Performance Evaluation of Smoothing Algorithms for Transmitting Prerecorded Variable-Bit-Rate Video

Wu-chi Feng *IEEE Member*, Jennifer Rexford *IEEE Member*

Abstract— **The transfer of prerecorded, compressed variable-bit-rate video requires multimedia services to support large fluctuations in bandwidth requirements on multiple time scales. Bandwidth smoothing techniques can reduce the burstiness of a variable-bit-rate stream by transmitting data at a series of fixed rates, simplifying the allocation of resources in video servers and the communication network. This paper compares the transmission schedules generated by the various smoothing algorithms, based on a collection of metrics that relate directly to the server, network, and client resources necessary for the transmission, transport, and playback of prerecorded video. Using MPEG-1 and MJPEG video data and a range of client buffer sizes, we investigate the interplay between the performance metrics and the smoothing algorithms. The results highlight the unique strengths and weaknesses of each bandwidth smoothing algorithm, as well as the characteristics of a diverse set of video clips.**

Keywords: **video-on-demand server, compressed video, bandwidth smoothing, video traces, traffic management**

I. Introduction

Many emerging multimedia applications, such as distance learning and entertainment services, rely on the efficient transfer of prerecorded video. Video-on-demand servers typically store video on large, fast disks [1], [2], [3]; the server may also include tertiary storage, such as tapes or optical jukeboxes, for holding less frequently requested data. A network connects the video servers to the client sites through one or more communication links. The network can help ensure the continuous delivery of the video data by including support for rate or delay guarantees [4], [5], based on resource reservation requests from the video server. Client sites include workstations and set-top boxes that have a playback buffer for storing video frames.

High-quality video requires a large amount of storage space and network bandwidth. Even effective compression techniques, such as MPEG [6] and motion-JPEG [7], still result in video streams with bandwidth requirements in the range of 2-10 megabits/second. Many video encoders generate constant-bit-rate (CBR) streams to simplify the allocation of disk, memory, and network resources. However, CBR-encoded video ultimately has variable quality, since the encoder is not permitted to increase the output bit rate during periods of action or detail, precisely when degradation in quality would be most noticeable to the viewer. Alternatively, video encoders can generate constant-quality video, resulting in a variable-bit-rate (VBR) stream. Constant-quality video typically has higher quality than a constant-bit-rate stream with the same average bandwidth [8], [9]. However, constant-quality video can exhibit significant burstiness on multiple time scales, due to the natural variations within and between scenes, as well as the frame structure of the encoding algorithm [10], [11], [12], [13], [14].

The efficient transfer of constant-quality video requires effective techniques for handling burstiness. Although the server, network, and client could conceivably allocate resources based on the peak bit rate of the stream, such over-provisioning is extremely wasteful and undermines the benefits of a constant-quality encoding. Alternatively, resources could be allocated based on certain assumptions about how a variable-bit-rate stream would multiplex with other traffic. Models based on statistical multiplexing, and particularly the theory of effective bandwidth, are useful for network provisioning, particularly on high-bandwidth links that carry a large number of traffic streams. However, statistical multiplexing does not offer deterministic guarantees and is less useful on lower-bandwidth links that multiplex a small or moderate number of streams. Despite rapid increases in backbone capacity in recent years, most broadband access networks cannot carry more than a handful of high-quality video streams at a time. Instead of relying on statistical multiplexing to handle burstiness, we believe that it is necessary to reduce the variability of individual video streams.

Prerecorded video offers a unique opportunity to reduce the variability of the network bandwidth requirements by transmitting frames to the client playback buffer in advance of each burst. Capitalizing on *a priori* knowledge of the number of bytes in each frame (the *frame size*), the server can precompute a transmission schedule that minimizes the bit rate while avoiding both underflow and overflow of the client buffer. This basic observation has been the underpinning of a class of *bandwidth smoothing* algorithms for stored video [15], [16], [17], [18], [19], [20], [21], [22]. Each of these algorithms can compute a transmission schedule for an N-frame video stream, given frame sizes f_i, $i = 1, 2, \ldots, N$, and a b-bit client buffer. Bandwidth smoothing offers substantial reductions in the peak and variability of bandwidth requirements for transmitting constant-quality video. These benefits come from removing short-term burstiness (e.g., at the MPEG group-of-pictures level), as well as the medium-term burstiness within and between scenes. The transmission of a smooth stream can

W. Feng is with the Comp. and Info. Sci. Dept. at Ohio State University. E-mail: wuchi@cis.ohio-state.edu. J. Rexford is with AT&T Labs Research. E-mail: jrex@research.att.com.

make efficient use of simple resource allocation models, such as constant-bit-rate or renegotiated constant-bit-rate services [17].

Reducing the peak transmission rate of the video stream is the primary goal of each of the smoothing algorithms. The algorithms, however, differ in what other performance metrics they consider. As a result, the various bandwidth smoothing algorithms generate transmission plans with different performance properties. The properties of the transmission schedules relate directly to the overhead of the transmission, transport, and playback of prerecorded constant-quality video. For example, this paper considers six smoothing algorithms that generate transmission schedules that (*i*) minimize the number of rate changes in transmission[16], (*ii*) minimize the variability of the bandwidth requirements[17], (*iii*) minimize the utilization of the client buffer[18], (*iv*) minimize the number of on-off segments in an on-off transmission model[23], (*v*) change transmission rates only at periodic intervals[19], or (*vi*) minimize general cost metrics through dynamic programming[20] subject to limiting the peak transmission rate in the stream and avoiding underflow and overflow of the b-bit client buffer.

The appropriate optimization criterion depends on the underlying resource allocation model at the server and client sites, as well as the network. In a realistic setting, however, more than one criterion may have impact on the resource requirements, particularly when the components in the system have different performance goals. For example, low buffer utilization may appeal to the clients, whereas low bandwidth variability may appeal to service providers that wish to multiplex as many streams as possible. As a result, it becomes important to understand how well each smoothing algorithm performs across the range of possible optimization criteria. An algorithm that has near-optimal performance according to several metrics may be preferable to an algorithm that is optimal for one metric and performs poorly for all the others. In this paper, we present a systematic comparison of bandwidth smoothing algorithms across a range of optimization criteria and video traces, with the goal of quantifying how well each algorithm performs according to each metric. Experimenting with a diverse set of video traces enables us to draw sound conclusions about the performance of the various algorithms and the effectiveness of bandwidth smoothing.

The performance comparison draws on our library of twenty full-length, constant-quality video clips[1]. These traces were generated using a PC-based, motion-JPEG video capture testbed, as described in [24], [25]. By studying a range of different video streams (including a range of educational videos, action movies, and animated films, as well as clips encoded with different quantizers settings), we can determine how much each type of video stream can benefit from each of the smoothing algorithms. For completeness, we also consider a number of publicly available MPEG-1 traces [14]. Throughout the paper, we show results for all of the traces to highlight both the general trends and the variation in the results for different video clips and different encoding schemes. Our detailed evaluation of bandwidth smoothing algorithms on a diverse set of video traces complements recent survey papers that focus more broadly, and in less detail, on the variety of techniques available for handling variable-bit-rate video [26], [27].

Section II surveys the six bandwidth smoothing algorithms, with an emphasis on the metrics they optimize as well as their computational complexity. Drawing on the video traces, Section III compares the smoothing algorithms and investigates the interaction between four key performance metrics: (i) peak bandwidth requirement, (ii) variability of transmission rates, (iii) number of rate changes, and (iv) client buffer utilization that relate directly to the server, network, and client resources required for transmitting the smoothed video stream. In addition to evaluating the bandwidth smoothing algorithms, these experiments also highlight unique properties of the underlying video clips. These results motivate several possible directions for future research on the efficient transmission of prerecorded variable-bit-rate video, as discussed in Section IV.

