A Storage and Retrieval Technique for Scalable Delivery of MPEG-Encoded Video[1]

H.J. Chen, T.D.C. Little, and D. Venkatesh

Multimedia Communications Laboratory
Department of Electrical, Computer and Systems Engineering
Boston University, Boston, Massachusetts 02215, USA
(617) 353-9877, (617) 353-6440 fax
{*huangjen,dinesha,tdcl*}*@bu.edu*

MCL Technical Report 03-22-1995

Abstract–Concurrent retrieval of continuous media from a physical storage device can be achieved by interleaving data and providing a suitable scheduling algorithm. Scheduling approaches that exploit gains from statistical multiplexing are susceptible to a non-zero probability of frame loss due to the variable-bit-rate characteristic of compressed video. With interframe encoding schemes (such as specified by the MPEG standard), the losses propagate, resulting in a net loss of frames that exceeds the fraction of missing data.

In this paper we describe a mechanism for the storage and retrieval of MPEG-encoded video from a single disk storage system. The scheme balances the need for the reliable delivery of MPEG frames with the desire to support the largest number of sessions. Our approach reorganizes the MPEG-encoded video stream based on the relative importance of the frames and maps them to the storage device geometry. The reorganization reduces the impact of frames lost due to missed deadlines and distributes the frame losses over time and among sessions. Simulation results show that the new approach improves performance when compared to conventional storage and scheduling schemes.

Keywords: Multimedia, physical data organization, scheduling, time-dependent audio and video data, secondary storage, performance modeling, priority retrieval, MPEG.

[1]In *Journal of Parallel and Distributed Computing* (Special Issue on Multimedia Processing and Technology), Vol. 30, No. 2, November 1995, pp. 180-189.

1 Introduction

Multimedia data differ significantly from conventional data types due to their strict timing and large bandwidth requirements. Table 1 illustrates some bandwidth and storage needs for several continuous-media data types. Typically, video data require more storage and communication bandwidth than text files do. When users request interactive on-demand sessions from a multimedia server, the system must ensure data delivery at the specified rate; the file system must ensure the availability of sufficient buffer space for the playback process. For example, to ensure the delivery of jitter-free NTSC-quality video, the file system must deliver data at a rate of 30 frame/s.

Data Type	**Bandwidth**	**Storage for 1 Hour**
Voice-quality audio (8 bits @ 8 KHz)	64 Kb/s	28.8 Mbyte
CD quality audio (stereo @ 44.1 KHz)	1.4 Mb/s	630 Mbyte
MPEG-I-compressed NTSC video	$\cong$ 1.5 Mb/s	675 Mbyte
MPEG-II-compressed video	$\leq$ 10 Mb/s	3,600 Mbyte

Table 1: Storage and Communication Requirements for Multimedia Data

Unlike traditional file systems, the goals for designing a multimedia system are not just the maximization of utilization and throughput, but also the maintenance of temporal relationships among multimedia data. However, a storage subsystem[2] accesses stored data by positioning its read heads at the desired location for a data block. A random allocation approach, regardless of the time-dependency for multimedia data, increases the disk head seek switching frequency and the resulting access latency. In addition, the electro-mechanical nature of secondary-storage devices requires the use of scheduling disciplines modified to meet the throughput and real-time requirements of multimedia data delivery [1]. Furthermore, it is desirable to design a multimedia-on-demand system that can support several concurrent interactive sessions from the same physical device.

The ability of the multimedia system to meet session timing requirements becomes difficult due to the unpredictable nature of disk seek latencies. Due to the time-varying characteristics of compressed video it is difficult to predict disk production and display consumption rates. Without proper scheduling, systems cannot support continuous playback for an unlimited number of sessions. The result is the inability to meet session deadlines and the subsequent loss of data. Scheduling is more critical for inter-frame encoding schemes such as

[2]In the rest of the paper we use the term "storage" to refer to an electro-mechanical storage device.

MPEG because even the loss of a small portion of data in retrieval can result in significant losses in the number of dropped frames at playout time.

In this paper we focus on data placement strategies within a single disk storage system and propose a new approach that improves a disk's ability to support multiple sessions. Our approach is suitable for video data compressed and stored using an inter-frame encoding scheme as specified by the MPEG (Motion Picture Experts Group) video compression standard. The proposed scheme employs a non-preemptive fixed-length-period flexible-service-time round-robin scheduling discipline to support multiple time-varying MPEG-compressed video sessions from a single physical device. This approach permits the disk to switch alternately between tasks to take advantage of the gains offered by statistical multiplexing.

The remainder of this paper is organized as follows. In Section 2 we investigate the characteristics of MPEG-compressed data including decoding dependence and time-varying behaviors. In Section 3 we develop a physical disk scheduling scheme which supports multiple real-time temporal data streams. In Section 4 we propose an alternate storage scheme for MPEG data and propose a priority access scheme for retrieving data. In Section 5 we evaluate the performance gains from our approach. We discuss related work in Section 7. Section 8 concludes the paper.

2 MPEG Video Characteristics

In order to understand the effects of MPEG-encoding on disk scheduling, we begin by examining inter-frame dependencies in an MPEG-compressed video stream. In MPEG compression, frames are typically arranged in fixed-size groups for the entire video stream [9]. Each group represents a logical entity for decoding purposes and are decompressed independently. Frames within a group are also considered logical units and depend on each other for decoding. Fig. 1 shows a sequence of MPEG encoded I, P and B frames for a group size of 10. As a result, losses are sensitive to the particular frame that is corrupted or dropped. For example, if an I frame at the head of a group is lost, all frames within its group must be discarded as they are dependent on the I frame for decoding. Therefore, frame losses in MPEG-encoded video are correlated within groups.

