Physical Storage Organizations for Time-Dependent Multimedia Data[1]

H.-J. Chen and T.D.C. Little

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

MCL Technical Report 10-01-1993

Abstract–Multimedia computing requires support for heterogeneous data types with differing storage, communication and delivery requirements. Continuous media data types such as audio and video impose delivery requirements that are not satisfied by conventional physical storage organizations. In this paper we describe a physical organization for multimedia data based on the need to support the delivery of multiple playout sessions from a single rotating-disk storage device. Our model relates disk characteristics to the different media recording and playback rates and derives their storage pattern. This storage organization guarantees that as long as a multimedia delivery process is running, starvation will never occur. Furthermore, we derive bandwidth and buffer constraints for disk access and present an approach to minimize latencies for non-continuous media media stored on the same device. The analysis and numerical results indicate the feasibility of using conventional rotating magnetic disk storage devices to support multiple sessions for on-demand video applications.

Keywords: Multimedia, physical data organization, file systems, data clustering, time-dependent data.

[1]In *Proc. 4th Intl. Conf. on Foundations of Data Organization and Algorithms (FODO'93)*, Evanston, IL, October 1993, pp. 19-34. This work is supported in part by the National Science Foundation under Grant No. IRI-9211165.

Table 1: Properties of multimedia data

Data Type	**Buffer/Bandwidth**
Voice-quality audio (8 bits @ 8 KHz)	64 Kb/s
CD quality audio (stereo @ 44.1 KHz)	1.4 Mb/s
NTSC-quality video (uncompressed @ 512 × 480 pixels, 24 bits/pixel)	5.9 Mb/frame (177 Mb/s)
JPEG-compressed NTSC video	≈ 7 Mb/s – 3.5 Mb/s
MPEG-I-compressed NTSC video	≤ 1.5 Mb/s
MPEG-II-compressed NTSC video	≤ 10 Mb/s
HDTV-quality video (uncompressed @ 1248 × 960 pixels, 24 bits/pixel)	28.7 Mb/frame (863 Mb/s)

1 Introduction

Files comprised of multimedia data are different from conventional data files in many respects. As shown in Table 1, multimedia data, and hence files, consume enormous space and bandwidth relative to text of program files. For example, a single feature-length JPEG-compressed movie can require over 2 gigabytes of memory for digital storage. Because multimedia data are also sensitive to timing during delivery, a multimedia file system must satisfy timing constraints of some data and not others. When a user *plays-out* or *records* a time-dependent multimedia data object, the system must consume or produce data at a constant, gap-free rate. This means that the file system must ensure the availability of sufficient data buffer space for the playback or recording process. For example, to maintain a continuous NTSC-quality video playback, a file system must deliver data at a rate of 30 frames/s. Moreover, the delivery mechanism must also satisfy the intermedia synchronization requirement among related media (e.g., the lip synchronization between audio, video, and subtitles).

A storage subsystem accesses 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 head and seek switching frequency and resultant access latency. In addition, the electromechanical nature of secondary-storage devices requires the use of scheduling disciplines modified to meet the throughput and real-time requirements of multimedia data delivery. When a multimedia file system transfers data from a disk, it must guarantee that multimedia data arrive at the consuming device on time. It must also meet the timing requirements of the multimedia object; however, this task is difficult due to the

unpredictability of disk seek latencies. Furthermore, in a multitasking system, more than one user can request multimedia or non-real-time services, thereby requiring multiple session management. In contrast, the data allocation and scheduling strategies for conventional file systems are only concerned about the throughput, latency, and storage utilization for random access to files.

Our objective is to provide real-time behavior for a set of multimedia *sessions* originating from a single, conventional rotating-disk magnetic storage device. Since conventional file systems can not satisfy the real-time requirements for multimedia applications, we propose a new file system to support multimedia applications.

A number of related works exist in this area. The problem of satisfying timing requirements for multimedia data has been studied as a conceptual database problem [7], as an operating system delivery problem [1, 10, 17, 9], and as a physical data organization and performance problem [3, 4, 5, 11, 8, 16, 18].[2] Rangan et al. [13] propose a model for storing real-time multimedia data in file systems. The model defines an interleaved storage organization for multimedia data that permits the merging of time-dependent multimedia objects for efficient disk space utilization. In a related work, Rangan et al. [12] develop an *admission control* algorithm for determining when a new concurrent access request can be accepted without violating the real-time constraints of existing sessions. Polimenis [11] shows that the hard requirement for the acceptance of a set of real-time sessions is the availability of disk bandwidth and buffer space. Gemmell and Christodoulakis [5] establish some fundamental principles for retrieval and storage of time-dependent data. A theoretical framework is developed for the real-time requirements of multimedia object playback. Storage placement strategies for multichannel synchronized data are also examined. P. Yu, Chen, and Kandlur [8] present an access scheme called the *grouped sweeping scheme* (GSS) for disk scheduling to support multimedia applications by reducing buffer space requirements. C. Yu et al. [16, 18] describe approaches to interleaving time-dependent data to support constant playout rates.

