*To appear in Proc. ACM SIGMETRICS, May 1996*

# 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
Department of Computer Science, University of Massachusetts, Amherst, MA, 01003 USA
{salehi,zhzhang,kurose,towsley}@cs.umass.edu

## 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, 9] and Renegotiated CBR (RCBR) service [8, 7]. 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, 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 [5, 8]. 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 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 dictates that their aggregate bandwidth requirements converge to a Normal distribution as the number of streams becomes large. This smoothing effect does not introduce delay. 3) Smoothing by *work-ahead* 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 [20] for a precise formulation of these constraints. Work-ahead 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 [4, 8, 9, 11, 17, 19, 20, 23]. In general, network service models may permit only temporal multiplexing [9], only statistical multiplexing [8], 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 work-ahead 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* [14] 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 suffices to say 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 for the stream? Intuitively, smoother transmissions should require fewer resources.

We begin 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, is easy to implement, and can be used in a manner that accommodates network jitter. We evaluate its performance on a diverse set of VBR-encoded MPEG-1 traces (one of which has been shown to exhibit *self-similarity* in its bit rate variation [5]), demonstrating reduction in peak rate of 75-87% and in standard deviation of 72-84% for a 1 MB client buffer. These benefits can be realized by small, uniprocessor servers as well as large, scalably-parallel ones.

We then evaluate the impact on network performance when streams are optimally smoothed, considering two network service models: *Deterministic Guaranteed* service [1, 9], which provides hard guarantees, and *Renegotiated CBR* (RCBR) [8, 7], which adds a renegotiation mechanism to traditional CBR service. We find that optimal smoothing yields dramatic improvement in performance

*This work was supported by NSF under grant NCR-9206908 and by ARPA under ESD/AVS contract F-19628-92-C-0089.



SIGMETRICS 96-5/96 Philadelphia, PA, USA

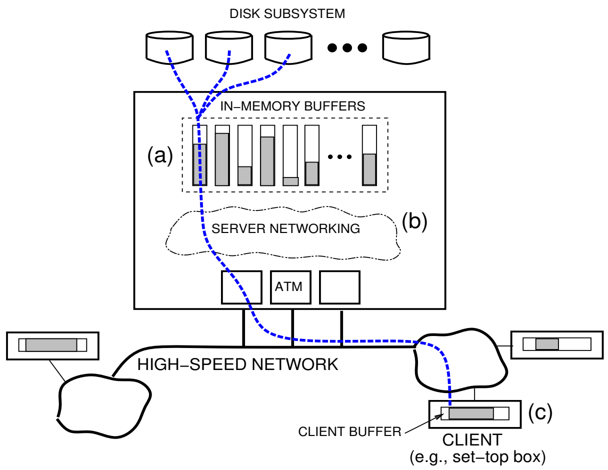



Figure 1: Overview of the problem setting

under both models.

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 4-6, we evaluate the impact on resource allocation under the two network service models. Related work is reviewed in Section 7.

## 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 traces, and show how to accommodate network jitter.

We begin with an overview of the 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 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 needs 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 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 $\tau$ 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 2.3.

### 2.1 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.

| | | |
|---|---|---|
| $b$ | : | 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 $t = 2, \ldots, N$, $B(1) = b$, $B(0) = 0$. (B(t) need not lie at a fixed offset above D(t); see text.) |

Table 1: Notation

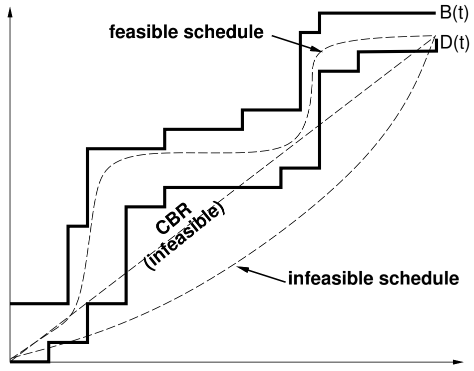



Figure 2: Various transmission schedules, in continuous time

We say the $N$-dimensional real vector $S = [a(1), \ldots, a(N)]$ represents a *feasible* server transmission schedule over $[1, N]$ if 1) the server sends exactly the amount of data required by the client, or $\sum_{t=1}^{N} a(t) = D(N)$, and 2) the client buffer neither starves nor overflows during stream playback, or $D(t) \le A(t) \le B(t)$ for all $t$ (see Table 1). 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 schedule with minimum variance and peak rate. The variance of schedule $S$ is given by $\mathrm{var}(S) = \sum_t (a(t) - \bar{a})^2/N$, 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 may not be feasible (Figure 2).