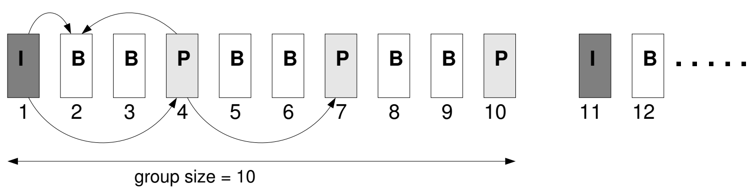


Figure 1: MPEG Frame Dependencies

2.1 Time-Varying MPEG Traffic

MPEG compression also yields video streams that exhibit highly time-varying bandwidth characteristics [7, 15]. In our example the video sources generate 30 frame/s with an image-aspect ratio of 320×240 pixels. The size (in bits) of a video frame depends on the compression algorithm and the activity within the video sequence. Fig. 2 shows the characteristics of video traffic for 1,500 frames sampled as groups containing 10 frames. The average rate μ_λ over all 1500 frames is 0.6369 Mb/s and the standard deviation σ_λ for the traffic rate is 0.2305 Mb/s with a sampling interval τ of 10 frames.

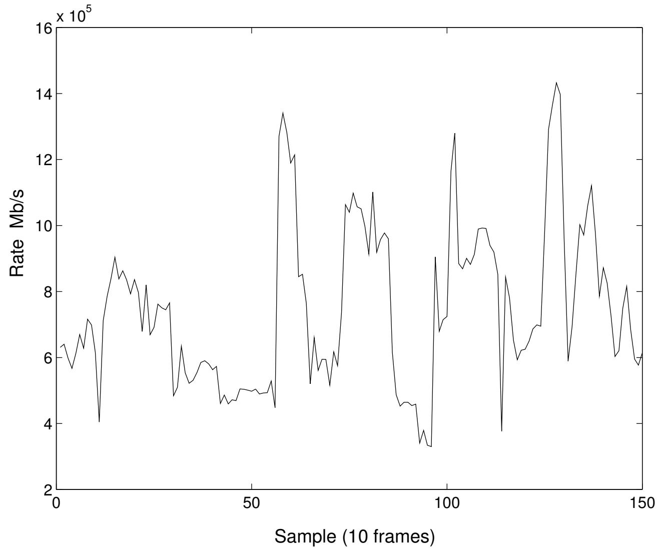


Figure 2: Traffic Rate for MPEG Video

The burstiness of MPEG-compressed traffic and interdependency among frames affect the ability of a disk to support concurrent access. This inter-frame dependency can cause a disk to perform poorly when it supports several concurrent sessions. In the next section, we examine the impact of disk scheduling and its effect on MPEG data.

3 Disk Access Scheduling and Bandwidth Requirements

In this section we describe the scheduling constraints for the acceptance of a set of multimedia sessions and the accompanying disk bandwidth requirements. Without any loss in generality, we choose the round-robin service scheme to support real-time multiple stream retrieval for a set of multimedia tasks as shown in Fig. 3.

For round-robin scheduling, we define a fixed length working period T_{period} during which the scheduler switches among all multimedia sessions [5]. It retrieves the exact amount of data necessary for each session to ensure continuous playout into the buffer. This keeps each session busy until the next retrieval period when the buffer is ready to be refilled. (The buffer management policies to ensure continuous playout are described elsewhere [8, 13].) If session k displays m frames per second, the file system must read exactly $m \times T_{period}$ frames for session k. Let $S(k)$ be the size of these $m \times T_{period}$ frames. If R is the entire disk I/O bandwidth available, each session k shares an interval $T(k)$ where $T(k) = \frac{S(k)}{R}$. As shown in Fig. 3, the total interval used for multimedia sessions plus the disk seek latency should be less than the working period T_{period} to accommodate the variances due to disk seek latencies and MPEG traffic. In other words, the preset period T_{period} must be greater than the time needed to transfer data from the disk for all sessions.

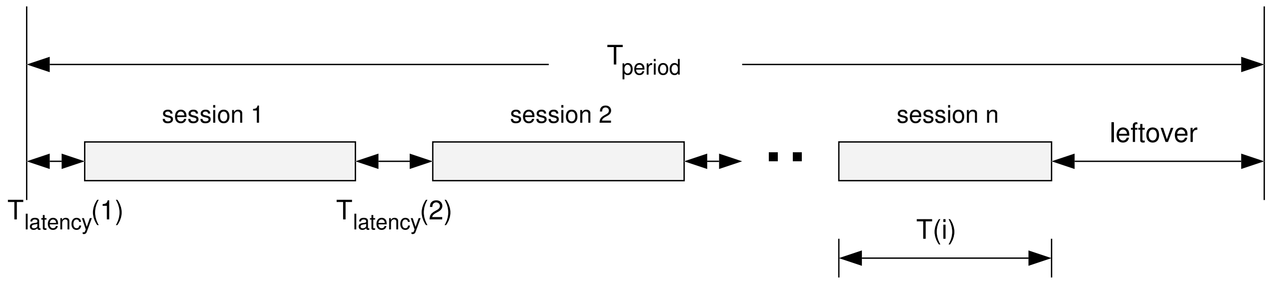


Figure 3: Round Robin Scheduling Model

To prevent starvation, the requirements for the length of T_{period} are [5]:

$$T_{period} \geq \sum_{1}^{n} \frac{S(k)}{R} + \sum_{1}^{n} T_{latency}(k). \qquad (1)$$

In Eq. 1, n is the number of concurrent sessions from the disk. $T_{latency}(k)$ is the switching latency from session k to $k+1$. We can rewrite Eq. 1 and derive the bandwidth requirement R for retrieving n simultaneous sessions of time-dependent data as [5]:

$$R \geq \frac{\sum_1^n S(k)}{T_{period} - \sum_1^n T_{latency}(k)}. \tag{2}$$

Eq. 2 represents an estimate of the sustained bandwidth the storage system must support in order to support the n sessions.

To guarantee the delivery of an MPEG stream without losses, the scheduling algorithm must reserve resources at the peak bandwidth. This approach is inefficient as it leads to wasted disk bandwidth. However, a scheduling approach that exploits gains from statistical multiplexing of sessions at the average bandwidth can suffer from missed deadlines and lost data. Because MPEG frame losses tend to propagate, the fraction of frames that cannot be decoded is greater than the fraction of lost data. These losses are estimated in the next section.