In this paper, we propose a physical data organization for multimedia data. We interleave different media objects within a block so as to maintain temporal relationships among those objects during retrieval (Fig. 1). We also define an allocation policy based on the contiguous approach to prevent frequent head movement that can cause significant seek latencies and can support editing on multimedia files. The behavior of a conventional magnetic rotating-disk storage device is analyzed with respect to the mean and variance of the seek latency.

[2]The communications view is not applicable here.

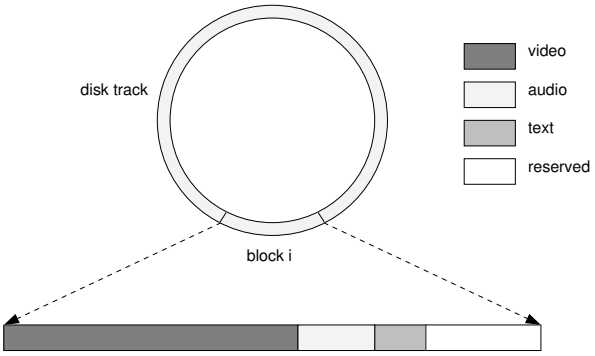

Figure 1: Physical storage organization for a rotating disk device

A round-robin scheduling discipline is chosen for the service of multimedia sessions as in other work [9, 11, 14], permitting the disk to switch alternately between multimedia tasks and other non-real-time tasks. We show the constraints which must be satisfied to permit the acceptance of a set of multimedia sessions including bandwidth and buffer considerations. We also evaluate the impact of the disk latency and establish a probabilistic model for our disk access schedule to guarantee that the frequency of starvation will be less than a specified rate.

The remainder of this paper is organized as follows. In Section 2, we describe the storage organization and allocation policy for multimedia objects to facilitate disk bandwidth utilization. In Section 3, we analyze the probabilistic behavior of seek latency for a disk. In Section 4, we show an access schedule for the disk and present a periodic service discipline for multimedia objects based on a probabilistic model of a disk, and show how this schedule reduces the required buffers and increases the number of supported multimedia sessions. Section 5, concludes the paper.

2 Storage Organization for Multimedia Objects

Most existing storage server architectures employ random allocation of blocks on a disk. This type of organization is not sufficient to meet the real time requirements for multimedia applications because the disk latency between blocks of a media object is unpredictable [14]. The file system cannot guarantee satisfaction of the deadline for the retrieval of multimedia data.

We view a multimedia object as consisting of components of any data type. Without loss

of generality, we model a typical multimedia object as being comprised of audio, video and text. These three components can be viewed as distinct even though they are simultaneously recorded, and as input, arrive at the file system as three different streams [14]. During retrieval, these three streams are sent to three output queues for playout and ultimately are experienced by the user. From a timing perspective, the data streams can arrive at the file system with specific implied timing (e.g., live audio) or can arrive at the file system arbitrarily. For example, live video and audio can be recorded at the same time while subtitles are recorded later.

This leads us to the issue of data interleaving for maintaining intermedia synchronization. The advantage of interleaving multiple data streams into a single layout is the preservation of timing between related steams. The penalty with this scheme is the overhead associated with data combination and redistribution. These layouts are also called *homogeneous* (non-interleaved) and *heterogeneous* (interleaved) layouts [14]. The homogeneous layout stipulates storage of single medium data in blocks without interleaving. Of course, for this layout scheme timing relationships between media are not implicitly stored with the interrelated media.

In the homogeneous approach, each medium requests a session in a round-robin schedule. When retrieving a multimedia object, the file system must switch between sessions which can consume additional disk bandwidth and degrade throughput. There is no such problem in the heterogeneous approach. We merge different media data within a block based on their temporal relationships and can treat the aggregation of data as a single media object. Therefore, there is only one session for each multimedia object for the heterogeneous approach. For this reason we use the heterogeneous layout approach in this work. In our approach, multiple media streams being recorded are stored within the same block and the length of each object is proportion to its consumption rate.

In terms of *intramedia* timing, interleaving of data becomes important to maintain smooth, gap-free playout. In the extreme case, contiguous space allocation yields the highest effective bandwidth from a disk, but with a penalty for costly reorganization during data insertions and updates:

1. With the interleaved policy, multimedia data are stored on disk in a interleaved fashion [13, 14, 16, 18]. This approach can guarantee continuous retrieval and smooth the speed gap between disk and multimedia devices. Therefore, it can reduce the buffer requirement significantly. Usually, it can be applied on optical disks or in a single user

environment.

2. With the contiguous policy, multimedia data are stored on a disk contiguously. This policy can also provide continuous retrieval, but entails enormous copying overhead during insertions and deletions [13]. However, it is the most efficient way for bandwidth utilization [11]. This approach can be used for data that is seldom modified (read-only) such as digital motion picture archives which do not need deletion and insertion.

In our approach, we refine the contiguous scheme using a two-tiered structure. On the first level, we propose a doubly-linked list which is built based on the temporal relations of a multimedia object [7]. Each item in the list contains a pointer which points to the disk address of a media block. The reason for the doubly-linked list structure is to support the ability to provide reverse playback of multimedia objects. On the second level, we store the multimedia data that are indicated in the first level, permitting the reversal of a multimedia presentation at any moment. Basically, multimedia objects are stored sequentially on the disk. Subsequent media blocks are put on adjacent, unoccupied blocks. If a disk track or cylinder becomes full (or the next block is occupied) this policy places the multimedia data in the next nearest available block.

