Balancing the Trade-Off Between Prefetch and Rewind Buffering in Video-on-Demand Services

Wu-chi Feng
Computer and Information Sciences
Ohio State University
Columbus, OH 43210
wuchi@cis.ohio-state.edu

Jennifer Rexford
Networking and Distributed Systems
AT&T Labs Research
Murray Hill, NJ 07974
jrex@research.att.com

Abstract

The efficient delivery of pre-recorded, compressed video plays an important role in a wide variety of emerging multimedia services. However, compressed video exhibits significant burstiness that complicates the transmission from the video server, transfer across the communication network, and playback at the client site. By capitalizing on buffer space at the client, the server can reduce transmission overheads and enhance playback performance. For example, the server can decrease the peak bandwidth requirement of the outgoing video stream by prefetching frames in advance of each burst. Alternatively, the service could use the buffer space to save previous video frames, in the hope of satisfying VCR operations directly at the client site. This paper investigates this fundamental trade-off between network bandwidth requirements and efficient VCR functionality in video-on-demand services. In particular, we characterize the relationship between prefetch buffer space and the peak bandwidth requirement. Using these results, we develop a cost model and efficient numerical techniques for balancing the use of prefetch and VCR space in the client playback buffer. Experiments with sixteen compressed, full-length movies demonstrate that this buffer trade-off is unique for each movie and can be accurately represented with a small number of parameters.

1 Introduction

Emerging multimedia applications, such as distance learning and video entertainment services, depend on the efficient storage and transmission of pre-recorded digital video. In these applications, servers typically store pre-recorded video on large, fast disks [1–4], as shown in Figure 1; the server may also include tertiary storage, such as tapes or optical jukeboxes, for holding less frequently requested data [5]. 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 [6, 7], based on resource reservation requests from the video server. Client sites include workstations and set-top boxes that have a playback buffer for storing one or more video frames. In this paper, we propose effective techniques for capitalizing on this buffer space to reduce transmission overheads and enhance playback performance.

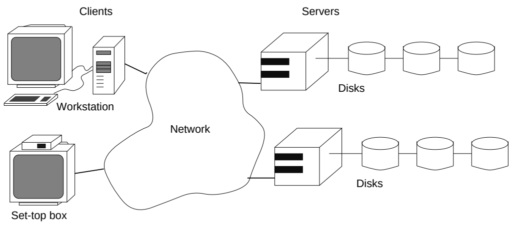


Figure 1: **Video-On-Demand Architecture:** This figure shows a basic video-on-demand architecture consisting of video servers, a communication network, and client sites. Possible clients include workstations and set-top boxes that contain hardware to interact with the network and a disk (or RAM) that can serve as a playback buffer.

To reduce the resource requirements for storing and transmitting video data, multimedia services typically employ compression techniques such as MPEG [8] and motion-JPEG. However, compressed video traffic typically exhibits significant burstiness on multiple time scales, due to the frame structure of the compression algorithm as well as natural variations within and between scenes [9–11]. This bursty, variable-bit-rate traffic complicates the effort to allocate server and network resources to ensure continuous playback at the client sites. For pre-recorded video, the server can reduce this variability by prefetching frames into the client playback buffer in advance of each burst. Based on the frame sizes in the video stream, a *bandwidth smoothing algorithm* [12–17] can schedule the transfer at a sequence of fixed rates, while avoiding overflow and underflow at the client site. With a larger prefetch buffer, bandwidth smoothing can achieve more dramatic reductions in the peak bandwidth requirement and the frequency of rate changes, resulting in lower demands on the server and network resources.

The availability of buffer space at the client site can also allow video-on-demand services to support efficient interactive operations, such as modest fast-forward and rewind functions. By storing multiple video frames, ahead and behind the playback point, the application can satisfy small requests directly at the client site. Although user behavior is not yet well-understood, video applications are likely to include requests to replay a short scene or scan past commercials. Satisfying these operations in the playback buffer provides a faster response, since the client can initiate the operation before contacting the server site and waiting for a response. For rewind operations, reading the frames directly from the client buffer avoids the server and network overheads of sending the data a second time. With a larger client buffer, the service can increase the likelihood of providing a rapid response to fast-forward and rewind operations.

Bandwidth smoothing and VCR functionality introduce a fundamental tension between prefetch and VCR space at the client site. Viewing the playback buffer as a circular queue, any prefetching of data can overwrite frames that may be necessary to satisfy a future rewind operation. Although aggressive prefetching can significantly reduce the peak bandwidth requirement, it can also result in times when no rewind space is available. This paper proposes a buffer management technique that balances the conflicting goals of minimizing the peak bandwidth requirement and maximizing the rewind/fast-forward space at the client. By reserving part of the space for rewind and fast-forward buffering, the algorithm ensures that the client can always satisfy small VCR operations. At times when the bandwidth smoothing algorithm does not exhaust the prefetch portion of the buffer, the service can field even longer interactive requests directly at the client site.

