Physical Storage Allocation Policies for Time-Dependent Multimedia Data[1]

Huang-Jen Chen and T.D.C. Little

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

MCL Technical Report 06-15-1994

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 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, scheduling, time-dependent audio and video data, secondary storage, performance modeling.

[1]In *IEEE Trans. on Knowledge and Data Engineering*, Vol. 8, No. 5, October 1996, pp. 855-864. Portions of this work were presented at the 4th International Conference on Foundations of Data Organization and Algorithms (FODO'93). This work is supported in part by the National Science Foundation under Grant No. IRI-9211165.

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 program files or "text" documents. For example, a single feature-length JPEG-compressed movie can require over 2 Gbytes of memory for digital storage. Multimedia data can also be sensitive to timing during delivery. 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).

Table 1: Properties of Multimedia Data

Data Type	**Buffer/Bandwidth**
Single text document (HTML)	$\approx$ 80 Kb/document
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 $\times$ 480 pixels, 24 bits/pixel)	5.9 Mb/frame (177 Mb/s)
JPEG-compressed NTSC video	$\approx$ 7 Mb/s — 3.5 Mb/s
MPEG-I-compressed NTSC video	$\leq$ 1.5 Mb/s
MPEG-II-compressed NTSC video	$\leq$ 10 Mb/s
HDTV-quality video (uncompressed @ 1248 $\times$ 960 pixels, 24 bits/pixel)	28.7 Mb/frame (863 Mb/s)

A multimedia file system must reconcile the deficiencies of conventional storage subsystems. A typical 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 frequencies and resultant 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. 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 the management of multiple *sessions*. In contrast, the data allocation and scheduling strategies for conventional file systems are only concerned with the throughput, latency, and storage utilization for random access to files. Therefore, we seek to provide real-time behavior for a set of multimedia sessions originating from a single storage system; typically a conventional rotating-disk magnetic storage device. Note that we constrain ourselves to cases in which the aggregate bandwidth of sessions is less than or equal to the capacity provided by a single device; we do not consider RAID or other data distribution approaches in this context.

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 [11], as an operating system delivery problem [1, 12, 13, 22], as a physical disk modeling problem [6, 9, 10, 18], and as a physical data organization and performance problem [5, 7, 8, 14, 21, 23, 24]. Rangan et al. [16] 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. [15] 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 [14] 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 [8] 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 [24] 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. [21, 23] describe approaches to interleaving time-dependent data to support constant playout rates. Tobagi et al. [20] develop a Streaming RAID approach to handle video traffic on a disk array. Chiueh and Katz [4] propose a multi-resolution video representation scheme based on Gaussian and Laplacian Pyramids, which allows the parallel disk array to deliver only the absolute minimum amount of data necessary.

In this paper, we propose a physical data organization and file system 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 to 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.