3.1 Frame Loss Estimation for MPEG Data

Consider an MPEG sequence in which frames are grouped based on a predetermined order. To simplify analysis, we assume that this group order is fixed for the stream under consideration. Let S_I, S_P, and S_B represent the average frame sizes and N_I, N_P, and N_B be the number of I, P, and B frames within a group, respectively. Also, the total number of frames in a group is

$$N = N_I + N_P + N_B$$

and the group size is given by

$$G = N_I S_I + N_P S_P + N_B S_B.$$

Losses within the MPEG bit stream due to missed deadlines and network errors are bursty by nature due to frame interdependency and the inability to exactly predict the beginning of a loss. Let k represent the size of a burst loss. The number of groups spanned by a burst is dependent on the first affected frame within the group in addition to the size of the burst.

The probability that a particular frame-type within a group is at the head of a burst depends on two factors: The frame type (I, B, or P) and its ordering within the group. From the discussion above, frame sizes within a group can be defined by a vector V_G,

$$V_G = [S_1, S_2, S_3, ..., S_N], \qquad S_j \in (S_I, S_P, S_B).$$

If we assume that the occurrence of losses cannot be predicted in advance, then the probability that a burst begins at frame j within the group can be specified as

$$p_j = \frac{S_j}{G}, \text{ where } \sum_{j=1}^{N} p_j = 1.$$

For a burst of size k beginning at frame j, the number of groups g_j spanned by the burst depends on the group size G. We derive the value of g_j to be

$$g_j = \left\lceil \frac{k - \sum_{t=j}^{N} V_G[t]}{G} + 1 \right\rceil. \tag{3}$$

In Eq. 3, the fraction denotes the number of groups spanned by the burst excluding the beginning group. The unitary term is added to account for the initial group affected by the burst.

From Eq. 3, the number of frames lost in the burst can be determined as

$$n_j = (g_j - 1)N + N - j. \tag{4}$$

For groups in which the leading I frame is lost, the entire group is unrecoverable, accounting for the first term in Eq. 4. The second term represents the frames lost within the group at the beginning of the burst. The average number of frames lost in burst k is then given by

$$\sum_{j=1}^{N} p_j n_j. \tag{5}$$

When the number of groups spanned by a burst $g_j \geq 2$, all leading frames in the subsequent burst-affected groups are lost. The frame loss value given in Eq. 4 is accurate to within a single frame. However, when $g_j = 1$, data loss takes place within a single group. Depending on the particular frame lost, some or all of the remaining data in the group can be recovered. For example, if an I frame is corrupted, all data within the group are lost. However, if data within a single B frame are corrupted, only one frame is lost and the remaining data can be decoded. The following analysis addresses this scenario and determines frame dependencies within a group and their effects on frame loss.

To account for intra-group losses we define a vector

$$D = [d_1, d_2, ...d_N]$$

where $d_j = 0$ if j is an I or a P frame. For a B frame, d_j equals the sum of the sizes of any immediately following B frames. For example, for the MPEG group shown in Fig. 1, d_2 is the size of the third frame and d_3 is 0. If we assume that all burst errors occur at the beginning of a frame boundary, the number of intra-group frames lost for bursts of size k starting at frame j is given by

$$n_j = \begin{cases} N - j + 1 & : \quad k > d_j \\ \left\lceil \frac{d_j}{k} \right\rceil & : \quad k \leq d_j \end{cases} \tag{6}$$

The first part of Eq. 6 represents the case when an I or P frame is corrupted, thereby making the entire group useless. The second part accounts for the case when only B frame is lost, and the remainder of the data in the group can be recovered. Eq. 6 can then be used to compute the average frame loss by substituting for n_j in Eq. 5. Our assumption simplifies the analysis and can yield reasonable results that help us understand the effects of data loss on an MPEG stream. In reality, the actual frame loss will be the weighted average of the values provided in Eqs. 4 and 6. However, this overly complicates analysis and is not considered further in this paper.

The aforementioned analysis demonstrates the disadvantage in organizing MPEG data in a contiguous fashion on a storage device. If the disk scheduler misses a deadline in the scheduling cycle, the number of lost frames is usually greater than the fraction of lost data. In the next section, we describe an MPEG stream reorganization and storage approach that reduces these effects of missed server deadlines.

4 Disk Storage Organization and Access Scheme

In this section, we propose a storage reorganization approach and access scheme that is adapted to MPEG stream dependencies and time-varying characteristics. The proposed mechanism provides a priority access mechanism that improves disk performance and minimizes the probability of frame loss.

4.1 Storage Pattern Reorganization

To improve disk performance and accommodate the MPEG decoding algorithm, we propose a fixed service period, T_{period} which is a multiple of the playout-time for an MPEG group. For example, for a group containing 10 frames (Fig. 1) and a video source that generates 30 frames per second, the length of T_{period} is a multiple of 10/30 second. The file system will read $30 \times T_{period}$ frames or $3 \times T_{period}$ MPEG groups per session.

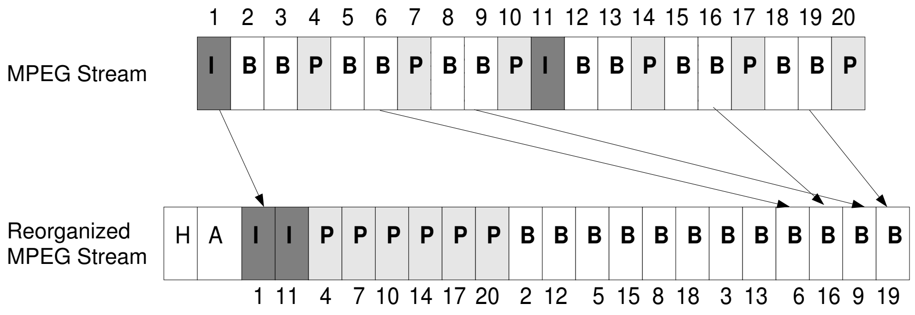


Figure 4: Storage Pattern for the Reorganized MPEG Frames

We can predefine a fixed length scheduling period T_{period} for a disk by analyzing the traffic characteristics of the supported video streams. [3] Once the appropriate period length is determined, we can define the number of frames or MPEG groups to be read every period. We reorganize the MPEG frame sequence as shown in Fig. 4. In the reorganized storage pattern, the most important data are placed at the beginning of the sequence. **A** denotes the audio data and **H** denotes a header containing information about the subsequent storage pattern (i.e., size of each frame). The reorganized pattern is then stored contiguously on the disk. The contiguous layout reduces seek latencies during continuous retrieval of video data. The header information helps the disk scheduler predict the access loads for subsequent

