TIME CONSTRAINED BANDWIDTH SMOOTHING FOR INTERACTIVE VIDEO-ON-DEMAND SYSTEMS

Wu-chi Feng
The Ohio State University
Department of Computer and Information Science
Columbus, OH 43210 USA

ABSTRACT

The use of a client-side buffer in the delivery of compressed prerecorded video can be an effective tool for removing the burstiness required of the underlying server and network by *smoothing* the bandwidth requirements for continuous delivery. Given a fixed-size smoothing buffer, several bandwidth smoothing algorithms have been introduced in the literature that are provably optimal under certain constraints, typically requiring large buffer residency times to realize their optimal properties. The large buffer residency times, however, make VCR functions harder to support. In this paper, we introduce the notion of *time constrained bandwidth smoothing*. Specifically, we introduce two new algorithms that, in addition to the size of the client side buffer, use a time constraint as a parameter in the bandwidth smoothing plan creation, making the plans more amenable to supporting VCR interactivity. Our results show that the buffer residency times can be reduced, while still allowing the bandwidth allocation to be smoothed for continuous video delivery.

1. Introduction

The use of video compression algorithms such as MPEG result in video streams that exhibit significant burstiness on multiple time scales. For stored video-on-demand (VOD) systems, this bursty variable-bit-rate video can complicate resource management for both the network and VOD server. To aid in the resource management, *bandwidth smoothing* techniques have been introduced to take advantage of both the *a priori* information that is available from stored video streams as well as a client-side buffer such as a disk or random access memory (RAM). By prefetching data into the client-side buffer before large bursts of frames occur, a client can smooth the resource requirements needed from the server and network, allowing it to serve a greater number of clients simultaneously.

Given a fixed size client, several bandwidth smoothing algorithms have been introduced that are provably optimal under certain conditions. In particular, a class of *river-charting* bandwidth smoothing algorithm have been introduced that, given a fixed size client-side buffer, result in plans that have the *minimum* peak bandwidth requirements for the continuous delivery of the video. While minimizing the peak bandwidth requirement, these *river-charting* algorithms may also minimize

- the number of rate increases [8],
- the total number of rate changes [9],
- the variability of bandwidth requirements [25], or
- the buffer residency times [13].

The first three techniques typically require large buffer residency times to achieve their optimal properties. As an example, using the *minimum changes bandwidth smoothing* algorithm and a 10 megabyte smoothing buffer results in buffer residency times on the order of half a minute (or more). With larger client-side buffers, greater reductions in peak bandwidth requirements are possible but require even larger buffer residency times.

For interactive VOD systems, the use of bandwidth smoothing algorithms complicate the ability to provide fully interactive functions. As an example, consider a long scan into a region of very large frame sizes. Under the bandwidth smoothing plans, these frames would have been prefetched in advance to smooth the bandwidth requirements. In order to support this scan, the server must then 1) allocate additional network and server resources to avoid buffer underflow, 2) make the user wait until some of the data can be prefetched into the client-side buffer, or 3) decrease the quality of the video stream to fit within the available resources. The first two approaches can be aided by the use of a *contingency channel*, which reserves part of the network resources for temporary actions such as VCR interactivity. The amount of additional resources required, however, can be minimized by minimizing the buffer residency times (or buffer utilization). The *minimum buffer residency* algorithm does this, however, it typically requires orders of magnitude more bandwidth changes than the other techniques.

In this paper, we introduce the notion of *time-constrained bandwidth smoothing* for the interactive delivery of prerecorded compressed video. We propose two new algorithms, a *time-constrained bandwidth smoothing* algorithm and a *rate/time constrained bandwidth smoothing* algorithm. By using a time constraint, *t,* as a parameter in the creation of the bandwidth plans, these algorithms allow for the advantages of small buffer residency times *as well as* smoothing

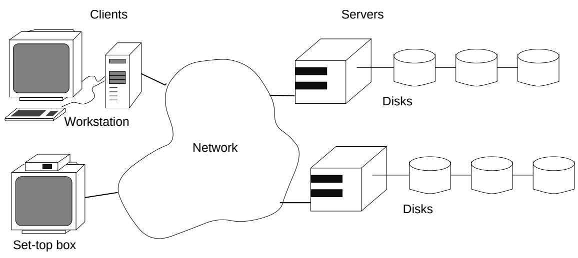


FIGURE 1: Video-On-Demand Architecture - This figure shows a basic video-on-demand architecture consisting of video servers, a communication network, and client sites. Possible clients include workstations and set-top boxes that contain hardware to interact with the network and a disk (or RAM) that can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

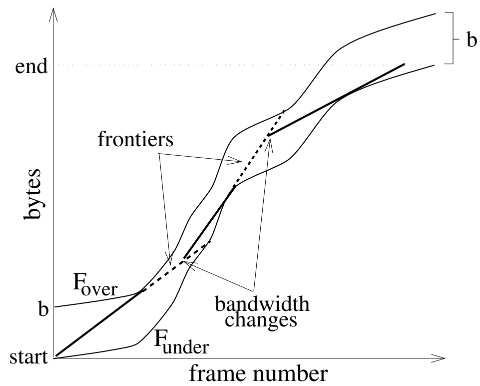


FIGURE 2: Sample Transmission Plan - This figure shows the buffer underflow and buffer overflow curves for a sample video stream. The plan consists of three constant-rate runs that serve as a schedule for transmitting video frames from the server.

the number of bandwidth changes that are required for continuous playback. More specifically, the time-constrained bandwidth smoothing algorithm takes two parameters, a maximum prefetch time t and a maximum buffer size b. The algorithm then creates a plan for the continuous delivery of the video that does not violate either constraint and results in a plan with the minimum peak bandwidth requirement, given the two constraints. The rate/time constrained bandwidth algorithm takes an additional parameter, r, the maximum allowable peak bandwidth requirement. Using r, t, and b, the rate/time constrained bandwidth smoothing algorithm allows the time constraint to be relaxed only when the rate constraint r would have otherwise been violated. A comparison with other bandwidth smoothing algorithms using both a Motion-JPEG encoded video and a MPEG encoded video are included. Our results show that the time-constrained bandwidth smoothing algorithms effectively balance the trade-off of small buffer residency times and the number of rate changes required for continuous playback.

In the next section, we describe some of the background and related work on bandwidth smoothing techniques that have appeared in the literature and discuss their implications on providing interactive video-on-demand services. In Sections 3 and 4, we present the time-constrained bandwidth smoothing algorithm and the rate/time constrained bandwidth smoothing algorithms, respectively. To compare and contrast the new algorithms, Section 5 provides an evaluation of the two new techniques. Finally, we summarize our results and provide directions for future research.

2. Background

The basic video-on-demand architecture that we assume is shown in Figure 1. Video servers typically store prerecorded video on large, fast disks [1, 14, 17, 23] and may also include tertiary storage, such as tapes or optical jukeboxes, for holding less frequently requested data [6]. A network connects the video servers to the client sites through one or more communication links; the network can help ensure continuous delivery of the smoothed video data by including support for rate or delay guarantees[2, 27], based on resource reservation requests from the video server. The client sites, such as workstations or set-top boxes, include a buffer for storing prefetched video frames; in addition, the client may interact with the server to compute video transmission plans, based on the buffer size and frame lengths.