Instead, we construct a feasible piecewise-CBR transmission schedule, relying on the following two insights. First, since a CBR transmission is as smooth as possible, in constructing the schedule we consider CBR transmission segments which are as long as possible. Second, when the transmission rate must be increased or decreased to ensure feasibility, we change the rate as *early* as possible. This ensures that the rate change is as *small* as possible, which is advantageous from the perspective of making the overall schedule as *smooth* as possible.

Figure 3 illustrates the construction of $S^*$. We begin at $t$=0 by identifying the longest feasible CBR segment, to (a). Since the client buffer would starve at (a), the rate must be increased at some

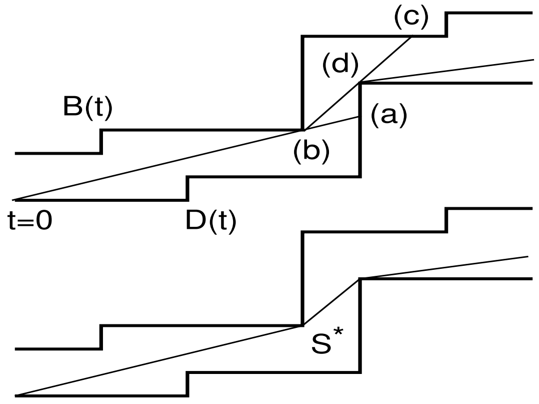



Figure 3: An illustration of the iterative construction of $S^*$. See text for discussion.

1. `PROCEDURE` Find_optimal_schedule $(d(t), b)$
2. $t_s = 0, t_e = 1, q = 0$
3. $cmax = b, t_B = 1, c_{min} = d(1), t_D = 1$
4. `REPEAT`
5. Set $t'_e = t_e + 1$
6. `IF` $cmax < \frac{D(t'_e)-(D(t_s)+q)}{t'_e - t_s}$, end segment at $t_B$:
7. `OUTPUT` segment $\langle t_B - t_s, cmax \rangle$
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 $cmax, t_B, c_{min}$, and $t_D$ over $[t_s, t_e]$
17. `UNTIL` $t_s = n$
18. `END PROCEDURE`

Figure 4: 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.

earlier point. To keep the rate change as small as possible, we increase the rate at the *earliest* possible time enabling extension of the overall schedule (b). Note this is the latest time at which the client buffer is *full* along the initial CBR segment, and that a rate increase prior to (b) would result in buffer overflow. The process is then repeated, identifying the longest feasible CBR segment from (b). Since the new segment would result in client buffer overflow at (c), the rate must be *decreased* before (c). Again, we change the rate as early as possible (d). Note this is the latest time at which the client buffer is *empty* along the CBR segment, and that a rate decrease prior to (d) would result in buffer underflow. The process continues iteratively until the entire video is segmented and $S^*$ is identified.

The optimal smoothing algorithm $\mathcal{A}$ is presented formally in Figure 4, and relies on the following notation. Let $cmax$ be the maximum rate at which the server may transmit over a given interval $[a, b]$, without overflowing the client buffer, starting from initial buffer level $q$:

$$cmax = \min_{a+1 \le t \le b} \frac{B(t) - (D(a) + q)}{t - a},$$

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

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

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 $q$:

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

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 $q$:

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

CBR transmission over the interval $[a, b]$ is feasible if and only if $cmax \ge c_{min}$ 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 $cmax, t_B, c_{min}$ 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 $cmax$ is too small to avoid starvation at $t'_e$ (see Figure 5a), 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, $cmax$. 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 $t'_e$ (see Figure 5b), 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, $c_{min}$. 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 $c_{min}$, $t_D$, $cmax$ and $t_B$ are computed (Lines 15-16). The process continues until the entire video is segmented. $S^*$ is identified as a series of CBR transmission segments, each ending with a change in transmission rate.