II. Bandwidth Smoothing Algorithms

A multimedia server can substantially reduce the rate requirements for transmitting prerecorded video by transmitting frames into the client playback buffer in advance of each burst. A class of bandwidth smoothing algorithms capitalizes on *a priori* knowledge of the prerecorded stream to compute a server transmission schedule, based on the size of the playback buffer.

A. Overflow and Underflow Constraints

A compressed video stream consists of n frames, where frame i requires f_i bytes of storage. Without loss of generality, we assume that time is measured in units of frame slots. To permit continuous playback at the client site, the server must always transmit enough data to avoid buffer underflow, where

$$L(k) = \sum_{i=0}^{k} f_i$$

indicates the amount of data consumed at the client by frame k, where $k = 0, 1, \ldots, n-1$. Similarly, the client should not receive more data than

$$U(k) = L(k) + b$$

by frame k, to prevent overflow of the playback buffer (of size b) [2]. Consequently, any valid server transmission plan should stay within the area enclosed by these vertically equidistant functions, as shown in Figure 1(a). That is,

$$L(k) \le \sum_{i=0}^{k} c_i \le U(k)$$

where c_i is the transmission rate during frame slot i of the smoothed video stream.

[1] http://www.cis.ohio-state.edu/∼wuchi/Video/index.html.

[2] This definition of the overflow constraint $U(k)$ assumes that the client removes frame k from the playback buffer into a separate decode buffer at time k. If the client has a single shared buffer, the underflow constraint is $U(k) = L(k-1) + b$.

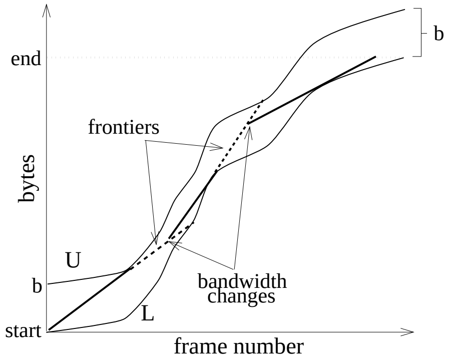


Fig. 1. **Computing Transmission Plans:** This figure shows the buffer underflow and overflow curves for a sample video stream. The resulting transmission plan consists of three constant-bit-rate runs that serve as a server schedule for transmitting video frames.

A transmission schedule consists of a sequence of m linear segments, each with a constant bandwidth allocation r_j, $j = 1, 2, \ldots, m$, where time is measured in discrete frame slots. At time i the server transmits at rate $c_i = r_j$, where slot i occurs during run j. Together, the m bandwidth runs must form a monotonically-nondecreasing, piecewise-linear path that stays between the $L(k)$ and $U(k)$ curves. For example, Figure 1 shows a plan with $m = 3$ runs, where the second run increases the transmission rate to avoid buffer underflow at the client playback buffer; similarly, the third run decreases the rate to prevent overflow. Bandwidth smoothing algorithms typically select the starting point for run $j + 1$ based on the trajectory for run j. By extending the fixed-rate line for run j, the trajectory eventually encounters either the underflow or the overflow curve, or both, requiring a change in the server transmission rate.

B. Selecting Long Trajectories

Several different smoothing algorithms have been introduced that use both the $L(k)$ and $U(k)$ curves in computing the bandwidth runs in the transmission plans, based on the size of the client playback buffer.

Given a starting point for run $j+1$, these algorithms select a trajectory that extends as far as possible to limit the number of bandwidth changes. As a result, the trajectory for each run must eventually reach both the overflow and the underflow curves, generating a *frontier* of possible starting points for the next run, as shown in Figure 1. The various bandwidth smoothing algorithms differ in how they select a starting point for run $j+1$ on rate increases and decreases, resulting in transmission plans with different performance properties:

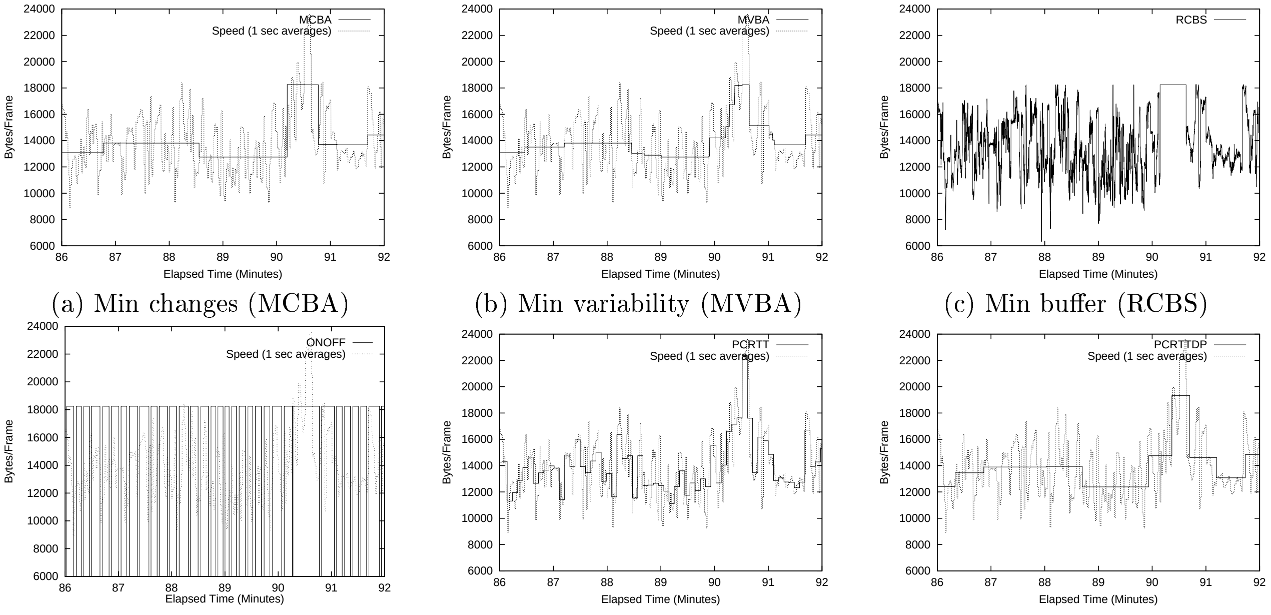
- **MCBA:** To minimize the number of rate decreases, the *minimum changes bandwidth allocation* (MCBA) algorithm [16] performs a search operation on the frontier of each rate change for a starting point for the next run. This results in a transmission plan with the smallest possible number of rate changes (minimizes m), as well as the minimum peak bandwidth requirement. When implemented with a binary search, the MCBA algorithm has a worst-case complexity of $O(n^2 \log n)$, where the $\log n$ term arises from performing a binary search along the frontier of each run; on average, the algorithms run in $O(n \log n)$ time. An alternate implementation, based on an algorithm from the robotics literature, has $O(n)$ complexity [21]. A sample schedule is shown in Figure 2(a).
- **MVBA:** Instead of minimizing the number of rate changes m, the *minimum variability bandwidth allocation* (MVBA) algorithm minimizes the variance in the rate requirements [17]. MVBA initiates bandwidth changes at the leftmost point along the frontier, for both rate increases and rate decreases. As a result, an MVBA transmission plan *gradually* alters the stream's rate requirement, sometimes at the expense of a larger number of small bandwidth changes. By avoiding a binary search along the frontier, the MVBA algorithm can have a worst-case complexity of $O(n^2)$. An alternate implementation of the algorithm can be derived that has $O(n)$ worst-case complexity [28]. A sample schedule in shown in Figure 2(b).