2.1 Creating Bandwidth Allocation Plans

A multimedia server can substantially reduce the rate requirements for transmitting prerecorded video by prefetching frames into the client playback buffer. In order to minimize these requirements, a bandwidth smoothing algorithm must capitalize on the *a priori* knowledge of the video and should compute a server transmission schedule based on the size of the prefetch buffer.

For a video that consists of n frames, where frame i requires f_i bytes of storage, 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 by the client by frame k, where $k=0,1,...,n-1$. In addition, a client should receive no more than

$$F_{over}(k) = b + \sum_{i=0}^{k} f_i$$

by frame k to prevent buffer overflow of the playback buffer (of size b). Consequently, any valid server plan should stay within the *river* defined by the vertically equidistant functions F_{under} and F_{over}. That is,

$$F_{under}(k) \le \sum_{i=0}^{k} c_i \le F_{over}(k)$$

where c_i is the transmission rate during frame slot i of the smoothed video stream. A sample valid bandwidth allocation plan is shown in Figure 2. Given the functions F_{under} and F_{over}, the *river-charting bandwidth smoothing* plans chart a path from the beginning of the movie to the end that stay within the river.

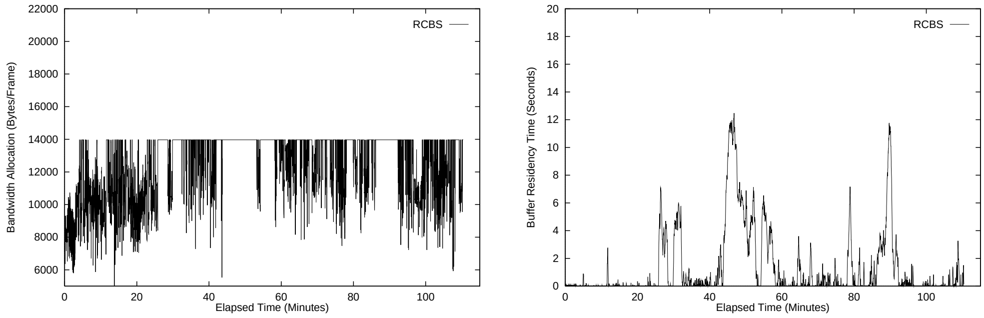


FIGURE 4: Rate Constrained Bandwidth Smoothing: This figure shows an example of the *rate-constrained bandwidth smoothing algorithm* for the Motion-JPEG compressed movie *Speed* using a 5 megabyte buffer. The right figure shows the buffer residency times for the same algorithm.

Based on this framework, several algorithms have been developed to traverse down the river defined by F_{under} and F_{over}. Most algorithms minimize the peak bandwidth requirement given the fixed client-side buffer[7, 8, 13, 25]. While minimizing the peak bandwidth requirement, a smoothing algorithm may also minimize the number of rate increases (CBA) [7, 8], minimize the total number of rate changes (MCBA) [9], or minimize the variability of rate changes (MVBA) [25] while providing for the continuous uninterrupted playback of video. To avoid a long discussion, these three algorithms select *runs* of constant bandwidth allocation that, given a starting point, extend as far into the video as possible. These runs create trajectories that touch both the F_{under} and F_{over} curves known as *frontiers* (see Figure 2). The three algorithms then select different starting points along the frontier for the next run. Depending on the chosen starting point, the algorithms will exhibit different properties. A more in-depth discussion of the pre-1997 algorithms can be found in reference [12]. Sample bandwidth allocation plans for these three algorithms (along with their buffer residency requirements) can be found in Figure 3. One important trait that these algorithms exhibit is that with small amounts of buffering significant reductions in the burstiness of the video delivery are possible. Note that in Figure 3 the one second frame averages represents smoothing across 30 frames. Thus, the actual frame sizes are more bursty than are shown in the figure.

As an alternative to the previous three algorithms, a bandwidth smoothing algorithm can strive to minimize the buffer residency times. The *rate-constrained bandwidth smoothing* (RCBS), given a maximum rate-constraint *r*, minimizes both the buffer size required and also reduces the buffer residency times[13]. The RCBS algorithm examines all the frames within the movie and prefetches any of the frames that would have otherwise violated the rate constraint. As a result, the RCBS algorithm *only* smooths the bandwidth requirements when the rate constraint would have otherwise been violated. Because of this property, the RCBS algorithm results in many more changes in bandwidth requirements than the other river-charting bandwidth smoothing algorithms. A sample RCBS plan similar to those shown in Figure 4 is shown in Figure 3. As shown by the figure, it appears as if a knife has been run over the frames of the movie from the right to left at the rate constraint, filling in valleys with frames that violate the rate constraint. As shown in Figure 4(b), the buffer residency times are much smaller than in those found in Figure 4, particularly in regions where the rate constraint is not violated (ex. time 0-10 minutes).

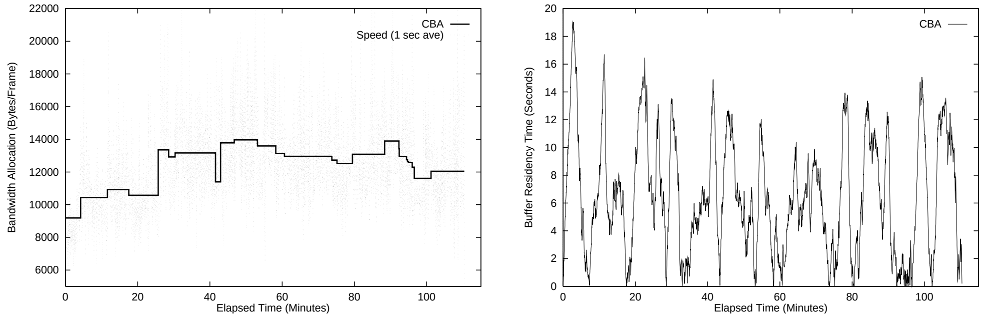
2.2 Delivering Video in Video-on-Demand Systems

For supporting the playback of stored video streams, it is commonly agreed that several levels of interactivity must be provided for in interactive video-on-demand systems[20]. For our purposes, we expect there to be at least three levels of service: Strict Playback, Quasi-Interactive Playback, and Interactive Playback. To understand why, we will discuss the necessary support for these three methods

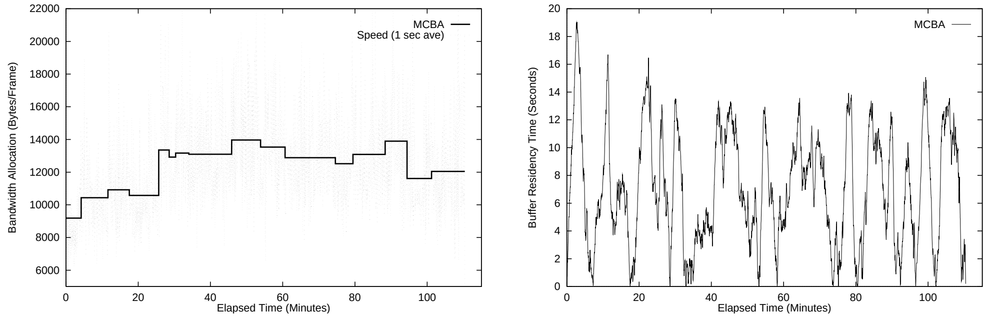
In the *strict playback* mode, the user is forced to watch the video from the beginning to end without any change in the consumption rate, allowing the network and server resources to be more efficiently allocated. As a result, the server is free to schedule bandwidth as tightly as possible by using peaks in bandwidth requirements of some plans to fill in valleys of other bandwidth plans.

