame sizes. The derived sequence is $(\sum_{j=1}^{m} x_i(j), \sum_{j=2}^{m+1} x_i(j), \ldots, \sum_{j=N_i-m+1}^{N_i} x_i(j))$. We derive the pmf of this new sequence using its histogram. Thus, we obtain a set of pmfs, $\{f_i(1),\ldots,f_i(N_i)\}$ for $v_i$.

In order to derive the pmf of the number of packets which arrived from the aggregated streams during a single frame period, we simply execute the convolution of $f_1(1), f_2(1), \ldots$, and $f_K(1)$. We denote the convolution of $a(1), a(2), \ldots$, and $a(K)$ by $convolution(a(1), a(1), \ldots, a(K))$. Then, the pmf of the number of packets which arrived from the aggregated streams during a single frame period, $g(1)$, is given by $convolution(f_1(1), f_2(1), \ldots, f_K(1))$. Similarly, the pmf of the number of packets which arrived from the aggregated streams during $m$ consecutive frame intervals, $g(m)$, is given by $convolution(f_1(m), f_2(m), \ldots, f_K(m))$, where $m = 2, \ldots, \min_i N_i$. If all $f_i(m)$ have $l$ elements, respectively, i.e., their corresponding histograms have $l$ bins, $g(m)$ has $l + (l-1)(K-1)$ elements.

By integrating $g(m)$, we can derive the PDF of the number of packets of the aggregated VoD streams that arrived during $m$ consecutive frame intervals where $m = 1, \ldots, \min_i N_i$. Let $G_m(\cdot)$ denote the PDF. Then, the statistical traffic envelope during $m$ frame intervals is given by

$$\tilde{A}(m) = G_m^{-1}(1-Z), \tag{26}$$

where $Z$ is the target loss tolerance.

Note that the computational complexity of $\tilde{A}(m)$ increases linearly with the size of observation period, $m$. Thus, we want to determine the maximum observation period for $\tilde{A}(m)$ which is needed to derive $B^*_{stat}$. Strictly speaking, it should be derived for the longest busy period of the aggregated VoD streams as shown in Eq. (17), which must be derived from the deterministic traffic envelope. However, for practical purposes, we may use the statistical traffic envelope for this purpose as an approximation. The length of this approximated longest busy period is given by

$$\tilde{m}^* = \min\{\eta : \tilde{A}(\eta) \leq R\eta\}. \tag{27}$$

## 5 Evaluation

In order to demonstrate the effectiveness of the proposed statistical traffic envelope, we conducted trace-driven simulations with the frame size sequence of the MPEG-coded *Star Wars* movie, available from an FTP site [19]. The movie was compressed at a rate of 24 frames/sec. The number of frames in the sequence is 174136, which corresponds to approximately a 2-hour runtime. The frame size is represented in number of packets, where the size of each packet is fixed to 48 bytes.[4] The mean of the sequence is 41.12 packets, and the maximum is 483 packets. The standard deviation is 47.31.

We considered three multiplexing scenarios in which 100, 50, and 20 *Star Wars* streams were multiplexed. Multiplexed streams have random start times. For each multiplexing scenario, we derived the statistical traffic envelopes of the aggregated streams for a set of loss tolerances (upper-bounds of the starvation probability) ranging from $10^{-1}$ to $10^{-8}$, using the procedure discussed in the previous section. The derived statistical traffic envelopes were then used to calculate server-buffer requirements for different server-buffer clearing rates, using Eq. (19). We employed a 10-bin histogram. Therefore, the pmf of the aggregated streams consisting of 100 *Star Wars* streams has 901 elements. In Fig. 8, estimated buffer requirements vs. loss tolerances are plotted for different server-buffer clearing rates,

[4] This choice is somewhat arbitrary, but our conclusions will not be affected by this choice.

$R$ when 100 *Star Wars* streams were multiplexed. In the figure, we show normalized buffer requirements, instead of actual server-buffer requirements, which are defined as actual server-buffer requirements divided by the number of multiplexed streams. By using normalized buffer requirements, we can compare the three different multiplexing scenarios in terms of per-stream buffer requirements. $R$ is shown on top of each figure, and set to 1.1, 1.2, 1.3, and 1.4 times the average packet arrival rate of the aggregated streams for each case. Note that the horizontal axis represents normalized buffer size while the vertical axis represents loss tolerance.

Using the estimated buffer sizes, we conducted simulations as follows: As shown in Fig. 7, we multiplexed *Star Wars* streams with each having random starting points onto a server-buffer with the estimated size, and measured the buffer overflow probability which represents the starvation probability, as we argued in Section 4.2. The length of the measured period is 20,000 frame intervals, and the measurement was taken for 10,000 multiplexing cases. The measured starvation ratios are also shown in Fig. 8 when 100 streams were multiplexed. The small vertical lines between the x marks indicate the 95 % confidence intervals of the measured starvation ratios.[5] As proved in the previous section, the measured starvation ratios are upper bounded by the loss tolerances. In the figure, the measured starvation ratios are missing for small starvation ratios (loss tolerances) for which we were unable to observe any starvation. For the purpose of comparison, we derived normalized buffer estimates for zero loss tolerance for each $R$, using the deterministic traffic envelope. They are $1.25 \times 10^7$, $8.81 \times 10^6$, $5.29 \times 10^6$, and $1.95 \times 10^6$ bytes for each case, respectively. This result shows that by allowing a small starvation probability, *e.g.*, $10^{-8}$, we can reduce the buffer requirement by as much as three orders of magnitude.