[3]This approach is reasonable for read-only VOD applications in which extensive preprocessing and analysis can be applied prior to rendering of storage organization

working periods and allows the scheduler to adapt quickly to any load changes and provide the best service.

4.2 Priority Access Scheme

To take advantage of the MPEG stream reorganization we propose a disk scheduling approach. As shown in Eq. 1, the fixed period T_{period} must be greater than the time necessary to transfer data for all sessions from the disk to prevent starvation. From Section 2 and Fig. 1 we know that the amount of data the must be read in T_{period}, $\sum_1^n S(i)$ is not constant. To ensure continuous playout, we can assign the peak rate for every session and guarantee that starvation will not occur. However, as shown in Fig. 1, the MPEG video traffic is highly time varying. In this example, the peak rate is almost 1.5 Mb/s but the average rate is only 0.6369 Mb/s. It is inefficient to assign the peak rate for every session due to the high peak/average ratio for bandwidth consumption.

The proposed file system can accept a set of sessions as long as the sum of their average rates does not exceed the data rate of the disk subject to the constraints of T_{period}. This means that several time varying MPEG video streams are multiplexed during the disk retrieval phase. However, occasionly the combined rates of the multiplexed streams can exceed the I/O rate of the disk. This results in some data starvation due to missed scheduler deadlines. If p is the maximum starvation frequency that can be tolerated, we have

$$Prob[(\sum_1^n \frac{S(i)}{R} + \sum_1^n T_{latency}(i)) > T_{period}] < p. \tag{7}$$

This means the probability that the preset period T_{period} is less than the time to transfer data from disk is less than p. $T_{latency}(i)$ represents the seek latency when the disk switches the service from session $i-1$ to session i. Because the subsequent data for session i can be placed anywhere on the disk, $T_{latency}$ is also a random variable. Chen and Little [5] derive the average seek latency and the variance of the seek latency as follows. Let $E(T_{latency})$ be the average seek latency, $\sigma^2_{latency}$ be the variance of seek latency, $E(S)$ be the average size of storage pattern, and σ^2_S be the variance of the size of storage pattern. The time to transfer data from the disk $T_{transfer}$ is then

$$T_{transfer} = \sum_1^n \frac{S(i)}{R} + \sum_1^n T_{latency}(i). \tag{8}$$

The mean $E(T_{transfer})$ and variance $\sigma^2(T_{transfer})$ of time $T_{transfer}$ can be derived as follows:

$$E(T_{transfer}) = n \times (\frac{E(S)}{R} + E(T_{latency})) \tag{9}$$

$$\sigma^2(T_{transfer}) = n \times (\frac{\sigma^2(S)}{R} + \sigma^2(T_{latency})) \tag{10}$$

From Eqs. 9 and 10, we know that $T_{transfer}$ is a sum of random variables. By the Central Limit Theorem, if the number of sessions n is large enough, we know that the distribution of $T_{transfer}$ can be approximated by a Normal distribution. We can plot the probability density of $T_{transfer}$ (Fig. 5) with a mean value $E(T_{transfer})$ and standard deviation $\sigma(T_{transfer})$. If the fixed length working period T_{period} and the number of session are given, we can predict the frequency of starvation p, as shown in Fig. 5.

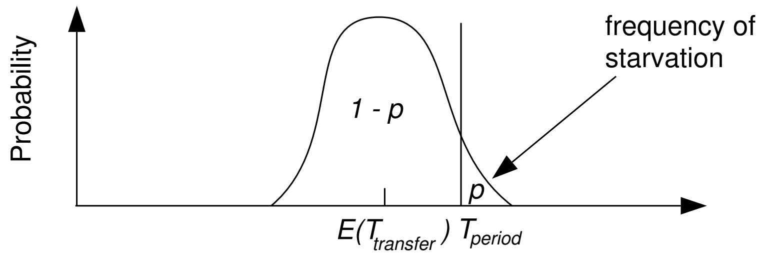


Figure 5: Distribution of $T_{transfer}$

4.3 Disk Scheduling in the Priority Access Scheme

As described in Section 4.1, we measure the size of every frame, encapsulate this information in a header and place it in the storage pattern for the previous group. With the reorganized storage layout, as soon as a period is complete the file system has sufficient information about the requirements for the next period. The file system can then calculate the total size of the data that need to be read for the next period and predict their total retrieval time. If starvation will occur in the coming period, it is known beforehand and the system can make adjustments.

When starvation occurs, priority access can be provided as follows. We first calculate the time for the file system to transfer data from a disk as derived in Eq. 8. If the transfer time $T_{transfer}$ is less than the fixed length working period T_{period} there is no starvation for the next period and the file system has sufficient time or bandwidth to retrieve data for all existing sessions. In this situation the file server scheduling process can ask the file system to read $S(i)$ of data for session i. $S(i)$ is the size of the data for session i in this period and the characteristics of $S(i)$ are already known by the scheduler at the end of previous period. If the transfer time $T_{transfer}$ is greater than the working period T_{period}, we predict the occurrence of a starvation for the coming period. We then calculate the shortage time for the file system for which it is unable to transfer data from the disk. We define the shortage time T_{fail} as

$$T_{fail} = T_{transfer} - T_{period}. \tag{11}$$

If the amount of data the file system fails to read is S_{fail}, we determine $S_{fail} = T_{fail}/R$. From Eqs. 8 and 11, we have

$$S_{fail} = \sum_{1}^{n} S(i) + R \sum_{1}^{n} T_{latency}(i) - R \times T_{period}. \tag{12}$$

We can then reschedule the access scheme by dropping frames equitably among all n sessions. As a result, any session i is allowed to read only $S(i) - S_{fail}/n$ amount of data. This means that every session drops S_{fail}/n of data and shares the impact of starvation. Since we use a reorganization of the storage pattern on the disk, the file system will automatically drop the less important frames. As shown in Fig. 4, the less important frames are placed at the end of each storage pattern. Therefore, B frames will be dropped first, and the dropped frames will be uniformly distributed across all retrieved blocks and among sessions.