Although theoretically the time complexity of $\mathcal{A}$ is O($N^2$), with $N$ the length of the video in frames, 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 less than 2 seconds for any given client buffer size in the range 64 KB – 16 MB. For the 174,000 frame *Star Wars* trace, optimal smoothing requires 6-8 seconds for client buffer sizes in this range.

($\mathcal{A}$ 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 using these upper and lower bounds which yields a feasible CBR segment over $[t_s, t_e + 1]$. With this approach, $t_e$ is never decremented, so backtracking (at Lines 9 and 13 in Figure 4) is

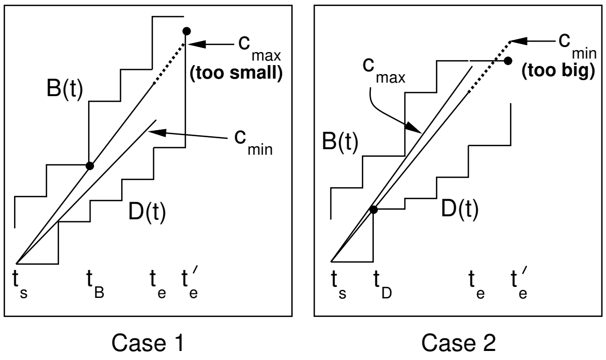



Figure 5: Operation of $\mathcal{A}$. In Case 1, the largest feasible constant rate over $[t_s, t_e]$, $c_{max}$, would result in *starvation* at $t'_e$. The CBR segment is ended at $t_B$, and a new segment established at higher rate. In Case 2, the smallest feasible constant rate over $[t_s, t_e]$, $c_{min}$, would result in *buffer overflow* at $t'_e$. The CBR segment is ended at $t_D$, and a new segment established at lower rate.

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 $t$); see [22] 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 *work-ahead constraint* (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.

## 2.2 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 [5, 9, 10, 21], who were kind enough to share them with us.

In Algorithm $\mathcal{A}$, a startup latency of $s$ frames can be introduced to smooth 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* [5], with 500 ms startup latency. Figure 6 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 smoothed 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.

[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.

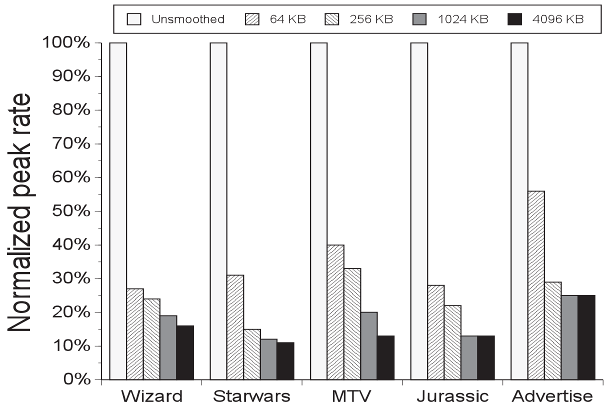



Figure 7: Reduction in peak rate enabled by optimal smoothing, with 500ms startup delay, across client buffer sizes ranging from 64-4096 KB.

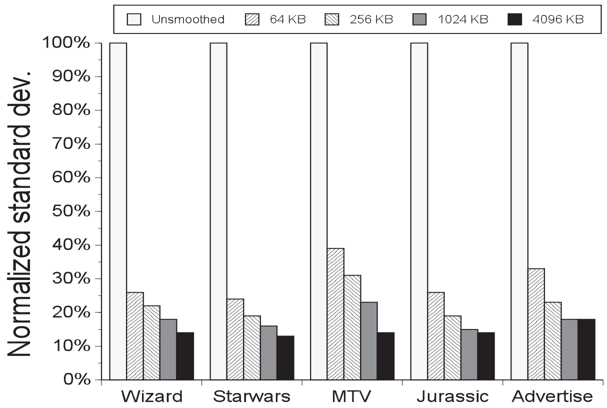



Figure 8: Reduction in standard deviation enabled by optimal smoothing (500ms startup delay).

Figures 7 and 8 present the reduction in peak rate and standard deviation across different videos,[2] in comparison to 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 for the given buffer sizes, since $\mathcal{A}$ is performing optimal smoothing.