Section 2 describes how a bandwidth smoothing algorithm can construct a server transmission plan, based on the video stream and the client buffer space, and discusses the additional complexity introduced by VCR functionality. In Section 3, we introduce a cost model that guides the partitioning of the client playback buffer for prefetching and VCR functionality. Experiments with sixteen compressed, full-length movies show that this trade-off can vary significantly across different video clips. To allow video servers to quickly select an appropriate buffer partitioning for each movie, Section 4 introduces a simple, two-parameter model that characterizes the relationship between prefetch buffer size and the peak bandwidth requirement. Simulation experiments show that the model can accurately represent the buffer allocation trade-offs for a wide range of movies. Section 5 concludes the paper with a discussion of future research directions.

2 Capitalizing on the Client Playback Buffer

Bandwidth smoothing algorithms generate server transmission schedules that substantially reduce the rate requirements of pre-recorded video streams. By prefetching frames into the client playback buffer, a video-on-demand server can send large frames at a slower rate without disrupting the client application. Depending on the buffer utilization, the service can also use this playback space to satisfy interactive requests, such as rewind and fast-forward operations, directly at the client site.

2.1 Bandwidth Smoothing

The potential benefits of bandwidth smoothing depend of the size of the client prefetch buffer. The server must limit the amount of prefetching to prevent buffer overflow; however, to avoid underflow, the server should transmit enough data to allow the client to display frames at the playback rate. To formalize the problem, consider a compressed video stream that consists of n frames, where frame i is f_i bytes long. To permit continuous playback at the client site, the server must always transmit quickly enough to avoid buffer underflow, where $F_{under}(k) = \sum_{i=0}^{k} f_i$ 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 than $F_{over}(k) = F_{under}(k) + b$ bytes of data by frame k, to prevent overflow of the prefetch buffer

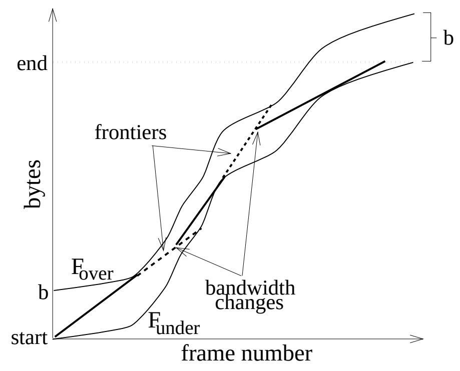
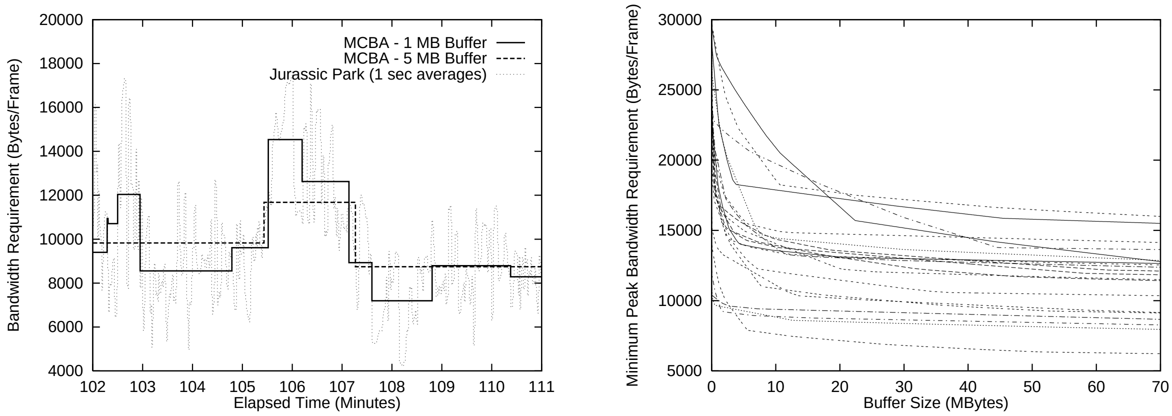


Figure 2: **Sample Transmission Plan:** This figure shows the buffer underflow and overflow curves for a sample video stream. To construct a server transmission plan, the smoothing algorithm constructs a monotonically-increasing, piecewise-linear curve that stays between F_{over} and F_{under}. The slope of each line represents the transmission rate during the corresponding time interval.

