Supporting Stored Video: Reducing Rate Variability and End-to-End Resource Requirements through Optimal Smoothing

James D. SALEHI, Zhi-LI ZHANG,
James F. KUROSE and Don TOWSLEY

CMPSCI Technical Report 95-98

November 1995

Technical Report 95-98, November 1995

Supporting Stored Video: Reducing Rate Variability and End-to-End Resource Requirements through Optimal Smoothing*

James Salehi, Zhi-Li Zhang, James Kurose and Don Towsley
Department of Computer Science
University of Massachusetts
Amherst, MA 01003

Abstract

VBR compressed video is known to exhibit significant, multiple-time-scale bit rate variability. In this paper, we consider the transmission of stored video from a server to a client across a high speed network, and explore how the client buffer space can be used most effectively toward reducing the variability of the transmitted bit rate.

We present two basic results. First, we present an optimal smoothing algorithm for achieving the *greatest possible reduction in rate variability* when transmitting stored video to a client with given buffer size. We provide a formal proof of optimality, and demonstrate the performance of the algorithm on a set of long MPEG-1 encoded video traces. Second, we evaluate the impact of optimal smoothing on the network resources needed for video transport, under two network service models: Deterministic Guaranteed service [1, 11] and Renegotiated CBR (RCBR) service [9, 8]. Under both models, we find the impact of optimal smoothing to be dramatic.

1 Introduction

An extraordinarily broad range of applications are enabled by the capacity to efficiently manage digital multimedia information. Examples include digital libraries, video and image servers, distance learning and collaboration, interactive virtual environments, and shopping and entertainment services. Many of these applications require the playback of stored video over a high-speed network. For continuous playback at the client, isochronous processing requirements and strict quality-of-service (QOS) must be provided in an end-to-end manner. Complicating the situation is the fact that variable-bit-rate (VBR) compressed video exhibits significant rate variability, often spanning time scales as coarse as several minutes [6, 9]. This burstiness is inherently at odds with the goal of designing efficient real-time storage, retrieval and transport mechanisms capable of achieving high resource utilization. As a general real-time design principle, less bursty (i.e., *smoother*) workloads are easier to manage. In light of the inherent multiple-time-scale burstiness of VBR video, techniques for reducing its rate variability are of significant interest.

There are three basic techniques available for reducing the rate variability of compressed video. 1) Smoothing by *temporal multiplexing* can be achieved by introducing a per-stream buffer somewhere along the end-to-end path.

*This work was supported by NSF under grant NCR-9206908 and by ARPA under ESD/AVS contract F-19628-92-C-0089. The authors can be contacted at [salehi,zhzhang,towsley,kurose]@cs.umass.edu.

This can smooth the stream's peak rate, but at the cost of introducing delay. 2) Smoothing by *aggregation* (or *statistical multiplexing*) can be achieved when multiple independent streams share a given resource (e.g., processor or network bandwidth), each with a statistical QOS requirement. When the individual streams have service times with finite variance, the Central Limit Theorem [4] dictates that their aggregage bandwidth requirements converge to a Normal distribution as the number of streams becomes large. This smoothing effect does not introduce delay. 3) Smoothing by *workahead* is possible for video data which can be sent *ahead of schedule* (with respect to its playback time), subject to the dual constraints that *i)* the data is available to be sent, and *ii)* the client has sufficient buffer space to receive it. See [22] for a precise formulation of these constraints. Workahead smoothing does not introduce delay.

These techniques are not mutually exclusive, and can be applied in combination and at various points along the end-to-end path from the server to the client [5, 9, 10, 11, 13, 19, 22, 21, 24]. In general, network service models may permit only temporal multiplexing [11], only statistical multiplexing [9], or both, as is the case whenever FIFO scheduling with shared buffers is employed at a network switch.

In this paper, we propose and evaluate a workahead smoothing technique which enables the server to achieve the *greatest possible reduction in rate variability* when transmitting stored video to a client with given buffer size. We focus on buffer sizes in the range of 64-1024 KB, reflecting the anticipated memory limitation of a low-cost set-top box. We study two fundamental questions. 1) Given a client buffer of size b, how can a given video be sent from the server such that the transmitted bit rate is as *smooth* as possible, while ensuring that the buffer neither starves nor overflows? We will turn to the theory of *majorization* [16] for precise definitions of "smooth" and "as smooth as possible". For ease of presentation, we defer discussion of majorization until Section 3; for now, it is sufficient to state that we seek the transmission schedule which minimizes both the variance and peak rate at which data is sent to the client. 2) Given that the server sends optimally-smoothed transmissions, what is the impact on the network resources required to transmit the stream? High-speed networks such as ATM typically employ temporal and/or statistical multiplexing to increase performance. Thus, to address this second question realistically, we need to consider network service models which employ these techniques.

We begin in Section 2 by developing an optimal smoothing algorithm which identifies, for a given video and client buffer size, the server transmission schedule S^* which is as smooth as possible. We establish the optimality of S^* formally, and show (as a simple consequence of a more general result) that S^* has minimum variance and peak rate among all feasible schedules—and in this sense is optimally smooth. The algorithm has low complexity, can be easily implemented, and can be used in a manner that accommodates network jitter. We then evaluate the algorithm on several long VBR-encoded MPEG-1 traces. For the diverse set of traces in our test suite (one of which has been shown to exhibit *self-similarlity* in its bit rate variation [6]), the peak rate is reduced by 75-87%, and the standard deviation by 72-84%, when smoothed into a 1 MB client buffer.

We then evaluate the impact on network performance when streams are optimally smoothed. We consider two network service models: *Deterministic Guaranteed* service [1, 11], which provides hard guarantees, and *Renegotiated CBR* (RCBR) [9, 8], which adds a renegotiation mechanism to traditional CBR service. We find that optimal smoothing yields dramatic improvement in performance under both models.

These results are important because 1) the processing and transmission of stored video is likely to play a central role in a wide variety of emerging multimedia applications, and 2) VBR-encoded video is known to exhibit significant, multiple-time-scale rate variability. By optimally reducing the stream rate variability, its resource requirements can be drastically reduced. Moreover, this improvement is made available by an algorithm that is easy to implement, has low complexity, and places no restriction on the multimedia server architecture (i.e., applies to small, uniprocessor servers, as well as to large, scalably-parallel ones).

The paper is organized as follows. We present the optimal smoothing algorithm in Section 2, and formally establish its optimality in Section 3. In Sections 4-7, we evaluate the impact on resource allocation under the two network service models. Related work is reviewed in Section 8.

2 Optimal smoothing of stored video

In this section, we present the optimal smoothing algorithm, demonstrate its performance on a set of long VBR MPEG-1 videos, and discuss how network jitter can be accommodated. We begin with an overview of the problem setting.

2.1 Problem setting

Consider Figure 1, which depicts a server concurrently transmitting multiple, distinct VBR streams to clients across a high-speed network. For each stream, data is retrieved from the disk subsystem and stored temporarily in server memory (a). Its eventual destination is the client buffer (c). The server network processing (b) is responsible for moving the data from the in-memory buffers onto the high-speed network, according to some server *transmission schedule*. The network software communicates with the disk subsystem to ensure that the data needed by the transmission schedule is in memory by the time it is scheduled to be sent. For simplicity, we will assume that each stream is served periodically at its frame rate, and what can vary is the amount of data sent when the stream is served.

Each client has buffer capacity b available for server workahead of video data. When the client decode/display process runs, with periodicity given by the stream frame rate, the next frame is selected from the client buffer, decoded, and displayed. If the frame is incomplete, the client displays partial data or pauses stream playback—in either case experiencing a *playback discontinuity*. We will develop the optimal smoothing algorithm in a manner

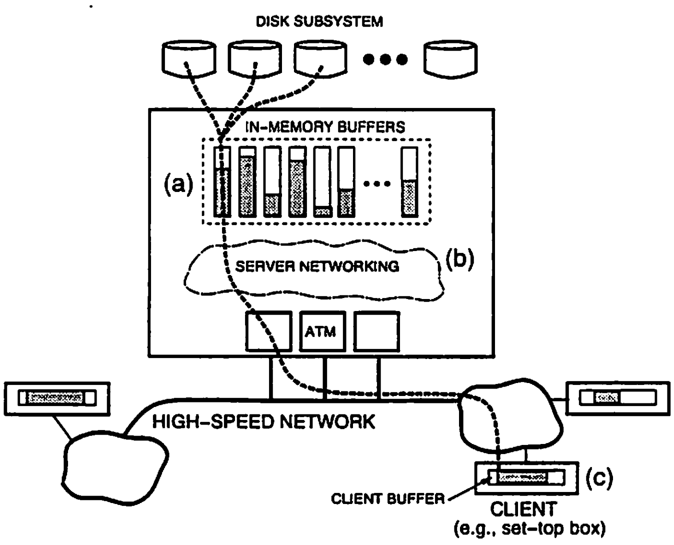


Figure 1: Overview of the problem setting

b	:	Client buffer capacity for storing unplayed frames.
$d(t)$	:	Amount of data consumed by the client at time t (size of the t^{th} frame). Data is consumed from the buffer after any arrivals at t.
$D(t)$	:	Cumulative data consumed by the client over $[1, t]$: $\sum_{i=1}^{t} d(i)$
$a(t)$	:	Amount of data sent by the server at time t.
$A(t)$	:	Cumulative data sent by the server over $[1, t]$: $\sum_{i=1}^{t} a(i)$
$B(t)$	:	Maximum cumulative data that can be received by the client over $[1, t]$, without buffer overflow: $B(t) = \min\{D(t-1) + b, D(N)\}$ for $t = 2, \ldots, N$, $B(1) = b$, $B(0) = 0$.

Table 1: Notation

that completely avoids playback discontinuities. We focus here on video playback, deferring issues introduced by client interactivity (e.g., VCR functions of fast-forward, rewind and pause) to future study.

Let τ denote the end-to-end delay over the network. For the time being, assume $\tau = 0$. We show how non-zero network delay can be accommodated in Section 2.4.

2.2 The optimal smoothing algorithm

In formulating the optimal smoothing algorithm $\mathcal{A}$, we consider a discrete-time model at the frame level. That is, $t \in \{1, \ldots, N\}$, where N is the length of the video in frames. A finer-granularity time discretization can be considered without loss of generality. Table 1 summarizes the relevant notation.

We say the N-dimensional real vector $S = [a(1), \ldots, a(N)]$ represents a *feasible* server transmission schedule over $[1, N]$ if 1) $\sum_t a(t) = D(N)$, and 2) $D(t) \leq A(t) \leq B(t)$ for all t (see Table 1 for definition of these quantities).

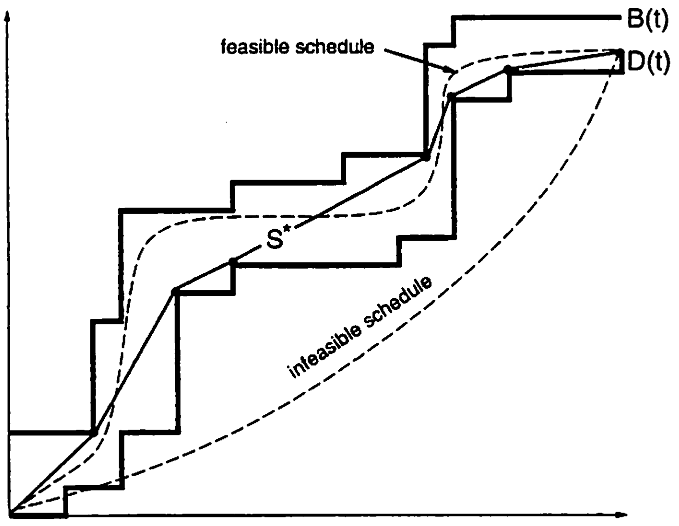


Figure 2: Various transmission schedules, in continuous time

Note the second constraint ensures that the client buffer neither starves nor overflows during stream playback. Let $\mathcal{S}$ denote the set of all feasible server transmission schedules. See Figure 2 for an illustration, in continuous time, of feasible and infeasible transmission schedules.

Our task is to devise an algorithm for finding the feasible schedule S^* which is as *smooth* as possible. While we formally define the smoothness criterion in Section 3, consider for the moment the problem of identifying the transmission schedule with minimum variance and peak rate. (The variance of S is given by $\text{var}(S) = \sum_t (a(t) - \bar{a})^2/N$, with $\bar{a} = \sum_t a(t)/N$; the peak rate is given by $\max(S) = \max_t a(t)$). How might we identify such a schedule? Intuitively, the CBR transmission schedule $\bar{S} = [\bar{a}, \ldots, \bar{a}]$ is optimally smooth; however, it is not necessarily feasible. As an alternative, construct a feasible piecewise-CBR transmission schedule in the following manner. Define $\langle n, c \rangle$ to indicate a "*feasible CBR segment over* $[a, b]$", for $b > a$, if $(b - a) = n$ and starting from a given buffer level q, the server can feasibly transmit at rate c over $[a + 1, b]$ (i.e., $D(t) \leq q + (t - a)c \leq B(t)$ for $t = \{a + 1, \ldots, b\}$). Now, starting from a given time t_s with initial client buffer level q, establish the longest possible feasible CBR segment, by finding the greatest t_e such that $\langle t_e - t_s, c \rangle$ is feasible for some c. Eventually this current segment must end, and a new one established at a higher or lower rate. Begin the new segment at the latest time at which the client buffer is *empty* along the current segment when the rate must be *decreased*, or *full* when the rate must be *increased*. This strategy minimizes the change in rate from the current CBR segment, which is desirable from the perspective of finding the smoothest transmission schedule overall. Figure 2 illustrates the S^* constructed in this manner. The schedule dictates that the server transmit in a piecewise-CBR manner, filling the client buffer before increasing the transmission rate, and allowing the buffer to empty prior to any rate decrease.