3 Disk Latency and Bandwidth

Multimedia data require large file sizes and consumption rates. The file system must organize the multimedia data on the disk for efficient use of the limited available space and bandwidth. To reach the highest bandwidth, a disk system must read (or write) contiguously. A discontinuous disk access can result in diminished disk bandwidth due to additional seek and rotational latencies involved in each discontinuity.

In our approach, there are two classes of disk latencies. The first one is caused by fragmentation inside the multimedia file. The file system can trace the file index and calculate the latencies. The second one is task switching latency. In our scheduling approach, the disk switches alternately to different multimedia tasks. Because one can *pause, stop* or *reverse* a multimedia presentation at any moment, and a multimedia object can be allocated anywhere in the disk, there are unpredictable and significant latencies during retrieval. In this section, we determine these disk latencies and their distributions through analysis for a typical hard disk storage unit suitable for a Unix workstation [15]. Parameters characterizing such a device are summarized in Table 2 using symbols adopted and extended from Kiessling [6].

Table 2: Disk parameters and derived statistical behavior

Symbol	**Identification**	**Value**	**Units**
S_{dt}	Size of a single track	54,900	bytes
N_{head}	Number of tracks in a cylinder (number of disk heads)	15	tracks
T_{hh}	Time to change head to the another surface	2,000	μs
T_{tt}	Time to cross a track	21	μs
T_{start}	Seek start-up time	11,000	μs
T_{rot}	Rotation time for a disk	16,700	μs
R_t	Data transfer rate within a track	3.29	Mbyte/s
c	Number of cylinders per disk	2,107	cylinders
$T_{latency}$	$= T_{cross} + T_{switch} + T_{rotate}$		ms
$E(T_{cross})$	$\cong \frac{1}{3}c \times T_{tt} + T_{start}$	25.7	ms
σ^2_{cross}	$\cong \frac{c^2}{18}T^2_{tt}$	108	ms^2
σ_{cross}	$\cong \frac{c}{\sqrt{18}}Ttt$	10.4	ms
$E(T_{switch})$	$= \frac{N_{head}-1}{N_{head}}T_{hh}$	1.86	ms
σ^2_{switch}	$= T^2_{hh}\frac{N_{head}-1}{N^2_{head}} \cong \frac{T^2_{hh}}{N_{head}}$	0.27	ms^2
σ_{switch}	$\cong \frac{T_{hh}}{\sqrt{N_{head}}}$	0.51	ms
$E(T_{rotate})$	$\cong \frac{1}{2}T_{rot}$	8.35	ms
σ^2_{rotate}	$\cong \frac{1}{3}T^2_{rot}$	92.96	ms^2
σ_{rotate}	$\cong \frac{1}{\sqrt{3}}T_{rot}$	9.64	ms
$E(T_{latency})$	$\cong \frac{1}{3}c \times T_{tt} + T_{start} + \frac{N_{head}-1}{N_{head}}T_{hh} + \frac{1}{2}T_{rot}$	35.9	ms
$\sigma^2_{latency}$	$\cong \frac{c^2}{18}T^2_{tt} + \frac{T^2_{hh}}{N_{head}} + \frac{1}{3}T^2_{rot}$	201.6	ms^2

3.1 Seek Delay Latency

When a user edits the multimedia file or the file system schedules another process to access the disk, the next block to be retrieved can be arbitrarily located anywhere on the device. The disk head must start up and cross a number of tracks, switch to a recording (write) surface and rotate to the indicated block. Assuming that the location of the desired block is uniformly distributed on the whole disk, then the total latency is $T_{latency} = T_{cross} + T_{switch} + T_{rotate}$, where T_{cross} is the arm positioning time for disk head move to the correct track, T_{switch} is the delay to switch the head to the other surface, and T_{rotate} is the delay for disk rotation. We have derived various statistical disk performance behaviors from these base parameters [2], and summarize them in Table 2.

3.2 Disk Bandwidth Normalization

In an ideal disk storage organization, data can be accessed without latencies, and the data transfer rate (or bandwidth) is dependent only on the disk rotational speed. In a real disk, latencies are introduced due to track and platter switching, and disk rotation. These latencies are determined by the layout of data on the disk and the scheduling policy for their access. We can normalize the data transfer rate based on a complete disk scan policy as follows: once the head reaches and retrieves the first block of an object, it retrieves the adjacent block in the same track. If the whole track has been retrieved, it switches to the next surface but remains on the same cylinder. If the whole cylinder has been retrieved, the disk arm crosses to the next track. We normalize the disk bandwidth by considering each of these head motions in the complete scan as:

$$R = \frac{1}{\frac{1}{R_t} + \frac{1}{S_{dt}} T_{hh} + \frac{1}{S_{dt} \times N_{head}} [T_{start} + T_{tt}]} \tag{1}$$

Therefore, we can use this derived value as the maximum effective bandwidth for data transfer from the disk.