In the *quasi-interactive playback* mode, the user is allowed to have limited VCR interactions. In this mode, a method called the *VCR window* can be used to allow users limited access to their videos[10]. This is based upon the observation that the common VCR functions rewind, examine, stop, and pause can be handled by the client-side buffer *without* requiring additional bandwidth from the network and server to service. The *VCR window* can be appended with additional *VCR buffering* in order to provide a larger VCR window for the user. Because the user can change its consumption rate, the streams cannot be scheduled as tightly as with in the strict playback case. However, in a system where the users abide by the *VCR window* the network and server resources can be allocated based on the peak bandwidth requirement *without* worrying about the user issuing a bandwidth request above and beyond what has been reserved.

Critical Bandwidth Allocation Algorithm



Minimum Changes Bandwidth Allocation Algorithm



Minimum Variability Bandwidth Allocation Algorithm

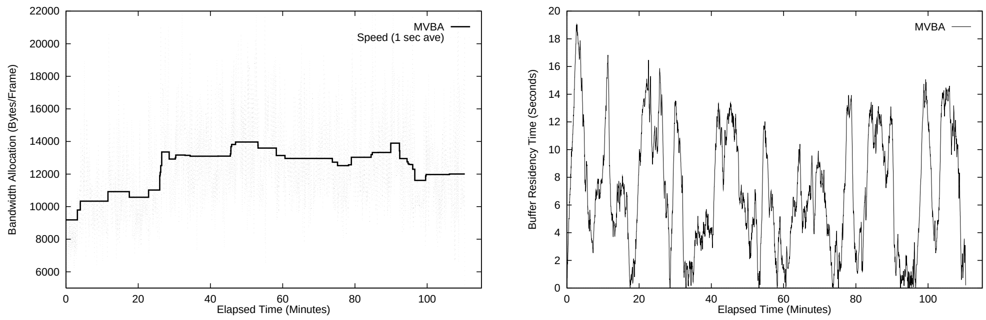


FIGURE 3: Bandwidth Smoothing Plans: This figure shows example bandwidth smoothing plans for the *critical bandwidth allocation algorithm,* the *minimum changes algorithm,* and the *minimum variability bandwidth smoothing algorithms* for the Motion-JPEG compressed movie *Speed* using a 5 megabyte buffer. The right figures show the buffer residency times that these algorithms use to achieve their smoothing.

In the *interactive playback* mode, the user is allowed to freely play, browse, fast-forward, and rewind through the video. Due to the non-deterministic playback rates, efficient allocation of resources is more difficult. As an illustration, consider the bandwidth smoothing plan in Figure 5. When a random access is made from VCR functions such as a long fast-forward, excess channel capacity may have to be allocated in order to resynchronize the plan with the original peak bandwidth requirement. In this example, the server needs to send $data_{resync}$ bytes to make up the data that the client would have had during normal playback. In addition, enough resources need to be allocated to avoid buffer underflow. A sample plan for resynchronization is shown by the dotted line in Figure 5. As another illustration, consider a scan into an area that has a large number of large frame sizes. Under bandwidth smoothing these frames would have been prefetched in order to reduce the bandwidth requirement. However, a random access to these frames will require that (1) excess channel capacity be allocated, (2) reducing the quality of video until the plans are resynchronized, or (3) making the user wait until the buffer is filled. Due to the undeterministic nature of the interactions, providing guaranteed VCR interactivity can be difficult while maintaining a high network utilization. To aid in VCR functionality, ideas such as *contingency channels* can be useful[4], where excess channel capacity is allocated for temporary allocation to VCR functionality. In addition, using a bandwidth algorithm that minimizes both the rate-constraint and buffer residency times can be useful by reducing the amount of data required on a resynchronization[13].

3. Time Constrained Bandwidth Smoothing

For the delivery of video in interactive VOD systems, we would like to have the properties that the MCBA, MVBA, and

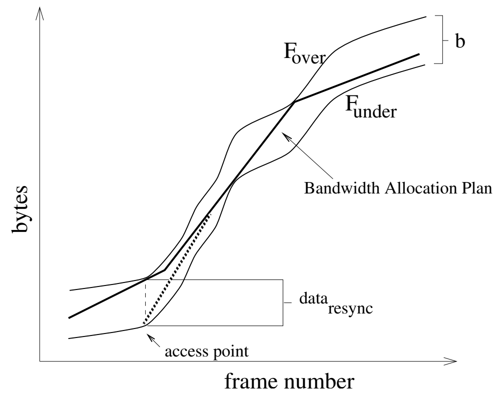


FIGURE 5: Supporting VCR Functionality: This figure shows the result of a scan to a random "access point". With no overlap of data in the buffer, the distance between the bandwidth smoothing plan and F_{under} at the access point must be made up in order to continue along the original bandwidth plan. The heavy dotted line shows a sample plan for resynchronizing to the original plan.

```
max_del = maximum time constraint (in frames)
buff_size = client buffer size in bytes;

for (i=0; i<N ;i ++)
   F_under(i) = summation frames 0 to i

for (i=0; i<N ; i++)
   if (F_under[i]+buff_size < F_under[i+max_del])
       F_over[i] = F_under[i]+buff_size;
   else
       F_over[i] = F_under[i+max_del];

run bandwidth smoothing algorithm using F_under
    and F_over to get bandwidth plan.
```

FIGURE 6: Time-constrained bandwidth smoothing pseudo-code. This figure shows the pseudo-code for the algorithm. Details of the implementation and optimizations have been omitted.

CBA smoothing algorithms offer (namely the relatively few bandwidth changes), but we would also like to set a maximum buffer residency time in order to limit the resychronization data required as in the RCBS algorithm. While the RCBS algorithm minimizes the buffer residency times, it results in a plan that can exhibit significant burstiness. For interactive video-on-demand systems, some delay may be acceptable to the user, allowing more of the burstiness to be removed. In this section, we propose a *time constrained bandwidth smoothing* algorithm that limits the buffer residency times to a user defined limit of *t* frames (or seconds).

For the time-constrained bandwidth smoothing algorithm (TCBA), we use the function $F_{under}()$ as the same function for the algorithms in Section 2. For the $F_{over}()$ curve, however, we use the following function to determine each point on the $F_{over}()$ curve:

$$F_{over}(j) = min\{F_{under}(j) + b, F_{under}(j+t)\} \qquad (1 \le j \le N)$$

where b is the buffer size in bytes, t is the time constraint in number of frames, and N is the number of frames in the video. Note, we assume that:

$$F_{under}(k) = F_{under}(N) \qquad (k > N)$$

where N is the number of frames in the movie. The pseudo-code for the time-constrained bandwidth smoothing algorithm is shown in Figure 6.