For a given client buffer size, the MCBA and MVBA bandwidth smoothing algorithms result in transmission plans that minimize the peak bandwidth and maximize the minimum bandwidth. Still, these algorithms differ in terms of rate variability, the frequency of rate changes, and client buffer utilization, as discussed in Section III.

C. Smoothing at the Peak Rate

In addition to generating transmission plans, the MCBA and MVBA algorithms provide an efficient way to compute the minimum achievable peak transmission rate. The next two algorithms use this rate to generate schedules with different performance properties:

- **RCBS:** Given a maximum bandwidth constraint r, the *rate-constrained bandwidth smoothing* (RCBS) algorithm generates a schedule with the smallest buffer utilization by transmitting frames as late as possible, subject to the rate constraint [18], [29]. Given the rate r, this algorithm minimizes the maximum buffer size required for the particular rate. This $O(n)$ algorithm starts with the last frame of the movie and sequences backwards toward the first frame. Any frame that exceeds the rate constraint is modified to the maximum rate constraint and then transmitted earlier. As shown in Figure 2(c), the RCBS plan follows the actual data rate for the movie rather closely, particularly for small buffer sizes.
- **ON-OFF:** Given a maximum bandwidth constraint r, the *on-off* algorithm generates a schedule that alternates between transmitting at the peak rate ("on" period) and not transmitting at all ("off" period), subject to the rate constraint [23]. The $O(n)$ on-off algorithm minimizes the number of on-off segments, subject to the rate constraint and the client buffer size, and results in transmission rates that fluctuate across time, as shown in Figure 2(d). Depending on the scheduling model at the server, and the availability of traffic shaping hardware, the on-off schedules may be easier to implement than the other transmission plans.



(a) Min changes (MCBA) (b) Min variability (MVBA) (c) Min buffer (RCBS)

(d) Min on-off (ON-OFF) (e) Periodic changes (PCRTT) (f) Dynamic prog. (PCRTT-DP)

Fig. 2. **Bandwidth Plans:** These graphs show the transmission plans generated by four different bandwidth smoothing algorithms, applied to the movie *Speed* and a 1 megabyte client playback buffer. For the PCRTT algorithm, the graph shows the plan with the largest possible interval size that would not overflow a 1 megabyte buffer.

Algorithm	**Optimization**	**Complexity**
MCBA [16], [21]	Minimum rate changes	$O(n)$
MVBA [17]	Minimum rate variability	$O(n)$
RCBS [18]	Minimum client buffer usage	$O(n)$
ON-OFF [23]	Minimum number of on/off periods	$O(n)$
PCRTT [19]	Maximum spacing of rate changes	$O(n)$
PCRTT-DP [20]	General optimization model	$O(n^3)$

TABLE I

Bandwidth Smoothing Algorithms

D. Periodic Time Intervals

Given the different starting points on the *frontiers*, the MCBA and MVBA algorithms select trajectories that extend as far as possible before reaching both the $L(k)$ and $U(k)$ curves. Other smoothing algorithms focus on the $L(k)$ curve in constructing a schedule; if necessary, these algorithms can iterate to compute a schedule that also satisfies the buffer constraint b for the $U(k)$ curve:

- **PCRTT:** In contrast to the four previous algorithms, the *piecewise constant rate transmission and transport* (PCRTT) algorithm [19] creates bandwidth allocation plans by dividing the video stream into fixed-size intervals. This $O(n)$ algorithm generates a single run for each interval by connecting the intersection points on the $L(k)$ curve, as shown in Figure 3; the slopes of these lines correspond to the rates r_j in the resulting transmission plan. To avoid buffer underflow, the PCRTT scheme vertically offsets this plan until all of the runs lie above the $L(k)$ curve. Raising the plan corresponds to introducing an initial playback delay at the client site; the resulting transmission curve also determines the minimum acceptable buffer size to avoid overflow given the interval size, as shown in Figure 3. The algorithm results in periodic rate changes, as shown by the example in Figure 2(e).
- **PCRTT-DP:** Instead of requiring a rate change for each time interval, an extension to the PCRTT algorithm employs dynamic programming (DP) to calculate a minimum-cost transmission plan that consists of m runs [20]. Although dynamic programming offers a general framework for optimization, we focus on the buffer size b as the cost metric to facilitate comparison with the other smoothing algorithms. The algorithm iteratively computes the minimum-cost schedule with k runs by adding a single rate changes to the best schedule with $k-1$ rate changes. However, an exact solution, permitting rate changes in any time slot, would introduce significant computational complexity, particularly for full-length video traces. To reduce the computational overhead, a heuristic version of the algorithm [20] groups frames into intervals, as in Figure 3, when computing each candidate schedule; then, the full frame-level information is used to determine how far to raise the schedule to avoid buffer underflow. This algorithm has a computational complexity of $O(n^3)$.

As shown in Figure 2(f), the resulting PCRTT-DP al-

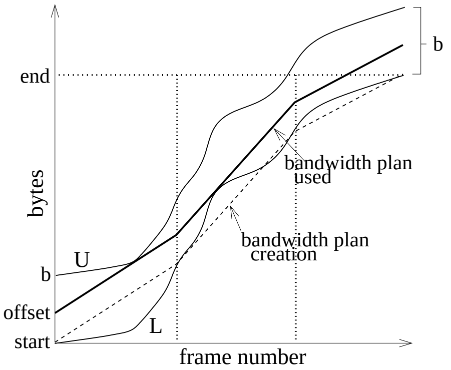


Fig. 3. **PCRTT Plan Creation:** This figure shows the creation of a PCRTT bandwidth allocation plan. First, the algorithm calculates the average frame size for each interval (dashed line). Then, the algorithm raises the plan to avoid buffer underflow. Based on the offset plan, the minimum buffer requirement is the maximum distance above the underflow curve.

gorithm, using a group size of 60 frames, produces bandwidth plans that are somewhat similar to MCBA plans, since both try to limit the number of rate changes. In contrast, the original PCRTT algorithm produces a schedule with a larger number of short runs, since the algorithm uses a single time interval throughout the video; in this example, a small interval size is necessary to avoid overflow of the client buffer. The RCBS plan changes the transmission rate in almost every time unit, except when large frames introduce smoothing at the peak rate. The next section compares the smoothing algorithms across a range of client buffer sizes, video clips, and performance metrics to evaluate these trade-offs in transmitting prerecorded, variable-bit-rate video. The properties of the various smoothing algorithms are summarized in Table I.

III. Performance Evaluation

Using the traces from the video library, this section compares bandwidth smoothing algorithms based on a collection of performance metrics. These metrics include the peak rate requirements, the variability of the bandwidth allocations, the number of bandwidth changes, and the utilization of the playback buffer. By applying these metrics to server transmission plans, across a wide range of realistic client buffer sizes, the simulation experiments show cost-performance trends that affect the transmission, transport, and playback of compressed video.

A. Experimental Set-Up

Since some of the algorithms implicitly introduce playback delay, we permit each algorithm to use the same playback delay to transmit data in advance for the first bandwidth run. For our experiments, we have allowed a maximum prefetch time (before playback begins) of 900 frames or 30 seconds. We note that depending on the movie, the size of the buffer, and the smoothing algorithms that the prefetch delay was chosen to optimize the metrics for which they were created. Except for PCRTT, most of the algorithms do not require such as large playback delay; typically a few frames, or at most a few seconds, of start-up delay are sufficient to remove the burstiness at the beginning of a video.