4 Disk Access Scheduling

In this section we show the constraints for the acceptance of a set of multimedia sessions and the requirements for buffer size and disk bandwidth.

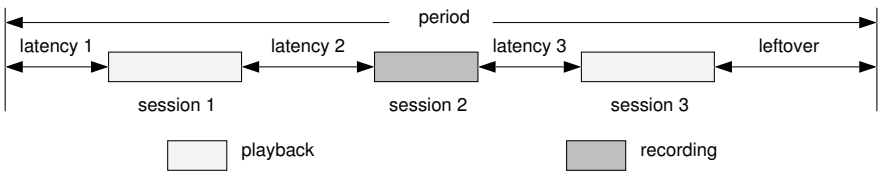

Figure 2: Layout model

4.1 Layout Model

In the layout model of Polimenis [11], a working period T_{period} is defined for a set of multimedia tasks and other non-real-time tasks as shown in Fig. 2.

During a working period, the schedule switches among all multimedia sessions. It carries enough data into the buffer for the ith session to keep task i busy until its term is active in the next working period. If R is the whole disk bandwidth that we derived in (1), then each session i shares an interval $T(i)$ proportional to its consumption rate $R^c(i)$. The amount of data accessed during $T(i)$ is equal to the amount consumed during the period T_{period}. Thus, we have $T(i) = \frac{R^c(i)}{R} T_{period}$.

In this equation, $R^c(i)$ represents the bandwidth requirement for session i. Let the ith session contain k different media data (video, audio, text, etc.). Each medium j requires $R^c_j(i)$ of bandwidth. Thus, the total bandwidth requirement $R^c(i)$ for session i is $\sum_{j=1}^{k} R^c_j(i)$. For viable multimedia data delivery, the bandwidth lost due to task switching latencies plus the bandwidth consumed by each multimedia session must be less than the normalized disk bandwidth (where the period is fixed unless we change the number of sessions).

4.2 Bandwidth Requirements

In this section, we derive the bandwidth constraint. Let $n(i)$ be the number of bytes accessed for medium i during a working period T_{period}. The total number of bytes n to be read during a period T_{period} is then $\sum_{i=1}^{m} n(i)$. Because the time interval $T(i)$ for each media is proportional to its bandwidth requirement and $n(i) = T(i) \times R$. Thus, we have $n(i) = T_{period} \times R^c(i)$.

As shown in Fig. 2, the total interval used for multimedia sessions plus the disk seek latency should be less than the working period T_{period} in order to have enough bandwidth for other non-real-time tasks. On the other hand, the period T_{period} must be greater than the

time needed in the worst case to transfer data from (or to) the disk for all sessions. Suppose we have m multimedia sessions. Let R be the total disk bandwidth and $T_{latency}(i)$ be the task switching latency between sessions $i-1$ and i. Then,

$$\frac{n}{R} + \sum_{i=1}^{m} T_{latency}(i) < T_{period} = \frac{n(i)}{R^c(i)}, \tag{2}$$

where $\frac{n(i)}{R^c(i)}$ should be equal to T_{period} to maintain a steady-state. This means that the amount of data read from the disk for each session i during a period is exactly equal to the amount of data consumed by the ith consumer process. Thus, by (2),

$$R > \frac{n}{\frac{n(i)}{R(i)} - \sum_{i=1}^{m} T_{latency}(i)} = \frac{1}{\frac{n(i)}{n}\frac{1}{R(i)} - \frac{\sum_{i=1}^{m} T_{latency}(i)}{n}}.$$

Since, $\frac{n(i)}{n} = \frac{R^c(i)}{\sum_{i=1}^{m} R^c(i)}$, then

$$R > \frac{1}{\frac{R^c(i)}{\sum_{i=1}^{m} R^c(i)}\frac{1}{R(i)} - \frac{\sum_{i=1}^{m} T_{latency}(i)}{n}} = \frac{1}{\frac{1}{\sum_{i=1}^{m} R^c(i)} - \frac{\sum_{i=1}^{m} T_{latency}(i)}{n}}.$$

The right hand side of the above equation can be divided into two parts. The first part is the bandwidth requirement of all multimedia sessions. The second part is the factor due to the seek latency between any two sessions. Thus,

$$R > \sum_{i=1}^{m} R^c(i) + \frac{(\sum_{i=1}^{m} R^c(i))^2 \times \sum_{i=1}^{m} T_{latency}(i)}{n - \sum_{i=1}^{m} R^c(i) \times \sum_{i=1}^{m} T_{latency}(i)}. \tag{3}$$

The last term of this equation describes the bandwidth wasted, or lost, when the disk head is moved, or seeked, between sessions.