## 2.3 Accommodating network jitter

We now address the question of how to accommodate network jitter. Let $\tau = \sigma + \nu$ denote the end-to-end network delay for a given packet, with $\sigma$ the sum of the fixed transmission and propagation delays, and $\nu$ the end-to-end queueing delay. Earlier, we assumed $\tau = 0$ to simplify presentation. If $\tau$ is nonzero but fixed, $S^*$ can be used directly. However, *network jitter*—variation in $\tau$ 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, $\nu_{max}$ would be specified at call admission; alternatively, $\nu_{max}$ might be determined via a measurement-

[2] All traces were VBR-encoded using an MPEG-1 software encoder. *The Wizard of Oz* trace [10] is 23 minutes in length and has mean rate 1.25 Mb/s. *MTV* and *Jurassic Park* [21] are each 28 minutes in length, and have mean rates of 0.737 Mb/s and 0.392 Mb/s, respectively. *Advertisements* [9] is 9 minutes in length and has mean rate 0.457 Mb/s.

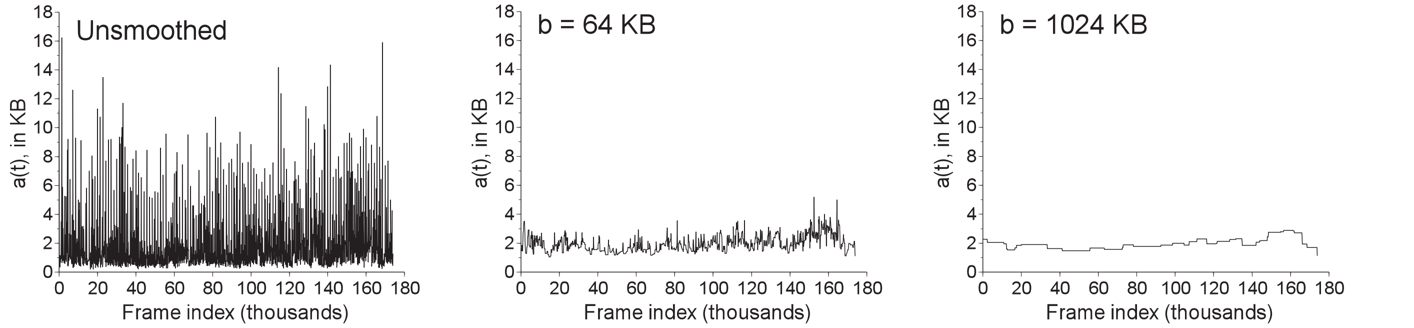



Figure 6: Optimal smoothing of a 2-hour MPEG-1 encoding of *Star Wars*, with 500 ms startup latency.

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 $\Delta$ 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 Deterministic Guaranteed network service, in Section 5.

## 3 Optimality proof

In this section we establish that $S^*$, the server transmission schedule generated by Algorithm $\mathcal{A}$, is optimally smooth. For a precisely-defined comparative measure of smoothness, we turn to the theory of *majorization* [14], 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 $n$-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. $X$ is said to be* 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 $Y \prec X$.

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 [14], 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 $\phi : \mathbf{R} \to \mathbf{R}$.*

[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 $[1, N']$.

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

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 show that $S^* \prec S$ for all feasible transmission schedules $S$.

**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 $S^* \prec S$.*

See [22] for proof. Theorem 1 establishes that, with respect to majorization, $S^*$ is optimally smooth. It follows that $S^*$ has minimum peak rate (by the definition of majorization) and minimum variance (by Result 1) among all feasible transmission schedules, and thus is optimally smooth according to these less general measures of variability as well. In addition, we can easily show that $S^*$ is unique; see [22] for details.

## 4 Impact on network resource requirements

We have established an optimal smoothing technique which, through work-ahead, 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? 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 [9], and Renegotiated CBR (RCBR) service, as formulated in [8]. Guaranteed and CBR service models have been widely studied, and are of particular interest for the transport of compressed video. The benefits of work-ahead 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 [9].
- RCBR is closely related to traditional CBR service, and employs no temporal multiplexing within the network. 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.

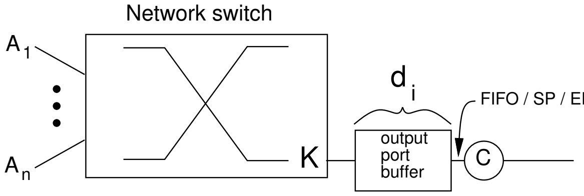



Figure 9: Guaranteed service at a single network switch, with $n$ streams destined for output port $K$. $A_i$ is the traffic constraint function for stream $i$, and $d_i$ is the negotiated bound on queueing delay at the output port.