Algorithm $\mathcal{A}$ (Figure 3) uses this strategy to iteratively construct S^*. Define c_{max} as the maximum rate at which

```
1.  PROCEDURE Find_optimal_schedule ($d(t), b$)
2.    $t_s = 0, t_e = 1, q = 0$
3.    $c_{max} = b, t_B = 1, c_{min} = d(1), t_D = 1$
4.    REPEAT
5.      Set $t'_e = t_e + 1$
6.      IF $c_{max} < \frac{D(t'_e)-(D(t_s)+q)}{t'_e-t_s}$, end segment at $t_B$:
7.        OUTPUT segment $\langle t_B - t_s, c_{max} \rangle$
8.        Start new segment at $t_B$:
9.          $t_s = t_B, t_e = t_B + 1, q = B(t_B) - D(t_B)$
10.     ELSE IF $c_{min} > \frac{B(t'_e)-(D(t_s)+q)}{t'_e-t_s}$, OR $t'_e = n$, end segment at $t_D$:
11.       OUTPUT segment $\langle t_D - t_s, c_{min} \rangle$
12.       Start new segment at $t_D$:
13.         $t_s = t_D, t_e = t_D + 1, q = 0$
14.     ELSE
15.       Set $t_e = t'_e$
16.       Compute $c_{max}, t_B, c_{min}$, and $t_D$ over $[t_s, t_e]$
17.   UNTIL $t_s = n$
18. END PROCEDURE
```

Figure 3: Algorithm $\mathcal{A}$. Optimal smoothing of stored video, given frame size trace $d(t)$ and client buffer capacity b. See text for description of data structures. S^*, the optimally smooth transmission schedule, is identified as a series of CBR segments, each of the form $\langle n_i, c_i \rangle$, where n_i is the segment length and c_i the segment transmission rate.

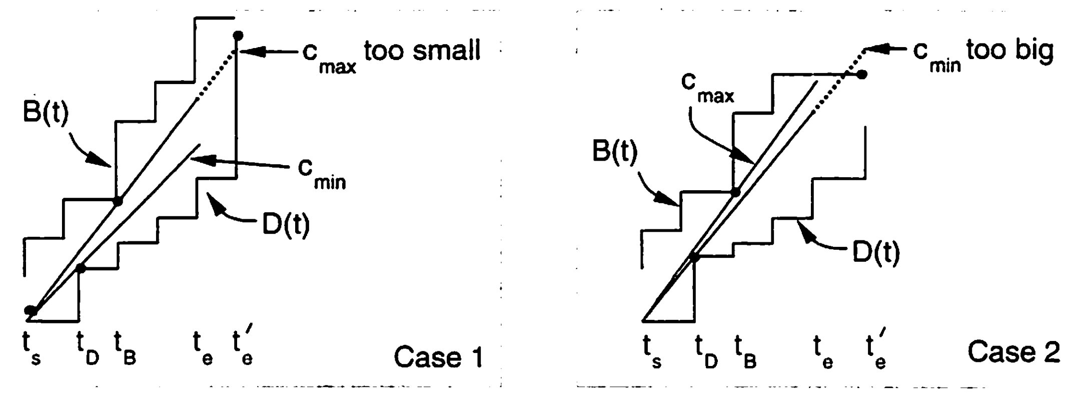


Figure 4: Operation of $\mathcal{A}$. In Case 1, the largest feasible constant rate over $[t_s, t_e]$, c_{max}, would result in *starvation* at t'_e. The CBR segment is ended at t_B, and a new segment established at higher rate. In Case 2, the smallest feasible constant rate over $[t_s, t_e]$, c_{min}, would result in *buffer overflow* at t'_e. The CBR segment is ended at t_D, and a new segment established at lower rate.

the server may transmit over a given interval $[a, b]$, without overflowing the client buffer, starting from initial buffer level q:

$$c_{max} = \min_{a+1 \leq t \leq b} \frac{B(t) - (D(a) + q)}{t - a},$$

and define t_B as the latest time t at which the client buffer is *full* when the server transmits at c_{max} over $[a, b]$, starting with initial buffer level q:

$$t_B = \max_{a+1 \leq t \leq b} \{t : \frac{B(t) - (D(a) + q)}{t - a} = c_{max}\}.$$

See Figure 4. Similarly, define c_{min} as the minimum rate at which the server may transmit over a given interval $[a, b]$ such that the client buffer never starves, starting from initial client buffer level q:

$$c_{min} = \max_{a+1 \leq t \leq b} \frac{D(t) - (D(a) + q)}{t - a},$$

and define t_D as the latest time t at which the client buffer is *empty* when the server transmits at c_{min} over $[a, b]$, starting with initial buffer level q:

$$t_D = \max_{a+1 \leq t \leq b} \{t : \frac{D(t) - (D(a) + q)}{t - a} = c_{min}\}.$$

CBR transmission over the interval $[a, b]$ is feasible if and only if $c_{max} \geq c_{min}$ when computed over $[a, b]$.

$\mathcal{A}$ operates as follows. Each iteration begins (Line 4) with a feasible CBR segment over $[t_s, t_e]$, and its corresponding values of c_{max}, t_B, c_{min} and t_D. (They are assigned initial values on Line 3, and subsequently recomputed for each iteration on Line 16). Let q denote the client buffer level at t_s. $\mathcal{A}$ considers whether the CBR segment can be extended by one time unit, by setting t'_e to $t_e + 1$ (Line 5) and testing the feasibility of the extension:

- *Case 1* (Line 6). If c_{max} is too small to avoid starvation at t'_e, the CBR segment must be ended before t'_e, and a new one established at a higher rate. The segment is ended at t_B and is assigned the highest feasible rate, c_{max} (see Figure 4a). In this manner, the rate *increase* for the next segment (beginning at t_B) will be as small as possible.

- *Case 2* (Line 10). Otherwise, if c_{min} would result in buffer overflow at t'_e, the CBR segment must be ended before t'_e, and a new one established at a lower rate. The segment is ended at t_D and is assigned the smallest feasible rate, c_{min} (see Figure 4b). In this manner, the rate *decrease* for the next segment (beginning at t_D) will be as small as possible.

Otherwise, there exists a feasible CBR segment over $[t_s, t_e + 1]$, so t_e is updated and new values of c_{min}, t_D, c_{max} and t_B are computed (Lines 15-16). The process continues until the entire video is segmented.

Thus S^* consists of some number of CBR transmission segments, each ending with a change in transmission rate. w is said to be a *change point* of $S^* = [a^*(1), \ldots, a^*(N)]$ if $a^*(w) \neq a^*(w+1)$, $w \in \{1, \ldots, N-1\}$. A change point w is said to be *convex* when $a^*(w) < a^*(w+1)$, and *convex* when $a^*(w) > a^*(w+1)$. Note that for S^*, the client buffer is full at the convex change points, and empty at the concave change points. These properties will prove useful in establishing the optimality of Algorithm $\mathcal{A}$ in the next section.

Although theoretically the time complexity of $\mathcal{A}$ is $O(N^2)$, in practice we find the execution time of the algorithm to be quite low. On an SGI workstation with R4400 processor running at 150 MHz, the 40,000-frame traces in our test suite can be optimally smoothed in 1.5-2 seconds for any given client buffer size in the range 64 KB – 16 MB. For the 174,000 frame *Star Wars* trace, which is about 4 times as long, optimal smoothing requires 6-8 seconds for client buffer sizes in this range.

In addition, $\mathcal{A}$ can be easily modified to be strictly $O(N)$. The idea is to construct both the convex lower bound of $B(t)$ and the concave upper bound of $D(t)$ online, at each step as t_e is extended to $t_e + 1$. Then, when the segment rate must be adjusted, a new t_s can be identified directly which yields a feasible CBR segment over $[t_s, t_e + 1]$. Under this approach, t_e is never decremented, so backtracking (at Lines 9 and 13 in Figure 3) is completely avoided. We have implemented this modification to $\mathcal{A}$, and measured its performance. Across the set of traces in our test suite, for a wide range of client buffer sizes, we found no significant difference in execution time when compared with the original implementation.

We close this subsection with two observations. First, $\mathcal{A}$ can be easily *quantized* (i.e., modified to produce an S^* which sends only integral numbers of bytes at each time t), while ensuring that the client buffer retains the property

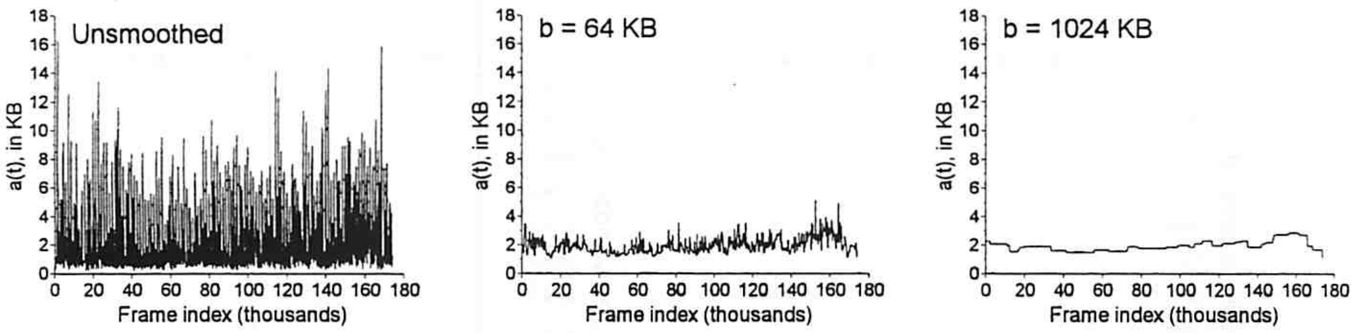


Figure 5: Optimal smoothing of a 2-hour MPEG-1 encoding of *Star Wars*, with 500 ms startup latency.

of being full/empty at the change points in S^*. See Appendix B for details. Second, $\mathcal{A}$ does not depend on $B(t)$ lying at a fixed offset above $D(t)$. Minimally, the client needs only sufficient buffer space to store the next frame at any given time (i.e., the constraint is $B(t) \geq D(t-1) + d(t)$, for all t). In particular, $B(t)$ could be set to reflect a *workahead constraint* (i.e., a limitation on the amount of data available to be sent ahead of schedule at any given time) without affecting the operation of the smoothing algorithm.

2.3 Empirical evaluation

In this section, we demonstrate the performance of $\mathcal{A}$ on several long MPEG-1 frame-size traces. We use traces captured by a number of other researchers [6, 11, 12, 23], who were kind enough to share them with us.

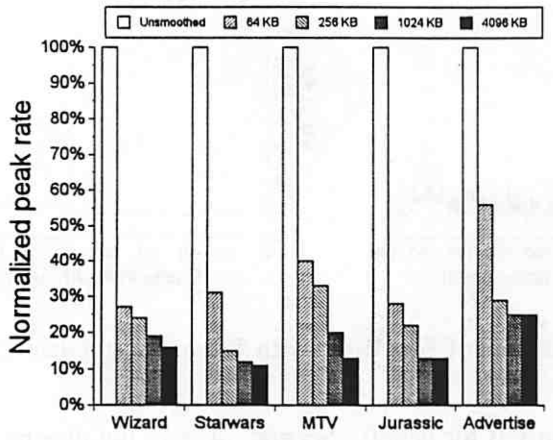
In Algorithm $\mathcal{A}$, a startup latency of s frames can be introduced to smooth out the transmission of the very first frame of the video, which (due to the MPEG compression technique) is typically much larger than the immediately subsequent frames. Otherwise, the transmission rate of the initial segment can be comparatively large. In $\mathcal{A}$, this is accomplished by extending the video length by s frames, and shifting both $D(t)$ and $B(t)$ by s units to the right.[1]

We begin by optimally smoothing a 2-hour MPEG-1 encoding of *Star Wars* [6], with a 500 ms startup latency. Figure 5 demonstrates the transmission sizes, over the entire video, for the unsmoothed transmission schedule (which has $a(t) = d(t)$), as well as optimally-smoothed schedules for client buffer sizes of 64 KB and 1024 KB. The figure visually demonstrates the tremendous reduction in rate variability achieved by $\mathcal{A}$. Smoothing reduces the range of transmission sizes from 0-16 KB, in the unsmoothed schedule, down to 1-5 KB in the 64 KB case. Further increases in buffer size yield smoother traffic, with the transmission schedule for a 1024 KB buffer consisting of a relatively small number of CBR segments.

Figure 6 presents the reduction in in peak rate and standard deviation across different videos,[2] in comparison to

[1]Specifically, set $N' = N + s$, $d'(t) = d(t-s)$ for $t \in [s+1, N']$, $d'(t) = 0$ for $t \in [1, s]$, and define $D(t) = \sum_t d'(t)$ and $B(t)$ over $[1, N']$. Note that the introduction of startup latency is optional.

[2]All traces were VBR-encoded using an MPEG-1 software encoder. *The Wizard of Oz* trace [12] is 23 minutes in length and has mean rate 1.25 Mb/s. *MTV* and *Jurassic Park* [23] are each 28 minutes in length, and have mean rates of 0.737 Mb/s and 0.392 Mb/s, respectively.



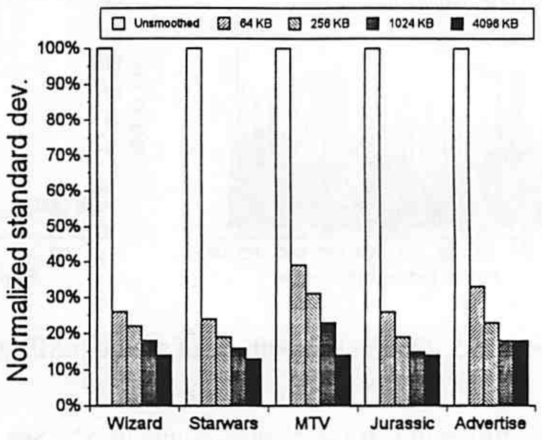


Figure 6: Reduction in rate variability enabled by optimal smoothing, with 500ms startup delay, across client buffer sizes ranging from 64-4096 KB.

the unsmoothed transmission schedule. For all traces, the peak rate is reduced substantially, by 45-70% for a 64 KB client buffer and by 75-95% for a 4096 KB buffer. The standard deviation exhibits similar behavior. Note that these performance numbers reflect the largest *possible* reduction in rate variability achievable for the given buffer sizes, since $\mathcal{A}$ is performing optimal smoothing.