5 Performance Evaluation

In this section we evaluate our proposed model by performing a series of experiments on a set of video data. We encoded a video clip with an image-aspect-ratio of 320×240 pixels and without audio at 15 frame/s and extraploated the resulting statistics to create a 30 frame/s MPEG-I stream. The size of each MPEG frame was measured for the test set. Table 2 illustrates the video characteristics and the disk device parameters used in the experiments.

Symbol	Identification	Value	Units
$E(T_{latency})$	average disk seek latency	35.9	ms
$\sigma^2_{latency}$	variance of disk seek latency	201.6	ms^2
R	normalized disk bandwidth	8	Mb/s
μ_λ	average data rate for MPEG video	0.6369	Mb/s
σ_λ	standard deviation of MPEG video traffic rate	0.2305	Mb/s

Table 2: Simulation Parameters

5.1 Performance Evaluation and Comparison

We first fixed the working period T_{period} to be 6 seconds, yielding storage patterns containing 180 MPEG frames each. We then measured the time for the file system to transfer data from the disk by changing the number of sessions from 9 to 12 for 500 periods. For each case we measured and plotted the value of $T_{transfer}$ used to read n video sessions from disk (Fig. 6). We found that whenever more sessions were accepted the file system required more time to transfer the data. For example, if the file system supports 8 or 9 sessions, $T_{transfer}$ was always less than our preset working period T_{period}. Therefore starvation never occured when there were 9 or fewer sessions supported. As we increased the number of sessions to 11, the average disk transfer time $T_{transfer}$ remained between 5.4 to 5.8 seconds. However, on occasion, $T_{transfer}$ exceeded the preset working period and starvation occured.

Using the same data set we also studied the effects of increasing the number of supported sessions on starvation frequency (Eq. 7). Figs. 6 and 7 show the frequency of starvation for a range of supported sessions. Fig. 8 shows the average shortage time and the data loss percentage for different number of sessions.

Additional simulations were performed to understand the effects of data loss on video playback. For the MPEG encoding scheme, we know that B frames are decoded by referencing the previous and following P or I frames and have a higher compression ratio than P and I frames. In our priority access scheme, we only drop B frames when starvation occurs. Typically, with MPEG-compressed video, we can expect the frame loss percentage to be higher than the data loss percentage. Fig. 9 illustrates the average frame loss under starvation for a range of supported sessions.

If the file system accepts 9 or fewer sessions there was no starvation. When the number of sessions increased to 10, starvation occured about 0.15 % of the time. This implies that there is a starvation every 667 periods (4,000 seconds) and the starvation is distributed across the