4.3 Buffer Requirements

In Section 4.1, we showed the bandwidth requirements for a set of multimedia sessions without considering their acceptability in terms of buffer utilization. In the layout model, each session i shares only part of a period (Fig. 2). Each session must carry enough data into the buffer to keep the process i busy until it is reserviced. Otherwise, the process starves. Therefore, the second condition to accept a set of multimedia sessions is the availability of

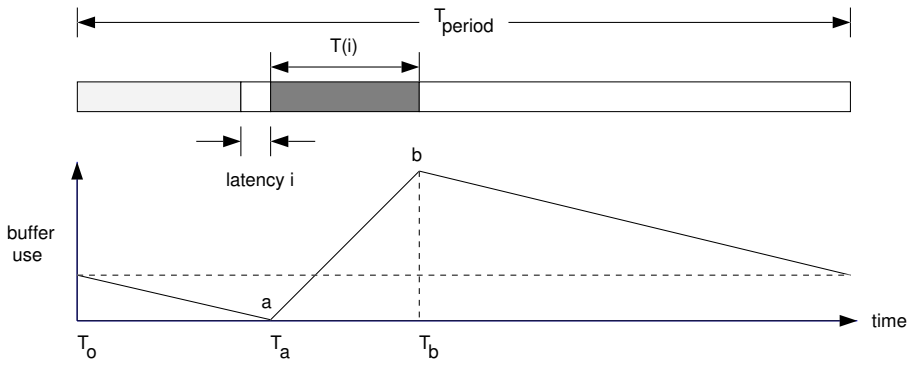

Figure 3: Buffer consumption

sufficient buffer space. As illustrated in Fig. 3, session i shares a duration $T(i)$ in a disk access.

When session i is active, its buffer size increases at a rate $R-R^c(i)$. Outside this duration, the buffer size shrinks at a rate $R^c(i)$. Let $B(i)$ be the buffer requirement for session i. Then $B(i) > (R - R^c(i)) \times T(i)$, or $B(i) > R^c(i) \times (T_{period} - T(i))$. If we let B be the total buffer requirement, then $B > \sum_{i=1}^{m}[(R - R^c(i)) \times T(i)]$. Rewriting, we get:

$$B > \sum_{i=1}^{m}[R^c(i) \times (T_{period} - T(i))] \tag{4}$$

Therefore, we have defined the buffer constraint that can be applied to determine the feasibility of adopting additional multimedia sessions.

4.4 Length of Period T_{period}

In Fig. 2 and (2), we show that the period T_{period} must be greater than the sum of all individual session periods in order to transfer data from (or to) disk for all sessions. Let D be the leftover duration as shown in Fig. 2. For each period, the disk spends $T_{transfer}$ to transfer data, where, $T_{transfer} = T_{period} - \sum_{i=1}^{m} T_{latency}(i) - D$. In a period, session i shares $T(i)$ duration based on its consuming rate $R^c(i)$. Therefore,

$$T(i) = [T_{period} - \sum_{i=1}^{m} T_{latency}(i) - D] \times \frac{R^c(i)}{\sum_{i=1}^{m} R^c(i)}$$

To maintain a steady-state for the system, the data read from the disk during $T(i)$ for session i must be equal to the amount consumed during the period T_{period}. Otherwise, the buffer can starve or grow without bound. Thus,

$$T_{period} \times R^c(i) < [T_{period} - \sum_{i=1}^{m} T_{latency}(i)] \times \frac{R^c(i)}{\sum_{i=1}^{m} R^c(i)} \times R$$

$$T_{period} > \sum_{i=1}^{m} T_{latency}(i) \times \frac{R}{R - \sum_{i=1}^{m} R^c(i)} = T \tag{5}$$

In (5), $T_{latency}(i)$ represents the seek latency when the disk switches the service from session $i-1$ to session i. Because the next retrieval data for session i can be allocated anywhere on the disk, the latency $T_{latency}$ is a random variable. In Section 3, we derive the average seek latency and the variance of the seek latency. Let $E(T_{latency})$ be the average seek latency and $\sigma^2_{latency}$ be the variance of seek latency (Table 2). The expectation $E(T)$ and variance $\sigma^2(T)$ of T in (5) are as follows:

$$E(T) = m \times E(T_{latency}) \times \frac{R}{R - \sum_{i=1}^{m} R^c(i)}$$

$$\sigma^2(T) = m \times \sigma^2_{latency} \times \frac{R}{R - \sum_{i=1}^{m} R^c(i)}$$

By the above equations, we know T is also a random variable, so we cannot assign T to be the lower bound of the period T^{min}_{period}. Let p be the probability of starvation that can be tolerated for the mth session, by Chebychev's Inequality we have $P[|T^{min}_{period} - E(T)| > k] \leq \frac{\sigma^2(T)}{k^2} = p$, and therefore,

$$T^{min}_{period} \geq E(T) + \frac{\sigma(T)}{\sqrt{p}} \tag{6}$$

This means that if the lower bound T^{min}_{period} is chosen, the probability for the mth session can be accepted successfully is greater than $1-p$.

By (6), if we chose T_{period} equal to the lower bound $E(T) + \frac{\sigma(T)}{\sqrt{p}}$, we can guarantee that the starvation rate for session m will be less than p. Equation 6 is always true; however, it does not mean that the starvation rate is equal to p. In the heavy load situation, when the number of multimedia sessions m is very large, by the Law of Large Numbers, the starvation

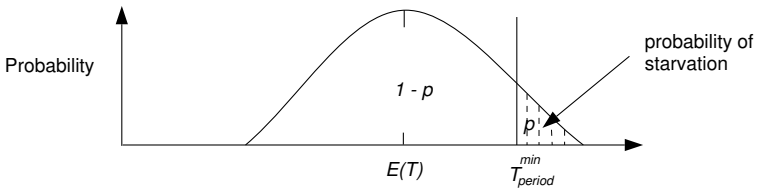

Figure 4: Distribution of T

rate will approach p. In the light load case, the starvation rate can be much lower than p. Conversely, we can use a shorter period T_{period} to keep the starvation rate under p.