For the PCRTT and PCRTT-DP algorithms, which determine the buffer size as a byproduct of computing the bandwidth plan, we vary the window size (for PCRTT) and the number of rate changes (for PCRTT-DP) to generate a collection of plans, each with a corresponding buffer size. The fixed window size in the PCRTT algorithm can result in fluctuations in the performance metrics as the buffer size increases, since a smaller window size can sometimes result in a larger buffer requirement. The PCRTT-DP heuristic computes bandwidth plans based on groups of 60 for the M-JPEG streams (due to their substantially large number of frames) and 12 frames for the MPEG streams (to match the Group of Pictures pattern) to reduce the computation time; sample experiments with smaller group sizes resulted in similar values for the performance metrics. However, the frame grouping does limit the ability of the algorithm to compute bandwidth plans for small buffer sizes; for small buffer sizes, a more exact (and computationally expensive) version of the PCRTT-DP heuristic should produce statistics that resemble the MCBA results, since both algorithms compute transmission plans that limit the number of rate changes. The frame-grouping and rate-change parameters both limit the algorithm's ability to compute valid plans for small buffer sizes, since smoothing into a small buffer requires bandwidth changes on a very small time scale.

For a typical two-hour video ($n = 216,000$ frames), the MCBA, MVBA, RCBS, and PCRTT algorithms require a few seconds of computation time on a modern workstation. The RCBS algorithm generally executes in the smallest amount of time (after determining the rate constraint), followed by the ON-OFF, PCRTT, MVBA, and MCBA algorithms (in that order); exact comparisons of computational overhead are difficult, since they would depend on the platform and the details of each implementation. The PCRTT-DP algorithm, using a group size of 60 frames for M-JPEG streams and allowing up to 3000 rate changes, requires about an hour to execute. Similarly, for the MPEG streams the PCRTT-DP algorithm uses group sizes of 12 and require about an hour to execute as well. Because the PCRTT-DP algorithm start with the number of rate changes $K = 1$ and iteratively calculates the minimal cost of each successive bandwidth change, calculating a plan that has 1000 bandwidth changes requires the calculation of all plans with fewer bandwidth changes. To speed this algorithm up, we calculated all the costs (in terms of the buffer size) for each sequence of frames (i, j), $0 < i < j \leq N$. This reduces the computational complexity of each bandwidth change to $O(n^2)$.

B. Peak Bandwidth Requirement

The peak rate of a smoothed video stream determines the worst-case bandwidth requirement across the path from the video storage on the server, the route through the network, and the playback buffer at the client site. Hence, most bandwidth smoothing algorithms attempt to minimize

$$\max_j \{r_j\}$$

to increase the likelihood that the server, network, and the client have sufficient resources to handle the stream. This is especially important if the service must reserve network bandwidth based on the peak rate, or if the client has a low-bandwidth connection to the network. In addition, reducing the maximum bandwidth requirement permits the server and the network to provide deterministic guarantees to a larger number of streams.

Figure 4 plots the peak rate $\max\{r_j\}$ as a function of the client buffer size for each of the motion-JPEG and MPEG clips. The graphs plot the minimum peak rate achieved by the MCBA, MVBA, RCBS, and ON-OFF algorithms. For each video, the peak rate decreases as the buffer size increases, with diminishing returns for larger buffer sizes. The same basic trends hold for each of the motion-JPEG and MPEG clips, as shown in Figure 4(a) and Figure 4(b), respectively, since motion-JPEG and MPEG encodings should have similar variability in frame sizes on the medium time scale. However, the MPEG clips typically show a more dramatic reduction in the peak rate for very small buffer sizes, since a small buffer allows the server to smooth over variations in frame sizes within an MPEG group-of-pictures. Also, the MPEG clips flatten sooner under large buffer sizes, due to the smaller average frame sizes and the shorter video lengths of the MPEG clips.

The graphs in Figure 4 also show some variation between the different video clips. Under small buffer sizes, the movies with the largest variations in frame sizes also tend to have the largest peak bandwidth requirements, due to the limited ability to smooth large peaks. In Figure 4(a), the *Beauty and the Beast*, *E.T. (high quality)*, and *NCAA Final Four* videos are the top three curves, while the *Seminar* videos have the lowest peak bandwidth requirements for buffer sizes less than 1 megabyte. For the three *E.T.* videos at different quality levels, the lower-quality encodings have lower peak rate requirements, due to the smaller frame sizes at each point in the video. In fact, under larger buffer sizes, the *E.T. (low quality)* video actually has a *lower* peak bandwidth than the *Seminar* videos. For large client buffers, smoothing removes nearly all of the burstiness in the stream, yielding a plan that stays very close to the mean frame size of 6305 bytes; the three *Seminar* videos, digitized with a quality factor of 90, have larger average frame sizes (8604, 8835, and 9426 bytes). Thus, for small buffer sizes, the peak bandwidth requirement is generally driven by the *maximum* frame size, while for larger buffer sizes, the peak rate is driven mostly by the *average* frame size.

While the MCBA, MVBA, RCBS, and ON-OFF algorithms minimize the peak bandwidth, PCRTT and PCRTT-DP transmission plans typically do not have substantially larger maximum bandwidth requirements. Figure 5 plots the peak bandwidth as a function of the client buffer size for two of the video clips. These results are representative of the performance of the PCRTT and PCRTT-DP algorithms on the other video clips. In general, the PCRTT algorithm is limited by its interval size, since it does not have full flexibility to smooth across intervals. Hence, the PCRTT algorithm has the most difficulty when a video clip has areas of sustained large frames followed by areas of small frames (or vice-versa). These regions require small interval sizes to avoid overflow and underflow of the client buffer. These small interval sizes, in turn, limit the algorithm's ability to smooth across larger time periods, resulting in a higher peak rate.

The PCRTT-DP plans often have smaller peak bandwidth requirements than the PCRTT algorithm, and are similar to the other algorithms. In fact, an exact version of PCRTT-DP algorithm, using a group size of 1 frame, would generate transmission plans that minimize the peak bandwidth. However, the grouping of frames can sometimes inflate the peak rate when a sequence of large frames fall within a single group.

C. Variability in Bandwidth Requirements

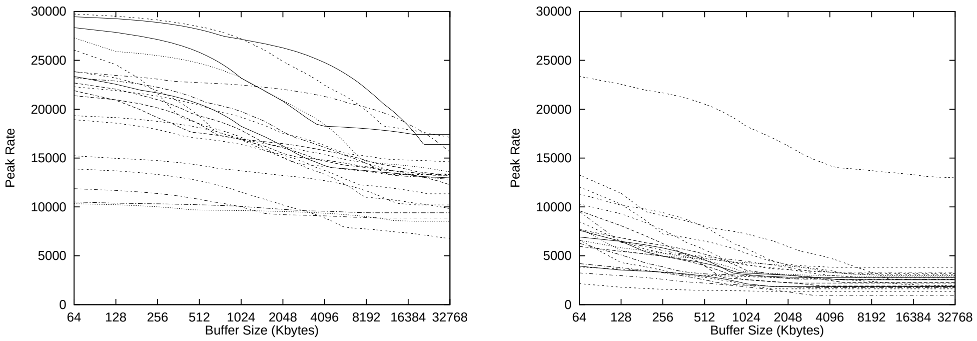
In addition to minimizing the peak bandwidth, a smoothing algorithm should reduce the overall *variability* in the rate requirements for the video stream [17]. Intuitively, plans with smaller rate variation should require fewer resources from the server and the network; more precisely, smoother plans have lower *effective bandwidth* requirements, allowing the server and the network to statistically multiplex the maximum number of streams [30]. Even under a deterministic model of resource reservation, the server's ability to change a stream's bandwidth reservation may depend on the *size* of the adjustment ($|r_{j+1} - r_j|$), particularly on rate *increases*. If the system does not support advance booking of resources, the server or the network may be unable to acquire enough bandwidth to start transmitting frames at the higher rate[3]. Since the video clips have different average rate requirements, Figure 6 plots the *coefficient of variation*