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 restrict attention to a sampling of the representative behavior. For a performance comparison of Guaranteed and RCBR services under optimal smoothing, see [22].

## 5 Guaranteed service

In this section, we evaluate the benefit of optimal smoothing under Guaranteed service. 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 [9]. Consider a single network switch, as depicted in Figure 9. Suppose the $n$ incoming streams are routed to the $K^{th}$ output port and that 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, and 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* [9]). Computed at the frame level for a stored video with $N$ frames, $\mathcal{E}(t)$ is given by:

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

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 use $A_i(t) = \mathcal{E}_i(t)$; in [22], we also consider concave upper bounds to $\mathcal{E}_i(t)$, each specified by a set of "$(\sigma, \rho)$" pairs.

For FIFO, SP, and EDF link scheduling policies, exact admission control tests are known [3, 9]. 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 $d$:

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

$Q$ 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

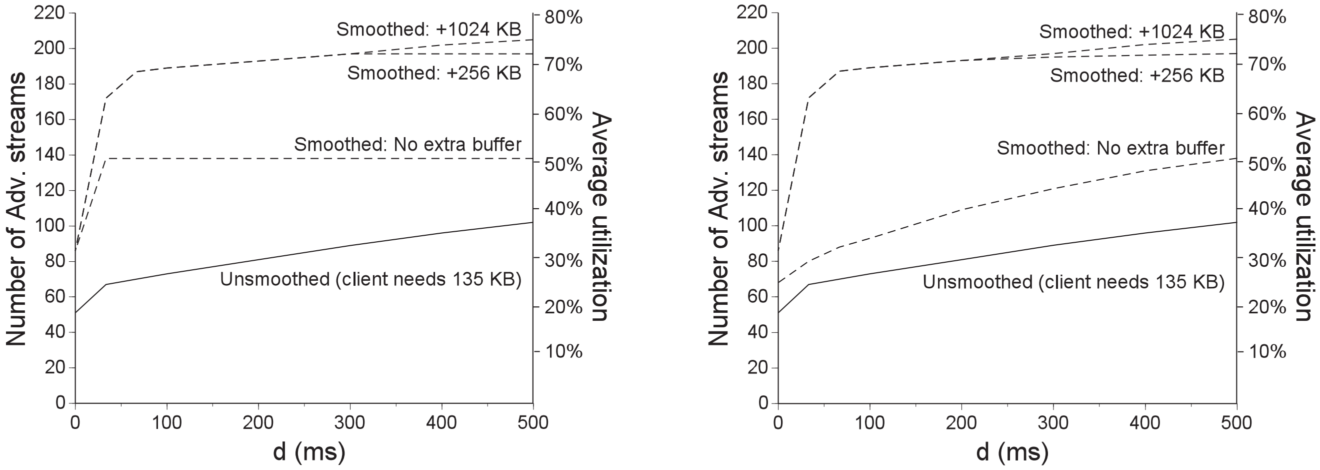
We evaluate the benefits of optimal smoothing under Guaranteed service in the following manner. First, for a given negotiated delay bound $d$, we derive a transmission schedule which accommodates the potential jitter of $d$. See discussion in Section 2.3. Second, to determine the maximum number of admissible streams for a given transmission 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 [13, 9] for other cases).

Consider the case of 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 [19]). 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 [9]).[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.

To achieve the performance indicated in Figure 10a, the server must either precompute and store 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 overconservative with respect to jitter work-ahead. But how much achievable utilization is sacrificed? Figure 10b provides the answer. Here, smoothed transmission schedules are computed with a 500 ms jitter work-ahead, 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

[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.




(a) $d$ ms jitter work-ahead

(b) 500 ms jitter work-ahead

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 work-ahead corresponding to the negotiated delay bound $d$. In (b), the server performs 500 ms jitter work-ahead for all values of $d$.

nominal requirement, the impact of the over-conservative jitter work-ahead is negligible.