A period T_{period} for a set of multimedia sessions must meet two hard requirements. In Section 4.2, we derived the bandwidth requirement, but it was not sufficient to accept a set of multimedia sessions. The system must provide enough buffer for each multimedia session. In the light load situation, there are always enough resources to allocate to multimedia sessions. Thus, we are more interested in the heavy load case. Let us assume the number of multimedia sessions m is large. Then, comparing with the period T_{period}, the duration $T(i)$ assigned to each multimedia session is small. We simplify (4) by ignoring the $T(i)$ and the result is still valid:

$$B \cong T_{period} \times \sum_{i=1}^{m} R^c(i) \tag{7}$$

By the equation above, we find that the buffer requirements are dependent on the length of period T_{period}. Let B_{max} be the maximum buffer space that is available. Obviously, there is a upper bound T^{max}_{period} for the period that can be accepted for a set of multimedia sessions. Otherwise, the total buffer requirements will exceed the available buffer space B^{max}. From (7), we have:

$$T^{max}_{period} = \frac{B^{max}}{\sum_{i=1}^{m} R^c(i)} \tag{8}$$

Equs. (6) and (8) derived above are for the general case where the consumption rates for multimedia sessions have different values. In real applications, the disk bandwidth requirements for multimedia sessions can have the same value. In the following example, we assume, for simplicity, that the consumption rates for all multimedia sessions are the same and evaluate the buffer consumption and number of sessions supported.

Table 3: File system performance for Example 1

	100 % Bandwidth Utilization		**100 % Buffer Allocation**	
N	T^{min}_{period} (ms)	Buffer Allocation (bytes)	T^{max}_{period} (ms)	Bandwidth Utilization
1	86	29,000	40,000	16.35 %
2	213	143,000	20,000	32.88 %
3	385	386,000	13,333	49.58 %
4	706	946,000	10,000	66.48 %
5	1,577	2,641,000	8,000	83.75 %
6	14,013	* 28,163,000	6,667	* 100.80 %

* Insufficient memory.

Example 1 Let us assume all multimedia sessions request the same disk bandwidth. Each multimedia session includes video data at a rate of 1.92 Mb/frame @ 30 frames/s with a 20:1 compression ratio, and audio data at a rate of 1.4 Mb/s with a 4:1 compression ratio. Each multimedia session consumes disk bandwidth at a rate of 0.4 Mbyte/s. Using the same disk parameters as in Table 2, the average disk latency $E(T_{latency})$ is equal to $35,965\mu s$ and the standard division $\sigma_{latency}$ is equal to $14,212\mu s$. In (6) we let p be 0.05. We then derive the lower bound for different numbers of supported sessions using (6) assuming that there are 16 Mbytes of main memory that can be assigned for buffering. The upper bound of a period is determined by (8).

Let N be the number of multimedia sessions and T^{min}_{period} be the lower bound for the period. In this example we assume all multimedia sessions request the same disk bandwidth R^c. If T^{min}_{period} is chosen then there is no disk bandwidth left. By (4) we know that the buffer requirement is minimized. By (4), we have

$$B = \sum_{i=1}^{N}\{R^c(i) \times [T^{min}_{period} - \frac{R^c(i)}{R}T^{min}_{period}]\} = N \times R^c \times T^{min}_{period} \times (1 - \frac{R^c}{R})$$

The results of this analysis are summarized in Table 3. The third column presents the buffer requirement for N multimedia sessions when we chose T^{min}_{period}. The fourth column indicates the upper bound for period. In this case, the whole 16 Mbytes of memory are assigned for buffering. This allows us to use the least amount of disk bandwidth.

In our layout model, a period T_{period} is equal to the sum of all durations assigned multimedia sessions and the disk latency for switching service between multimedia sessions plus the leftover used for other non-real-time process (Fig. 2). The percentage P of disk bandwidth