$$\frac{\mathrm{stdev}\{c_0, c_1, \ldots, c_{n-1}\}}{\frac{1}{n}\sum_{i=0}^{n-1} c_i}$$

to normalize the variability metric across the different streams, where the server transmits at rate c_i at time i. Although the curves have a similar shape to the peak-rate graphs in the previous subsection, the variability metric continues to decrease over a wider range of buffer sizes. In general, the bandwidth variability metric is more affected by the variation in frame sizes across the entire clip, rather than the scene (or set of scenes) with the largest bandwidth requirements.

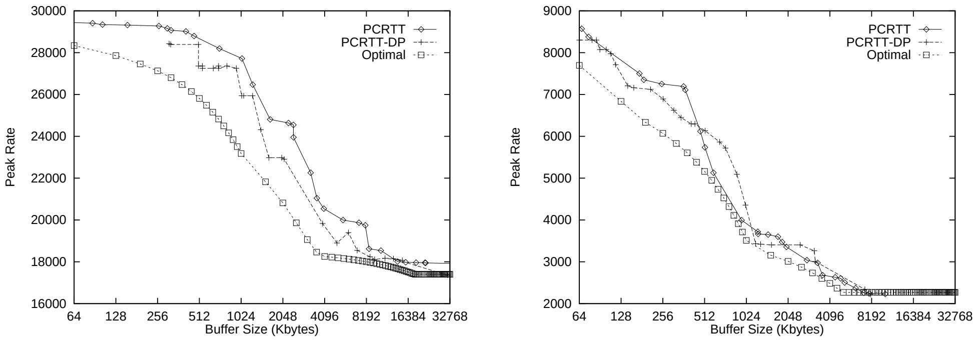
For each of the video clips, the coefficient of variation decreases as a function of the buffer size. In general, the motion-JPEG and MPEG clips have similar performance

[3] If the system cannot reserve resources for the higher bandwidth r_{j+1}, the video stream may have to adapt to a smaller rate to avoid terminating the remainder of the transfer. For example, with a *layered* encoding of the video stream, the server could reduce the transmission rate by sending only the higher priority components of the stream [31], [32]. To limit the degradation in video quality at the client site, the server can raise the stream's rate as close to r_{j+1} as possible.



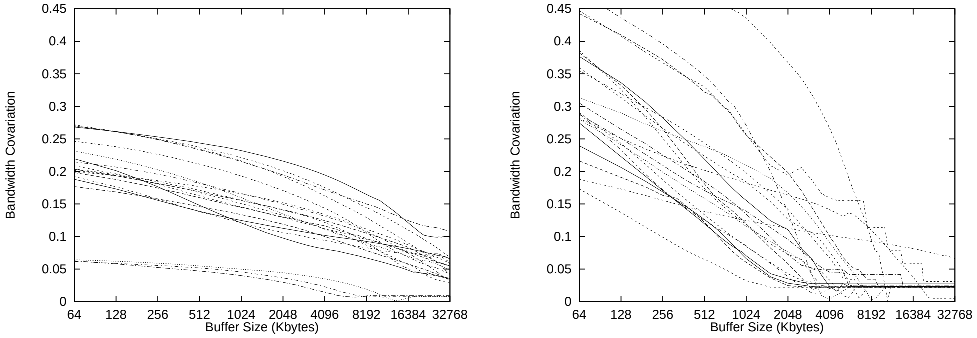
(a) Motion-JPEG clips (b) MPEG clips

Fig. 4. **Minimum Peak Bandwidth Requirement:** These graphs plot the peak bandwidth as a function of the client buffer size for each of the motion-JPEG and MPEG video clips. The graphs plot the peak rate achieved by the MCBA, MVBA, RCBS, and ON-OFF algorithms, which minimize this metric.



(a) *Final Four* (M-JPEG) *Mr. Bean* (MPEG)

Fig. 5. **Peak Bandwidth Requirement Across All Algorithms:** These graphs plot the peak bandwidth as a function of the client buffer size for each of the bandwidth smoothing algorithms for two video clips.



(a) Motion-JPEG clips (b) MPEG clips

Fig. 6. **Minimum Bandwidth Variability:** These graphs plot the coefficient of variation (standard deviation normalized by the mean) of the bandwidth requirements as a function of the client buffer size for the MVBA algorithm, which minimizes this metric.

trends, though the MPEG clips have higher variability under small buffer sizes, due to the differences in frame sizes within a group-of-pictures. In Figure 4(a), the *Beauty and the Beast*, *E.T. (quality 75)*, and *E.T. (quality 90)* videos exhibit the most bandwidth variability. Interestingly, the *E.T.* streams with *lower* quality factors have greater variability in the bandwidth requirements. Although a coarser encoding reduces the *average* frame size, some frame lengths decrease more than others, depending on the scale of detail in the scene.

While the MVBA plans consistently have the smallest variability in bandwidth requirements, the MCBA plans typically have similar performance, as shown in Figure 7. In fact, the coefficient of variation for the MCBA plans is rarely more than 5% higher than the corresponding MVBA plans, across the range of buffer sizes and collection of video clips in Figure 6. The videos that show a higher difference in bandwidth variability also show a higher difference in the number of bandwidth changes in the MCBA and MVBA plans. This suggests that the higher variability in the MCBA plans stems from its attempts to combine multiple bandwidth changes into a single transmission rate, rather than making gradual adjustments. Still, both algorithms achieve low variability across a wide range of video clips, while minimizing the peak rate and maximizing the minimum transmission rate.

In contrast, the RCBS plans have higher rate variability, particularly under larger buffer sizes, since the algorithm only transmits data early when it is necessary to avoid exceeding the peak rate later in the schedule. As a result, an RCBS plan often transmits small frames at a low rate, resulting in a much lower minimum bandwidth than the MVBA and MCBA algorithms. Hence, the increase in rate variability under the RCBS algorithm actually stems from the *small* transmission rates. The PCRTT and PCRTT-DP schedules have larger variability in bandwidth allocations. Because the PCRTT algorithm smooths bandwidth requests based on fixed interval lengths, it cannot smooth burst of large frames beyond the size of the interval, resulting in higher peaks and lower valleys. Under larger buffer sizes, the partitioning of the frames into fixed intervals plays a large role in determining the minimum amount of buffering required to have continuous playback of the video. Finally, the ON-OFF schedules have very high variability, since the transmission rates alternate between zero and the peak rate.