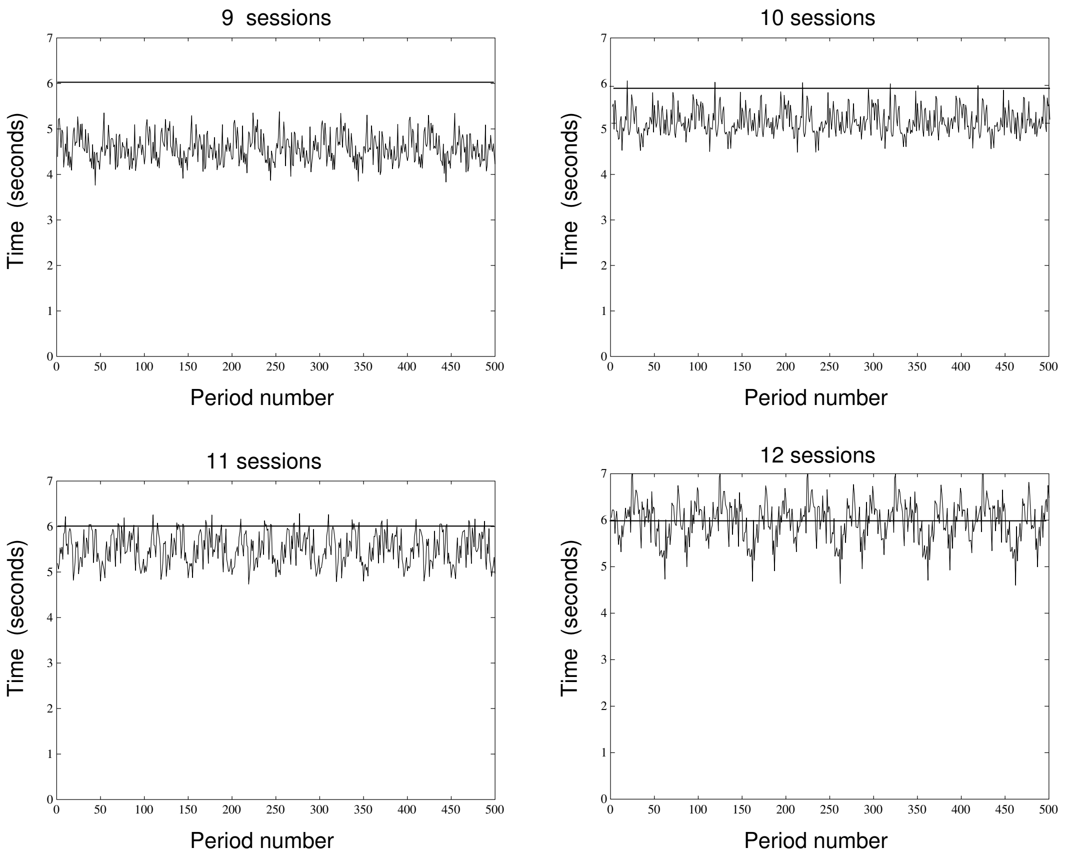


Figure 6: Effects of Traffic Multiplexing on Starvation

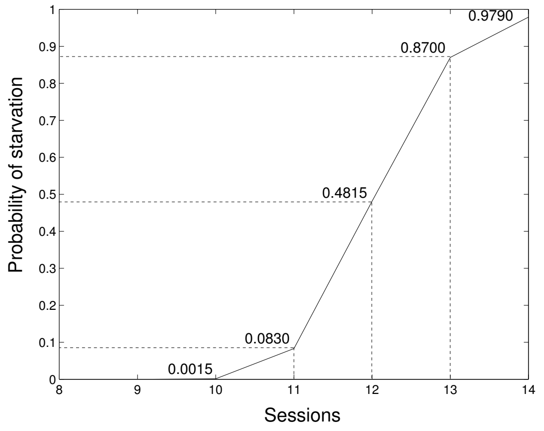
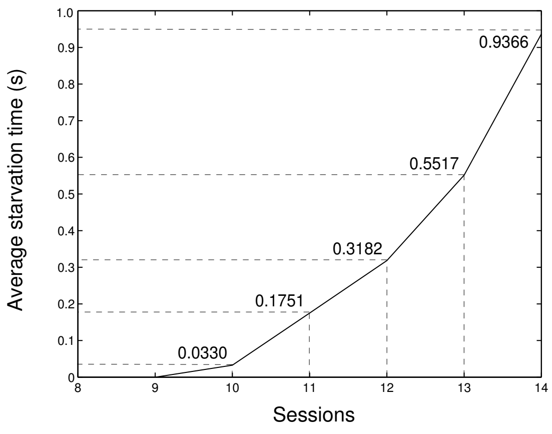


Figure 7: Starvation Frequency: $T_{period} = 6$ Seconds



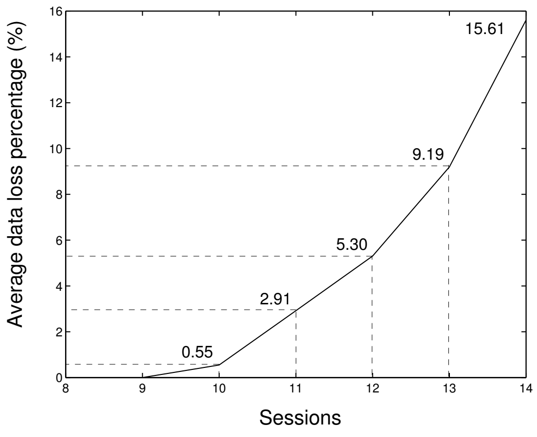


Figure 8: Average Shortage Time and Data Loss Percentage Under Starvation: $T_{period} = 6$ Seconds

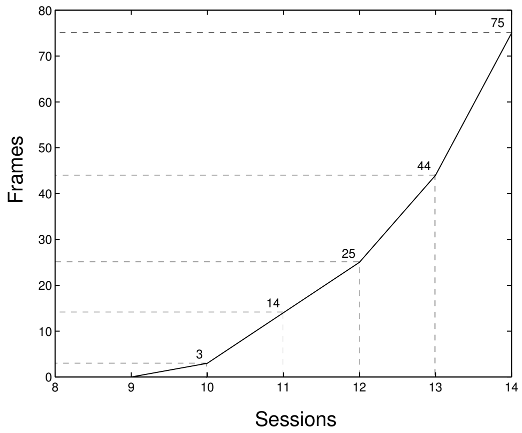


Figure 9: Average Frame Loss Under Starvation: $T_{period} = 6$ Seconds

entire period. During this starvation period three frames per session are lost as the video object is played back at 30 frame/s. Because the storage pattern is reorganized, the three lost frames are uniformly distributed across the 6 second period. In this case the file system loses only one frame in two seconds.

For 13 sessions the average video rate μ_λ was 0.6369 Mb/s and the normalized disk bandwidth was 8 Mb/s. The total bandwidth requirement for 13 sessions was 8.2797 Mb/s which was higher than the available disk bandwidth. In this case the starvation frequency was 55.17 % and starvation occured every two periods. During the starvation period 44 frames are lost over 6 seconds. This means that the file system can still provide 30 frame/s 45 % of the time and 23 frame/s 55 % of the time.

5.2 Considerations for the Selection of T_{period}

Fig. 3 and Eq. 1 show that the working period T_{period} must be greater than the sum of all individual session retrieval times and seek latencies in order to transfer data from the disk for all sessions. From Eq. 11 and Fig. 5, we know that if $T_{transfer}$ is less than the fixed length working period T_{period} there is no starvation. However, this does not mean that we can extend the length of the working period to the maximal value of transfer time $T_{transfer}$ to prevent any starvation. Since the value of $T_{transfer}$ depends on T_{period}, if we extend the length of period T_{period}, the file system needs to transfer more data for every session which increases the length of transfer time $T_{transfer}$. As shown in Fig. 3, if we extend T_{period} the probability density of $T_{transfer}$ will shift to the right. In other words, when we change the length of period T_{period} the shape for $T_{transfer}$'s probability density will change and affect the starvation frequency.

Fig. 10 shows a comparison of starvation frequency and frame losses for a range of scheduling periods. We find that if T_{period} is very small (for example, less than one second), no matter how many sessions there are, the starvation frequency and frame losses are extremely high. As we increase the duration of T_{period} the starvation frequency and frame loss rates diminish. The total latency $\sum_{i=1}^{n} T_{latency(i)}$ for switching the disk head between sessions is primarily dependent on the number of sessions n rather than the duration of T_{period}. This means that there is no change to the total latency $\sum_{i=1}^{n} T_{latency(i)}$ with T_{period} being 1 second or 10 seconds. Most of the disk bandwidth is wasted on switching the disk head when T_{period} is very small. This will introduce frequent starvation and frame losses.