Note the performance dip at low delay bounds. Under 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). In [22], we show that when a 100 ms (3-frame) startup latency is added, the low-delay anomaly disappears. The implication is that under optimal smoothing with sufficient client buffer space, Guaranteed service need not introduce delay in order to achieve high utilization.

Further performance results can be found in [22]. There, we also explore the impact of optimal smoothing for a practical traffic constraint function (the empirical envelope $\mathcal{E}$ is difficult to police [9]), finding that optimal smoothing enables simple, policable constraint functions to achieve near-comparable performance. In addition, we consider the case of heterogeneous streams with distinct delay bounds, as well as practical switch scheduling policies and call admission algorithms. Overall, the results show that under optimal smoothing, Guaranteed service can achieve high network utilization for bursty VBR video while utilizing efficient switch scheduling, call admission, and traffic policing mechanisms.

## 6 RCBR service

We now turn to an evaluation of optimal smoothing under RCBR service. We will consider the improvement in network utilization when individual streams are optimally smoothed, as a function of the number of renegotiations allowed over the length of the video. We begin with an overview of the service model.

### 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 [6]), 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.

RCBR adds a bandwidth renegotiation mechanism to static CBR [8, 7]. 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][6]; MPEG-2 contains support for limited scalable compression. Another possibility is to dynamically requantize the stream at a lower bit rate. See [8] 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/client a given reservation schedule [8]) of the 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, $\gamma$ 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) [8], 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

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

reservation schedules that can be carried over a fixed capacity link at a given QOS. For the impact at the client, we examine the cost 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 $R$. In [22], we present Algorithm $\mathcal{B}$ for identifying the minimum-cost reservation schedule for $S$ with $R$ or fewer renegotiations. The algorithm has time complexity O($Rn^2$), where $n$ is the number of *segments* in the transmission schedule. Since the complexity depends on the number of CBR segments, Algorithm $\mathcal{B}$ runs much faster on optimally-smoothed transmission schedules. For example, for 40,000-frame MPEG-1 traces, we have measured its computation time (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.

While we use $\mathcal{B}$ in the evaluation of optimal smoothing under RCBR throughout the remainder of this section, in [22] we discuss and present several extensions to the algorithm. First, the additional constraint of a minimum time between renegotiations is easily 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 [22]). Finally, note that $\mathcal{B}$ finds the minimum-cost reservation schedule for a *given* transmission schedule. In [22] 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.[7] 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 $C(N) < L$. (For additional discussion of the simulation, see [22].)

Figure 11a 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 an issue under CBR service [6], 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 11 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.

Is there incentive to send smoothed video, thus resulting in these high network utilizations? Consider Figure 11b, which shows the cost to the client of receiving the video. Here, we have normalized by the cost of sending an unsmoothed video without renegotiations. Optimal smoothing results in a remarkable drop in transmission cost. 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 $\phi(c_i) = c_i$).

From Figures 11a and 11b, 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 client, in terms of reducing the cost of receiving the video. The second is that when streams are optimally smoothed, relatively infrequent renegotiations (on the order of once a minute) yield high performance.

## 7 Related work

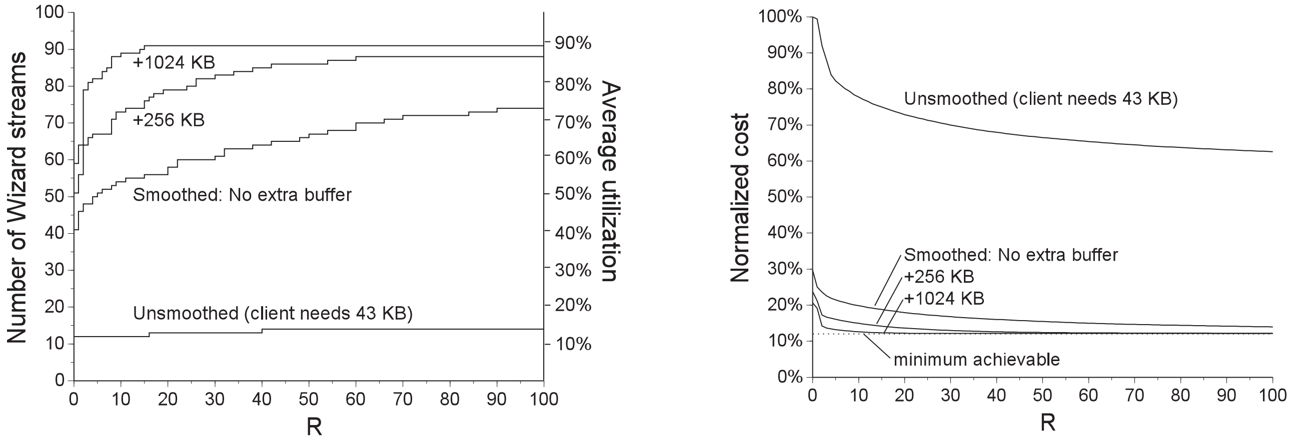
Reibman and Berger [19, 18] have conducted an extensive study of compressed video transport over ATM, for video generated online by delay-sensitive applications such as teleconferencing. One of their objectives is to evaluate smoothing techniques within the video encoder which reduce the stream's peak 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 work-ahead smoothing over the range of delays (0-4 frame times) which can be tolerated by the application. By contrast, 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 [19], 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 [12] in the context of the encoder and decoder buffer timing constraints [20] to derive a schedule which reduces the peak rate at which frames are drained from the encoder buffer.[8] 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 Euclidean distance sense, as in