D. Number of Bandwidth Changes

In addition to reducing the variability in the resource requirements, bandwidth smoothing also decreases the frequency of rate changes. Decreasing the number of rate changes reduces the cost of negotiating with the network [13] to reserve link bandwidth for transporting the stream[4]. In addition, reducing the number of rate changes may also reduce complexity at the server, which must retrieve data from disk for each stream based on its scheduled rate. As a minor point, reducing the number of rate changes also decreases the size of the transmission schedule, though this is typically small in comparison to the size of the actual video frames. Figure 8 plots the minimum frequency of bandwidth changes, achieved by the MCBA algorithm. Since the video clips have different durations, the graphs plot the *frequency* of bandwidth changes

$$\frac{m}{n}$$

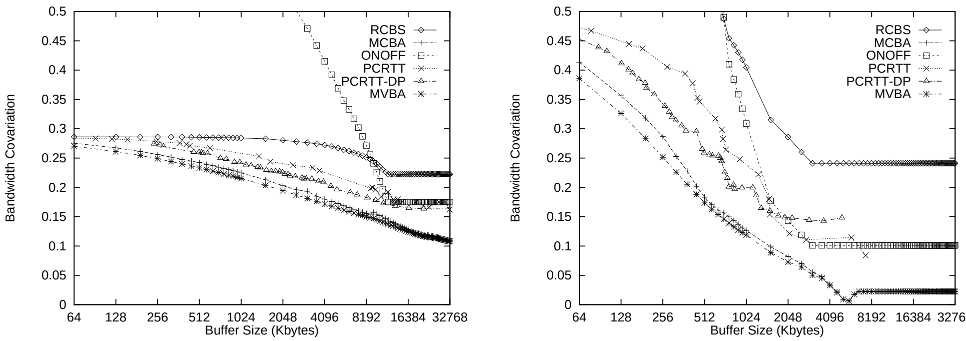
in changes per minute across a range of client buffer sizes, where m is the number of piecewise-linear segments in the smooth transmission schedule and n is the number of frames in the video clip.

For all of the smoothing algorithms and video traces, the client playback buffer is effective in reducing the frequency of rate change operations. In Figure 8(a), the bottom three curves correspond to the *Seminar* videos, which do not require many rate changes due to their small frame sizes and the low variability in their bandwidth requirements. The *NCAA Final Four* video requires the highest rate of bandwidth changes, due to the large frame sizes and long-term variations in scene content. For a 64 kilobyte buffer, this stream requires an average of 1.8 rate changes per minute under the MCBA plan. In contrast, the corresponding MVBA plan requires 8.5 rate changes per minute. In general, MCBA requires much fewer rate changes than the other algorithms, as shown in Figure 9. For some movies and buffer sizes, the MVBA plans have up to 14 times as many bandwidth changes as the corresponding MCBA plans. This occurs because the MVBA algorithm introduces a larger number of small rate changes to minimize the variability of bandwidth requirements in the server transmission plan.

As an example, we compare the MCBA and MVBA algorithms on the 23-second video trace shown in Figure 10(a). For a 128 kilobyte buffer, the MVBA algorithm introduces 104 rate changes (55 increases and 49 decreases), while the MCBA plan has just three bandwidth changes, as shown in Figure 10(b). During the first 400 frames of the video segment, the frame sizes gradually increase over time. On this long stretch of increasing bandwidth requirements, the MVBA algorithm tends to follow the "curve" of the increase by generating a sequence of small rate increases. A similar effect occurs during the gradual decreases in frame sizes for the remainder video segment. In Figure 10(b), note that the area between the two plans, in the range of frames 12720 to 12900, is approximately equal to the size of the smoothing buffer. This suggests that the MVBA plan has filled the client buffer, requiring a more gradual response to the rate increases in the video segment. In contrast, the MCBA plan has a nearly empty buffer, giving the algorithm greater latitude in adjusting the server transmission rate; referring to Figure 1, this is a case where the MCBA algorithm selects a starting point at the *rightmost* point along the frontier whereas the MVBA algorithm selects the *leftmost* point.

Although the MVBA plans often have fewer rate changes than the corresponding PCRTT plans, the PCRTT algorithm sometimes generates fewer rate changes under moderate buffer sizes. For these buffer sizes, the PCRTT algorithm is effective at combining several bandwidth runs of the MVBA algorithm into a single rate interval. For example, in Figure 10(c), the PCRTT algorithm generates only 31 rate changes, in contrast to the 104 changes in the corresponding MVBA plan. The PCRTT-DP algorithm produces bandwidth allocation plans that are very similar to the MCBA algorithm, since they both strive to minimize the number of rate changes; however, under smaller buffer sizes, the PCRTT-DP heuristic generate more bandwidth changes due to the frame-grouping factor. In contrast to the PCRTT algorithms, the RCBS plans tend to follow the sizes of the individual frames for most of the stream, except when some workahead transmission is necessary to avoid increasing the peak rate for transmitting the video. With a small client buffer, the RCBS algorithm requires nearly 1800 rate changes per minute (i.e., one per frame!). Although the number of rate changes decreases as the buffer size grows, the RCBS algorithm still generates significantly

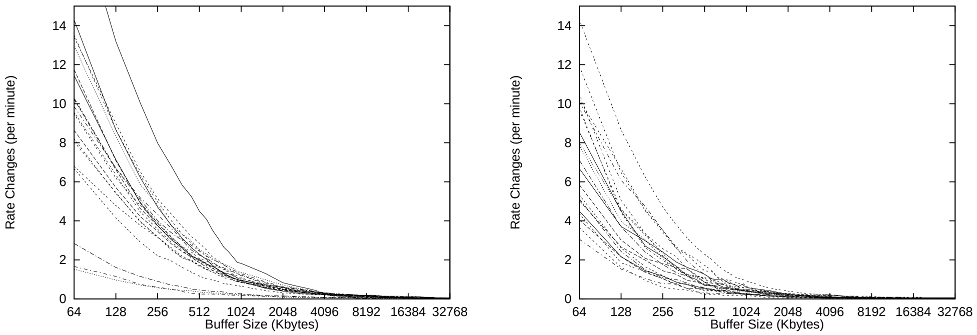
[4] To further reduce interaction with the network, each video stream could have a separate *reservation plan* for allocating network resources along the route to the client. This reservation plan could have fewer rate changes than the underlying transmission plan, at the expense of reserving excess link bandwidth [13], [17].



(a) *Jurassic Park* (M-JPEG)

(b) *Soccer 2* (MPEG)

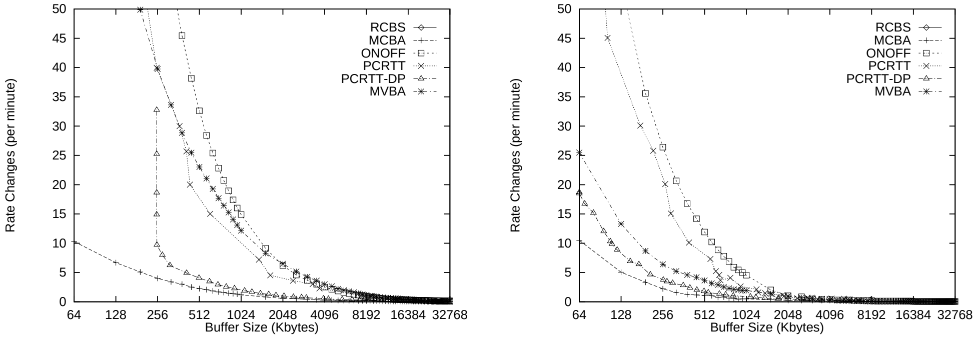
Fig. 7. **Bandwidth Variability Across All Algorithms:** These graphs plot the coefficient of variation (standard deviation normalized by the mean) of the bandwidth requirements as a function of the client buffer for each of the bandwidth smoothing algorithms for two video clips.