2.4 Accommodating network jitter

In this subsection, we address the question of how can network jitter be accommodated by the optimal smoothing algorithm. Let $\tau = \sigma + \nu$ denote the end-to-end network delay for a given packet, with σ the sum of the fixed transmission and propagation delays, and ν the end-to-end queueing delay. Earlier, we assumed $\tau = 0$ to simplify presentation. If τ is nonzero but fixed, S^* can be used directly. However, *network jitter*—variation in τ resulting from variable queueing delay—must be explicitly addressed.

Suppose some mechanism exists for determining (or estimating) the maximum network jitter, $\nu_{max} = \tau_{max} - \tau_{min}$. For hard-guaranteed service, ν_{max} would be specified at call admission; alternatively, ν_{max} might be determined via a measurement-based algorithm. Network jitter can then be accommodated by ensuring that the server is always at least $d = \lceil \frac{\nu_{max}}{\Delta} \rceil$ time units ahead of the client, where Δ is the unit of time discretization (e.g., $\frac{1}{30}$ of a second). In the unsmoothed case, client playback is delayed by d, and the server simply follows the unsmoothed transmission schedule. Under optimal smoothing, we shift $D(t)$ and $B(t)$ by d units to the right, and perform smoothing between B and the *unshifted* D.[3]

We will employ this jitter-handling technique in our evaluation of the impact of optimal smoothing under

Advertisements [11] is 9 minutes in length and has mean rate 0.457 Mb/s.

[3]Specifically, set $N' = N + \omega$, $d(t) = d(N)$ for $t \in [N+1, N']$, $d'(t) = d(t - \omega)$ for $t \in [\omega + 1, N']$ $d'(t) = 0$ for $t \in [1, \omega]$, and define $D(t) = \sum_{i=1}^{t} d(t)$ and $B(t) = \sum_{i=1}^{t} d'(t) + b$ over $[1, N']$.

Determinstic Guaranteed network service, in Section 5.

3 Optimality proof

In this section we show that the server transmission schedule generated by Algorithm $\mathcal{A}$, S^*, is optimally smooth. For a precisely-defined comparative measure of smoothness, we turn to the theory of *majorization* [16], which formalizes the intuitive notion that the elements of one vector are more "evenly distributed" than those of another. Majorization techniques have been applied in a wide variety of problem domains. In our context, we use majorization to compare the smoothness of N-dimensional vectors representing server transmission schedules.

Definition 1 *Given two n-dimensional real vectors $X = [x_1, \ldots, x_n]$ and $Y = [y_1, \ldots, y_n]$, let $x_{[1]}, \ldots, x_{[n]}$ and $y_{[1]}, \ldots, y_{[n]}$ denote the non-increasing orderings (i.e., arrangement of vector elements from largest to smallest) of X and Y, respectively. We say that X is* majorized *by Y, or $X \prec Y$, if $\sum_{i=1}^{t} x_{[i]} \leq \sum_{i=1}^{t} y_{[i]}$, $t = 1, 2, \ldots n-1$, and $\sum_{i=1}^{n} x_{[i]} = \sum_{i=1}^{n} y_{[i]}$.*

For example, if $X = [2,3,3,2]$ and $Y = [1,8,1,0]$, then $X \prec Y$. X does appear to be smoother than Y. If $X = [12,3,7]$ and $Y = [2,10,10]$, neither $X \prec Y$ nor $Y \prec X$. One way to visualize majorization is to consider a graphical view of the function $\omega(t)$, defined as the sum of a vector's t largest elements. ω is always concave (Figure 7a). If $X \prec Y$, ω_X and ω_Y will have the same endpoints over $[0, n]$, and ω_Y will never fall below ω_X over the entire interval.

Thus, we consider a transmission schedule S_1 to be *smoother* than schedule S_2 if $S_1 \prec S_2$. While variance and peak are commonly-used comparative measures of smoothness, such metrics are in fact subsumed by majorization. Since $x_{[1]}$ is by definition $\max(X)$, it follows that $X \prec Y$ implies $\max(X) \leq \max(Y)$. To see that $X \prec Y$ implies $\text{var}(X) \leq \text{var}(Y)$ (where $\text{var}(X) = \sum_i (x_i - \bar{x})^2/n$, with $\bar{x} = \sum_i x/n$), consider the following fundamental result, the proof of which is given in [16], p. 108.

Result 1 *$X \prec Y$ if and only if $\sum_i \phi(x_i) \leq \sum_i \phi(y_i)$, for all continuous convex[4] functions $\phi : \mathbb{R} \to \mathbb{R}$.*

Since $\phi(x) = (x - \bar{x})^2/N$ is a continuous convex function in x, it follows that $X \prec Y$ implies $\text{var}(X) \leq \text{var}(Y)$.

To establish the majorization-based optimality of S^*, we will show (Theorem 1 below) that $S^* \prec S$ for all feasible transmission schedules S. This will imply that S^* has minimum peak rate and minimum variance among all feasible transmission schedules by Result 1, and thus that it is optimally smooth according to these less general measures of variability as well.

[4]A continuous function ϕ is *convex* if, for all x, y, $0 \leq \alpha \leq 1$, $\phi(\alpha x + (1-\alpha)y) \leq \alpha\phi(x) + (1-\alpha)\phi(y)$.

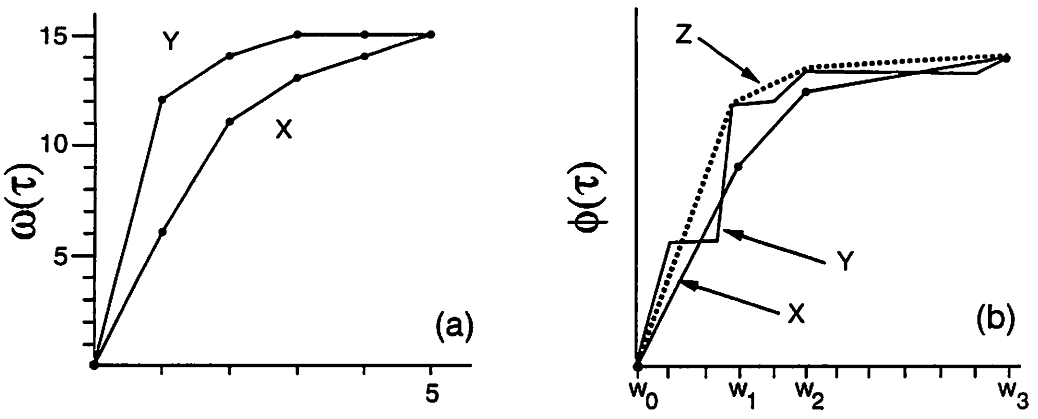


Figure 7: (a) Graphical view of "X is majorized by Y" ($X \prec Y$), for $X = [2,1,6,5,1]$ and $Y = [0,12,0,1,2]$. $\omega(\tau)$ is the sum of the τ *largest* vector elements. (b) The idea behind Lemma 1. $\phi(\tau)$ is the sum of the first τ vector elements. X is concave 3-segment piecewise linear, Y is at or above X at change points w_k, and $\sum_i y_i = \sum_i x_i$. Z is constructed such that $X \prec Z$ and $Z \prec Y$.

3.1 Preliminaries

Our proof will depend on several known facts about majorization, which we establish here for the purposes of illustration. The first is that majorization is closed under concatenation.

Fact 1 *Let U^k and V^k be m_k-dimensional real vectors, $k = 1, 2, \ldots n$, such that $U^k \prec V^k$. If we form $(\sum_{i=k}^n m_k)$-dimensional real vector U (V) by concatenating $U^1, \cdots, U^k$ ($V^1, \cdots, V^k$), then $U \prec V$.*

Proof: From Result 1, we have that $\sum_{i=1}^{m_k} \phi(u_i^k) \le \sum_{i=1}^{m_k} \phi(v_i^k)$ for any continuous convex function $\phi : I\!R \to I\!R$. It follows that $\sum_{j=1}^k \sum_{i=1}^{m_j} \phi(u_i^j) \le \sum_{j=1}^k \sum_{i=1}^{m_j} \phi(v_i^j)$. Thus $U \prec V$, by Result 1. ∎

Fact 2 *If X is* linear *(i.e., if $x_1 = \cdots = x_n = c$), then for all Y such that $\sum_{i=1}^n y_i = \sum_{i=1}^n x_i = nc$, $X \prec Y$.*

Proof: Suppose $X \not\prec Y$. Then $\exists j \in [1, n)$ such that $\sum_{i=1}^j x_{[i]} = cj > \sum_{i=1}^j y_{[i]}$. We must have $y_{[j]} < c$, implying $\sum_{i=j+1}^n y_{[i]} < c(n-j)$. Together, these imply $\sum_{i=1}^n y_{[i]} < nc$, a contradiction. Thus $X \prec Y$. ∎

Definition 2 $X = [x_1, \ldots, x_n]$ *is said to be* m-segment piecewise linear *if there exists a set of* change points *of X, $\{w_k\}$, $k = 0, 1, \ldots, m$, with $w_0 = 0 < w_1 < \cdots < w_{m-1} < w_m = n$, such that $x_i = x_j$ for all $i, j \in \{w_{k-1} + 1, \ldots, w_k\}$.*

Definition 3 $X = [x_1, \ldots, x_n]$ *is said to be* concave (convex) m-segment piecewise linear, *if X is m-segment piecewise linear and $x_{w_k} > x_{w_{k+1}}$ ($x_{w_k} < x_{w_{k+1}}$) for $k = 1, 2, \ldots m-1$.*

We can now present Lemma 1, which will be used in establishing the optimality of S^*.

Lemma 1 *Given two n-dimensional real vectors X and Y such that X is concave (convex) m-segment piecewise linear with change points w_k, $k = 0, 1, \ldots m$. If $\sum_{i=1}^{w_k} x_i \leq \sum_{i=1}^{w_k} y_i$ ($\sum_{i=1}^{w_k} x_i \geq \sum_{i=1}^{w_k} y_i$) for $k = 0, 1, \ldots m-1$, with strict equality for $k = m$, then $X \prec Y$.*

Proof: We prove the Lemma for the case in which X is concave (Figure 7b); a similar argument can be given for the convex case. We establish $X \prec Y$ by constructing an m-segment piecewise linear Z such that $X \prec Z$ and $Z \prec Y$. $X \prec Y$ then follows by the transitivity of majorization. $Z = [z_1, \ldots, z_n]$ is constructed in the following manner. Let $p_k = \sum_{i=w_{k-1}+1}^{w_k} y_i$ for $k = 1, 2, \ldots m$, and $z_i = p_k/(w_k - w_{k-1})$ for $i = \{w_{k-1}+1, \ldots, w_k\}$. Since Z is linear and has the same partial sum as Y over intervals $w_{k-1}+1, \ldots, w_k$, $Z \prec Y$ follows from Fact 2 and Result 1. $X \prec Z$ follows from $\sum_{i=1}^{j} x_{[i]} = \sum_{i=1}^{j} x_i \leq \sum_{i=1}^{j} z_i \leq \sum_{i=1}^{j} z_{[i]}$ for $j = 1, 2, \ldots n-1$, with equality for $j = n$. ∎

3.2 The optimality of S^*

In Section 2, we established that 1) the optimal transmission schedule $S^* = [a^*(1), \ldots, a^*(N)]$ is m-segment piecewise linear, and 2) at convex change points in S^* (i.e., at times t where $a^*(t) < a^*(t+1)$), the client buffer is full, whereas at concave change points (i.e., at times t where $a^*(t) < a^*(t+1)$) it is empty. Let $0 = w_0 < w_1 \cdots < w_m = N$ denote the change points of S^*. Define w_0 and w_N as concave change points.

Consider groups of consecutive convex change points.

Definition 4 *$\langle w_i, n \rangle$ is said to denote a* convex region *if $w_{i+1}, \ldots, w_{i+n-1}$ are convex change points and both w_i and w_{i+n} are concave change points.*

S^* contains a finite number K of convex regions, possibly none. If $K \geq 1$, let $\langle w_{i_k}, n_k \rangle$ denote the k^{th} convex region, $k = 1, \ldots K$.

We are now in a position to establish the main theorem.

Theorem 1 *Let $S^* = [a^*(1), \ldots, a^*(N)]$ denote the transmission schedule generated by $\mathcal{A}$ for a given video and client buffer size. If $S = [a(1), \ldots, a(N)]$ is any arbitrary feasible schedule for the video, then $S^* \prec S$.*

Proof: Our approach will be to construct an $S' = [a'(1), \ldots, a'(N)]$ such that $S^* \prec S'$, and $S' \prec S$. Let $A^*(t) = \sum_{i=1}^{t} a^*(t)$, $A(t) = \sum_{i=1}^{t} a(t)$, and $A'(t) = \sum_{i=1}^{t} a'(t)$. If S^* contains no convex regions (i.e., if $K = 0$), Theorem 1 follows directly from Lemma 1, with $X = S^*$ and $Y = S$. Therefore, let us assume $K \geq 1$. See Figure 8.