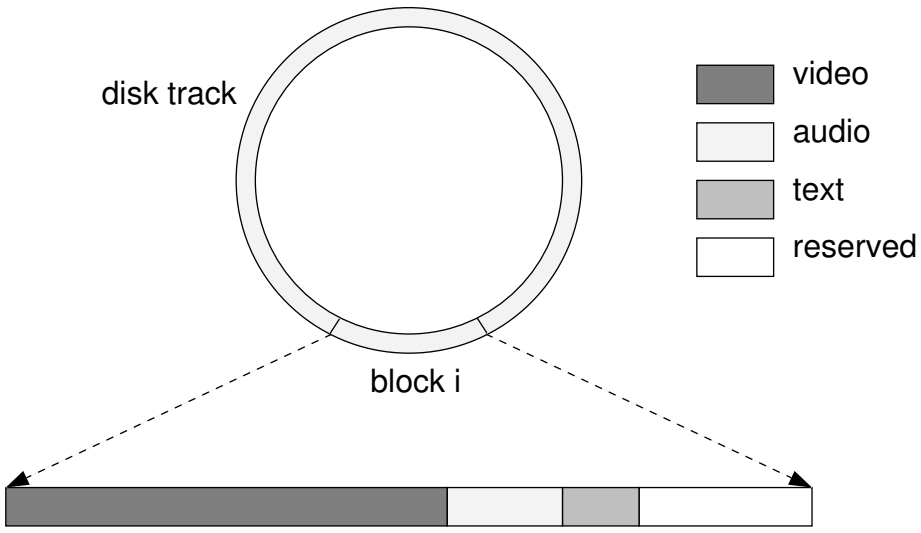

Figure 1: Physical Storage Organization for a Rotating Disk Device

A round-robin scheduling discipline is chosen for the service of multimedia sessions as in other work [12, 14, 17], permitting the disk to switch alternately between multimedia tasks and other non-real-time tasks. The file system achieves a high disk bandwidth utilization by assigning long disk reads or writes and thus sharing the seek and latency delays among a large number of bits read or written, resulting in a small overhead per transferred unit. We introduce a disk access schedule which is a refined model based on the work of Polimenis [14]. We show the constraints which must be satisfied to permit the acceptance of a set of multimedia sessions including bandwidth and buffer considerations. This work differs from other approaches in that we establish a probabilistic model for our disk access schedule to accept a set of sessions rather than using a guarantee of a worst case for the frequency of starvation.

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 disk seek latency. In Section 4 we show an access schedule for the disk and present a periodic service discipline for multimedia objects based on the probabilistic model. In Section 5 we describe how this schedule reduces the required buffering and increases the number of supported multimedia sessions.

Section 6 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 of multimedia applications because the disk latency between blocks of a media object is unpredictable [17]. The file system cannot guarantee satisfaction of the deadline for the retrieval of multimedia data.

We view a multimedia object as an entity comprised of mixed-type data components. 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 might be recorded at the same time [17]. 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 [17]. The homogeneous layout stipulates storage of single medium data in blocks without interleaving. However, timing relationships among media are stored as part of 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 proportional to its consumption rate.

In terms of *intra-media* 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 [16, 17, 21, 23]. 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 [16]. However, it is the most efficient method to utilize bandwidth [14]. This approach can be used for data that are seldom modified such as read-only digital entertainment video.

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 created based on the temporal relations for a multimedia object [11]. 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 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. 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

To support multimedia data requires the manipulation of large files and the support for large data consumption rates. It is the responsibility of the file system to organize the data for efficient storage and delivery within space and I/O bandwidth limitations. In most disk drive subsystems, the dominant inhibitor to achieving maximum disk I/O bandwidth is seek latency. However, seek latency can be reduced through contiguous writes or reads of

Table 2: Disk Parameters

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

time-dependent multimedia data. When these data become fragmented and discontinuous, effective disk bandwidth diminishes due to additional seek and rotational latencies involved in each discontinuity.

In our modeling approach, we consider latencies attributed to data fragmentation as well as session switching latencies. In the proposed scheduling approach, the disk is cycled through a set of independent multimedia sessions. Because sessions exist for many cycles and their access is unpredictable due to user interaction (e.g., start, stop, reverse), there are significant session switching latencies. 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 [19]. Parameters characterizing such a device are summarized in Table 2 using symbols adopted and extended from Kiessling [10].

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, cross a number of tracks, switch to a recording (writing) 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 the 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, and summarize them in Table 3.

Table 3: Derived Statistical Disk Behavior

Symbol	Equation	Value	Units
$T_{latency}$	$= T_{cross} + T_{switch} + T_{rotate}$		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

3.2 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 by considering each of these head motions in the complete scan.