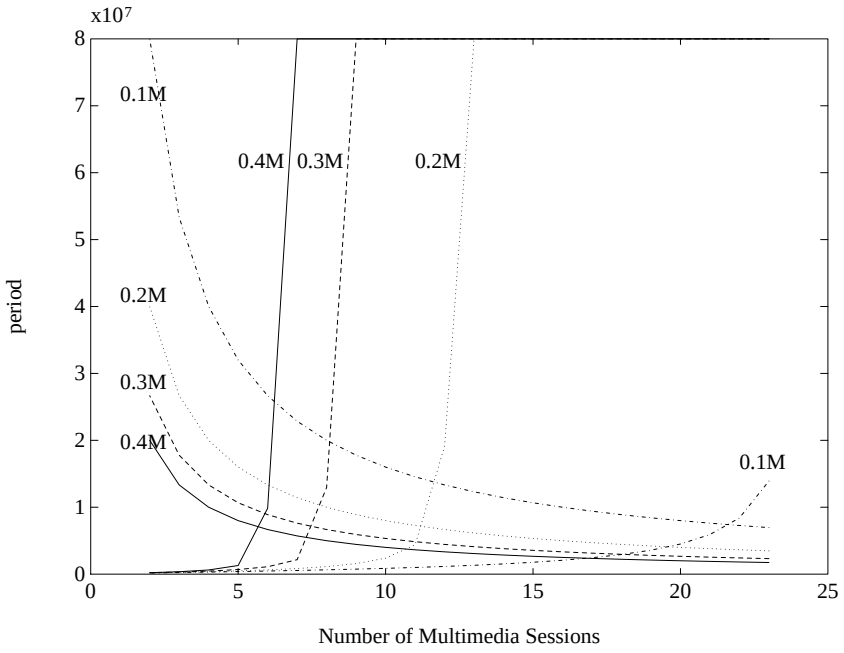

Figure 5: Number of sessions vs. period length

consumed by multimedia sessions can be considered as the interval assigned to multimedia session, plus disk latency lost in task switching between multimedia sessions, divided by the length of the period:

$$P = \frac{\sum_{i=1}^{N} T(i) + \sum_{i=1}^{N} T_{latency}(i)}{T_{period}^{max}} = \frac{N \times T_{period} \frac{R^c}{R} + N \times T_{latency}}{T_{period}^{max}}$$

In the fifth column, we show the percentage of disk bandwidth consumed by the multimedia sessions when the upper bound T_{period}^{max} is chosen.

In Fig. 5, when we increase the number of supported sessions, the buffer requirement for lower bound and disk bandwidth requirement for upper bound increases. If there are five multimedia sessions accessing the file system, the system can perform within these constraints, but it cannot accept additional multimedia sessions. An additional session causes the request for a 28,163,000 byte buffer and 100.8% of disk bandwidth, both of which exceed the capacity of the system.

Fig. 5 also shows the effect of varying the compression rate to reduce the bandwidth required for any (video) session and increase the number of multimedia sessions supported per

device. For video data, a compression ratio in the range of 1:10 to 1:100 is not unreasonable.

4.5 Consideration for Choosing a Period

Two hard requirements must be met when choosing the length of a period, otherwise the system cannot work. A period must be greater than T_{period}^{min} to meet the bandwidth requirement and less than T_{period}^{max} to meet the buffer requirement. These constraints are summarized as:

$$T_{period}^{max} > T_{period} > T_{period}^{min} \tag{9}$$

A new multimedia session can be accepted only it satisfies this relationship. Fig. 5 illustrates the ranges of sessions supported that satisfy these constraints. The region enveloped by the lower bound and upper bound is safe. In Table 3 for the sixth session the lower bound of period T_{period}^{min} is $14,013\ ms$, the upper bound T_{period}^{max} is $6,667\ ms$. Since $T_{period}^{min} > T_{period}^{max}$, we know the file system can not accept six multimedia sessions at the same time.

We estimate the upper and lower bound very conservatively (due to the large m assumed). The real upper bound can be larger and the lower bound can be lower than we derived. However, when the number of sessions increases, our estimates approach the real upper and lower bounds. There are two reasons to support our assumption. First, in the light load case, there are always enough resources for use. We are more concerned about the heavy load situation when the number of multimedia sessions m is large. Second, it is not necessary or wise to chose a period T_{period} close to either the upper or lower bound because of the degradation of the throughput of other non-real-time data transfers. For a general-purpose machine, a multimedia file system not only has to meet the hard requirements above, but also must leave enough bandwidth for these other non-real-time transfers. Let $A = D/T_{period}$ be the the percentage of disk bandwidth used to carry data from disk for non-multimedia job during every period T_{period}. For a set of multimedia sessions, we have the maximum value of A when $T_{period} = T_{period}^{max}$ [11]. This means if we increase the period T_{period} we can have more disk bandwidth left for the non-multimedia task.

From a memory perspective, a multimedia file system must minimize its buffer utilization to provide more free memory for other system tasks. By (7), when period $T_{period} = T_{period}^{min}$, the buffer requirement is minimized. By the above two results, we intend to increase the period to have more disk bandwidth left for non-multimedia traffic but minimize the period

to get more free memory for non-multimedia tasks. In the extreme case, if we minimize the T_{period} value, we minimize the buffer requirement and maximize free memory for other non-multimedia tasks. At the same time, the leftover for disk bandwidth is zero. Similarly, maximizing the T_{period} can free the maximum disk bandwidth for other non-multimedia processes to use but will also result in complete memory consumption. In this case, even if the disk has ample bandwidth available, no non-multimedia process can use it. Thus, these two soft requirements are in conflict.

To improve the response time for non-multimedia processes, we can change the period T_{period} dynamically with feedback from operating system to balance the resources allocation. For example, if there are tasks suspended due to disk bandwidth shortage and there is free memory space available, the file system can extend the period T_{period} in order to have more disk bandwidth to assign to the non-multimedia processes. On the other hand, if there are non-multimedia processes waiting for memory space and the disk is idle during the leftover interval. The file system can shrink the period T_{period} in order to free more memory space and to load more processes in the memory.