To graphically understand this procedure, Figure 7 shows the creation of a time-constrained bandwidth smoothing plan using the F_{under} and F_{over} curves. To create the bandwidth plan, each frame i is examined and the point $F_{over}()$ is calculated for that point. The figures (a) and (b) show the buffer requirement and time requirement calculations, respectively. The buffer residency constraint is vertically equidistant from $F_{under}()$ while the time constraint is horizontally equidistant from $F_{under}()$. As shown in Figure 7(c), the new $F_{over}()$ results in areas determined by the time constraint t as well as the buffer constraint b. In particular, in regions of very small frame sizes, as in the left side of Figure 7(c), $F_{over}()$ is deter-

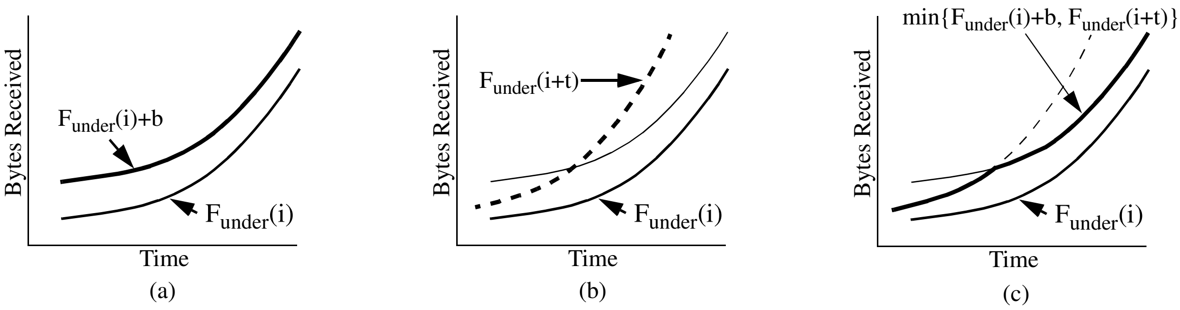


FIGURE 7: Time Constrained Bandwidth Smoothing Example. This figure shows how the function $F_{over}()$ is calculated for the time-constrained bandwidth smoothing algorithm for a buffer of size *b* bytes and a maximum time constraint of *t* frames. Figure (a) shows the curves that are used in the traditional bandwidth smoothing algorithms. Figure (b) shows the curve that is required to maintain the time constraint *t* frames. Figure (c) show the final function $F_{over}()$ that is used for the calculation of the bandwidth plan (heavy solid line). The adjustment of $F_{over}()$ guarantees that the prefetch buffer at frame *i* does not prefetch too much to violate the time constraint.

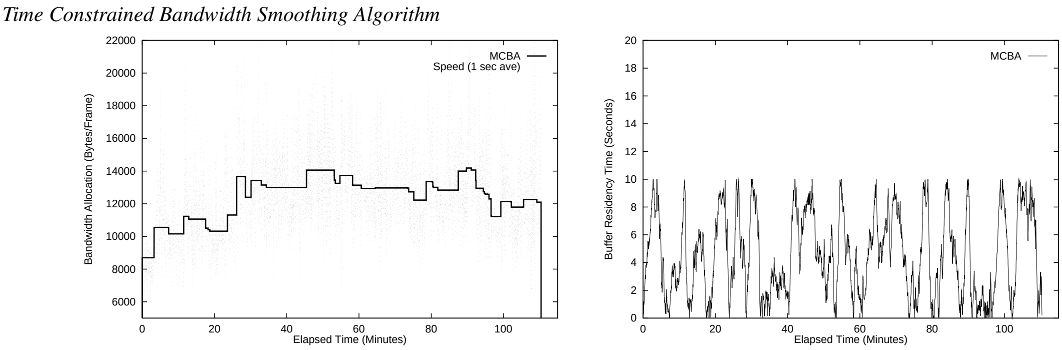


FIGURE 8: Time Constrained Bandwidth Smoothing Example - The left figure shows the *time-constrained* version for the *critical bandwidth allocation* algorithm for the movie *Speed* using a 5 megabyte buffer and a time constraint of 10 seconds. The right figure shows the buffer residency times for the same algorithm. Note how the buffer residency times were modified over those in Figure 4.

mined by the time constraint *t*. In regions of large frames, $F_{over}()$ is determined by the buffer constraint *b*.

Once the curve $F_{over}()$ has been determined, *any* of the river-traversing bandwidth techniques can be applied to generate the bandwidth allocation plan used for playback. As an example, the *minimum changes bandwidth smoothing* algorithm can be used to generate the bandwidth plan with the new $F_{under}()$ and $F_{over}()$ curves. For a maximum time constraint *t* and a maximum buffer size *b*, the resulting plan then has the smallest peak bandwidth requirement and the smallest number of changes in bandwidth that meets *both* the time constraint and buffer constraint. Similarly, the *minimum variability bandwidth smoothing* algorithm may be used to create a plan that has the minimum peak bandwidth requirement and smallest variability of bandwidth requests for plans that meet the time and buffer constraints. A sample time-constrained bandwidth plan is shown in Figure 7, for the *minimum increases bandwidth algorithm*, a 5 megabyte smoothing buffer, and a time constraint of 300 frames. Compared with the bandwidth smoothing algorithm presented in the background section, the time-constrained bandwidth smoothing algorithm results in a small increase in the peak bandwidth requirement from 13968 bytes/frame to 14186 bytes/frame. The maximum buffer residency time, however, has been limited to 300 frames (10 seconds).

At this point, several interesting points are worth discussing. Without the buffer time constraint *t*, it is still possible to enforce the buffer time constraint by adjusting the size of the buffer *b*, resulting in a $F_{over}()$ curve as with the traditional bandwidth smoothing techniques. However, this results in a time constraint that is determined by a few regions within the movie. Second, as shown by the buffer residency figures in Figure 4, the largest buffer residency requirements can occur at times where the bandwidth requirement is actually at its minimum (E.g. 0 to 10 minutes). As a result, using the time-constraint can minimize the prefetched data during non-peak bandwidth requirement segments as well. Third, the time constraint *t* does not necessarily imply that the user will have to wait *t* time units for every VCR function. The time constraint, rather, is a worst case delay when the network and server cannot allocate any additional resources to the client and where the buffer residency times happen to be at the maximum time constraint *t*. Finally, very large values of *t* will result in plans that are determined completely by the buffer constraint *b*, while very small values of *t* will result in a buffer that never fills completely due to the timing constraint.

```
max_del = maximum time constraint (in frames)
buff_size = client buffer size in bytes;
for (i=0; i<N ;i ++)
   F_under(i) = summation frames 0 to i

for (i=0; i<N ; i++)
   if (F_under[i]+buff_size < F_under[i+max_del])
       F_over[i] = F_under[i]+buff_size;
   else
       F_over[i] = F_under[i+max_del];

Determine minimum peak bandwidth req. based on buffer size b
Calculate Rate-constrained bandwidth plan at minimum peak bandwidth

For each region in the Rate-constrained bandwidth plan in
   which all frames do not violate the time constraint
          run bandwidth smoothing algorithm using F_under and F_over
          to get bandwidth plan for the region.
```

FIGURE 9: Pseudo-code for Rate/Time Constrained Bandwidth Smoothing Algorithm. Details of exact implementation have been omitted for brevity.