From the server-buffer sizes, we derived build-up delays using Eq. (18). When server-buffer clearing rate was given as 1.1 times the average packet arrival rate of the multiplexed streams, the maximum build-up delay was only 4.64 seconds which was obtained for the loss tolerance of $10^{-8}$. When server-buffer clearing rate was larger than 1.1 times the average packet arrival rate, build-up delays were $1/24$ sec. or one frame interval, for all loss tolerances, meaning that only a single frame is required to be stored at the receiver. These small build-up delays directly affect receiver-buffer requirements since they are linearly proportional. Thus, when the server-buffer clearing rate was given as 1.1 times the average packet arrival rate of the multiplexed streams, the maximum buffer requirement was only $6.12 \times 10^5$ bytes which was obtained for the loss tolerance of $10^{-8}$. In other cases, buffer requirement was $2.318 \times 10^4$ bytes. As we claimed in Section 4.2, receiver-buffers are small enough to use for regular set-top boxes.

[5] In Figs. 8(b), 8(c), and 8(d), the lower bounds of some 95 % confidence intervals are shown to be $10^{-9}$, but their actual values are zeros.

# 6 Conclusion

In this paper, we presented a new scheme called PBRT-MRG for transporting VBR VoD streams. PBRT-MRG is simple to implement since its transport mechanism does not require calculation of any complex transport schedule while enhancing resource utilization, by allowing multiple streams to be transported concurrently. In order to increase resource utilization, PBRT-MRG exploits both intra-stream and inter-stream correlations in its admission control. In particular, resource utilization can be improved dramatically by making statistical (instead of deterministic) QoS guarantees. To provide statistical QoS guarantees using PBRT-MRG, we proposed and used the concept of a statistical traffic envelope, and derived it using histograms of individual VoD streams. Even with a 10-bin histogram, the statistical approach is shown to provide tight starvation probability estimates as compared to the deterministic approach.

# References


[1] P. Skelly, S. Dixit, and M. Schwartz, "A histogram-based for video traffic behavior in an ATM network node with an application to congestion control," in *Proc. of IEEE INFOCOM*, pp. 95–104, 1992.

[2] M. Krunz, R. Sass, and H. Hughes, "Statistical characteristics and multiplexing of MPEG streams," in *Proc. of IEEE INFOCOM*, pp. 455–462, 1995.

[3] N. Shroff and M. Schwartz, "Video modeling within networks using deterministic smoothing at the source," in *Proc. of IEEE INFOCOM*, pp. 342–349, 1994.

[4] S.-K. Kweon and K. G. Shin, "Transport of mpeg video with statistical loss and delay guarantees in atm networks using a histogram-based source model," in *Proc. of IEEE RTSS*, December 1999.

[5] J. M. McManus and K. W. Ross, "Video-on-Demand over ATM: constant-rate transmission and transport," *IEEE J. Select. Areas Commun.*, vol. 14, no. 6, pp. 1087–1098, August 1996.

[6] J. D. Salehi, Z.-L. Zhang, J. Kurose, and D. Towsley, "Supporting stored video: reducing rate variability and end-to-end resource requirements through optimal smoothing," in *Proc. of ACM SIGMETRICS*, pp. 222–231, 1996.

[7] W.-C. Feng, F. Jahanian, and S. Sechrest, "Optimal Buffering for the Delivery of Compressed Prerecorded Video," in *Proc. of IASTED/ISMM International Conference on Networks*, 1996.

[8] Z. Jiang and L. Kleinrock, "A general optimal video smoothing algorithm," in *Proc. of IEEE INFOCOM*, pp. 676–684, 1998.

[9] M. Krunz and H. Tripathi, "Impact of video scheduling on bandwidth allocation for multiplexed MPEG streams," *Multimedia Systems Journal*, vol. 5, no. 6, pp. 47–57, December 1997.