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

[8] 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.




(a) Network impact

(b) Client impact

Figure 11: 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.

[12]), 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.[9] By appropriately defining $D(t)$ and $B(t)$ to reflect the encoder/decoder timing constraints,[10] 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 [19] 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., [11, 17, 23]). These approaches always involve the introduction of delay to enable smoothing; see [22] for discussion.

For stored video, Feng and Sechrest [4] have also considered work-ahead 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 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, minimizing the peak rate and burstiness of the stream for which resources are allocated.

McManus and Ross have also considered work-ahead smoothing of stored video [16, 15]. They do not consider the problem of optimally-reducing rate variability. In [16], 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 [15] 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.

In light of the multiple-time-scale variability of VBR video, there is a growing interest in renegotiated services (e.g., [8, 7, 24]). While our study of renegotiated CBR is based directly on the work of Grossglauser, Keshav and Tse [8, 7], 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; see [22] for discussion.

By contrast, under our formulation of RCBR, we consider only work-ahead smoothing of stored video, which can be performed without introducing delay in client playback. We present three optimal reservation algorithms: 1) Algorithm $\mathcal{B}$, which finds the minimum-cost reservation schedule for a given feasible transmission schedule, with 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 work-ahead, unlike the algorithm in [8].

## 8 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

[9] 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 $\phi$) 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.

[10] 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.

have assumed a certain availability of client buffer space for server work-ahead, 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 questions: 1) Given buffer size $b$ available for server work-ahead, 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 work-ahead, and introduces no delay into client playback. We provide a formal proof of optimality, based on the theory of majorization. As a simple consequence of the more general result, we show that $S^*$ has minimum variance and peak rate among all feasible schedules for the video.

The smoothing algorithm is efficient and easy to implement, and network jitter can be accommodated in a straightforward manner. We have demonstrated its 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 work-ahead 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, 9]) and Renegotiated CBR service [8]. The impact of work-ahead 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. The benefit is achievable for small client buffer sizes (e.g., when the video customer is equipped with a low-cost set-top box) and is independent of the 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 remain central for some time.

## Acknowledgements

We would like to thank the researchers who generously shared their MPEG traces. In particular, the contributions of Ed Knightly [9], Marwan Krunz [10], Mark Garrett [5] and Oliver Rose [21] are gratefully acknowledged.

## References

[1] C. S. Chang. "Stability, Queue Length, and Delay of Deterministic and Stochastic Queueing Networks". *IEEE Transactions on Automatic Control*, 39(5):913-931, May 1994.

[2] E. Chang and A. Zakhor. "Scalable Video Data Placement on Parallel Disk Arrays". *IS&T/SPIE Symposium on Electronic Imaging Science and Technology*, Feb. 1994.

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

[4] W. Feng and S. Sechrest. "Smoothing and Buffering for Delivery of Prerecorded Compressed Video". *IS&T/SPIE Multimedia Computing and Networking*, p. 234-232, Feb. 1995.

[5] M. Garrett and W. Willinger. "Analysis, Modeling and Generation of Self-Similar VBR Video Traffic". *Proc. ACM SIGCOMM*, p. 269-280, Aug. 1994.