We define the size of a block as M. The frequency for switching the head to the other disk P_{switch} is

$$P_{switch} = \frac{M}{S_{dt}}$$

The size of a cylinder is $S_{dt} \times N_{head}$. Thus, the frequency P_{cross} for the arm to cross to the next track is $P_{cross} = \frac{M}{S_{dt} \times N_{head}}$. Let T_M be the time to transfer a block from disk in the

optimal case. Then

$$T_M = \frac{M}{R_t} + \frac{M}{S_{dt}} T_{hh} + \frac{M}{S_{dt} \times N_{head}} [T_{start} + T_{tt}] = M \times T$$

T represents the minimum transfer time to transfer a single byte from the disk:

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

Let $R = \frac{1}{T}$ be the maximum transfer rate onto the disk. We normalize the disk bandwidth R as:

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

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.

4.1 Scheduling Layout Model

In the layout model of Polimenis [14], 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 Equ. 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} as follows:

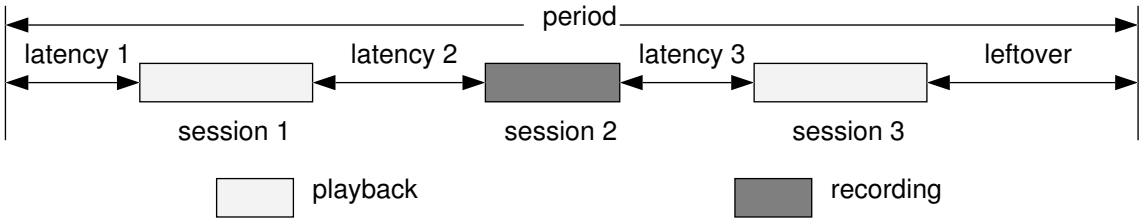

Figure 2: Layout Model

$$T(i) = \frac{R^c(i)}{R} T_{period} \tag{2}$$

In this equation, $R^c(i)$ represents the consumption rate for session i. Let the ith session contain k different media data (video, audio, text, etc.). 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 based on the round-robin scheduling model. 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 $n(i) = T_{period} \times R^c(i)$, then

$$\frac{n(1)}{R^c(1)} = \frac{n(2)}{R^c(2)} = ... = \frac{n(i)}{R^c(i)} \tag{3}$$

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 sufficient 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,

$$\frac{n}{R} + \sum_{i=1}^{m} T_{latency}(i) < T_{period} = \frac{n(i)}{R^c(i)} \tag{4}$$

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 Equ. 4,

$$\begin{aligned} R &> \frac{n}{\frac{n(i)}{R(i)} - \sum_{i=1}^{m} T_{latency}(i)} \\ &= \frac{1}{\frac{n(i)}{n}\frac{1}{R(i)} - \frac{\sum_{i=1}^{m} T_{latency}(i)}{n}} \end{aligned}$$

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

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

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,

$$R > \sum_{i=1}^{m} R^c(i) + R_{seek} \tag{5}$$

and

$$R_{seek} = \frac{(\sum_{i=1}^{m} R^c(i))^2 \times \sum_{i=1}^{m} T_{latency}(i)}{n - \sum_{i=1}^{m} R^c(i) \times \sum_{i=1}^{m} T_{latency}(i)} \tag{6}$$

The R_{seek} is the bandwidth wasted, or lost, when the disk head is switched between sessions.