(of size b). Consequently, any valid server transmission plan should stay within the river outlined by these vertically equidistant functions, as shown in Figure 2. For greater latitude in constructing the plan, the smoothing algorithm can start the first bandwidth run at any point in $[0, b]$ on the y-axis to allow more prefetching at the start of the video at the expense of an initial playback delay at the client site.

Based on this framework, several bandwidth smoothing algorithms have been developed to address different performance goals [12–16]. Most smoothing algorithms minimize the peak rate in the transmission plan [12–15] to limit the worst-case resource requirements across the path from the video storage on the server, the route through the network, and the prefetch buffer at the client site. With smaller peak rate requirements, the server and the network can handle a larger number of video streams simultaneously[1]. Figure 3(a) shows two sample bandwidth plans for a segment of a motion-JPEG encoding of the movie *Jurassic Park*, using a smoothing algorithm that minimizes the peak bandwidth requirement and the number of rate changes for a given buffer size [14]. Compared even to a 1 megabyte prefetch buffer, a 5 megabyte buffer significantly reduces the burstiness of the video stream. Figure 3(b) shows the peak bandwidth as a function of buffer size for a collection of full-length, motion-JPEG encoded movies [18, 19]. Although a large prefetch buffer can substantially reduce the peak rate, the addition of more buffer space offers diminishing returns beyond a certain point.

[1]While minimizing the peak bandwidth requirement, a smoothing algorithm may also minimize the number of rate increases [12, 13], the total number of rate changes [14], or the *variability* of the rate requirements [15]. Alternatively, an algorithm could generate bandwidth plans that change rates only at periodic intervals [16]. For a detailed comparison of existing bandwidth smoothing algorithms, we point the interested reader to reference [18].



(a) Bandwidth plan for *Jurassic Park* (b) Peak bandwidth requirement vs. buffer size

Figure 3: **Bandwidth Smoothing:** On the left side, this figure shows sample bandwidth plans for the movie *Jurassic Park*, with 1-megabyte and 5-megabyte prefetch buffers. A second plot shows the peak bandwidth as a function of the buffer size for fifteen full-length, motion-JPEG encoded movies.

2.2 Rewind and Fast-Forward Functionality

In addition to transmitting variable-bit-rate video, many emerging multimedia services need efficient ways to give users more control over video playback. For example, a video-on-demand application could allow the user to pause, rewind, fast-forward, or play in slow motion. In the simplest case, the client can forward these interactive requests to the server site; then, the server can reinitiate transmission at a different point in the video stream or at a different rate [20–22]. However, depending on the network latency, this approach may introduce substantial delay in satisfying the client request. The use of bandwidth smoothing further increases this delay if the new playback point requires the server to (pre)fetch large frames at a slower rate. To avoid this extra latency, the server could transmit the new frames at a higher rate, temporarily ignoring the bandwidth plan; however, sending at a higher rate would increase the demands on the server and network resources, particularly when multiple users request VCR functions simultaneously.

To improve response time, the client can try to satisfy the user request directly from the playback buffer. Even for long rewind and fast-forward operations, the prefetch buffer may hold the first few frames that the client should display for the user. By retrieving part of the new data from the prefetch buffer, the client can start responding without waiting for a reply from the server. If the buffer holds a reasonable number of frames, the server may even have enough time to use the bandwidth plan to prefetch the new frames, instead of retrieving and transmitting the video based on the size of the largest frame. Viewing the client playback buffer as a circular queue, the amount of fast-forward and rewind space depends on how much prefetching occurs. At any given time, the fast-forward (rewind) space is the distance that the bandwidth plan lies above F_{under} (below F_{over}). Video-on-demand services can improve interactive performance, while decreasing the load on the network and the server, by dedicating a minimum portion of the client playback buffer to supporting fast-forward and rewind functionality.