[6] M. Grossglauser and S. Keshav. "On CBR Service", *Proc. IEEE INFOCOM*, Mar. 1996.

[7] M. Grossglauser, S. Keshav and D. Tse. "The Case Against Variable Bit Rate Services". *Proc.* $5^{th}$ *Workshop on Network and Operating Systems Support for Digital Audio and Video*, p. 307-310, Apr. 1995.

[8] M. Grossglauser, S. Keshav and D. Tse. "RCBR: A Simple and Efficient Service for Multiple Time-Scale Traffic". *Proc. ACM SIGCOMM*, p. 219-230, Aug. 1995.

[9] E. W. Knightly, D. E. Wrege, J. Liebeherr and H. Zhang. "Fundamental Limits and Tradeoffs of Providing Deterministic Guarantees to VBR Video Traffic". *Proc. ACM SIGMETRICS*, p. 98-107, May 1995.

[10] M. Krunz and H. Hughes. "A Traffic Model for MPEG-Coded VBR Streams". *Proc. ACM SIGMETRICS*, p. 47-55, May 1995.

[11] S. S. Lam, S. Chow and D. K. Y. Yau. "An Algorithm for Lossless Smoothing of MPEG Video". *Proc. ACM SIGCOMM*, Aug. 1994.

[12] D. T. Lee and F. P. Preparata. "Euclidean Shortest Path in the Presence of Rectilinear Barriers". *Networks*, 14:393-410, 1984.

[13] J. Liebeherr, D. Wrege and D. Ferrari. "Exact Admission Control in Networks with Bounded Delay Services", Tech. Report CS-94-29, University of Virginia, Jul. 1994.

[14] A. W. Marshall and I. Olkin. "Inequalities: Theory of Majorization and its Applications". New York, Academic Press, 1979.

[15] J. M. McManus and K. W. Ross. "Prerecorded VBR Sources in ATM Networks: Piecewise-Constant-Rate Transmission and Transport". *Submitted for publication*, Sep. 1995.

[16] J. M. McManus and K. W. Ross. "Video on Demand over ATM: Constant-Rate Transmission and Transport". *Proc. IEEE INFOCOM*, Mar. 1996.

[17] T. Ott, T. V. Lakshman and A. Tabatabai. "A Scheme for Smoothing Delay-Sensitive Traffic Offered to ATM Networks". *Proc. IEEE INFOCOM*, p. 776-785, May 1992.

[18] A. R. Reibman and A. W. Berger. "On VBR Video Teleconferencing over ATM Networks". *Proc. IEEE GLOBECOM*, p. 314-319, 1992.

[19] A. R. Reibman and A. W. Berger. "Traffic Descriptors for VBR Video Teleconferencing over ATM Networks". *IEEE/ACM Transactions on Networking*, 3(3):329-339, Jun. 1995.

[20] A. R. Reibman and B. G. Haskell. "Constraints on Variable Bit-Rate Video for ATM Networks". *IEEE Transactions on Circuits and Systems for Video Technology*, 2(4):361-372, Dec. 1992.

[21] O. Rose. "Statistical Properties of MPEG Video Traffic and their Impact on Traffic Modeling in ATM Systems". Tech. Report 101, Univ. Würzburg Institute of Computer Science, Feb. 1995.

[22] J. Salehi, Zhi-Li Zhang, J. Kurose and D. Towsley. "Supporting Stored Video: Reducing Rate Variability and End-to-End Resource Requirements through Optimal Smoothing". Tech. Report 95-98, Department of Computer Science, U. Massachusetts at Amherst, Nov. 1995 (http://www.cs.umass.edu/~salehi or ftp://gaia.cs.umass.edu/pub/Sale95:Supporting.ps.Z).

[23] N. Shroff and M. Schwartz. "Video Modeling within Networks using Deterministic Smoothing at the Source". *Proc. IEEE INFOCOM*, p. 342-349, 1994.

[24] H. Zhang and E. W. Knightly. "A New Approach to Support Delay-Sensitive VBR Video in Packet-Switched Networks". *Proc.* $5^{th}$ *Workshop on Network and Operating Systems Support for Digital Audio and Video*, p. 275–286, Apr. 1995.