We can improve the quality of video presentation for the system by decreasing the

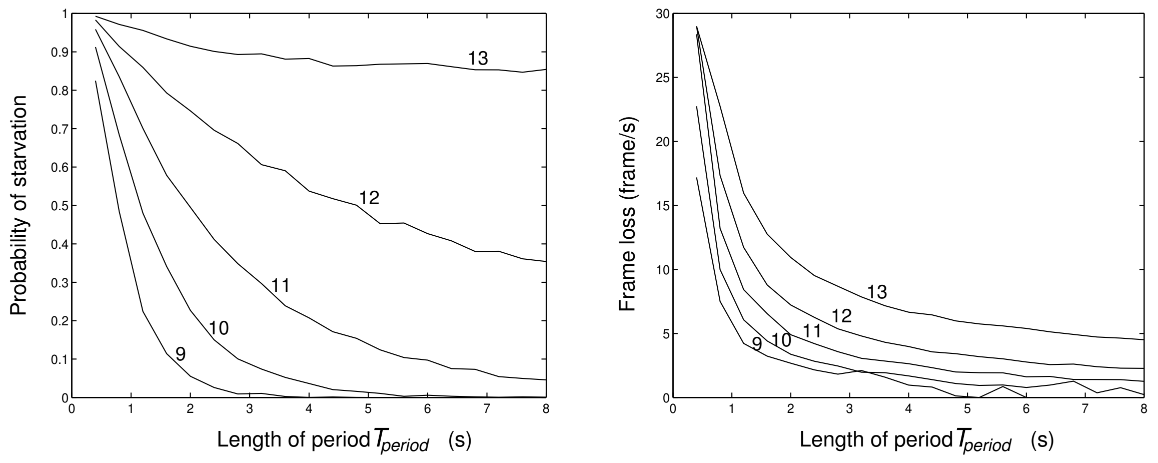


Figure 10: Starvation Frequency and Average Frame Loss Under Starvation

throughput or by extending the length of the scheduling period. When the length of period is sufficiently long (e.g., greater than 6 seconds) the starvation frequency and frame loss rate do not change significantly. At this point, even if we extend T_{period}, the improvement in the quality of the video stream is limited because disk seek latencies become less important and the variance in video data rates is reduced.

6 Discussion

Our earlier work demonstrated that a longer scheduling period requires more memory for buffering data and introduces a larger response time. Our goal is to minimize the total cost of such a system and still provide an acceptable level of service quality. We can improve the loss behavior by extending the length of a period, but this leads to a longer response delay. From the system's perspective, the gains resulting from accepting more sessions by extending the length of a period is offset by the increased buffering requirements. If a local file server is expected to provide m sessions for one particular movie, the interesting problem is to balance the different resources to minimize operational costs [11]. Our proposed model provides an option that allows the file system the flexibility of tolerating some data loss during the disk retrieval phase without degrading the quality of service dramatically. This is impossible with conventional access schemes due to the characteristics of MPEG data.

With the proposed mechanism, we can optimize the length of T_{period} to yield reduced buffering costs and improved performance. For example, in Fig. 10, supporting 10 sessions without starvation requires a T_{period} of 6 seconds using the traditional allocation scheme. Using the priority access scheme we can tolerate some data loss and still provide acceptable service. If we set the length of a period to be 3 seconds we only experience 10 % starvation and lose only 2 to 3 frames per second. In this example, we can save 50 % of memory costs and reduce the response time offered to the user to 3 seconds.

Our priority access scheme also provides a service scaling mechanism. If the file system bandwidth is sufficient, we can provide the desired resources for all sessions. Whenever the bandwidth is insufficient, the file system still has an option to accept new service requests by linearly degrading the quality of existing sessions to take advantage of the gains from statistical multiplexing. This means that the quality of the existing sessions will not degrade abruptly when the file system has transient bandwidth shortages.

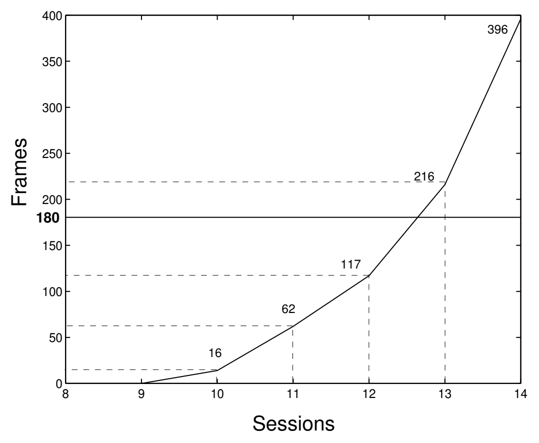


Figure 11: Frame Loss Per Period without Priority Access Scheme: T_{period}

With the proposed storage reorganization and priority access schemes, the file system has sufficient information to predict any starvation in advance. Without this scheme the file system switches service among sessions period by period. If there is no starvation (the length of period is enough to transfer data for all sessions), there are little performance gains from the proposed approach. With starvation, however, when a period elapses the current session must be preempted by the following session. All residual frames must be dropped. Since the file system cannot predict starvation, the last few sessions in the period will absorb any starvation. We illustrate this behavior in Fig. 11. In this example, if there are 10 sessions in a disk when starvation occurs, the last session will lose 16 consecutive frames during the 6 second period while the other 9 sessions do not lose any frame. Moreover, the lost frames are contiguous, i.e., the last session suffers a blank period of 0.5 seconds. As we increase the number of supported sessions to 13, the file system will lose 216 frames during a period. One session will lose everything (180 frames) and another session will lose 36 contiguous frames. Even if these losses are distributed across all sessions, frame loss propagation without reorganization causes the number of lost frames to be higher than with reorganization and priority access.