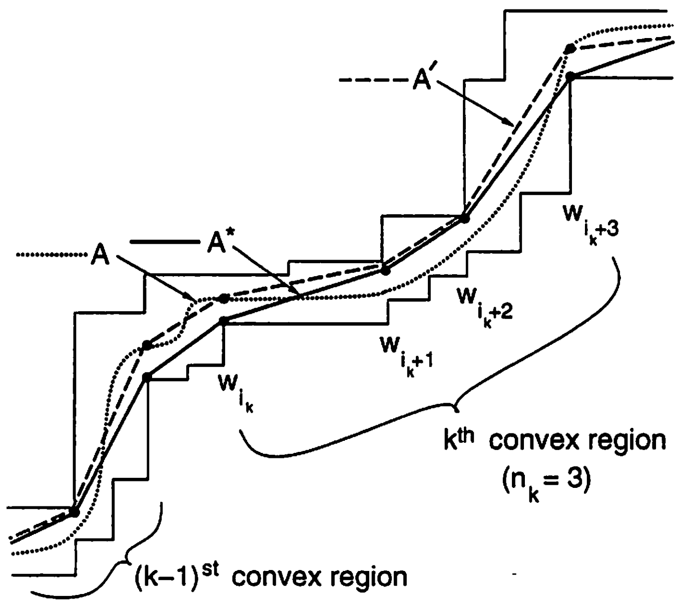


Figure 8: Establishing $S^* \prec S$, via construction of S'

The basic idea is to construct an m-segment piecewise linear S' such that A' meets A^* at the convex change points of S^* (i.e., at w_{i_k+1} and w_{i_k+2} in the figure), and meets A at the concave change points of S^* (i.e., at w_{i_k} and w_{i_k+3}).

Formally, define S' in the following way. Let w_i and w_{i+1} be consecutive change points of S^*, $i = 0, \ldots, m-1$. For $t \in \{w_i + 1, \ldots, w_{i+1}\}$, define

$$a'(t) = \begin{cases} (A(w_{i+1}) - A(w_i))/(w_{i+1} - w_i), & w_i, w_{i+1} \text{ both concave} \\ (A^*(w_{i+1}) - A^*(w_i))/(w_{i+1} - w_i), & w_i, w_{i+1} \text{ both convex} \\ (A^*(w_{i+1}) - A(w_i))/(w_{i+1} - w_i), & w_i \text{ concave, } w_{i+1} \text{ convex} \\ (A(w_{i+1}) - A^*(w_i))/(w_{i+1} - w_i), & w_i \text{ convex, } w_{i+1} \text{ concave} \end{cases}$$

We establish $S' \prec S$ as follows. Let $X[i,j] = [x_i + 1, \ldots, x_j]$ denote the $(j-i)$-dimensional subvector of X. Over the k^{th} convex region of S^*, $\langle w_{i_k}, n_k \rangle$, we have $S'[w_{i_k}, w_{i_k+n_k}] \prec S[w_{i_k}, w_{i_k+n_k}]$ by Lemma 1. (Note that A and A' meet at w_{i_k} and $w_{i_k+n_k}$, and that A' meets A^* at the convex change points of S^* within the region). This holds for all convex regions $k = 1, \ldots, K$. The fact that A' meets A at all concave change points, coupled with the fact that A' is linear between change points, implies by Fact 2 that $S' \prec S$ outside of the convex regions. Thus, by Fact 1, we have $S' \prec S$ over $[1, N]$.

To show $S^* \prec S'$, consider that for each region $[i,j] \in \{[0, w_{i_1+1}], [w_{i_1+n_1-1}, w_{i_2+1}], \ldots, [w_{i_k+n_k-1}, w_m]\}$ (which are essentially the concave regions of S^*), we have directly by Lemma 1 that $S^*[i,j] \prec S'[i,j]$. This follows from the facts that A^* and A' meet at the endpoints of these regions, all change points of S^* within the regions are concave, and A' is at or above A^* at any of the concave change points of S^*, since for A^* the client buffer is empty there. Outside these regions, $S' = S^*$. Thus, by Fact 1, we have $S^* \prec S'$. Together with $S' \prec S$, this completes the proof. ■

Thus, we have shown that $S^* \prec S$ for any arbitrary feasible transmission schedule S, establishing with respect to majorization that S^* is optimally smooth. It follows by the definition of majorization that S^* has minimum peak rate, and by Result 1 that S^* has minimum variance, among all feasible transmission schedules.

Finally, we can show that S^* is unique. Suppose there exists another feasible transmission schedule S_1 such that $S_1 \neq S^*$ and $S_1 \prec S$ for all feasible S (call this Property 1). If S_1 meets S^* at all change points of S^*, it is easy to see that $S_1 \not\prec S^*$, which contradicts Property 1. Otherwise, following the approach outlined in the proof of Theorem 1, we can construct an S' such that $S' \prec S_1$ yet $S' \not\prec S^*$; Property 1 then implies $S' \prec S^*$, a contradiction. Thus S^* is unique.

4 Impact of optimal smoothing on network resource requirements

We have established an optimal smoothing technique which, through workahead, enables the server to maximally reduce stream rate variability when sending stored video to a client across a high-speed network. We now ask the question: Given that the server performs optimal smoothing, what is the impact on the *network* resources required to transmit the stream? We consider the increase in number of streams which can be supported in the network, the improvement in achievable network utilization, and the reduction in cost to the server of sending the video. Our motivation is given by the fact that network performance depends fundamentally on the peak rate and burstiness characteristics of the admitted traffic, which are minimized under optimal smoothing.

In the following sections, we consider two network service models: Deterministic Guaranteed service (or simply "Guaranteed service") as formulated in [11], and Renegotiated CBR (RCBR) service, as formulated in [9]. Guaranteed and CBR service models have been widely studied, and are of particular interest for the transport of compressed video. Interestingly, the benefits of workahead smoothing have not been previously evaluated under either model.

These service models are disjoint with respect to the types of smoothing they permit within the network.

- Guaranteed service gains performance by introducing delay (i.e., temporal multiplexing) in the network. It permits no statistical multiplexing, since statistical performance is incompatible with hard guarantees. The advantage is guaranteed performance; the disadvantages are added delay and potentially low utilization for bursty VBR streams under moderate delay bounds [11]. As we shall see, under optimal smoothing the performance of Guaranteed service can improve dramatically for bursty VBR-encoded streams.

- RCBR is closely related to traditional CBR service, and employs no temporal multiplexing within the network.

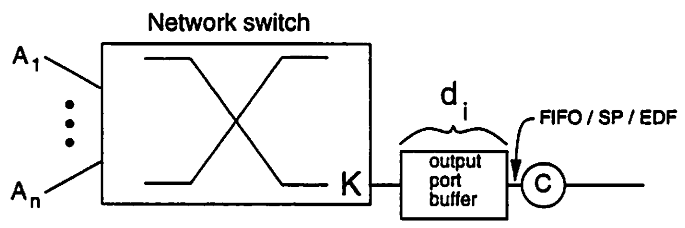


Figure 9: Guaranteed service at a single network switch, with n streams destinated for output port K. A_i is the traffic constraint function for stream i, and d_i is the negotiated bound on queueing delay at the output port.

Instead, performance gains are achieved strictly through statistical multiplexing of network resources via a bandwidth renegotiation mechanism. The advantage is low delay and potentially high utilization; the disadvantages are the renegotiation overheads and the fact that hard guarantees are not provided. Workahead smoothing has not been previously considered under RCBR; the authors in [9] assume per-stream buffering (i.e., temporal multiplexing) at the server, which introduces delay. As we shall see, under optimal smoothing, high performance can be obtained at very low renegotiation rates, without introducing delay.

The performance results we present in the next three sections have been verified to hold across the set of videos in our trace suite. For space considerations, we will restrict attention to a sampling of the representative behavior.

5 Guaranteed service

In this section, we evaluate the benefit of optimal smoothing under Guaranteed service. We consider the improvement in achievable network utilization when individual streams are optimally smoothed, as well as the impact on performance under practical design alternatives. Our objective is to demonstrate that under optimal smoothing, Guaranteed service can achieve high network utilization for bursty VBR video.

We begin with an overview of the service model, and then turn to performance results.

5.1 Overview

Under Guaranteed service, the network provides hard guarantees. Suppose the user requires *bounded delay* service [1]: for each packet sent, the end-to-end delay through the network must not exceed some *a priori* bound. (This implies that no packets are lost). A *call admission algorithm* admits the stream into the network only if it can guarantee that the delay bound will never be exceeded.

We use the formulation of bounded-delay Guaranteed service given in [11]. Consider a single network switch, as depicted in Figure 9. Suppose the n incoming streams are routed to the K^{th} output port, and assume the delay through the switch fabric is constant. Let the capacity of the outgoing link be C bits/s. A packet arriving at the output port is transmitted immediately if the link is idle, or is buffered otherwise. Some link scheduling policy, such as FIFO, Static Priority (SP), or Earliest Deadline First (EDF), governs the selection of which buffered packet to transmit next onto the output link. Each stream i has a negotiated delay bound d_i, indicating the maximum amount of time that any of its packets will wait for transmission. In this simple formulation, the maximum jitter across the switch is also d_i, since the packet queueing delay can range anywhere from 0 to d_i.

To enable call admission at the switch, a traffic description must be specified for each stream. When deciding whether to admit a new stream k, the call admission algorithm considers its traffic description, those of the admitted streams, and the switch scheduling policy. Guaranteed service can be provided if each stream i specifies a time-invariant traffic constraint function, $A_i(t)$, bounding the maximum number of bits that will arrive at the switch from the stream over any interval of length t. The least upper bound, $\mathcal{E}(t)$, is known as the *minimum envelope process* [1] (or *empirical envelope* [11]). Computed at the frame level for a stored video with N frames, $\mathcal{E}(t)$ is given by:

$$\mathcal{E}(t) = \max_{0 \leq k \leq N-t} \sum_{j=k+1}^{k+t} f(j), \tag{1}$$

where $f(j)$ is the size of the j^{th} frame. Thus $A_i(t) = \mathcal{E}_i(t)$ is the exact time-invariant traffic constraint function. Toward the other extreme is *peak rate* allocation, which captures only the worst-case behavior over a single frame time, i.e., $A_i(t) = t\mathcal{E}_i(1)$. In this section, we will use $A_i(t) = \mathcal{E}_i(t)$, as well as concave upper bounds to $\mathcal{E}_i(t)$, each specified by a set of (σ, ρ) pairs (see Section 5.2.2).

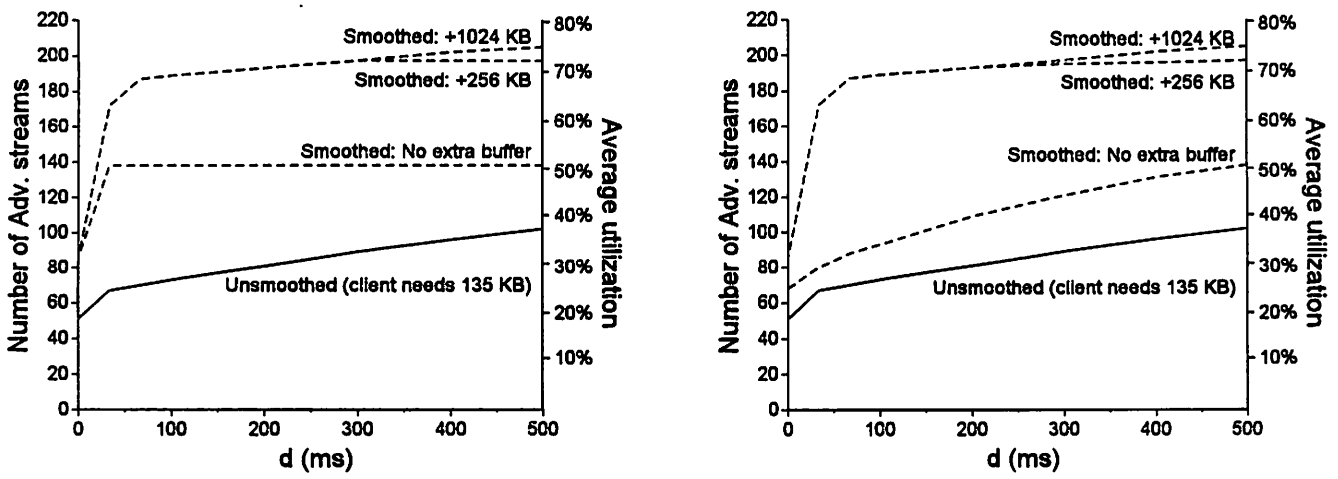
For FIFO, SP, and EDF link scheduling policies, exact admission control tests are known [3, 11]. Assume the n streams are homogeneous (i.e., for all i, $A_i(t) = A(t)$) and require common delay bound d. In this case, FIFO, SP and EDF behave identically, and the admission control test is as follows. When a packet arrives at the switch, its maximum waiting time—the time to clear the greatest possible backlog at the switch (Q bits), plus the time to clear the largest possible packet currently in service ($A(1)$ bits)—cannot exceed d:

$$\frac{Q + A(1)}{C} \leq d \tag{2}$$

Q is given as the maximum, over all t, of the number of bits which can arrive from the n streams within t units of time, minus the number of bits which can be serviced in that time: $Q = \max_t\{nA(t) - ct, 0\}$.

5.2 Performance under optimal smoothing

We evaluate the benefits of optimal smoothing under Guaranteed service in the following manner. First, for a given negotiated delay bound d, we derive a transmission schedule which accommodates the potential jitter of d. See discussion in Section 2.4. Second, to determine the maximum number of admissible streams for a given transmission



(a) *d* ms jitter workahead (b) 500 ms jitter workahead

Figure 10: Optimal smoothing under Guaranteed service, for homogeneous streams with identical delay bounds and $\mathcal{E}(t)$ as the traffic constraint function. The *Advertisements* stream is encoded at 0.457 Mb/s, and the link bandwidth is 127.155 Mb/s. In (a), the server performs jitter workahead corresponding to the negotiated delay bound *d*. In (b), the server performs 500 ms jitter workahead for all values of *d*.

schedule and delay bound, we compute a traffic constraint function for the schedule, and apply the admission control test (Equation 2 for homogeneous streams with identical delay bounds; see [15, 11] for other cases).