(a) Motion-JPEG clips

(b) MPEG clips

Fig. 8. **Minimum Frequency of Bandwidth Changes:** These graphs plot the rate of bandwidth changes as a function of the client buffer size for the MCBA algorithm, which minimizes this metric.



(a) *Jurassic Park* (M-JPEG)

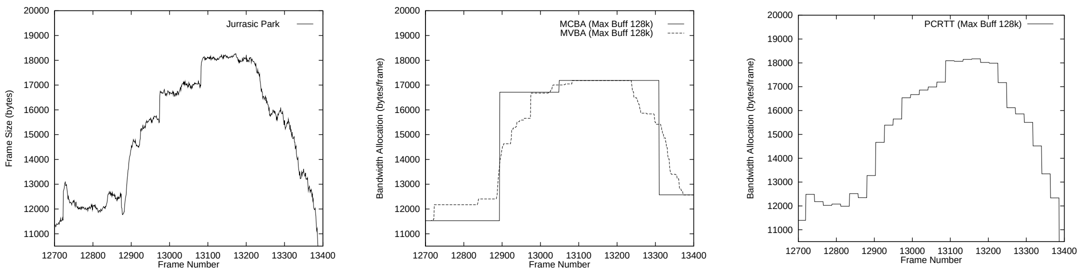
(b) *MTV 1* (MPEG)

Fig. 9. **Bandwidth Change Frequency Across All Algorithms:** These graphs plot the rate of bandwidth changes as a function of the client buffer size for each of the bandwidth smoothing algorithms, except the RCBS algorithm. The rage change frequency for the RCBS plans remains above 800/minute for *Jurassic Park* and above 600/minute for *MTV 1*, even for very large buffer sizes.

more bandwidth changes than the other algorithms except for extremely large buffer sizes.

E. Buffer Utilization

Although bandwidth smoothing reduces the rate requirements for transmitting stored video, workahead transmission may consume significant buffer resources at the client site. For a given size b for the playback buffer, a smoothing algorithm could strive to limit buffer utilization while still minimizing the peak rate [18]. Reducing the buffer utilization allows the client to statistically share the playback space between multiple video streams, or even other applications. If the client application can perform VCR functions, such as rewinding or indexing to arbitrary points in the video stream, a bandwidth plan that limits buffer utilization also avoids wasting server and network resources on transmitting frames ahead of the playback point. With fewer future frames in the playback buffer, the client can cache multiple frames behind the current playback point, allowing the service to satisfy small VCR rewind requests directly at the client site. Although the client could statically allocate buffer space for rewind operation, this would



(a) Segment of *Jurassic Park* (b) MCBA and MVBA plans (c) PCRTT plan

Fig. 10. **Gradual Changes in Frame Sizes:** This figure highlights the differences between the MCBA and MVBA plans for a 23-second segment of *Jurassic Park* under a 128-kilobyte playback buffer. The MVBA algorithm performs a large number of small bandwidth changes to track the gradual increases (decreases) in the frame sizes. In contrast, the MCBA plan initiates a smaller number of larger rate changes (3 changes vs. 104 in the MVBA plan). The corresponding PCRTT plan has 31 rate changes.

reduce the effectiveness of bandwidth smoothing by offering a smaller value of b to the smoothing algorithm. Instead, we focus on the utilization of the playback buffer across time, to determine the average amount of rewind space available.

The utilization of the client buffer corresponds to how far the transmission schedule lies above the lower constraint curve $L(k)$. By design, RCBS plans stay as close to the $L(k)$ curve as possible, without violating the peak rate restriction. The RCBS plan has less than 15% buffer utilization for most of the video clips in Figure 11. In fact, an RCBS plan only reaches the $U(k)$ curve during the bandwidth runs that must transmit frames at the peak rate. For example, Figure 12(a) shows the buffer utilization across time for the RCBS algorithm for the motion-JPEG video *Crocodile Dundee* and an 11-megabyte client playback buffer. For comparison, the graph also includes the buffer utilization for a transmission schedule that sends frames as *late* as possible, subject to the buffer and peak-rate constraints. This *inverse* RCBS algorithm maximizes client buffer utilization, and may be useful in supporting efficient fast-forward operations at the client and tolerating variation in network latency[5]. Any transmission schedule that minimizes the peak rate must have a buffer utilization that lies between these two curves. Note that the two curves meet at a common buffer utilization of 100% when both schedules must have a full client buffer to avoid exceeding the peak transmission rate.

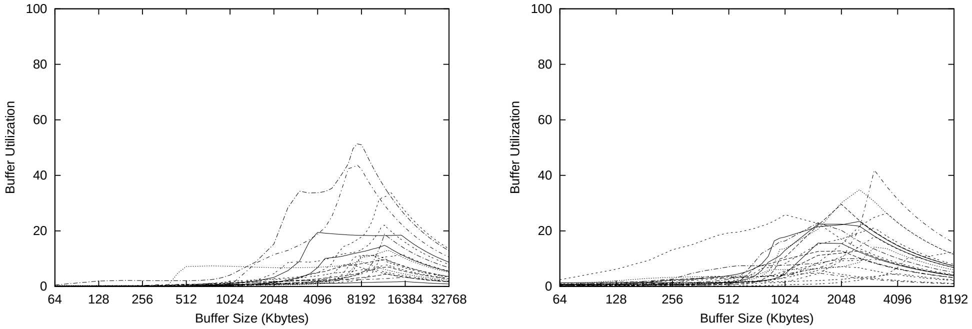
IV. Conclusions and Future Work

In this paper, we have compared a collection of bandwidth smoothing algorithms for compressed, prerecorded video. By capitalizing on the *a priori* knowledge of frame lengths, these algorithms can significantly reduce the burstiness of resource requirements for the transmission, transfer, and playback of prerecorded video. For small buffer sizes, the PCRTT algorithm is useful in creating plans that have near optimal peak bandwidth requirements. However, the PCRTT algorithm limits the ability of the server to smooth frames across interval boundaries. The MCBA and MVBA algorithms exhibit similar performance for the peak rate requirement and the variability of bandwidth allocations; the MCBA algorithm, however, is much more effective at reducing the total number of rate changes. The RCBS algorithm introduces a large number of rate changes, and a wide variability in the bandwidth requirements, to minimize the utilization of the client playback buffer. The ON-OFF algorithm also introduces a lot of variability, though the ON-OFF plans may be well-suited for systems that have hardware support for traffic shaping.

Future work can consider new smoothing algorithms that enforce a lower bound on the time between rate changes. The PCRTT algorithm serves as an initial approach to this problem, with some limitations in exploiting the playback buffer. In our experiments, we attempted to find the best interval size to use for the PCRTT algorithm given a fixed buffer size by calculating many interval lengths. Creating an efficient algorithm to find the best interval and interval offset, given a fixed buffer size, is a possible avenue for research. More generally, the use of dynamic programming in the PCRTT-DP algorithm offers a valuable framework for minimizing "costs" that are functions of multiple performance metrics. Similarly, hybrids of the other smoothing algorithm should be effective in balancing the trade-offs between different metrics. For example, extensions to RCBS (or inverse RCBS) algorithm could operate over coarser time intervals to reduce the variability in the transmission plans without significantly changing the buffer utilization properties.