7 Related Work

The problem of building storage systems for continuous media has seen a great deal of recent research activity. Lougher and Shepherd [12] have described the general principles in designing a continuous media file system. Gemmell and Christodoulakis [8] and Anderson et al. [1] have established some basic principles for retrieval and storage of delay-sensitive multimedia data. These principles can be used by a system designer to estimate hardware requirements and to evaluate design strategies. Rangan et al. [13] have developed an admission control algorithm for determining when a new concurrent access request can be accepted without violating the real-time constraints of existing sessions. Chen et al. [6] show an optimum grouped sweeping scheduling (GSS) procedure to support heterogeneous video streams. Reddy and Wyllie [14] proposed a SCAN-EDF algorithm which can support larger numbers of video streams. Related work on storage systems for continuous-media data on disk arrays includes Keeton and Katz's [10] placement scheme for multi-resolution video on disk arrays for providing scalable services. Chang and Zakhor [2] have proposed a placement strategy and an admission control strategy that provides scalable MPEG video service from disk arrays.

Our work is most closely related to the work of Chang and Zhakor [2] on the storage of MPEG compressed video on disk arrays. The authors use a MPEG data reorganization access scheme similar to ours. However, our scheduling and storage policies make information about the possibility of missed deadlines known a priori to the scheduler using an additional header. This allows the scheduler to scale all sessions uniformly. Furthermore, it is significant to study the performance of a single disk from the perspective of providing scalable services. Our work complements the earlier work in this respect.

8 Conclusion

When a video-on-demand file server transfers data from a disk, it must guarantee that real-time multimedia data can be read at a sustained rate and satisfy the timing requirements. Disk seek latencies and time varying traffic characteristics typical of VOD have a significant impact on these systems.

In this paper we have proposed a MPEG frame storage reorganization to complement a priority access scheme for video delivery from a single disk storage system. In the proposed placement scheme, MPEG data are prioritized and organized contiguously on the disk. The new placement policy enables the retrieval process to ensure that the most important frames are delivered during periods of disk overload, improving the quality of video presentation. Such a scheme is useful in building a file server that is tolerant of data losses and still providing an acceptable quality of service. Moreover, this scheme can be easily extended to support VCR-like functions such as fast-forward and fast-backward [4].

Performance evaluation indicates that the proposed mechanism reduces buffering requirements and improves response times significantly. Test results show the proposed scheme yields a better throughput, and increases the quality of presentation. It is an efficient means for supporting multiple sessions from a disk.

References

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

[2] Chang, E. and A. Zakhor, "Variable Bit Rate MPEG Video Storage on Parallel Disk Arrays," *Proc. 1st Intl. Workshop on Community Networking,* San Francisco, CA, July 1994, pp. 127-137.

[3] Chen, H.J., and T.D.C. Little, "A Scalable Video-on-Demand Service for the Provision of VCR-Like Functions," submitted to U.S. Patent Office, Nov. 1994.

[4] Chen, H.J., A. Krishnamurthy, D. Venkatesh, and T.D.C. Little, "A Scalable Video-on-Demand Service for the Provision of VCR-Like Functions," To appear in *Proc. 2nd IEEE Intl. Conf. on Multimedia Computing and Systems,* Washington D.C., May 1995.

[5] Chen, H.J. and T.D.C. Little, "Storage Allocation Policies for Time-Dependent Multimedia Data," To appear in *IEEE Trans. on Knowledge and Data Engineering,* 1995.

[6] Chen, M.S., D. D. Kandlur, and P. S. Yu, "Optimization of Grouped Sweeping Scheduling (GSS) with Heterogeneous Multimedia Streams," *Proc. 1st ACM Intl. Conf. on Multimedia,* Anaheim, CA, August 1993, pp. 235-242.

[7] Cidon, I., A. Khamisty, and M. Sidi, "Analysis of Packet Loss Processes in High-Speed Networks," *IEEE Trans. on Information Theory,* Vol. 39, No. 1, January 1993, pp. 98-108.

[8] Gemmell J., and S. Christodoulakis, "Principles of Delay-Sensitive Multimedia Data Storage and Retrieval," *ACM Trans. on Information Systems,* Vol. 10, No. 1, January 1992, pp. 51-90.

[9] ISO/IEC DIS 11172, "Information Technology – Coding of Moving Pictures and Associated Audio for Digital Media at up to about 1.5 Mbit/s," November 1992.

[10] Keeton, K. and R.H. Katz, "The Evaluation of Video Layout Strategies on a High-Bandwidth File Server," *Proc. 4th Intl. Workshop Network and Operating Systems Support for Digital Audio and Video*, Lancaster, UK, November 1993, pp. 237-248.

[11] Little, T.D.C., and D. Venkatesh, "Popularity Based Assignment of Movies to Storage Devices in a Video-on-Demand System," *ACM/Springer Multimedia Systems*, Vol 2, No. 6, January 1994, pp. 280-287.

[12] Lougher, P., and D. Shepherd, "The Design of a Storage Server for Continuous Media," *The Computer Journal*, Vol. 36., No. 1, February 1993, pp. 32-42.

[13] Rangan, P.V., H.M. Vin, and S. Ramanathan, "Designing an On-Demand Multimedia Service," *IEEE Communications Magazine,* Vol. 30, No. 7, July 1992 pp. 56-64.

[14] Reddy, A.L.N., and J. Wyllie, "Disk Scheduling in Multimedia I/O Systems," *Proc. 1st ACM Intl. Conf. on Multimedia,* Anaheim, CA, August 1993, pp. 225-233.

[15] Sen, P., M. Maglaris, N. E. Rikli, and D. Anastassiou, "Models of Packet Switching of Variable-Bit-Rate Video Source," *IEEE Journal on Selected Areas in Communication,* Vol. 7, No. 5, 1989, pp. 865-869.

[16] Vin, H.M., Al. Goyal, An. Goyal, and P. Goyal, "An Observation-Based Admission Control Algorithm for Multimedia Servers," *Proc. 1st IEEE Intl. Conf. on Multimedia Computing and Systems,* Boston, MA, May 1994, pp. 234-243.