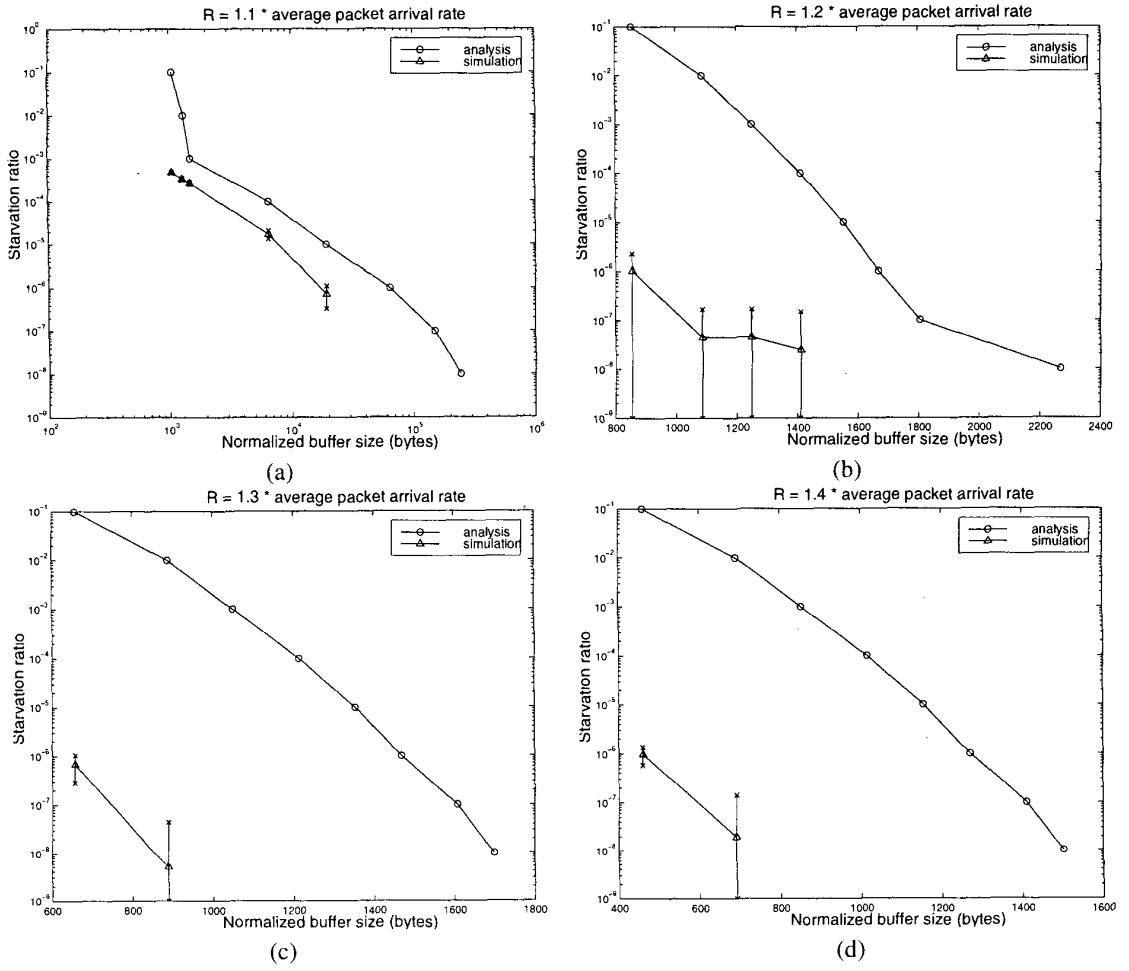


**Figure 8. Starvation ratio vs. normalized server-buffer size when 100 streams are multiplexed**


[10] M. Reisslein and K. W. Ross, "Call admission for prerecorded sources with packet loss," *IEEE J. Select. Areas Commun.*, vol. 15, no. 6, pp. 1167–1180, August 1997.

[11] S. C. Liew and H. H. Chan, "Lossless aggregation: a scheme for transmitting multiple stored VBR video streams over a shared communications channel without loss of image quality," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 6, pp. 1181–1189, August 1997.

[12] H. Zhang and E. Knightly, "RED-VBR: A renegotiation-based approach to support delay-sensitive VBR video," *ACM/Springer-Verlag Multimedia Systems Journal*, vol. 5, no. 3, pp. 164–176, May 1997.

[13] C. Chang, "Stability, queue length, and delay of deterministic and stochastic queueing networks," *IEEE Trans. on Automatic Control*, vol. 39, no. 5, pp. 913–931, May 1994.

[14] E. Knightly, D. Wrege, J.Liebeherr, and H. Zhang, "Fundamental limits and tradeoffs of providing deterministic guarantees to VBR video traffic," in *Proc. of ACM SIGMETRICS*, pp. 98–107, 1995.

[15] J. Kurose, "On computing per-session performance bounds in high-speed multi-hop computer networks," in *Proc. of ACM SIGMETRICS*, pp. 128–139, 1992.

[16] H. Zhang and E. Knightly, "Providing end-to-end statistical guarantees using bounding interval dependent stochastic models," in *Proc. of ACM SIGMETRICS*, pp. 211–220, 1994.

[17] A. Elwalid, D. Mitra, and R. Wentworth, "A new approach for allocating buffers and bandwidth to heterogeneous, regulated traffic in an ATM node," *IEEE J. Select. Areas Commun.*, vol. 13, no. 6, pp. 1115–1127, August 1995.

[18] E. Knightly, "H-BIND: A new approach to providing statistical performance guarantees to vbr traffic," in *Proc. of IEEE INFOCOM*, pp. 1091–1099, 1996.

[19] M. W. Garrett and W. Willinger, "Analysis, modeling, and generation of self-similar VBR video traffic," in *Proc. of ACM SIGCOMM*, pp. 269–280, September 1994.