4. Rate/Time Constrained Bandwidth Smoothing

Using the time-constrained bandwidth smoothing algorithm results in plans for the delivery of data that adhere to both a buffer constraint *b* and a time constraint *t*. For a given buffer size *b* and the time-constraint *t*, an increase in the peak bandwidth requirement may be required for continuous delivery over a plan determined by only the buffer size *b*. For smaller buffer sizes, as in our examples, the increase in the peak bandwidth requirement may not be that large. In other cases, however, a larger peak bandwidth requirement may be required. As a result, the user may want the bandwidth smoothing algorithm to both minimize the peak bandwidth requirement as well as adhere to the time-constraint *during non-peak bandwidth* segments. That is, the time-constraint is relaxed only during the peak bandwidth allocation segments.

Given a buffer size *b,* a time-constraint *t*, and a rate-constraint *r,* the *rate/time constrained bandwidth smoothing* (RTCBA) algorithm first adheres to the rate-constraint *r*, assuming that the rate-constraint is achievable with the given size buffer *b*. It accomplishes this by first using the RCBS algorithm to rate-constrain the frames in the movie. It then applies the time-constrained algorithm to the resultant plan. Two methods can be used to determine the rate-constraint *r*. First, the *r* can be determined by running one of the river-charting bandwidth smoothing algorithms from Section 2.1 using the buffer constraint *b*, resulting in the minimum peak bandwidth requirement for continuous delivery. Second, the rate constraint can be based on other factors such as the cost for peak bandwidth. To check whether or not the rate constraint is feasible with the buffer size *b*, the RCBS algorithm can be run to verify it the rate is possible.

To create the rate/time constrained bandwidth smoothing algorithm, a bandwidth plan is created using the O(n) RCBS algorithm that adheres to the rate-constraint *r*. The resulting plan is then examined to find regions in which the time-constraint *t* is violated. Note, the time constraint can only be violated in regions that use the peak bandwidth requirement *r*. If the time-constraint is violated, the section of the video is then marked as *untouchable*. Finally, the time-constrained bandwidth smoothing algorithm is run on the segments that are not marked *untouchable*. The pseudo-code for the algorithm is shown in Figure 9. A sample rate constrained bandwidth smoothing plan is shown in Figure 10. In this example, the regions around 43 minutes and 90 minutes are allowed to violate the time constraint of 10 seconds in order to minimize the peak bandwidth requirement for the 5 megabyte smoothing buffer.

5. Evaluation

To evaluate the time-constrained bandwidth smoothing algorithms, we selected the Motion-JPEG encoded movie *Speed* and the MPEG encoded movie *Star Wars*. We selected two different types of encodings to highlight the effect that the encoding has on the time-constrained algorithms. In particular, the Motion-JPEG compression standard has each frame compressed independently, resulting in a bit stream that does not take advantage of inter-frame redundancy. The MPEG video clip has three frame types that allow for greater compression ratios to be achieved. The important point here is that the time-constraint is somewhat correlated to the bit-rate of the compressed video stream, assuming a fixed size buffer. For these video streams, the video *Speed* has an average bit rate of 3 megabits per second while the video *Star Wars* has an average bit rate of approximately 500 kilobits per second.

For the delivery of the compressed video stream, several measures are important for the end-to-end resource guarantees. Among these are the peak bandwidth requirement, the number of bandwidth changes required, the variability of the bandwidth requests, and the average amount of data in the

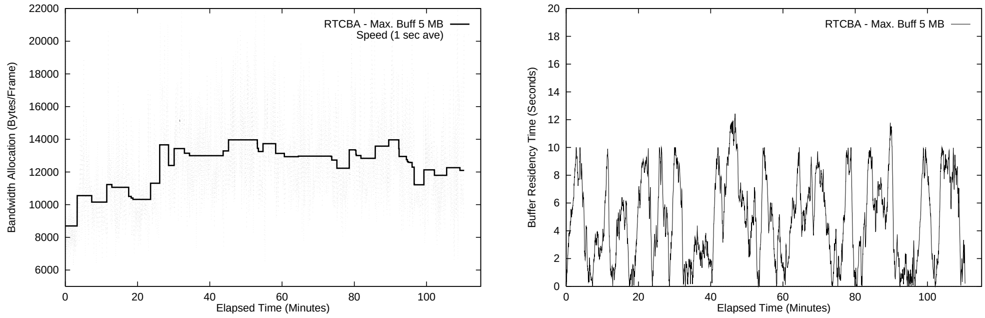


FIGURE 10: Rate/Time Constrained Bandwidth Smoothing Example - The left figure shows the *rate/time-constrained* version for the *critical bandwidth allocation* algorithm for the movie *Speed* using a 5 megabyte buffer and a time constraint of 10 seconds. The right figure shows the buffer residency times for the same algorithm. Note how the buffer residency times were modified over those in Figure 4.

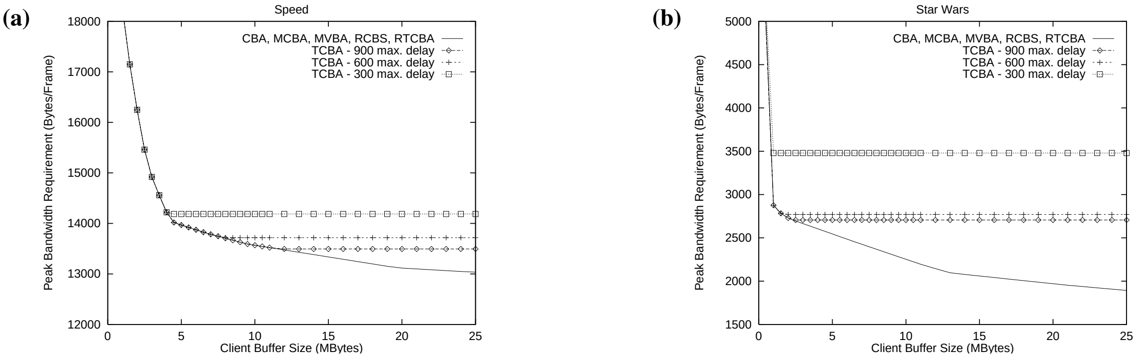


FIGURE 11: Peak Bandwidth Requirements - This figure shows the peak bandwidth requirements that are required for the various algorithms and the compressed video streams *Speed* and *Star Wars*.

buffer (its utilization). In the comparison of the various algorithms, we have used the RCBS, MCBA, and MVBA algorithms. The results for the CBA algorithm (minimum increases algorithm) always falls in between the MVBA and the MCBA algorithm. Finally, for the TCBA and RTCBA algorithms, we have used the MCBA algorithm to create the actual bandwidth allocation plans once the time-constraint process has been run. Recall, the TCBA and RTCBA are multi-step algorithms that first determine F_{under} and F_{over} based on the time constraint and then use any of the river-charting bandwidth plans to create the actual bandwidth allocation plan.