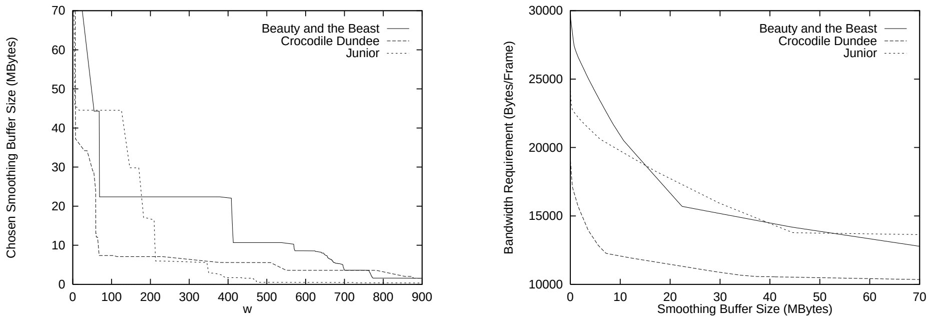
3 Interactive Buffering

To balance the trade-off between bandwidth smoothing and VCR functionality, a video-on-demand service can partition a B-byte client playback buffer into a b-byte prefetch area and a $(B-b)$-byte region dedicated to satisfying interactive requests, where $b \in [0, B]$. The appropriate value for b depends on a variety of factors, including the cost of network bandwidth and the frequency/duration of interactive requests, as well as the user's willingness to wait for playback to continue after selecting a VCR function. Note that the service can divide the $(B-b)$-byte interactivity space into separate rewind and fast-forward regions, depending on the expected mix of these two operations.

3.1 Provisioning the Prefetch Space

To guide the selection of b, we introduce a cost model that captures the trade-off between the peak bandwidth requirement and the importance of supporting efficient VCR functionality. The peak bandwidth $\text{peak}(b)$ represents the network and server costs for transmitting the video stream, in the absence of rewind and fast-forward operations. However, selecting a large value of b reduces $\text{peak}(b)$ at the expense of efficient VCR functionality. Hence, our cost model includes an additional term wb, where the constant w loosely represents the "importance" or frequency of interactive operations. Although mapping w into user-perceived quality requires extensive experimental studies, note that a larger value of w increases the incremental cost of prefetch space in the playback buffer. Under this cost model, selecting a value for b reduces to minimizing the function $\text{peak}(b) + wb$ over the interval $b \in [0, B]$.

To evaluate how w affects the cost of delivery, Figure 4 plots the smoothing buffer size b for three movies as a function of w, where B is arbitrarily large; lower values of B effectively truncate the curve on the y-axis, resulting in $b = B$ when w is small. Each of the movies experience an initially rapid drop in the size of the smoothing buffer as the weight w increases. The long plateaus in Figure 4(a) correspond to the "knee" of the peak bandwidth curves, where additional prefetch space offers diminishing returns. For example, in transmitting *Beauty and the Beast* a video server can extract most of the benefit of bandwidth smoothing by using 22 megabyte prefetch buffer, as shown in Figure 4(b). Similarly, an 8-megabyte prefetch buffer can extract most of benefit for *Crocodile Dundee*.



(a) Optimal prefetch size b vs. w (b) Peak bandwidth requirement vs. b

Figure 4: **Smoothing Buffer Size Selection:** These graphs plot the chosen smoothing buffer size for varying interactivity weights w for three full-length, motion-JPEG encoded movies.

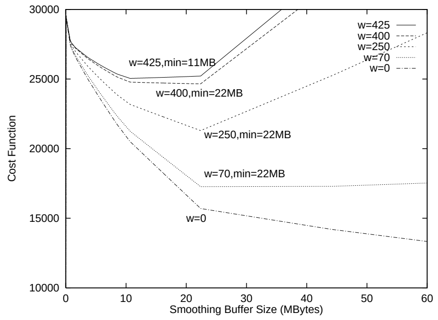
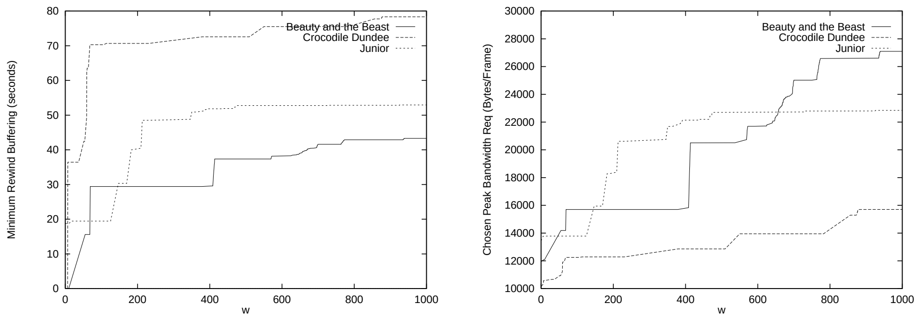


Figure 5: **Cost Function Examples:** This graph plots the cost function for varying smoothing buffer sizes b and various interactivity weights w for the movie *Beauty and the Beast*.