We consider first the case of homogeneous streams, with $\mathcal{E}(t)$ as the traffic constraint function.

5.2.1 Homogeneous streams with $\mathcal{E}(t)$ as the traffic constraint function

Figure 10a shows the number of admissible *Advertisements* streams as a function of the negotiated delay bound *d*, over the range 0-500 ms, for an OC-3 link providing 127.155 Mb/s for video data transport (i.e., at the ATM AAL layer [21]). Each stream has a mean bit rate of 0.457 Mb/s. We assume the client is equipped with sufficient buffer space to accommodate the worst-case jitter over the range of delay bounds that may be negotiated with the network. Here, this translates to assuming the client has a nominal buffer capacity of $\mathcal{E}$(500 ms) = 135 KB.

The figure demonstrates the benefit of optimal smoothing, in comparison to the case in which the server follows an unsmoothed transmission schedule, as in [11].[5] With optimal smoothing into the nominal buffer space, the number of admissible streams increases by *as much as 100%*. With an additional 256 KB at the client, performance improves even further, with network utilization rising above 70%. For this stream, further increases in buffer capacity yield only small improvements in performance.

Of course, to achieve the performance indicated in Figure 10a, the server must either precompute and store

[5] Performance improves as *d* increases, since the higher delay bound translates to greater per-stream buffering (i.e., temporal multiplexing) within the switch, which increases achievable switch utilization.

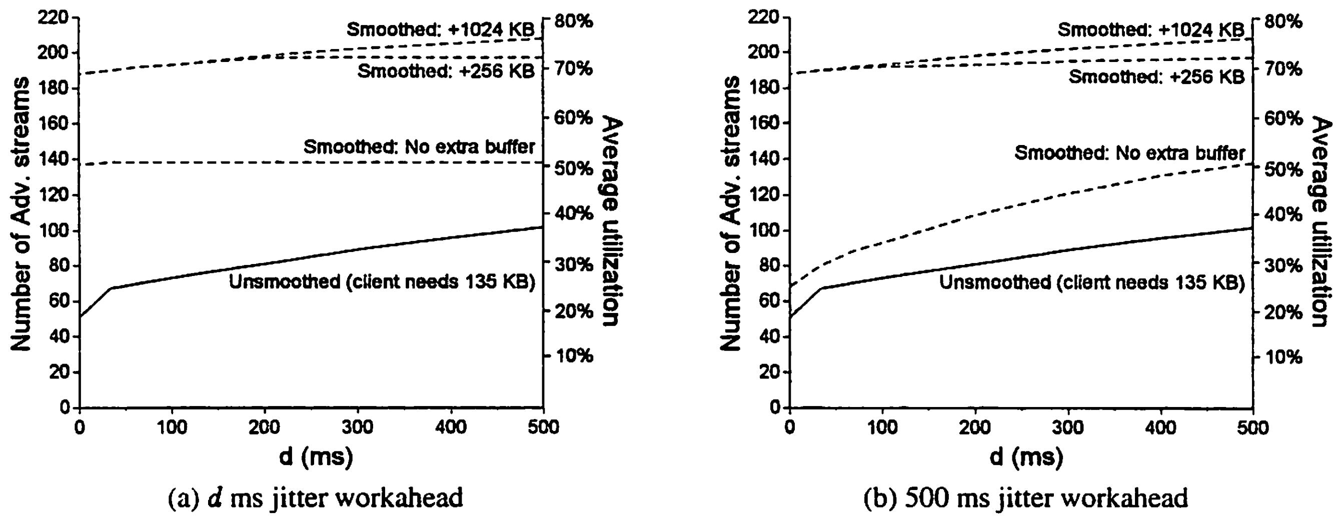
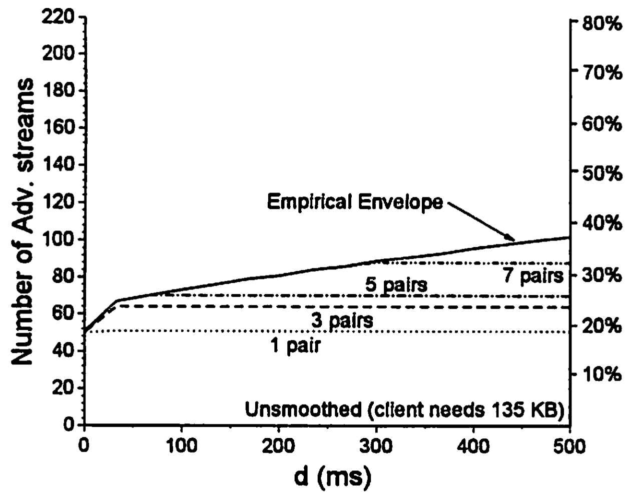


Figure 11: Under optimal smoothing, a small startup latency can improve performance at low delay bounds. Here, a 100 ms (3-frame) startup latency is introduced. in addition to the delay required for jitter workahead. See text for discussion of why the "Smoothed: No extra buffer" curve does not improve in (b).

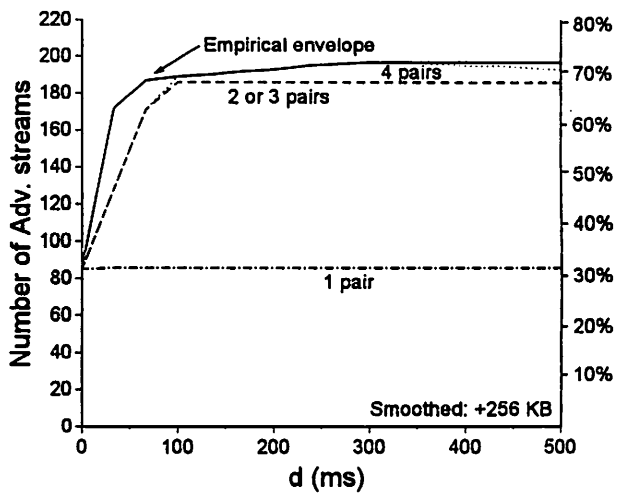
smoothed schedules across the range of possible delay bounds, or compute them online. An alternative is to store one smoothed schedule, for the anticipated *worst-case* delay bound (500 ms in Figure 10). Then, for lower negotiated delay bounds, the transmission schedule is simply over-conservative with respect to jitter workahead. But how much achievable utilization is sacrificed? Figure 10b provides the answer. Here, smoothed transmission schedules are computed with a 500 ms jitter workahead, across all values of *d*. While the drop in performance is most pronounced for the "Smoothed: No extra buffer" curve, its improvement over the unsmoothed curve is still significant. Moreover, when the client is equipped with buffer capacity beyond the nominal requirement, the impact of the over-conservative jitter workahead is negligible.

Note the performance dip at low delay bounds. For optimal smoothing, the transmission of the very first frame of the video, which is typically large for MPEG streams, cannot be smoothed without a startup latency (Section 2). When a small startup latency is added, low-delay-bound performance improves significantly. Figure 11 shows the impact of a 100 ms (3-frame) startup latency.[6] The low-delay performance anomaly disappears. (The apparent exception is the "Smoothed: No extra buffer" curve in Figure 11b. But its performance is lower because the worst-case jitter workahead leaves the client with very little buffer space available for smoothing, not because of the large initial frame, so the additional startup latency has little effect.) These results suggest that under optimal smoothing with sufficient client buffer space, Guaranteed service need not introduce delay in order to achieve high utilization.

[6]The jitter workahead is unaffected. In Figure 11a, client playback is delayed d+100 ms; in 11b, playback is delayed 500+100 ms.



(a) Unsmoothed



(b) Optimally smoothed with/ an additional 256 KB

Figure 12: (a) When the server sends unsmoothed transmissions and a (σ, ρ) traffic constraint function is used, a large number of (σ, ρ) pairs are needed to achieve comparable performance. (b) For optimally smooothed transmissions, a small number of (σ, ρ) pairs yield near-comparable performance. (In both (a) and (b), the server performs jitter workahead corresponding to the negotiated delay bound; no additional startup latency is introduced).

5.2.2 Homogeneous streams with a practical traffic constraint function

The empirical envelope $\mathcal{E}$ is not a practical traffic constraint function, since it is difficult to police [11]. Cruz [3] showed that a concave, piecewise-linear traffic constraint function can be policied by an n-level leaky bucket (where n is the number of linear segments), which can be implemented efficiently. Thus, an alternative to using $\mathcal{E}(t)$ is to form its *concave least upper bound*, i.e., to identify the decreasing piecewise-linear function $C(t)$ which meets $\mathcal{E}(t)$ at $t = 0$, $t = N$, and each of the "change points" of $C(t)$.[7] $C(t)$ can be described by a set of m (σ, ρ) pairs, $\{(\sigma_i, \rho_i)\}$, $i = 1, \ldots, m$, with $C(t) = \min\{\sigma_i + \rho_i t\}$. Here, $\rho_{i-1} > \rho_i$ and $\sigma_{i-1} < \sigma_i$ for $i = 2, \ldots, m$, and $\sigma_1 = 0$. Choosing the pairs with the k-smallest values of σ yields a traffic constraint function $A(t) = \{(\sigma_j, \rho_j)\}$, $j = 1, \ldots, k$, which is a concave upper bound of $\mathcal{E}(t)$ and can thus be efficiently policed. The number of pairs to choose, k, is a design and performance issue.

To achieve comparable performance to the case in which $\mathcal{E}(t)$ is used as the constraint function, a large number of (σ, ρ) pairs are needed when the server sends VBR-compressed video and follows the unsmoothed transmission schedule [11]. Figure 12a demonstrates this phenomenon for the *Advertisements* stream. For delay bounds over 0-500 ms, eight (σ, ρ) pairs are needed to match the performance achieved when the empirical envelope is used. On the other hand, when the individual streams are optimally smoothed, a much lower number of (σ, ρ) pairs are needed, since the empirical envelope of a smoothed stream is itself much smoother. Figure 12b demonstrates that when the client has 256 KB beyond the nominal 135 KB and the server performs optimal smoothing, the traffic constraint

[7] C is unique for a given $\mathcal{E}$.

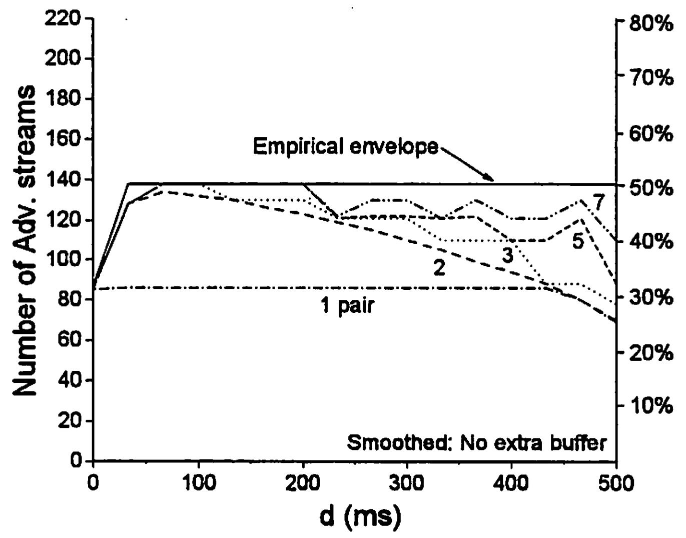


Figure 13: When the client buffer capacity is close to minimum required for jitter workahead and the server performs jitter workahead corresponding to the negotiated delay bound, there are two interesting effects. 1) A large number of (σ, ρ) pairs are needed for performance comparable to the case in which the empirical envelope is used as the traffic constraint function. 2) For a given number of (σ, ρ) pairs, higher utilization is generally achieved when the network provides *lower* delay service (!). See text for explanation. (No additional startup latency has been introduced).

function using only two (σ, ρ) pairs achieves near-comparable performance.

When the client buffer capacity is close to the minimum required for jitter workahead, a large number of (σ, ρ) pairs are needed even when the server performs optimal smoothing. Figure 13 demonstrates this case, under the assumption that the server performs jitter workahead corresponding to the negotiated delay bound. Surprisingly, as the delay bound increases, the performance of a given number of (σ, ρ) pairs generally decreases. The reason is that since more jitter workahead is required as d increases, less buffer space remains for smoothing. As a result, the smoothed stream is burstier for larger d, resulting in a looser traffic constraint function (for a given number of (σ, ρ) pairs) and lower achievable utilization. Thus, we have the somewhat remarkable anomaly: under optimal smoothing, when the client buffer capacity is close to the nominal requirement and a (σ, ρ) traffic constraint function is used, *the network can generally admit more streams by providing lower-delay service.*

5.2.3 Heterogeneous streams

To evaluate the impact of optimal smoothing when the network carries multiple heterogeneous streams, we follow the methodology developed in [11]. We consider EDF and SP switch scheduling policies. Under EDF, each arriving packet is assigned a deadline based on its arrival time and delay bound. An exact (i.e., necessary and sufficient) admission control test for EDF is known [15, 11]. But while EDF is optimal, in the sense that it achieves the highest possible switch utilization, it is complex to implement since a search for the packet with the lowest deadline is required [11]. SP, on the other hand, maintains a FIFO queue for each supported priority level, and simply transmits the next packet at the head of the highest priority queue. Both exact and sufficient admission control tests for SP are known [15, 11]. But while SP is more efficient to implement, it is not optimal.

As in [11], we consider both the "Benchmark case" of EDF scheduling, exact admission control, and the exact traffic constraint function $\mathcal{E}(t)$, as well as the "Practical case" of SP scheduling, sufficient admission control, and a 3-pair (σ, ρ) traffic constraint function. Our goal is to show the impact of optimal smoothing on 1) the achievable utilization under Guaranteed service for heterogeneous streams, and 2) the difference in achievable utilization between the Benchmark and Practical cases.