For a multimedia on-demand server, the file system need only provide service to multimedia processes. In this situation, we chose the lower bound to achieve the highest disk utilization. Given the physical disk characteristics, we can determine the buffer requirements. By Fig. 3 and (4), we know the buffer consumed is dominated by the period length T_{period}. By (5), the period length depends on the sum of random variables $T_{latency}(i)$. We assume the worst case, take the maximum value for for all task switching latencies $T_{latency}(i)$, and decide the period length. This assumes that starvation can never happen, when in practice it will only rarely happen. In a refined model, we define an acceptable rate $q = 1 - p$ of non-starvation, and derive the period length which guarantees a set of multimedia session can be accepted with at least a value of q for the probability of not starving. For example, we define $q = 95\%$. In this case, if there are five multimedia sessions in the system we can save 20.8% of available memory.

5 Conclusion

When a multimedia file system transfers data from a disk, it must guarantee that multimedia data arrive at the playout device with a minimum latency. It must also satisfy the timing requirements implied by the nature of the multimedia object (e.g., intermedia synchronization among media). However, disk seek latency is very significant and can be unpredictable

in a general-purpose file system.

In this paper we presented a physical data organization for supporting the storage of time-dependent multimedia data. We interleaved different media objects within a block to maintain timing among the objects during data storage and retrieval. We also refined existing contiguous allocation approaches to maximize disk bandwidth utilization and to prevent the enormous copying overhead during editing. Furthermore, we introduced a probabilistic model as a refinement of the round-round scheduling discipline that supports concurrent multimedia processes. Moreover, it reduces the amount of required buffering. We showed the acceptance conditions for additional multimedia sessions which include bandwidth and buffer constraints, and a means for balancing these two parameters to support the largest number of multimedia sessions originating from a single device.

References

[1] Anderson, D.P., Homsy, G.: A continuous media I/O server and its synchronization mechanism. Computer **24** (1991) 51-57

[2] Chen, H.J. Little, T.D.C.: A file system for multimedia applications. Tech. Rept. 12-09-1992, Multimedia Communication Laboratory, Boston University (1992)

[3] Christodoulakis, S., Faloutsos, C.: Design and performance considerations for an optical disk-based, multimedia object server. Computer **19** (1986) 45-56

[4] Ford, D.A., Christodoulakis, S.: Optimal placement of high probability randomly retrieved blocks on CLV optical disks. ACM Trans. on Information Systems **9** (1991) 1-30

[5] Gemmell, J., Christodoulakis, S.: Principles of delay-sensitive multimedia data storage and retrieval. ACM Trans. on Information Systems. **10** (1992) 51-90

[6] Kiessling, W.: Access path selection in databases with intelligent disc subsystems. The Computer Journal **31** (1988) 41-50

[7] Little, T.D.C., Ghafoor A.: Interval-based conceptual models for time-dependent multimedia data. To appear in IEEE Trans. on Data and Knowledge Engineering (1993)

[8] Yu, P.S., Chen, M.S., Kandlur, D.D.: Design and analysis of a grouped sweeping scheme for multimedia storage management. Proc. 3rd Intl. Workshop on Network and Operating System Support for Digital Audio and Video, San Diego (1992) 38-49

[9] Lougher, P., Shepherd, D.: The design and implementation of a continuous media storage server. Proc. 3rd Intl. Workshop on Network and Operating System Support for Digital Audio and Video, San Diego (1992) 63-74

[10] Nakajima, J., Yazaki, M., Matsumoto, H.: Multimedia/realtime extensions for the mach operating system. Proc. Summer 1991 Usenix Conf., Nashville, Tennessee (1991) 183-198

[11] Polimenis, V.G.: The design of a file system that supports multimedia: ICSI Tech. Rept. TR-91-020 (1991)

[12] Rangan, P.V., Vin, H.M., Ramanathan, S.: Designing an on-demand multimedia service. IEEE Communications Magazine **30** (1992) 56-64

[13] Rangan, P.V., Kaeppner, T., Vin, H.M.: Techniques for efficient storage of digital video and audio. Proc. Workshop on Multimedia Information Systems, Tempe, Arizona (1992) 68-85

[14] Rangan, P.V., Vin, H.M.: Designing file systems for digital video and audio. Proc. 13th Symposium on Operating Systems Principles (SOSP'91), Operating Systems Review **25** (1991) 81-94

[15] Seagate Technology: Seagate Wren 8 ST41650N product manual (volume 1). Publication No. 7765470-A (1991)

[16] Wells, J., Yang, Q., Yu, C.: Placement of audio data on optical disks. Proc. Intl. Conf. on Multimedia Information Systems, Singapore (1991) 123-134

[17] Wolf, L.C.: A runtime environment for multimedia communications. Proc. 2nd Intl. Workshop on Network and Operating Support for Digital Audio and Video, Heidelberg, Germany (1991)

[18] Yu, C., Sun, W., Bitton, D., Yang, Q., Bruno, R., Tullis, J.: Efficient placement of audio data optical disks for real-time applications. Communications of the ACM **32** (1989) 862-871