Interestingly, the curves in Figure 4 often have long plateaus that cover a wide range of w values, as well as sharp transitions between different buffer sizes b. To help explain this phenomenon, Figure 5 shows the shape of the cost functions for the movie *Beauty and the Beast*. For the $w = 0$ curve, consider the region between $b = 11$ and $b = 22$ megabytes. As the weight w increases, the cost curve tends to rise on the right side (i.e., for larger values of b). When $w = 70$, the minimum cost occurs at 22 megabytes. For larger values of w, the cost curve continues to rise on the right (e.g., see the $w = 250$ line); however, even at $w = 400$, the minimum cost still occurs at 22 megabytes. As w continues to increase, the cost curve between 11 and 22 megabytes is actually flat, coinciding with where the slope of peak(b) and wb are equal with opposite signs. As w grows above 413, the optimal buffer size changes to 11 megabytes, because the portion of the cost curve between $b = 11$ and $b = 22$ finally has a positive slope (derivative).

3.2 Smoothing vs. Interactivity

To quantify the trade-off between VCR buffering and bandwidth smoothing, Figure 6 plots the minimum rewind space and the peak bandwidth requirement as a function of the interactivity weight

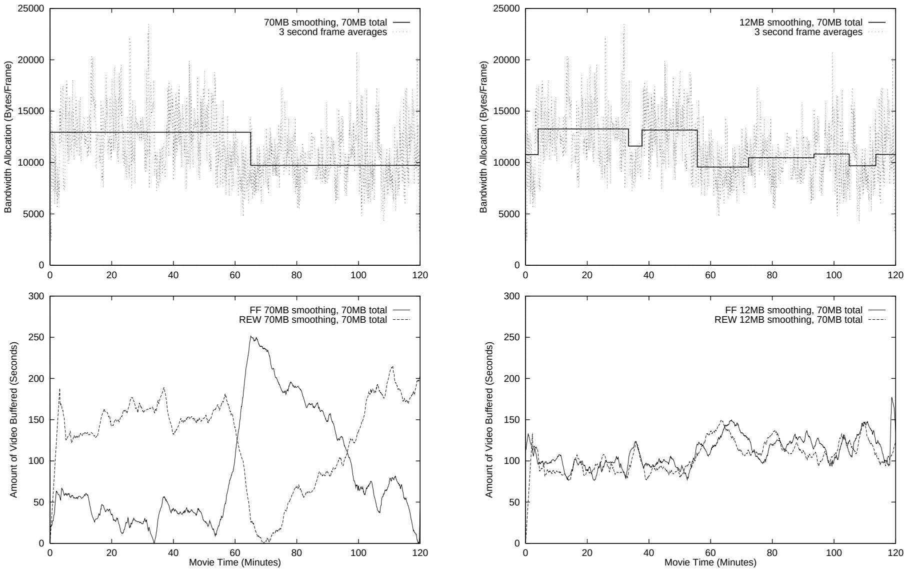


(a) Minimum rewind time vs. w (b) Peak bandwidth vs. w

Figure 6: **Rewind and Fast-Forward Buffering:** This graph plots the minimum available rewind time and the peak bandwidth requirement as a function of the interactivity weight w, for a $B = 70$ megabyte client buffer. Fast-forward times (not shown) are nearly identical to rewind times, since the service divides the $(B-b)$-byte VCR buffer into equal rewind and fast-forward regions.

w. The experiment evaluates a $B = 70$ megabyte playback buffer, with the interactive space $B - b$ evenly divided between rewind and fast-forward space. As expected, larger values of w increase the amount of rewind time available in the client buffer, at the expense of a higher peak bandwidth. For the movie *Crocodile Dundee*, a weight of $w = 100$ results in a 7-megabyte smoothing buffer, reserving 31.5 megabytes each for rewind and fast-forward operations. This ensures that the service has a minimum of 70 seconds of rewind time at the client buffer, at the expense of increasing the peak bandwidth requirement by 1700 bytes/frame. It is important to note that Figure 6 plots results for motion-JPEG encoded movies. An MPEG encoding would capitalize on inter-frame redundancy to (1) decrease the bandwidth requirement by a factor of 4 to 7 and (2) increase the minimum VCR guarantee by a factor of 4 to 7, resulting in approximately 5 minutes of rewind and fast-forward space available at all times.