5.1 Peak Bandwidth Requirements

For VOD systems that allocate resources based on the peak bandwidth requirements, minimizing the peak bandwidth requirement can increase the likelihood that the server and network have sufficient resources to handle the stream. In addition, a low peak rate may reduce the total cost of the data transfer. In Figure 11, we have graphed the peak bandwidth requirements for the movies *Speed* and *Star Wars* for the various bandwidth smoothing algorithms TCBA, RTCBA, MCBA, MVBA, and RCBS. As shown in Figure 11 (a), the MCBA, MVBA, RCBS, and RTCBA algorithms results in the same *minimum* peak bandwidth requirement given the fixed client-side buffer size. Given a time constraint *t* and the client side buffer size, the TCBA algorithm, results in plans that are buffer constrained for small buffer sizes and time constrained for larger buffer sizes, as expected. As an example, consider the time constraint *t=300* frames. For the movie *Speed* and buffer sizes less than 4 megabytes, the buffer size limits the amount of data that can be prefetched, resulting in the same peak bandwidth requirements as using just the buffer size. Note, for buffer sizes less than 4 megabytes, there may be times when the time constraint is indeed violated, but it does not occur during the run(s) which force the minimum peak bandwidth requirement. For buffer sizes larger than 4 megabytes, the client-side buffer is large enough that the time constraint can be violated more often. Because the TCBA algorithm enforces this time constraint for all frames within the movie, the peak bandwidth curve is flat for all buffer sizes greater than 4 megabytes. By increasing the time constraint *t*, the point at which the TCBA algorithm switches from the buffer constraint to the time constraint also increase (although not linearly) as shown by the figure.

For the movie *Star Wars*, the TCBA algorithm results in buffer constraint to time constraint cross-over points that are

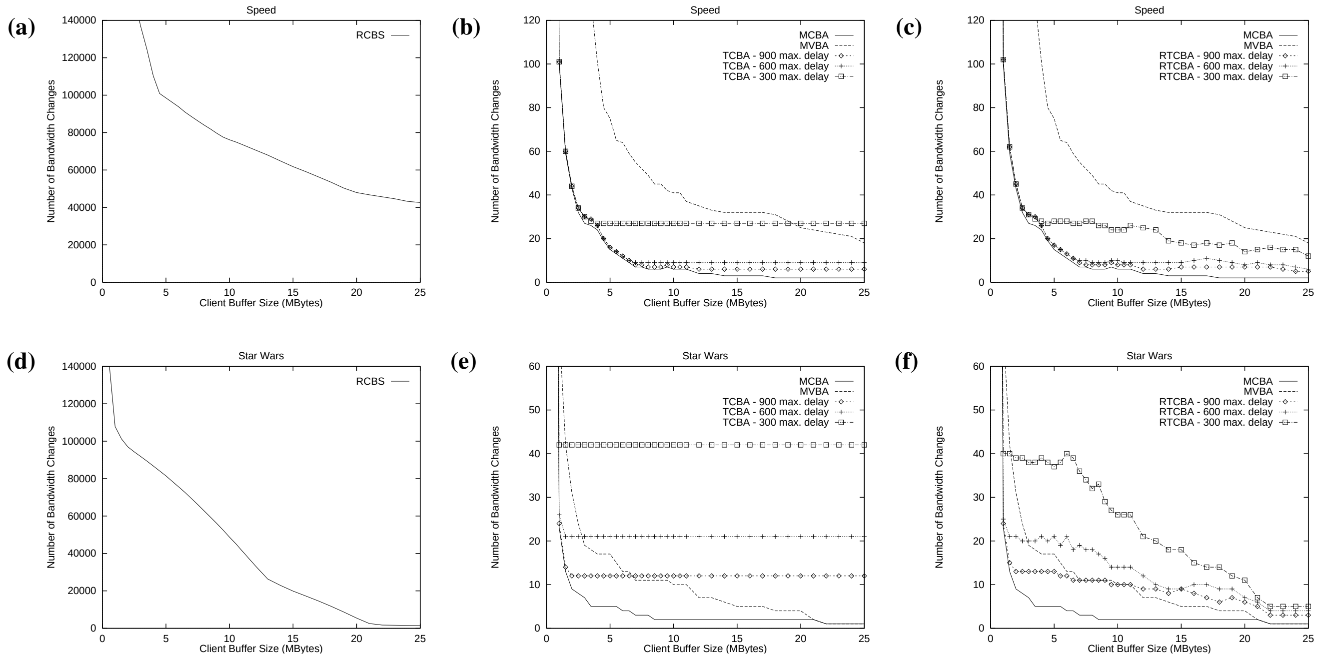


FIGURE 12: Number of Bandwidth Changes - This figure shows the number of bandwidth changes that the various algorithms require for the compressed video streams *Speed* and *Star Wars*.

smaller than in the movie *Speed* (in terms of the buffer size). The main reason for this is that with smaller frame sizes, a given buffer size can, on average, hold more frames. This results in time constraints that are violated with smaller buffer sizes. Finally, we see in Figure 11 (b) that the asymptotic value for the 600 and 900 frame time constraints are much closer to each other than the 300 frame time constraint. Once the short term burstiness has been removed (as in the MPEG frame patterns), the time-constraint becomes more critical in the determination of the peak bandwidth requirement.

5.2 Number of Bandwidth Changes

Minimizing the number of bandwidth changes that a stream requires can reduce the overhead that is involved in handling a video stream. For example, using a 5 megabyte buffer and the minimum changes bandwidth allocation algorithm for the delivery of the Motion-JPEG compressed video *Speed* results in a plan for the delivery of the video that has only 12 changes in bandwidth over the 2 hour duration of the movie. As a result, the network and server resources can be allocated approximately every 10 minutes before a change in bandwidth is required. Figure 12 shows the required bandwidth changes for the various bandwidth smoothing algorithms.

As shown in Figure 12 (a) and (d), we see the main drawback of the *rate-constrained bandwidth smoothing* (RCBS) algorithm. For the movie *Speed*, the RCBS algorithm requires more than 3 orders of magnitude more bandwidth changes than the other river-charting bandwidth plans. For the movie *Star Wars*, the RCBS algorithm requires nearly 4 orders of magnitude more bandwidth changes. In general, the RCBS algorithm requires a bandwidth change per frame for approximately 75% of the frames within the movie for small buffer sizes. In comparing the RCBS algorithm for the two movies, we see that the movie *Star Wars* has a lower asymptotic value than the movie *Speed.* The main reason for this is that much of the burstiness (due to the frame patterns in MPEG) are removed with small buffers. In addition, because *Star Wars* is smaller on average, the large buffer sizes can smooth much more of the data.

The number of changes required by the TCBA algorithm (using the minimum changes bandwidth allocation algorithm to create the bandwidth plan) are shown in Figure 12 (b) and (e). In Figure 12 (b), we see that the TCBA algorithm results in a graph that is similar to its peak bandwidth requirement graph. That is, for smaller buffer sizes, the graph is determined primarily by the buffer constraint, while for larger buffer sizes, the graph is determined primarily by the time constraint. Figure (b) also shows the main advantage of using the TCBA algorithm over the RCBS algorithm; it results in a total number of bandwidth changes that is at least the same order of magnitude as the MCBA algorithm (although slightly higher).