Figure 14 shows the achievable performance when two types of streams are carried over an OC-3 link. Each 0.457 Mb/s *Advertisements* stream has a 200 ms delay bound, and each 1.25 Mb/s *Wizard* stream has a 100 ms delay bound. The server performs jitter workahead corresponding to the negotiated delay bound; no additional startup latency has been introduced. We consider three cases: 1) unsmoothed transmissions, as in [11], 2) optimal smoothing into the nominal client buffer space (135 KB for *Advertisements* clients, 218 KB for *Wizard* clients), and 3) optimal smoothing when clients have an additional 1024 KB of buffer capacity. From the figure we can draw two broad conclusions. First, as in the case of homogeneous streams, the number of admissible streams and achievable network utilization increase dramatically when the individual streams are optimally smoothed. Second, under optimal smoothing, performance of the Practical case approaches that of the Benchmark case as the client buffer size increases. When individual clients are equipped with an additional megabyte of buffer and streams are optimally smoothed, the difference in achievable utilization between the two cases is at most 5%.

In general, these results indicate that under optimal smoothing, high network utilization can be achieved by Guaranteed service for originally bursty VBR video, while utilizing efficient switch scheduling, call admission, and traffic policing mechanisms.

6 Renegotiated CBR service

We now turn to an evaluation of optimal smoothing under Renegotiated CBR (RCBR) service. We will consider the improvement in network utilization when individual streams are optimally smoothed, as a function of the number of renegotiations which are allowed over the length of the video. Our objectives are to demonstrate that 1) optimal smoothing enables RCBR to achieve high network utilization at very low renegotiation rates, and 2) by sending optimally-smoothed transmissions, the server can achieve near-optimal reduction in the cost of sending the video.

We begin with an overview of the service model.

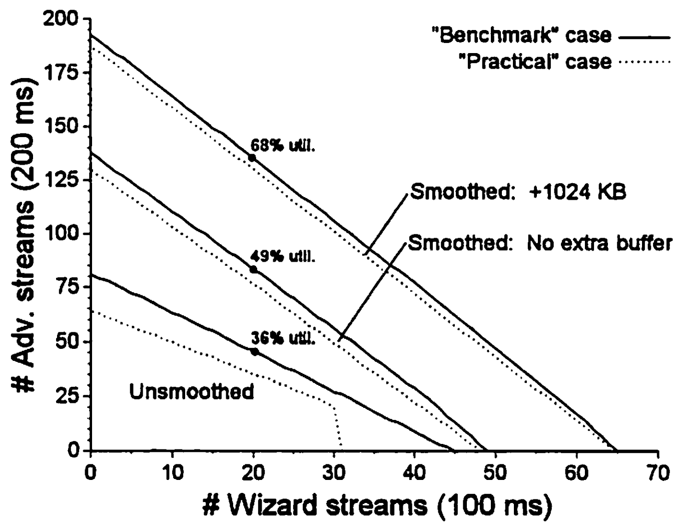


Figure 14: When individual streams are optimally smoothed, Guaranteed service can achieve high network utilization for heterogeneous streams, while implementing efficient switch scheduling, call admission, and traffic policing mechanisms. (The server performs jitter workahead corresponding to the negotiated delay bound; no additional startup latency has been introduced).

6.1 Overview

Under CBR network service, the server reserves network resources at the stream's peak rate. While CBR service offers many advantages (e.g., call admission and switch scheduling are simplified [7]), static CBR—in which the server provides a one-time specification of the peak rate over the duration of the session—does not perform well for VBR-compressed video, since the peak can be much higher than the mean.

Renegotiated CBR (RCBR) adds a bandwidth renegotiation mechanism to static CBR [9, 8]. Under RCBR, the server reserves bandwidth at peak rate over intervals of time, and at the end of each interval, renegotiates a new bandwidth reservation. The renegotiation is actually a signaling request to adjust the reserved bandwidth along the end-to-end path. A request to increase bandwidth may be rejected; thus, RCBR provides a *statistical* QOS, not a guaranteed one. The call admission algorithm must ensure that the probability of rejection is acceptably low. On a renegotiation failure, the server must adapt to the allocated bandwidth. For stored video, one alternative is to transmit a subset of the stream if it has been *scalably compressed* [2][8]; MPEG-2 contains support for limited scalable compression. Another possibility is to dynamically requantize the stream at a lower bit rate. See [9] for references and further discussion.

To evaluate the impact of optimal smoothing under RCBR, we assume a discrete-time model at the frame level, and a network pricing function (indicating the cost to the server of following a given reservation schedule [9]) of the

[8]Scalable compression enables decoding at multiple frame rates, resolution granulatirites and/or spatial sizes.

form:

$$\rho = \sum_{i=1}^{N} \phi(c_i) + \gamma \sum_{i=1}^{N-1} \delta_{c_i, c_{i+1}} \tag{3}$$

where N is the length of the session, c_i is the bandwidth reserved at time i, $\phi(x)$ is the cost of reserving bandwidth x over one time interval, γ is the fixed renegotiation cost, and $\delta_{a,b} = 1$ if $a \neq b$, 0 otherwise.

Renegotiations incur signaling overhead at each switch along the end-to-end path. The network can limit the renegotiation frequency indirectly, through pricing (i.e., by making the cost of renegotiations high) [9], or directly, by imposing a bound R on the number of allowable renegotiations over the length of the session. We will assume the latter scenario. To avoid any *a priori* assumptions about the feasible range of renegotiation frequency, we treat R as an independent parameter.

Our approach is as follows. We first identify an algorithm for determining, for a given transmission schedule S, the minimum-cost *reservation* schedule with R or fewer renegotiations. We apply this algorithm to the smoothed and unsmoothed transmission schedules for a given video. To establish the impact of optimal smoothing on the network, we evaluate the maximum number of reservation schedules that can be carried over a fixed capacity link at a given QOS. For the impact at the server, we examine the cost to the server of following a given reservation schedule, both with and without optimal smoothing.

6.2 The optimal reservation algorithm $\mathcal{B}$

Let S be any feasible transmission schedule, expressed as a sequence of n CBR segments, each with length L_i and transmission rate c_i. (Note that the unsmoothed video can be cast in this form). Assume a network pricing function of the form (3), and that the maximum number of allowable renegotiations is R. Algorithm $\mathcal{B}$ (Figure 15) identifies the minimum-cost reservation schedule for S which has R or fewer renegotiations.

In the algorithm, $T_{i,j}$ is the minimum cost reservation schedule over segments $[i, j]$ without any renegotiations:

$$T_{i,j} = \phi(\max_{i \leq k \leq j}(c_k))(\sum_{k=i}^{j} L_k).$$

Let $T_{i,j}$ be 0 if $i > j$. $\omega_{1,i}^{(r)}$ denotes the cost of cheapest reservation schedule over segments $[1, i]$ with r or fewer renegotiations. The algorithm begins by computing, for each segment i, the minimum-cost reservation from the beginning of the video through i (i.e., over $[1, i]$) with no renegotiations (Lines 1-2). It then enters the main loop. The cheapest reservation over $[1, i]$ with r or fewer renegotiations is computed as the minimium cost, over all $j \in \{1, \ldots, i\}$, of the cheapest reservation over $[1, j]$ with $(r - 1)$ renegotiations, followed by a renegotiation after segment j if $j \neq i$, followed by the cheapest reservation over $[j + 1, i]$ without renegotiations. Once $\mathcal{B}$ is finished,

```
1. for $i = 1$ to $n$
2.     $\omega_{1,i}^{(0)} = T_{1,i}$
3. for $r = 1$ to $R$
4.     for $i = r + 1$ to $n$
5.         $\omega_{1,i}^{(r)} = \min_{1 \le j \le i}\{\omega_{1,j}^{(r-1)} + \gamma\delta_{i,j} + T_{j+1,i}\}$
```

Figure 15: Algorithm $\mathcal{B}$, for finding the minimum-cost reservation schedule, with R or fewer renegotiations, for given transmission schedule S. See text for discussion of notation.

the cost of the cheapest reservation schedule over $[1, n]$, with R or fewer renegotiations, is given in $\omega_{1,n}^{(R)}$. $\mathcal{B}$ requires O(n) space to compute $\omega_{1,n}^{(R)}$, since only $\omega^{(r-1)}$ is needed to compute $\omega^{(r)}$. The time complexity is O(Rn^2), where n is the number of *segments* in the transmission schedule.

Actually *generating* the minimum-cost reservation schedule can be accomplished by keeping track of the segment index j which minimizes the expression in Line 5. That is, by defining $p_i^{(r)}$ as the segment index $j \in \{1, \ldots, i\}$ which minimizes $(\omega_{1,j}^{(r-1)} + \gamma\delta_{i,j} + T_{j+1,i})$. The minimum-cost reservation schedule can then be generated from the $p_i^{(r)}$ and the segment rates c_i.

Algorithm $\mathcal{B}$ runs much faster on optimally-smoothed transmission schedules. For 40,000-frame MPEG-1 traces, we have measured the computation time of Algorithm $\mathcal{B}$ (on an SGI workstation with 150 MHz MIPS R4400 processor) to be less than one minute when the video is smoothed into 64 KB, but over 22 *hours* when unsmoothed.

Several extensions to $\mathcal{B}$ are directly possible. First, the additional constraint of a minimum time between renegotiations can be incorporated, while retaining the O(Rn^2) time complexity. If the network does not impose a bound on the number of renegotiations, the time complexity can be reduced from O(Rn^2) to O(n^2). (See Algorithm $\mathcal{C}$ in Appendix A). Last, it is an easy generalization to support a more general renegotation cost function.

Note that Algorithm $\mathcal{B}$ finds the minimum-cost reservation schedule for a *given* transmission schedule. In Appendix A we present Algorithm $\mathcal{D}$, which finds the minimum-cost reservation schedule over *all* feasible transmission schedules for the given video and client buffer size.

6.3 Performance under optimal smoothing

We now turn to an empirical evaluation of optimal smoothing under RCBR. For simplicity, we assume $\phi(c_i) = c_i$ and $\gamma = 0$. For a given transmission schedule S, we first use Algorithm $\mathcal{B}$ to generate its minimum-cost reservation schedule. To determine the maximum number of admissible streams N_{max} for fixed link capacity L, we use a simple simulation that multiplexes N of the reservation schedules and determines their maximum aggregate bandwidth requirement $C(N)$. A large number of iterations, 500, are performed.[9] In each iteration, random starting points for each stream are computed, and the N reservation schedules are multiplexed over the duration of the video. $C(N)$ is computed as the maximum reserved bandwidth seen during a single frame time, over all iterations. To determine the probability of renegotiation failure p, 500 additional independent iterations are performed, and the fraction of time $C(N)$ is exceeded is computed. (This provides an empirical upper bound on p). Across all experimental data presented below, this bound is on the order of 10^{-5}. Finally, a binary search yields N_{max}, i.e., the largest N such that $C(N) < L$.

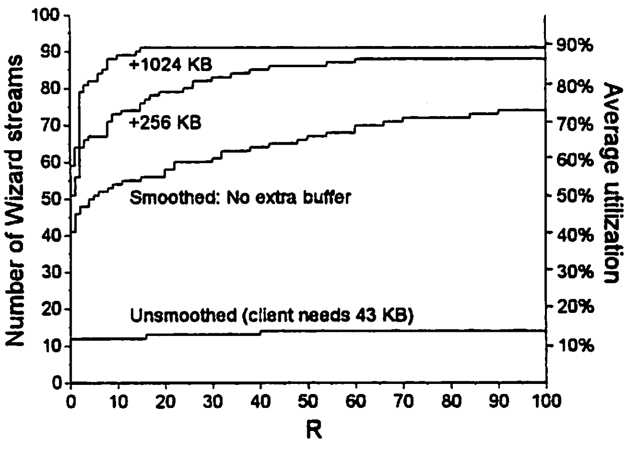
Two notes on the simulation. First, since it operates at the frame level, it yields a conservative estimate on the aggregate bandwidth requirements, and thus may underestimate the maximum number of admissible streams. Second, although synchronization of the multiplexed streams within the simulation is possible, it is unlikely due to the random shifting. This is not a problem, since in a real system this type of synchronization can be avoided or reduced at call admission time.

Figure 16a presents the simulation results for the MPEG-1 encoding of *The Wizard of Oz*, showing the maximum number of supportable streams over an OC-3 link for R ranging from 0 up to 100. This corresponds to a mean renegotiation time ranging from 23 minutes down to 14 seconds. The maximum number of unsmoothed streams which can be admitted is also shown. Since network jitter is a much less of anissue under CBR service [7], we assume it is zero. Thus, we consider as the base case that the client has sufficient buffer space to store only the largest frame (43 KB for this video), which is the minimum needed to support the unsmoothed transmission schedule in the absence of network jitter.

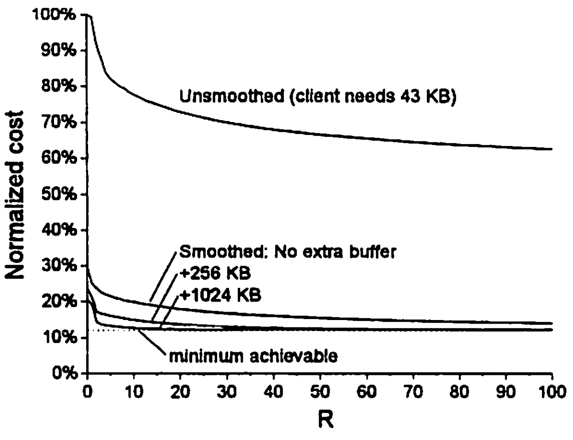
The figure shows that RCBR performs very poorly for the unsmoothed transmission schedule over this range of R. This argues that some sort of smoothing is absolutely necessary unless renegotiations can be performed very frequently. By optimally smoothing into the minimum client buffer size of 43 KB, network utilization increases dramatically, into the range of 50-70%. Performance improves even more when clients are given additional buffer space, with the addition of one megabyte realizing nearly the maximum possible gain. Perhaps the most striking aspect of Figure 16 is that optimal smoothing enables very high network utilization—in the range of 80-90%—at very low renegotiation rates, for clients equipped with 256-1024 KB of buffer space.