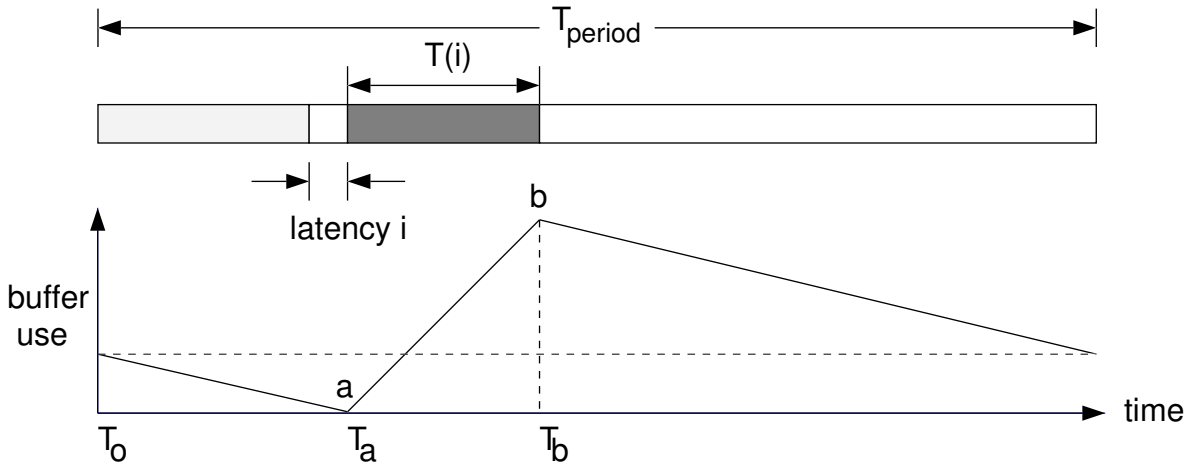

Figure 3: Buffer Consumption

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 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 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:

$$B > \sum_{i=1}^{m}[R^c(i) \times (T_{period} - T(i))] \tag{7}$$

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 Equ. 4, 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,

$$T_{period} > \sum_{i=1}^{m} T_{latency}(i) \times \frac{R}{R - \sum_{i=1}^{m} R^c(i)} = T \tag{8}$$

If we let U be the utilization, where $U = R/\sum_{i=1}^{m} R^c(i)$ and let C be the total latencies, then the minimum period for a set of multimedia sessions is [14]:

$$T_{period}^{min} = \frac{C}{1 - U} \tag{9}$$

In Equ. 8, $T_{latency}(i)$ represents the seek latency corresponding to the switch from session $i-1$ to session i. Because the next retrieval 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 3). The expectation $E(T)$ and variance $\sigma^2(T)$ of T in Equ. 8 are as follows:

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

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

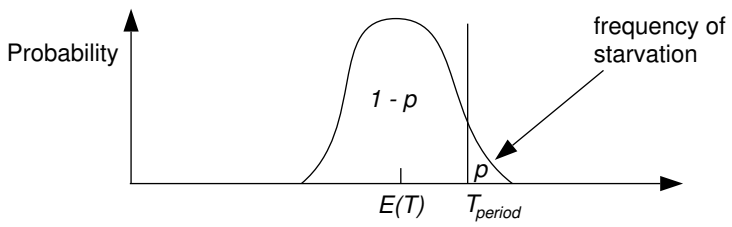

Figure 4: Distribution of T

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_{period}^{min}. Let p be the probability of starvation that can be tolerated for the mth session. By Chebychev's Inequality we have $P[|T_{period}^{min} - E(T)| > k] \leq \frac{\sigma^2(T)}{k^2} = p$, and therefore,

$$T_{period}^{min} \geq E(T) + \frac{\sigma(T)}{\sqrt{p}} \tag{10}$$

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

By Equ. 10, if we choose 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 10 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 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 determine whether to accept a set of multimedia sessions. The system must also provide sufficient buffering for each multimedia session. In the lightly loaded situation, there are always enough buffering to support multimedia sessions. However, buffering becomes significant when the number of multimedia sessions m is large. In this case, compared to the period T_{period}, the duration $T(i)$ assigned to each multimedia session is small. We simplify Equ. 7 by ignoring the $T(i)$ and the result is still valid:

$$B \cong T_{period} \times \sum_{i=1}^{m} R^c(i) \tag{11}$$

From the equation above, we see that the buffer requirements are dependent on the length of period T_{period}. Let B_{max} be the maximum buffer space that is available. There is an upper bound T_{period}^{max} 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 Equ. 11, we have:

$$T_{period}^{max} = \frac{B^{max}}{\sum_{i=1}^{m} R^c(i)} \tag{12}$$

Equs. 10 and 12 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.