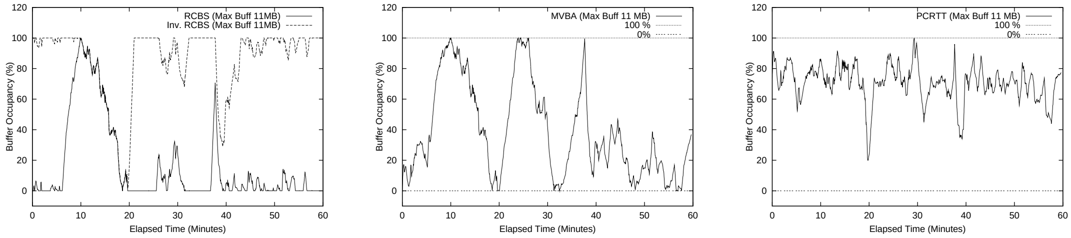
Ultimately, the construction of server transmission plans should depend on the actual configuration of the server and client sites, as well as the degree of network support for resource reservation and performance guarantees. For example, the server may have additional latitude in smoothing video streams if the client is willing to tolerate some loss in quality; for example, the server could avoid rate change operations by occasionally dropping frames, particularly if the stream has a layered encoding. These new schemes can broaden the family of bandwidth smoothing algorithms to tailor video transmission protocols to delay, throughput, and loss properties along the path from the server, through the communication network, to the client sites.

[5] Bounded jitter can also be tolerated by incorporating the maximum delay variation into the smoothing constraints [17].



(a) Motion-JPEG clips (b) MPEG clips

Fig. 11. **Minimum Average Buffer Utilization:** These graphs plot the average buffer utilization as a function of the client playback buffer size for the RCBS algorithm, which minimizes this metric.



(a) RCBS and Inverse RCBS (b) MVBA (c) PCRTT

Fig. 12. **Buffer Utilization:** This figure shows the buffer utilization over time for smoothing the movie *Crocodile Dundee* with an 11-megabyte playback buffer.

References

[1] D. Anderson, Y. Osawa, and R. Govindan, "A file system for continuous media," *ACM Transactions on Computer Systems*, pp. 311–337, November 1992.

[2] P.Lougher and D. Shepard, "The design of a storage server for continuous media," *The Computer Journal*, vol. 36, pp. 32–42, February 1993.

[3] D. Gemmell, J. Vin, D. Kandlur, P. Rangan, and L. Rowe, "Multimedia storage servers: A tutorial," *IEEE Computer Magazine*, vol. 28, pp. 40–49, May 1995.

[4] C. Aras, J. Kurose, D. Reeves, and H. Schulzrinne, "Real-time communication in packet switched networks," *Proceedings of the IEEE*, vol. 82, pp. 122–139, January 1994.

[5] H. Zhang, "Service disciplines for guaranteed performance service in packet-switching networks," *Proceedings of the IEEE*, vol. 83, pp. 1374–1396, October 1995.

[6] D. L. Gall, "MPEG: A video compression standard for multimedia applications," *Communications of the ACM*, vol. 34, pp. 46–58, April 1991.

[7] G. K. Wallace, "The JPEG still picture transmission standard," *Communications of the ACM*, vol. 34, pp. 30–44, April 1991.

[8] I. Dalgic and F. A. Tobagi, "Performance evaluation of ATM networks carrying constant and variable bit-rate video traffic," *IEEE Journal on Selected Areas in Communications*, vol. 15, August 1997.

[9] T. V. Lakshman, A. Ortega, and A. R. Reibman, "Variable bit-rate (VBR) video: Tradeoffs and potentials," *Proceedings of the IEEE*, vol. 86, May 1998.

[10] E. P. Rathgeb, "Policing of realistic VBR video traffic in an ATM network," *International Journal of Digital and Analog Communication Systems*, vol. 6, pp. 213–226, October–December 1993.

[11] S. S. Lam, S. Chow, and D. K. Yau, "An algorithm for lossless smoothing of MPEG video," in *Proc. ACM SIGCOMM*, pp. 281–293, August/September 1994.

[12] A. R. Reibman and A. W. Berger, "Traffic descriptors for VBR video teleconferencing over ATM networks," *IEEE/ACM Transactions on Networking*, vol. 3, pp. 329–339, June 1995.

[13] M. Grossglauser, S. Keshav, and D. Tse, "RCBR: A simple and efficient service for multiple time-scale traffic," *IEEE/ACM Transactions on Networking*, December 1997.

[14] O. Rose, "Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems," in *Proceedings of Conference on Local Computer Networks*, pp. 397–406, October 1995.

[15] W. Feng and S. Sechrest, "Critical bandwidth allocation for delivery of compressed video," *Computer Communications*, vol. 18, pp. 709–717, October 1995.

[16] W. Feng, F. Jahanian, and S. Sechrest, "Optimal buffering for the delivery of compressed prerecorded video," *ACM Multimedia Systems Journal*, September 1997.

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

[18] W. Feng, "Rate-constrained bandwidth smoothing for the delivery of stored video," in *Proceedings of IS&T/SPIE Multimedia Networking and Computing*, pp. 58–66, February 1997.

[19] J. M. McManus and K. W. Ross, "Video on demand over ATM: Constant-rate transmission and transport," *IEEE Journal on Selected Areas in Communications*, pp. 1087–1098, August 1996.

[20] J. M. McManus and K. W. Ross, "A dynamic programming methodology for managing prerecorded VBR sources in packet-switched networks," *Telecommunications Systems*, vol. 9, 1998.

[21] J. Zhang and J. Hui, "Applying traffic smoothing techniques for quality of service control in VBR video transmissions," *Computer Communications*, vol. 21, pp. 375–389, April 1998.

[22] Z. Jiang and L. Kleinrock, "General optimal video smoothing algorithm," in *Proc. IEEE INFOCOM*, pp. 676–684, April 1998.

[23] J. Zhang and J. Hui, "Traffic characteristics and smoothness criteria in VBR video transmission," in *Proc. IEEE International Conference on Multimedia Computing and Systems*, June 1997.

[24] W. Feng, *Video-On-Demand Services: Efficient Transportation and Decompression of Variable-Bit-Rate Video.* PhD thesis, University of Michigan, April 1996.

[25] W. Feng and J. Rexford, "A comparison of bandwidth smoothing techniques for the transmission of prerecorded compressed video," in *Proc. IEEE INFOCOM*, April 1997.

[26] S. Gringeri, K. Shuaib, R. Egorov, A. Lewis, B. Khasnabish, and B. Basch, "Traffic shaping, bandwidth allocation, and quality assessment for MPEG video distribution over broad-

band networks," *IEEE Network Magazine*, pp. 94–107, November/December 1998.

[27] M. Krunz, "Bandwidth allocation strategies for transporting variable-bit-rate video traffic," *IEEE Communication Magazine*, pp. 40–46, January 1999.

[28] J. Salehi, *Scheduling Network Processing on Multimedia and Multiprocessor Servers*. PhD thesis, University of Massachusetts – Amherst, September 1996.

[29] J. K. Dey, S. Sen, J. Kurose, D. Towsley, and J. Salehi, "Playback restart in interactive streaming video applications," in *Proc. IEEE Conference on Multimedia Computing and Systems*, pp. 458–465, June 1997.

[30] Z.-L. Zhang, J. F. Kurose, J. D. Salehi, and D. Towsley, "Smoothing, statistical multiplexing, and call admission control for stored video," *IEEE Journal on Selected Areas in Communications*, August 1997.

[31] L. Rowe, K. Patel, B. Smith, and K. Liu, "MPEG video in software: Representation, transmission, and playback," in *Proc. High Speed Networking and Multimedia Computing Symposium*, February 1994.

[32] P. Pancha and M. E. Zarki, "Prioritized transmission of variable bit rate MPEG video," in *Proc. IEEE GLOBECOM*, pp. 1135–1139, 1992.