Does the server have an incentive to send smoothed video, thus resulting in these high network utililizations? Consider Figure 16b, which shows the cost to the server of sending the stream. Here, we have normalized by the cost of sending an unsmoothed stream without renegotiations. Optimal smoothing results in a remarkable drop in the cost to the server. In particular, note that optimal smoothing into a client buffer approximately one megabyte in size realizes the minimum achievable cost for $R > 55$ (where this cost is given by the total size of the video, since we assume that $\gamma = 0$ and $\phi(c_i) = c_i$).

[9]This number was arrived at experimentally as yielding stability in the measurement of $C(N)$.



(a) Network impact



(b) Server impact

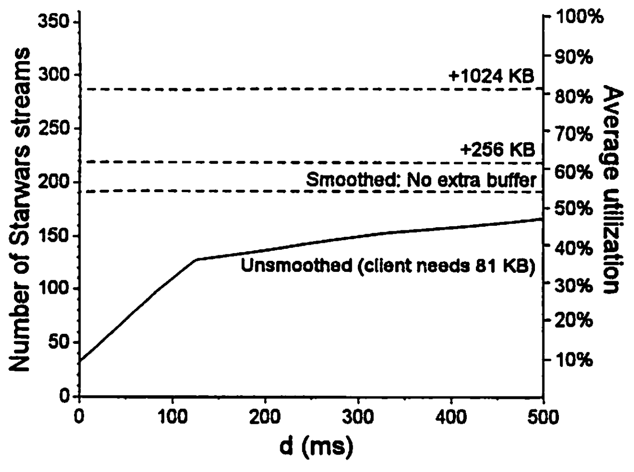
Figure 16: Performance of optimal smoothing under RCBR service, for the 1.25 Mb/s MPEG-1 encoding of *The Wizard of Oz* over an OC-3 link (127.155 Mb/s available for video data transport). In (b), costs have been normalized by the cost of sending an unsmoothed video without renegotiations.

From Figures 16a and 16b, we can draw two broad conclusions. The first is that optimal smoothing yields significant benefit under RCBR, both in the network, in terms of increasing the maximum number of admissible streams, as well as at the server, in terms of reducing the cost of sending the video. The second is that when streams are optimally smoothed, relatively infrequent renegotations (on the order of once a minute) yield high performance.

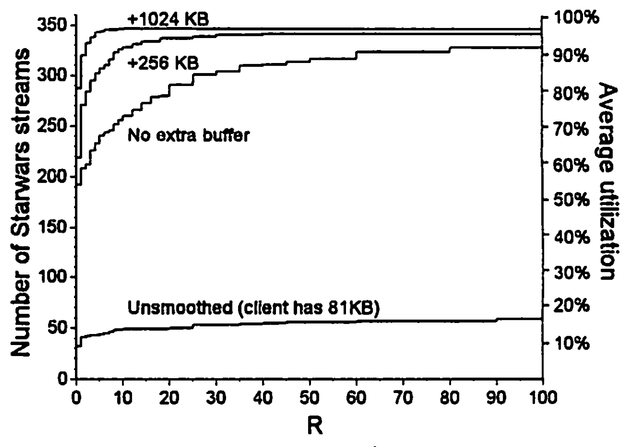
7 Comparison between Guaranteed service and RCBR

In this section, we compare the performance of Guaranteed and RCBR network service models under optimal smoothing. In conducting such a comparison, we must keep in mind that the two models differ fundamentally in the type of service that is being provided. Guaranteed Service provides hard guarantees on network performance, but relies on delay in the network to increase utilization. RCBR provides very-low-delay service, but relies on renegotiations and statistical multiplexing to increase network performance. This introduces end-to-end signaling overhead and the probability of renegotation failure.

Figure 17 provides a basis for comparison between the two service models, and shows the maximum number of *Star Wars* streams which can be concurrently carried over an OC-3 link. For both models, the figure shows the performance for unsmoothed and optimally-smoothed transmissions into various client buffer sizes. Under Guaranteed service, the client needs a nominal 81 KB to accommodate potential network jitter of 500 ms, so we consider this buffer size as the base case. The Guaranteed service curves are computed with exact jitter workahead (i.e., the server performs workahead corresponding to the negotiated delay bound d). No additional startup latency has been introduced; high performance is achieved at low negotiated delay bounds because the first frame of the



(a) Guaranteed service



(b) RCBR service

Figure 17: Comparison between Guaranteed and RCBR service models under optimal smoothing, for a 28-minute, 0.356 Mb/s MPEG-1 encoding of *Star Wars* over an OC-3 link (127.155 Mb/s available for video data transport). To reduce the computation required to generate these curves, we have considered only the first 40,000 frames of the longer 174,000-frame trace.

video is not particularly large. The empirical envelope is used as the traffic constraint function. For RCBR, the number of admissible streams is determined via simulation (as described in Section 6); the empirically-determined probability of renegotiation failure is on the order of 10^{-5}.

For Guaranteed service, network utilization of 80% is achievable across the broad range of negotiated delay bounds when the client is equipped with an addition 1 MB of buffer space. In fact, for this video, Guaranteed service need not introduce *any* delay to achieve high network utilization. For RCBR, network utilization can exceed 90% when clients renegotiate about once a minute. While RCBR achieves higher utilization, the RCBR curves should be considered an upper bound on achievable performance. A practical call admission algorithm which achieves high utilization under RCBR has not yet been demonstrated. To keep the comparison more balanced, the Guaranteed service curves have been computed using the empirical envelope; a more realistic traffic constraint function would lower utilization somewhat.

8 Related work

Reibman and Berger [21, 20] have conducted an extensive study of compressed video transport over ATM, for video generated online by delay-sensitive applications such as teleconferencing. One of their objectives is to evaluate smoothing techniques within the video encoder which reduce the stream's peak rate, and thus increase the number of streams that can be carried over a CBR network service. To this end, they introduce delay at the encoder, and perform workahead smoothing over the range of delays (0-4 frame times) which can be tolerated by the application. By constrast, since we consider stored video and much larger buffer capacity at the client, we can perform smoothing on a much broader time scale, thereby achieving more effective reduction in the stream's slow time-scale rate variability.

In the more recent study [21], Reibman and Berger assume for the sake of analysis that the stream frame sizes are known *a priori*. They then apply Lee and Preparata's *shortest path* algorithm [14] in the context of the encoder and decoder buffer timing constraints [22] to derive a schedule which reduces the peak rate at which frames are drained from the encoder buffer.[10] While they suggest as motivation that the shortest-path schedule should maximize the effectiveness of the allowable delay, they state without proof that it minimizes the peak transmission rate.

Our results imply not only that the schedule used by Reibman and Berger has minimum peak, but that in fact it is *optimally smooth*—and thus has minimum variance as well. If we define the *length* of the "path" of transmission schedule $S = [a(1), \ldots, a(N)]$ to be $\sum_{t=1}^{N} \sqrt{1 + a^2(t)}$ (i.e., in the Euclidian distance sense, as in [14]), it follows as a direct consequence of Theorem 1 (Section 3) that the smoothest schedule S^* has the shortest length path among all feasible schedules.[11] By appropriately defining $D(t)$ and $B(t)$ to reflect the encoder/decoder timing constraints,[12] Algorithm $\mathcal{A}$ can be used to identify the smoothest possible transmission schedule in Reibman and Berger's online formulation. Since S^* is unique (Section 3), we conclude that they are in fact using the *smoothest possible* transmission schedule. Thus, with respect to reducing rate variability, the results in [21] can be viewed as establishing an upper bound on the performance of any online smoothing algorithm.

Several other papers have examined smoothing of delay-sensitive, online compressed video (e.g., [13, 19]). The approach always involves the introduction of delay to enable smoothing, and can be coupled with adaptive congestion control techniques to dynamically adjust the encoder output rate [10]. Heuristics for forecasting future frame sizes may be employed (e.g., based on the specific compression pattern, as in [13]). At the level of cell smoothing, Shroff and Schwartz [24] have established that deterministic injection of cells into the network approaches minimum achievable loss under multiplexing and routing.

For stored video, Feng and Sechrest [5] have also considered workahead smoothing into client buffer space. They do not, however, consider the problem of optimally-reducing stream variability. Instead, they focus on reducing the number of rate *increases* required during stream transmission. Their approach is to segment the video into intervals over which CBR transmission is enabled, with possible rate decreases. The heuristic is not optimal from the perspective of minimizing rate variability. (We have reproduced their experimental results to verify this fact). In

[10]Since this algorithm is not *causal* (i.e., considers future frame sizes in determining the transmission rate at any particular time), the assumption that stream frame sizes are known a priori is thus necessary. As Reibman and Berger stress, the shortest path algorithm therefore cannot actually be used to identify a transmission schedule for video generated online.

[11]Theorem 1 ($S^* \prec S$ for any feasible schedule S), in conjunction with Result 1 ($X \prec Y$ *iff* $\sum_i \phi(x_i) \leq \sum_i \phi(y_i)$, $\forall$ continuous convex ϕ) and the fact that $\phi(x) = \sqrt{1 + x^2}$ is convex in x, together imply that S^* has the shortest path of any feasible schedule.

[12]For simplicity, suppose both encoder and decoder have buffer size b, and assume zero network delay. Let $E(t)$ denote the size of the frame generated by the encoder at time t, and f denote the delay introduced to enable smoothing. Defining $D(t) = E(t - f)$ and $B(t) = \min(E(t), E(t - f) + b)$ captures the encoder/decoder timing and buffer constraints.

general, it is somewhat unclear that the strategy of reducing the *number* of rate increases, at the expense of increasing the size of each request, will yield better network performance. Instead, our strategy is to keep the transmitted bit rate as smooth as possible, which minimizes the peak rate and burstiness of the stream for which resources need to be allocated.

McManus and Ross have also considered workahead smoothing of stored video [18, 17]. They do not consider the problem of optimally-reducing rate variability. In [18], they consider a client startup latency during which the server transmits the first d frames of the video. The focus is on establishing the relationship between d and the amount of client buffering b which together enable the server to transmit at a constant bit rate over the remainder of the video. The authors present 1) a method of determining, for a given startup latency, whether CBR transmission is feasible; 2) an algorithm for finding the transmission rate which minimizes the client startup delay, when it is feasible; and 3) an algorithm for finding the minimum client buffer size b such that the server can feasibly transmit at CBR. They also consider piecewise-CBR transmissions of a VBR stream. In [17], the authors focus on identifying the transmission schedule which minimizes the client buffer requirements, for a constraint K on the number of CBR transmission segments over the length of the video. They present an optimal algorithm, as well as a heuristic with lower time complexity, and demonstrate the performance of both for a VBR-encoded MPEG-1 video trace. Finally, they discuss support for limited VCR features (e.g., fast-forward, rewind and pause) when the server transmits in a piecewise-CBR fashion.

In light of the multiple-time-scale variability of VBR video, there is a growing interest in renegotiated services (e.g., [9, 8, 25]). While our study of renegotiated CBR is based directly on the work of Grossglauser, Keshav and Tse [9, 8], our formulation is somewhat different. They simultaneously consider both stored video and video generated online. To accommodate the online case, they assume that a fixed amount of per-stream buffering is available, at the server, to smooth the outgoing network transmissions. Arrivals into the buffer occur periodically at the stream frame rate; the buffer is serviced at a renegotiated constant rate whenever it contains data. The same setup is used for the stored video case.

One limitation of this formulation is that it does not explicitly consider the client's timing constraint. To ensure continuous playback, the client would need to be delayed long enough to accommodate the maximum jitter introduced by the server's buffer, a length of time which depends on the specific video and client buffer size. Formally, let b denote the server buffer size, $e(t)$ denote the size of the frame arriving into the buffer at time t, and set $E(t) = \sum_{i=1}^{t} e(t)$. The authors in [9] ensure that 1) the server buffer never overflows, and 2) the server never sends more data than is available to be sent. This can be expressed as the constraint $F(t) \leq A(t) \leq E(t)$, where $F(t) = \max\{E(t) - b, 0\}$. It is within the context of this constraint that the authors present an algorithm for identifying the minimum-cost reservation schedule for a given video and server buffer size. But note that to ensure continuous playback at the client, the client consumption curve $D(t)$ (which has the same shape as $E(t)$) must lie entirely below $F(t)$ for all t. Intuitively, the minimum startup delay is given by the smallest shift of $E(t)$ to the right such that the resulting curve lies entirely below $F(t)$ for all t. This minimum startup latency depends on the shape of $E(t)$ (i.e., the particular video) and the server buffer size b.

By constrast, under our formulation of RCBR, we consider only workahead smoothing of stored video, which can be performed without introducing delay in client playback. We have presented three optimal reservation algorithms: 1) Algorithm $\mathcal{B}$, which finds the minimum-cost reservation schedule for a given feasible transmission schedule, with a constraint on the maximum number of allowable renegotiations, 2) Algorithm $\mathcal{C}$, which finds the minimum-cost reservation schedule for a given feasible transmission schedule, with no constraint on the maximum number of allowable renegotiations, and 3) Algorithm $\mathcal{D}$, which finds the minimum-cost reservation schedule over *all* feasible transmission schedules for the video. Algorithm $\mathcal{D}$ explicitly considers both the client timing constraints and any constraint on server workahead (i.e., through appropriate definition of $B(t)$).

9 Summary

In this paper, we have considered the problem of transmitting stored video from a server to a client across a high-speed network. We have assumed a certain availability of client buffer space for server workahead, and studied the issue of how the server can use this space most effectively toward reducing the variability of the transmitted bit rate. We are motivated by the fact that VBR compressed video exhibits significant, multiple-time-scale rate variability. Specifically, we have examined two fundamental issues: 1) Given buffer size b available for server workahead, how can the video be sent from the server such that the transmitted bit rate is as *smooth* as possible, while ensuring continuous playback at the client? 2) Given the server sends optimally-smoothed transmissions, what is the impact on the network resources required to transmit the stream?