The number of changes required by the RTCBA algorithm (using the minimum changes bandwidth allocation algorithm to create the bandwidth plan) are shown in Figure 12 (c) and (f). Here, we see that the RTCBA algorithm has fewer bandwidth changes than the TCBA algorithm, mainly due to the time-constraint being relaxed to meet the minimum peak bandwidth requirements. Figure 12 (f) shows that the RTCBA algorithm reduces the number of bandwidth changes at a

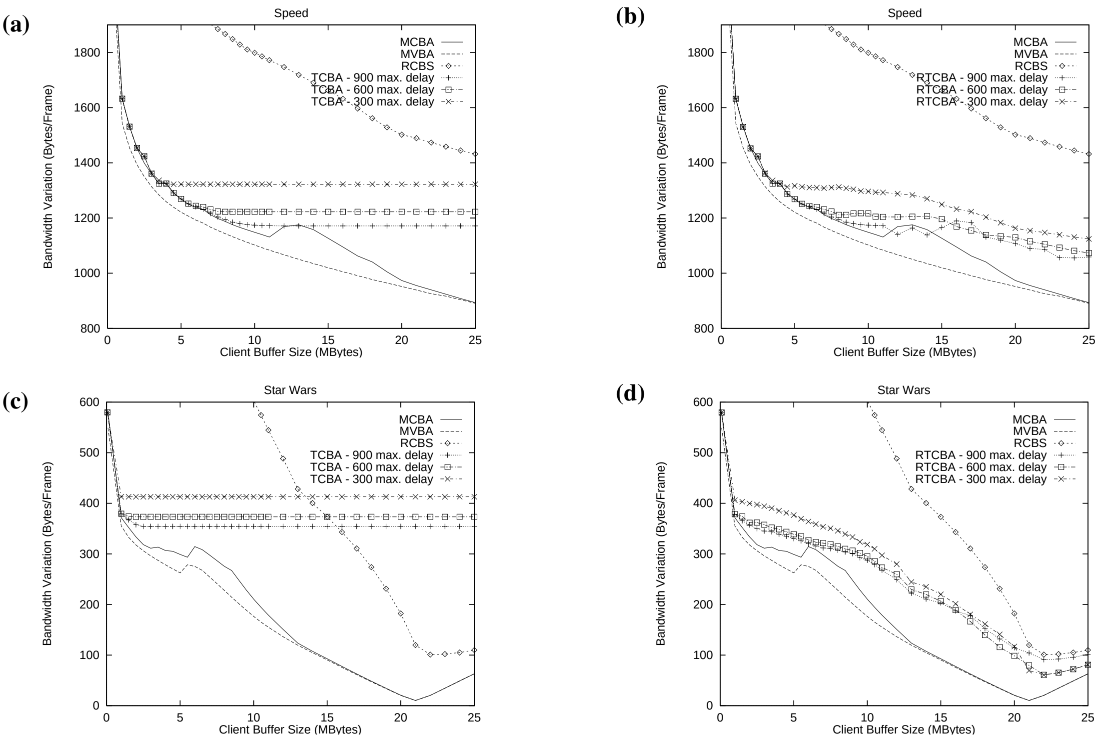


FIGURE 13: Bandwidth Variation - This figure shows the variation in bandwidth that each of the various algorithms require for the compressed video streams *Speed* and *Star Wars*.

quicker rate for the movie *Star Wars* than for the movie *Speed* (E.g. time constraint 900 in figure (f)). This is again due to the smaller frame sizes in the *Star Wars* video, resulting in larger regions that are at the minimum peak bandwidth requirement than in the movie *Speed* for a given buffer size.

5.3 Bandwidth Variation

In Figure 13, we have graphed the bandwidth variability exhibited by the various smoothing algorithms for the movies *Speed* and *Star Wars.* The bandwidth variability is the standard deviation of the rate requests on a per frame basis. As shown in Figure 13, we see that another advantage of using the time-constrained bandwidth smoothing algorithms over the RCBS algorithm is that the time-constrained algorithms result in similar bandwidth variability as the minimum variability bandwidth algorithm (MVBA). For the TCBA algorithms, the asymptotic values are horizontal as in the other performance metrics. Again, this is due to the fact that once the time-constraint is reached, adding more buffer does not change the bandwidth plan, resulting in the same variability measurements.

5.4 Buffer Utilization

The RCBS algorithm was introduced to minimize the buffer residency requirements for the delivery of stored video. As a result, it is not particularly well suited for reducing the total number of rate changes or reducing the variability of bandwidth requirement of the network. In Figure 14, we have graphed the buffer utilizations for the various bandwidth smoothing algorithms. Here we see that the bandwidth smoothing algorithms that minimized the variability or minimized the number of bandwidth changes have the largest buffer utilization measurements. In addition, we see that the RCBS algorithm has the smallest buffer utilization measurements, as expected. The TCBA algorithm has buffer utilizations that approach 0 for large buffer sizes, however, this is due to the fact that it has much higher peak bandwidth requirements in these areas, and hence, prefetches much less data. By using the RTCBA algorithm (and hence the same minimum peak bandwidth requirement), we see that the time-constrained bandwidth smoothing algorithms are in between the RCBS algorithm and the MCBA and MVBA algorithms. In particular, consider the graph shown in Figure 14 (d), we see that for buffer sizes in the range of 4-10 megabytes that the buffer utilizations of the RTCBA algorithms approach that of the RCBS algorithm, while still achieving similar number of bandwidth changes of the MCBA bandwidth smoothing algorithm. In general, as the time constraint is decreased for the RTCBA algorithm, the bandwidth plan used will continue to approach the RCBS algorithm. Using large time constraints for the RTCBA algorithm, the bandwidth plan used will approach the minimum changes or minimum variability algorithms.

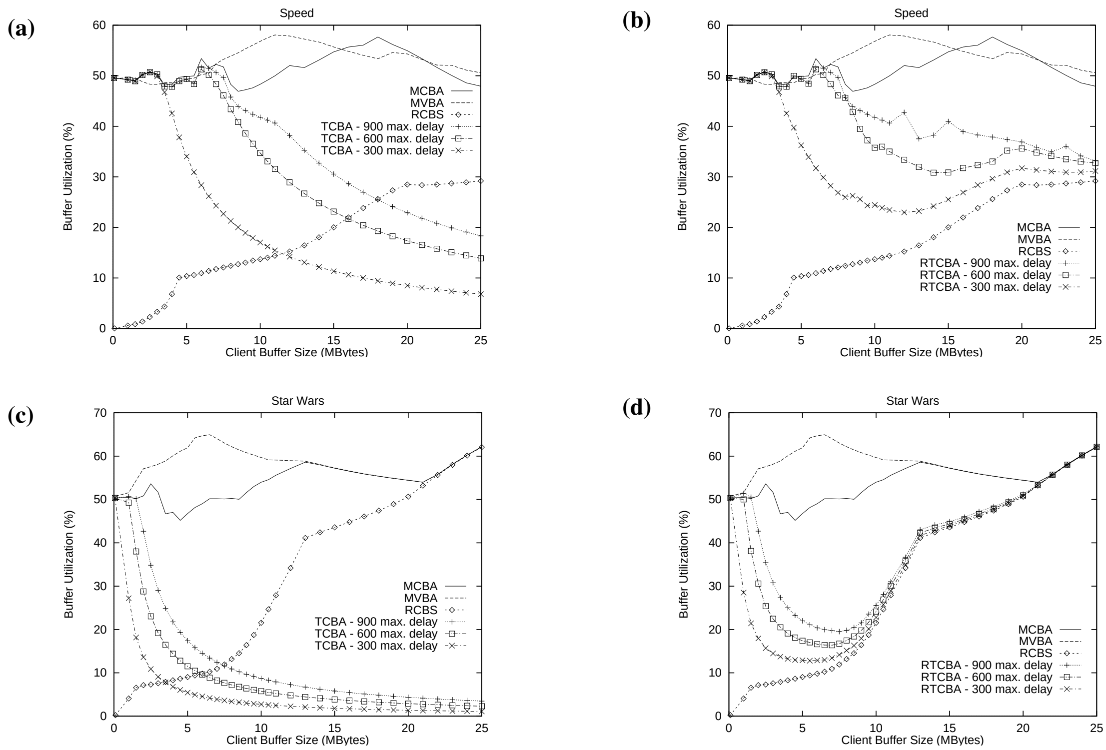


FIGURE 14: Buffer Utilization - This figure shows the buffer utilization for the various algorithms for the compressed video streams *Speed* and *Star Wars*.