To highlight the average performance of the buffer partitioning scheme, Figure 7 evaluates the movie *Jurassic Park* with a $B = 70$ megabyte playback buffer. When the service does not reserve any space interactive operations, the client experiences wide variations in the available VCR space, as shown in Figure 7(a). At various times, the client playback buffer does not have *any* rewind (fast-forward) frames; at these points in the movie, no rewind (fast-forward) request, however small, could be satisfied directly at the client site. In contrast, by dedicating 58 megabytes to VCR functionality, the client buffer has an average of 106 seconds of rewind time and 110 seconds of fast-forward time, as shown in Figure 7(b). The smaller prefetch buffer only slightly increases the peak bandwidth and the number of bandwidth changes in the server transmission plan. A 12-megabyte prefetch buffer results in a bandwidth plan with 10 rate changes and a peak bandwidth of 13273 bytes/frame, in comparison with 2 rate changes and a peak bandwidth of 12950 bytes/frame.



(a) $b = 70$ and $B - b = 0$ (b) $b = 12$ and $B - b = 58$

Figure 7: **Smoothing Versus Interactivity:** This figure shows the benefits of dedicating a portion $B-b$ of the playback buffer to supporting VCR functionality for the movie *Jurassic Park*.

4 Approximate Cost Model

To efficiently select an appropriate partitioning of the playback buffer, video-on-demand services need effective techniques for estimating the relationship between prefetch space and the peak bandwidth. This section presents a two-parameter model that accurately characterizes the peak bandwidth requirements of most videos. Using this approximation, the server can use a simple analytic expression to determine the prefetch size, instead of iterating through multiple calculations of $\text{peak}(b) + wb$.

4.1 Characterization of the Peak Bandwidth Curve

Although the relationship between b and $\text{peak}(b)$ varies across different video streams, the peak bandwidth curves have a common shape, as shown in Figure 3(b). For a given movie, the value of $\text{peak}(b)$ varies from the maximum frame size (when $b = 0$) to the average frame size (when b is large). As b increases, the reduction in the peak bandwidth depends on the time scale of burstiness in the underlying video stream. For a wide range of prefetch sizes, the $\text{peak}(b)$ curve is nearly logarithmic in b (i.e., $\text{peak}(b) \approx c\ln(b) + d$), as shown in Table 1. To determine the slope c and the intercept d, we calculated the peak bandwidth curve for buffer sizes ranging from 64 kilobytes to 100 megabytes and used the least-squares linear regression method to fit the data. In order to measure the accuracy of the approximation, we computed the standard error of estimation and the coefficient of correlation for each of the movies; similar to the standard deviation, the standard error of estimation quantifies

	Rate (Mbps)	$y = c\ln(b) + d$ c	d	Std. Error	Correl. Coeff.
Beauty and the Beast	3.04	-3642.85	28229.91	799.15	-0.980
Big	2.96	-1200.02	17254.66	96.51	-0.996
Crocodile Dundee	2.59	-1113.32	14948.04	298.80	-0.960
E.T.	2.17	-1303.68	14549.78	404.58	-0.950
Home Alone 2	2.73	-1269.66	16777.11	152.48	-0.990
Honey, I Blew Up the Kid	3.32	-854.32	17622.05	361.99	-0.930
Hot Shots 2	3.06	-1755.62	20003.28	719.34	-0.930
Jurassic Park	2.73	-1284.38	17526.46	560.94	-0.930
Junior	3.36	-2445.69	24179.49	713.58	-0.950
NCAA Final Four	3.95	-1477.07	21666.36	420.84	-0.960
Rookie of the Year	2.98	-825.25	15910.25	516.18	-0.860
Sister Act	2.86	-1339.65	17012.46	388.28	-0.960
Sleepless in Seattle	2.28	-1020.20	13499.37	177.88	-0.980
Speed	2.97	-861.72	16113.84	507.96	-0.860
Total Recall	2.34	-1182.25	16902.26	232.97	-0.980
Star Wars (MPEG)	0.47	-385.05	3140.42	107.45	-0.970

Table 1: **Video Statistics:** This table shows the results of approximating the peak bandwidth as a log-linear function of the prefetch buffer size for a collection of full-length video clips.

the spread of the data around the regression line.