In response to the first question, we have established an *optimal smoothing algorithm* which identifies, for a given video, the feasible server transmission schedule S^* which is as smooth as possible. The algorithm relies strictly on smoothing by workahead, and introduces no into client playback. We provide a formal proof of optimality, based on the theory of *majorization*. As a simple consequence of a general majorization result, we show that S^* has minimum variance and minimum peak rate among all feasible schedules for the video.

The smoothing algorithm is efficient, easy to implement, and network jitter can be accommodated in a straightforward manner. We have demonstrated the algorithm's performance on a set of long VBR-encoded MPEG-1 traces, and found in all cases that significant reduction in rate variability is achievable for client buffer sizes in the range of 64-1024 KB, or larger. In general, under optimal smoothing, the capacity to buffer 5-10 seconds of workahead data appears to be sufficient to effectively smooth the slow time-scale rate variability of VBR compressed video.

To address the reduction in network resources required under optimal smoothing, we have considered two important network service models: Guaranteed service (e.g., [1, 11]) and Renegotiated CBR service [9]. The impact of workahead smoothing (optimal or otherwise) has not previously been evaluated under either model. We find performance of Guaranteed service improves dramatically under optimal smoothing, with network utilizations increasing by *as much as a factor of 3* for bursty VBR streams. Our findings indicate that Guaranteed service can achieve high network utilization for these streams, even when practical switch scheduling, call admission, and traffic policing mechanisms are used. For RCBR service, we find that optimal smoothing enables very high network utilization *at very low renegotiation rates* (e.g., on the order of once a minute), thereby lowering the overall end-to-end network signaling overhead. Moreover, the server can achieve a near-optimal reduction in the cost of sending the video, by following an optimally-smoothed transmission schedule.

Overall, our findings indicate that optimal smoothing can result in a dramatic reduction in the network resources required for bursty VBR video. Moreover, the benefit is achievable for client buffer sizes in the range of 64-1024 KB (i.e., when the video customer is equipped with a low-cost set-top box), and is independent of any particular server architecture. Given the projected growth in applications which store, processes and transmit compressed video, the issue of reducing rate variability is likely to be a central one for quite some time.

Acknowledgements

We would like to thank the researchers who generously shared their MPEG traces. In particular, the contributions of Ed Knightly [11], Marwan Krunz [12], Mark Garrett [6] and Oliver Rose [23] are gratefully acknowledged.

A RCBR: Two additional optimal reservation algorithms

Algorithm $\mathcal{B}$, presented in Section 6, finds the minimum-cost reservation schedule for a *given* feasible transmission schedule, subject to a constrant on the maximum number of allowable renegotiations R. In this appendix, we present two additional optimal reservation algorithms: 1) Algorithm $\mathcal{C}$, which relaxes the constraint on R, and 2) Algorithm $\mathcal{D}$, which finds minimum-cost reservation schedule over *all* feasible transmission schedules for the video.

Figure 18 presents Algorithm $\mathcal{C}$, following the notation introduced earlier. The network pricing function is of the form given in Equation 3 (Section 6): $\phi(c)$ is the cost of reserving bandwidth c over one time interval, γ is the fixed renegotiation cost, and $\delta_{a,b} = 1$ if $a \neq b$, 0 otherwise. Assume the given feasible transmission schedule is expressed as a sequence of n CBR segments, each of length L_i with transmission rate c_i. Let $T_{i,j} = \phi(\max_{i \leq k \leq j}(c_k))(\sum_{k=i}^{j} L_k)$ denote the minimum-cost reservation schedule over segments $[i, j]$ without renegotiations, defined to be 0 if $i > j$. $\omega_{i,j}$ denotes the cost of the minimum-cost reservation schedule over $[1, i]$, with no restriction on the number of renegotiations.

The algorithm initializes $\omega_{1,i}$ to $T_{1,i}$, $1 \leq i \leq n$, and then iterates over segments 2 through n. $\omega_{1,i}$ is computed as the minimium cost, over all $j \in \{1, \ldots, i\}$, of the minimum-cost reservation over $[1, j]$ (computed on an earlier

```
1.  for $i = 1$ to $n$
2.      $\omega_{1,i} = T_{1,i}$
3.  for $i = 2$ to $n$
4.      $\omega_{1,i} = \min_{1 \le j \le i}\{\omega_{1,j} + \gamma\delta_{i,j} + T_{j+1,i}\}$
```

Figure 18: Algorithm $\mathcal{C}$, for finding the minimum-cost reservation schedule for given transmission schedule, with restrictions on the number of allowable renegotiations. See text for discussion of notation.

iteration for $j \neq i$ or during the initialization for $j = i$), followed by a renegotiation after segment j if $j \neq i$, followed by the cheapest reservation over $[j + 1, i]$ without renegotiations. Once $\mathcal{C}$ is finished, the cost of the cheapest reservation schedule over $[1, n]$ is given in $\omega_{1,n}$. The algorithm requires O(n) space and O(n^2) time. To actually generate the minimum-cost reservation schedule requires keeping track of the segment index j which minimizes the expression in Line 4, which can be done in O(n) space.

We now present Algorithm $\mathcal{D}$, which finds the minimum-cost reservation schedule over *all* feasible transmission schedules for the video. Recall a transmission schedule is said to be *feasible* if it ensures the client buffer never starves nor overflows during playback, i.e., $D(t) \leq A(t) \leq B(t)$ for all t (see Table 1 for discussion of this notation). To develop $\mathcal{D}$, we will assume that the video frame sizes, the client buffer size b, and the set of reservable bandwidths $\{r_1, \ldots, r_M\}$, are all discretized with and measured in a common granularity Δ. Note that a single-bit granularity could be specified.

Figure 19 presents the algorithm, which operates at the frame level. $C_{n,i,m}$ denotes the cost of the cheapest reservation schedule over frames $[1, n]$, ending with buffer level i at time n (*after* consumption of $d(n)$), with r_m indicating the rate which was reserved by the server for transmission of $a(n)$. N is the length of the video in frames. The algorithm iterates over n ranging from 1 to N. At each step, it computes $C_{n,i,m}$ for all i and m; upon completion, the minimum-cost reservation schedule over all feasible transmission schedules is given by $\min_m\{C_{N,0,m}\}$. For a given i and m, O(M) earlier costs are considered when $i < B(n) - D(n)$ (Line 9), and O(bM) earlier costs when $i = B(n) - D(n)$ (Line 15). Thus, the time complexity of the algorithm is O(bM^2N). To actually generate the minimum-cost schedule, the history of reservations leading to buffer level i at time n must be maintained for all m, so the space complexity is O(bMN).

$\mathcal{D}$ solves a different optimization problem than the ones considered in [9] and [17]. See Section 8 for discussion. Note that the operation of Algorithm $\mathcal{D}$ does not depend on $B(t)$ lying at a fixed offset above $D(t)$. In particular, $B(t)$ can be chosen to reflect a workahead constraint at the server.

```
for $m = 1$ to $M$
    $C_{0,0,m} = 0$
for $n = 1$ to $N$
    for $i = 0$ to $B(n) - D(n) - 1$
        for $m = 1$ to $M$
            $C_{n,i,m} = \infty$
            $j = i - r_m + d(n)$
            if $(0 \leq j \leq B(n-1) - D(n-1))$
                $C_{n,i,m} = \phi(r_m) + \min_{1 \leq s \leq M}\{C_{n-1,j,s} + \gamma\delta_{m,s}\}$
    $i = B(n) - D(n)$
    for $m = 1$ to $M$
        $C_{n,i,m} = \infty$
        $j = \max(i - r_m + d(n), 0)$
        if $(j \leq B(n-1) - D(n-1))$
            $C_{n,i,m} = \phi(r_m) + \min_{j \leq l \leq b,\ 1 \leq s \leq M}\{C_{n-1,l,s} + \gamma\delta_{m,s}\}$
```

Figure 19: Algorithm $\mathcal{D}$, for finding the minimum-cost reservation schedule over all feasible transmission schedules, for a given video and client buffer size. See text for discussion of notation.

B Quantizing S^*

To address the quantization of S^*, denote by c_i the real-valued transmission rate (in bits per frame time) of its i^{th} CBR segment. One method of quantizing S^* is as follows. The server can maintain a real-valued "partial byte counter", p, which is initially zero. At each frame time within segment i, the server increments p by $c_i - \lfloor c_i \rfloor$, sends $s = \lfloor c_i \rfloor + \lfloor p \rfloor$ bytes, and decrements p by 1 if $p \geq 1$. Note that p is zero after the last frame of the segment is sent, since the size of each frame is an integral number of bytes. (Over ATM, the server would pad out s to a 48-byte boundary, in order to send an integral number of ATM cells). Since this quantized schedule is always less than one byte behind the original S^*, and the size of each frame is at least one byte, the client buffer never starves. Further, overflow cannot occur under the quantized schedule, since it is never ahead of the original S^*.

References

[1] C.S. Chang. Stability, queue length, and delay of determinstic and stochastic queueing networks. *IEEE Transactions on Automatic Control*, 39(5):913–931, May 1994.

[2] Ed Chang and Avideh Zakhor. Scalable video data placement on parallel disk arrays. In *IS&T/SPIE Symposium on Electronic Imaging Science and Technology*, San Jose, CA, February 1994.

[3] R. Cruz. A Calculus for Network Delay, Part I: Network Elements in Isolation. *IEEE Transactions on Information Theory*, 37(1):114–131, January 1991.

[4] William Feller. *An Introduction to Probability Theory and its Applications*, Volume I. John Wiley and Sons, 1968.

[5] Wu-chi Feng and Stuart Sechrest. Smoothing and buffering for delivery of prerecorded compressed video. In *IS&T/SPIE Multimedia Computing and Networking*, pages 234–232, San Jose, CA, February 1995.

[6] Mark Garrett and Walter Willinger. Analysis, modeling and generation of self-similar VBR video traffic. In *Proc. ACM SIGCOMM*, pages 269–280, London, England UK, August 1994. ACM.

[7] M. Grossglauser and S. Keshav. On CBR service. In *Proc. IEEE INFOCOM*, San Francisco, CA, March 1996. To appear.

[8] M. Grossglauser, S. Keshav, and D. Tse. The case against variable bit rate services. In *Proc. 5th Workshop on Network and Operating Systems Support for Digital Audio and Video*, pages 307–310, Durham, NH, April 1995.

[9] M. Grossglauser, S. Keshav, and D. Tse. RCBR: A simple and efficient service for multiple time-scale traffic. In *Proc. ACM SIGCOMM*, pages 219–230, Boston, MA, August 1995.

[10] H. Kanakia, P. P. Mishra, and A. Reibman. An adaptive congestion control scheme for real-time packet video transport. In *Proc. ACM SIGCOMM*, pages 20–31, San Francisco, CA, September 1993.

[11] Edward W. Knightly, Dallas E. Wrege, Jörg Liebeherr, and Hui Zhang. Fundamental limits and tradoffs of providing deterministic guarantees to VBR video traffic. In *Proc. ACM SIGMETRICS*, pages 98–107, Ottawa, Canada, May 1995.

[12] Marwan Krunz and Herman Hughes. A traffic model for MPEG-coded VBR streams. In *Proc. ACM SIGMETRICS*, pages 47–55, Ottawa, Canada, May 1995.

[13] Simon S. Lam, Simon Chow, and David K. Y. Yau. An algorithm for lossless smoothing of MPEG video. In *Proc. ACM SIGCOMM*, London, England, August 1994.

[14] D. T. Lee and F. P. Preparata. Euclidean shortest path in the presence of rectilinear barriers. *Networks*, 14:393–410, 1984.

[15] J. Liebeherr, D. Wrege, and D. Ferrari. Exact admission control in networks with bounded delay services. Technical Report CS-94-29, University of Virginia, July 1994.

[16] Albert W. Marshall and Ingram Olkin. *Inequalities: Theory of Majorization and its Applications*. New York, Academic Press, 1979.

[17] Jean M. McManus and Keith W. Ross. Prerecorded VBR sources in ATM networks: Piecewise-constant-rate transmission and transport. *Submitted for publication*, September 1995.

[18] Jean M. McManus and Keith W. Ross. Video on demand over ATM: Constant-rate transmission and transport. In *Proc. IEEE INFOCOM*, San Francisco, CA, March 1996. To appear.

[19] Teunis Ott, T. V. Lakshman, and Ali Tabatabai. A scheme for smoothing delay-sensitive traffic offered to ATM networks. In *Proc. IEEE INFOCOM*, pages 776–785, Florence, Italy, May 1992. IEEE.

[20] Amy R. Reibman and Arthur W. Berger. On VBR video teleconferencing over ATM networks. In *Proc. IEEE GLOBECOM*, pages 314–319, 1992.

[21] Amy R. Reibman and Arthur W. Berger. Traffic descriptors for VBR video teleconferencing over ATM networks. *IEEE/ACM Transactions on Networking*, 3(3):329–339, June 1995.

[22] Amy R. Reibman and Barry G. Haskell. Constraints on variable bit-rate video for ATM networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 2(4):361–372, December 1992.

[23] Oliver Rose. Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems. Technical Report 101, University of Würzburg Institute of Computer Science, February 1995. Many MPEG-1 traces are available via FTP from `ftp-info3.informatik.uni-wuerzburg.de` in `pub/MPEG`.

[24] Ness Shroff and Mischa Schwartz. Video modeling within networks using deterministic smoothing at the source. In *Proc. IEEE INFOCOM*, pages 342–349, 1994.

[25] Hui Zhang and Edward W. Knightly. A new approach to support delay-sensitive VBR video in packet-switched networks. In *Proc. 5th Workshop on Network and Operating Systems Support for Digital Audio and Video*, pages 275–286, Durham, NH, April 1995.