6. Conclusion

In this paper, we have introduced the notion of *time-constrained* bandwidth smoothing. The *time-constrained bandwidth smoothing* algorithm allows the user to specify a maximum time constraint *t*, for which a frame can sit within the client-side buffer during continuous playback. By adjusting the calculation of the bandwidth plans, any of the well-known bandwidth smoothing techniques can be used to create plans that adhere to time and buffer requirement. For smaller values of *t*, the peak bandwidth requirement is typically defined by a single region within the video. To allow for more flexibility, we have also introduced a *rate/time constrained bandwidth smoothing* algorithm that, given a fixed size client buffer, determines a plan that has the minimal peak bandwidth requirement and that adheres to the time constraint *t*. This algorithm then results in areas that may violate the time constraint which are allocated at the maximum rate constraint *r*. By using the time constraint *t*, a bandwidth plan can be created that takes advantage of the properties that the RCBS, MCBA, MVBA, and CBA algorithms have to offer, namely, reducing the number of rate changes while keeping the buffer residency times small (as in the RCBS algorithm). As our results have shown, the time constrained bandwidth smoothing algorithms effectively balance the trade-off between smoothing (reducing the number of bandwidth changes) and buffer utilization (reducing the buffer residency times) for interactive playback of stored video.

REFERENCES

[1] D. Anderson, Y. Osawa, R. Govindan, “A File System for Continuous Media”, *ACM Transactions on Computer Systems*, Vol. 10, No. 4, Nov, 1992, pp. 311-337.

[2] C.M. Aras, J.F. Kurose, D.S. Reeves, H. Schulzrinne, “Real-time Communication in Packet Switched Networks”, *Proceedings of the IEEE*, Vol. 82, No. 1, pp. 122-139, Jan. 1994.

[3] M. Chen, D. Kandlur, P. Yu, “Support for Fully Interactive Playout in a Disk-Array-Based Video Server”, *Proc. of ACM Multimedia 1994*, San Francisco, CA, Oct. 1994, pp. 391-398.

[4] A. Dan, D. Sitaram, P. Shahabuddin, “Scheduling Policies for an On-Demand Video Server with Batching”, *Proc. of ACM Multimedia 1994*, San Francisco, CA, Oct. 1994, pp. 15-23.

[5] J. Dey-Sircar, J. Salehi, J. Kurose, D. Towsley, “Providing VCR Capabilities in Large-Scale Video Servers”, *Proc. of ACM Multimedia 1994*, San Francisco, CA, Oct. 1994, pp. 25-32.

[6] C. Federighi, L. Rowe, “A Distributed Hierarchical Storage Manager for a Video-on-Demand System”, *Proc. of 1994 IS&T/SPIE Symposium on Electronic Imaging: Science and Technology*, San Jose, CA Feb. 1994.

[7] W. Feng, S. Sechrest, “Smoothing and Buffering for Delivery of Prerecorded Compressed Video”, *Proc. of IS&T/SPIE Multimedia Computing and Networking* , Feb. 1995, San Jose, CA, pp. 234-242.

[8] W. Feng, S. Sechrest, “Critical Bandwidth Allocation for the Delivery of Compressed Prerecorded Video”, *Computer Communications*, Vol. 18, No. 10, Oct. 1995, pp. 709-717.

[9] W. Feng, F. Jahanian, S. Sechrest, “Optimal Buffering for the Delivery of Compressed Prerecorded Video”, *Proc. of IASTED International Conf. on Networks*, Jan. 1996, Orlando, Florida.

[10] W. Feng, F. Jahanian, S. Sechrest, “Providing VCR Functionality in a Constant Quality Video-On-Demand Transportation Service”, In *Proc. of 3rd IEEE Inter. Conf. on Multimedia Computing and Systems*, Hiroshima, Japan, June 1996.

[11] W. Feng, “Video-on-Demand Services: Efficient Transportation and Decompression of Variable Bit Rate Video”, Ph.D. Thesis, University of Michigan, April 1996.

[12] W. Feng, J. Rexford, “A Comparison of Bandwidth Smoothing Techniques for the Transmission of Prerecorded Compressed Video”, *Proc. IEEE INFOCOM 1997*, April 1997, Kobe, Japan.

[13] W. Feng, “Rate-Constrained Bandwidth Smoothing for the Delivery of Stored Video”, In *IS&T/SPIE Multimedia Computing and Networking Conf.*, San Jose, CA, Feb. 1996.

[14] D.J. Gemmell, H.M. Vin, D. Kandlur, P.V. Rangan, L.A. Rowe, “Multimedia Storage Servers: A Tutorial”, *IEEE Computer*, Vol. 28, No. 5, May 1995, pp. 40-49.

[15] Pawan Goyal, Harrick M. Vin, “Network Algorithms and Protocol for Multimedia Servers”, In *Proceedings of INFOCOM 1996*, San Francisco, CA, March 1996, pp. 1371-1379.

[16] M. Grossglauser, S. Keshav, and D. Tse, “RCBR: A simple and efficient service for multiple time-scale traffic,” In *Proceedings of ACM SIGCOMM*, pp. 219-230, August/September 1995.

[17] D. Kandlur, M. Chen, Z.Y. Shae, “Design of a Multimedia Storage Server” , In *IS&T/SPIE Symposium on Electronic Imaging Science and Technology*, San Jose, CA, Feb. 1994.

[18] D.J. LeGall, “A Video Compression Standard for Multimedia Applications,” *Communications of the ACM*, Vol. 34, No. 4, (Apr. 1991), pp. 46-58.

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

[20] T.D.C. Little, D. Venkatesh, “Prospects for Interactive Video-on-Demand”, *IEEE Multimedia*, Vol. 1, No. 3, Fall 1994, pp. 14-24.

[21] P. Lougher, D. Shepherd, “The Design of a Storage Sever for Continuous Media”, *The Computer Journal*, Vol. 36, No. 1, Feb. 1993, pp. 32-42.

[22] J. M. McManus, K. W. Ross, “Video on demand over ATM: Constant-rate transmission and transport”, *Proc. of IEEE INFOCOM*, pp. 1357-1362, March 1996.

[23] P. Venkat Rangan, H.M. Vin, “Designing File Systems for Digital Video and Audio”, In *Proceedings of the 13th ACM Symposium on Operating Systems Principles*, Operating Systems Review, Vol. 25, No. 5, October 1991, pp. 81-94.

[24] 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, December 1993.

[25] 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 *Proceedings of ACM SIGMETRICS*, pp. 222-231, May 1996.

[26] P. J. Shenoy, H. M. Vin, “Efficient Support for Scan Operations in Video Servers”, In P*roceedings of the 3rd ACM Conference on Multimedia*, October, 1995.

[27] H. Zhang, S. Keshav, “Comparison of Rate-Based Service Disciplines,” In *Proceedings of ACM SIGCOMM*, Sept. 1991, pp. 113-121.