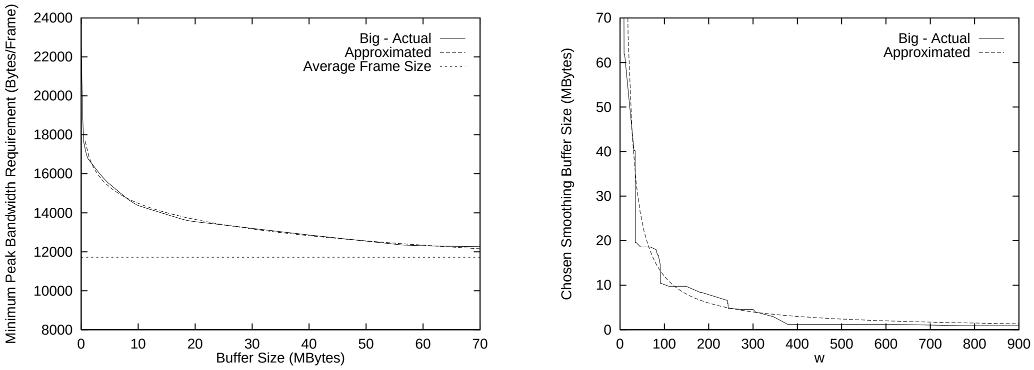
The sample video streams, including one MPEG trace [23] and fifteen motion-JPEG traces [18, 19], vary substantially in terms of their durations, average bandwidth requirements, and subject matter. Still, the log-linear approximation closely matches the peak bandwidth curve for most of the movies, with most of the videos having a correlation coefficient of better than -0.93. The standard error estimation is typically less than 500 bytes/frame, resulting in an accurate prediction of peak(b) as a function of b. Using this two-parameter model for peak(b), the cost function from Section 3 reduces to

$$(c\ln(b) + d) + wb, \quad b \in [0, B].$$

To minimize this function, we find the value of b where the derivative (with respect to b) equals zero. This results in $b = \min\{-c/w, B\}$. Consequently, the prefetch buffer size scales as $1/w$, which is consistent with the experimental trends in Figure 4 in Section 3.

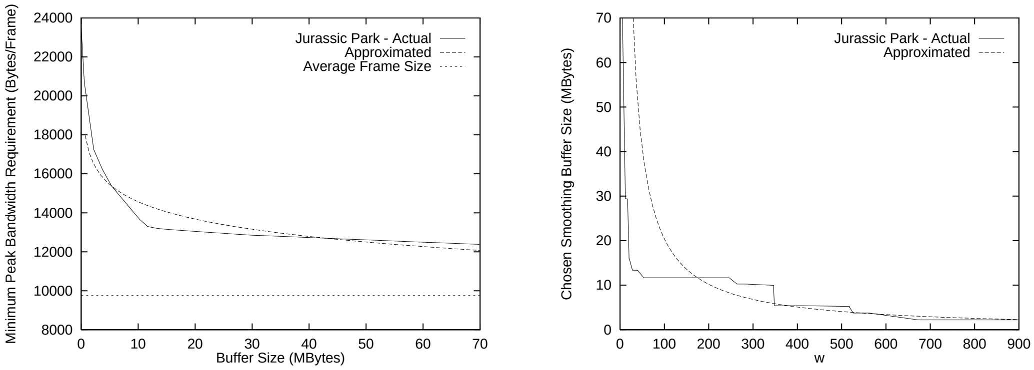
4.2 Simulation Experiments

To evaluate the accuracy of the approximate cost function, Figures 8 and 9 plot the peak bandwidth and chosen buffer size as a function of w for two movies. For the movie *Big*, the approximation generates results that match closely to the exact cost model, as shown in Figure 8. For a given peak bandwidth, the approximate curve has a standard error of estimation of 97 bytes/frame. However, the two-parameter approximation is not as accurate for the *Jurassic Park* video, particularly for



(a) Peak bandwidth vs. b (b) Prefetch buffer size vs. w

Figure 8: **Approximate Cost Function for *Big***: This figure shows the chosen prefetch buffer size for varying interactivity weights w, using both the exact and approximate cost models.



(a) Peak bandwidth vs. b (b) Prefetch buffer size vs. w

Figure 9: **Approximate Cost Function for *Jurassic Park***: This figure shows the chosen buffer size for varying interactivity weights w, using both the exact and approximate cost models.

larger buffer sizes, as shown in Figure 9. The main reason for this is that *Jurassic Park* has areas of large frames for the first half of the movie followed by smaller frame sizes, on average, for the remainder of the movie, as shown in Figure 7. As a result, the exact curve tends to be linear, instead of logarithmic, for large buffer sizes. An extremely large prefetch buffer would be necessary to smooth the burstiness on such a long time scale; as shown in Figure 7, changing from $b = 12$ to $b = 70$ only slightly decreases the height (and only slightly increases the length) of the peak bandwidth run.

In general, the movies that do not have long, sustained areas of large frames have better log-linear approximations, like in the movie *Big*. We are currently examining more complex models to more accurately describe individual movies. In particular, we have found that, while the peak bandwidth requirement versus minimum smoothing buffer size curve is roughly proportional to the logarithm of b, the curve seems to consist of two distinct parts, corresponding to small and large buffer sizes. These two regions are for the smoothing of short-term and long-term burstiness, respectively. Once b grows large enough to remove the short and medium scale burstiness, adding more prefetch space offers diminishing returns (as in *Jurassic Park*), resulting in a nearly flat region in the peak(b) curve. More generally, the peak bandwidth curves can have regions with different slopes, depending on the degree of burstiness on different time scales. Additional parameters in the model can capture these effects to improve the selection of a suitable size for the prefetch buffer; for example, Figure 4 suggests that, for some movies, the server may be able to store a few buffer sizes b that each cover a fairly wide interval of weights w.

5 Conclusion

Video-on-demand services can capitalize on client-side buffering to reduce transmission overheads and enhance playback performance. In this paper, we have investigated the tension between bandwidth smoothing and efficient VCR functionality. Although prefetch buffering reduces the peak bandwidth requirement for a video stream, reserving part of the client playback buffer for rewind and fast-forward operations can significantly improve the responsiveness to interactive requests. To balance on these conflicting trade-offs, we presented a simple cost function based on measurements of the peak bandwidth requirements of the underlying video streams. Then, we presented effective numerical techniques for characterizing the peak bandwidth curve, resulting in an efficient analytic expression. Experiments with sixteen compressed full-length videos demonstrate the effectiveness of the buffer partitioning scheme.

As future work, we are considering more complex models to characterize the relationship between the peak bandwidth requirement and the size of the prefetch buffer. By capturing the different time scales of burstiness, these models can accurately guide the division of the client buffer space. Often, a relatively small portion of the movie determines the peak bandwidth requirement. Instead of statically reserving a portion of the buffer for VCR functionality, new bandwidth smoothing algorithms could dynamically balance the trade-off between rewind/fast-forward space and the peak rate requirement in constructing the server transmission plan, based on the framework in Figure 2. Finally, additional simulation experiments with full-length MPEG traces would lend further insight into the trade-off between prefetch and rewind/fast-forward space in emerging video-on-demand services.

References

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

[2] 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.

[3] D. Kandlur, M. Chen, and Z. Shae, "Design of a multimedia storage server," in *Proc. IS&T/SPIE Symposium on Electronic Imaging Science and Technology*, February 1994.

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

[5] C. Federighi and L. Rowe, "A distributed hierarchical storage manager for a video-on-demand system," in *Proc. IS&T/SPIE Symposium on Electronic Imaging Science and Technology*, February 1994.

[6] 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.

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

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

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

[10] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of Ethernet traffic (extended version)," *IEEE/ACM Trans. on Networking*, vol. 2, pp. 1–15, February 1994.

[11] M. Grossglauser, S. Keshav, and D. Tse, "RCBR: A simple and efficient service for multiple time-scale traffic," in *Proc. ACM SIGCOMM*, pp. 219–230, August/September 1995.

[12] W. Feng and S. Sechrest, "Smoothing and buffering for delivery of prerecorded compressed video," in *Proc. IS&T/SPIE Symposium on Multimedia Computing and Networking*, pp. 234–242, February 1995.

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

[14] W. Feng, F. Jahanian, and S. Sechrest, "Optimal buffering for the delivery of compressed prerecorded video," in *Proc. IASTED/ISMM International Conference on Networks*, January 1995. Extended version to appear in *ACM/Springer-Verlag Multimedia Systems Journal.*

[15] 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," in *Proc. ACM SIGMETRICS*, pp. 222–231, May 1996.

[16] J. M. McManus and K. W. Ross, "Video on demand over ATM: Constant-rate transmission and transport," in *Proc. IEEE INFOCOM*, pp. 1357–1362, March 1996.

[17] J. M. del Rosario and G. C. Fox, "Constant bit rate network transmission of variable bit rate continuous media in video-on-demand servers," *Multimedia Tools and Applications*, vol. 2, pp. 215–232, May 1996.

[18] W. Feng and J. Rexford, "A comparison of bandwidth smoothing techniques for the transmission of prerecorded compressed video." To appear in *IEEE INFOCOM 1997.* Postscript is available at `http://www.eecs.umich.edu/~wuchi/pubs.html`, April 1997.

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

[20] M. Chen, D. Kandlur, and P. Yu, "Support for fully interactive playout in a disk-array-based video server," in *Proc. ACM Multimedia*, pp. 391–398, October 1994.

[21] J. Dey-Sicar, J. Salehi, J. Kurose, and D. Towsley, "Providing VCR capabilities in large-scale video servers," in *Proc. ACM Multimedia*, pp. 25–32, October 1994.

[22] P. Shenoy and H. Vin, "Efficient support for scan operations in video servers," in *Proc. ACM Multimedia*, October 1995.

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