Example 1 In this example, we assume all multimedia sessions request the same disk bandwidth R^c. 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 disk parameters from Tables 2 and 3 we pick the average disk latency $E(T_{latency})$ equal to $35,965\mu s$ and the standard deviation $\sigma_{latency}$ equal to $14,212\mu s$. For Equ. 10 we let p be 0.05. We then derive the lower bound for different numbers of supported sessions using Equ. 10 assuming the availability of 16 Mbytes of main memory that can be assigned for buffering. The upper bound of a period is then determined by Equ. 12.

Let N be the number of multimedia sessions and T_{period}^{min} be the lower bound for the period. If T_{period}^{min} is chosen then there is no disk bandwidth left. By Equ. 7 we know that the buffer requirement is minimized and we have

$$\begin{aligned} B &= \sum_{i=1}^{N} \{R^c(i) \times [T_{period}^{min} - \frac{R^c(i)}{R} T_{period}^{min}]\} \\ &= N \times R^c \times T_{period}^{min} \times (1 - \frac{R^c}{R}) \end{aligned}$$

Table 4: File System Performance for Example 1

N	**100 % Bandwidth Utilization**		**100 % Buffer Allocation**	
	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.

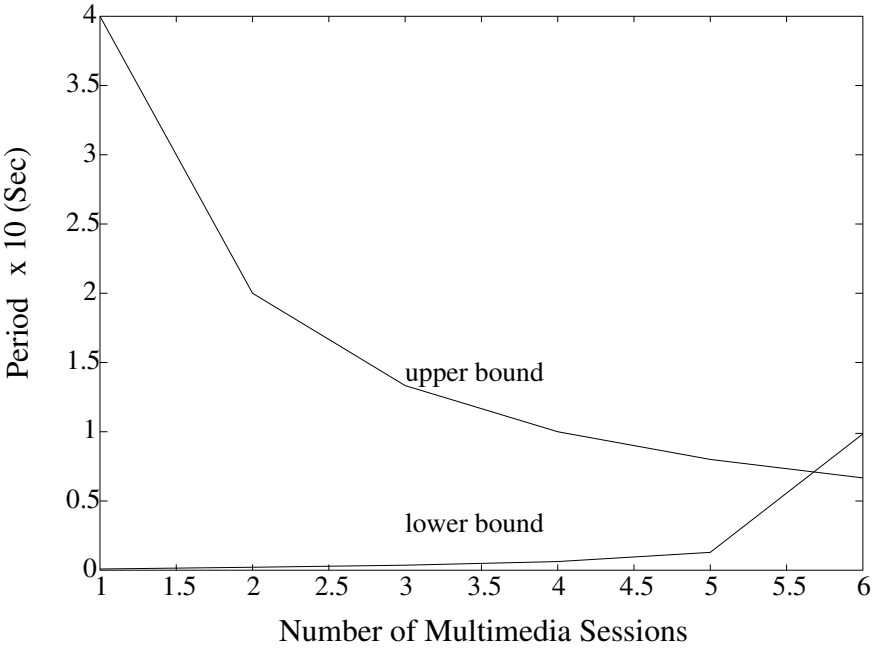

Figure 5: Number of Sessions vs. Period Length

The results of this analysis are summarized in Table 4. The third column presents the buffer requirement for N multimedia sessions when we chose T_{period}^{min}. The fourth column indicates the upper bound for period. In this case, the entire 16 Mbytes of memory are assigned to buffering, allowing us to minimize the use of disk bandwidth given the constraints.

In our layout model, a period T_{period} is equal to the sum of all durations assigned to multimedia sessions plus the session switching latency between sessions plus the leftover used for other non-real-time process (Fig. 2). The percentage P of disk bandwidth consumed by multimedia sessions can be considered as the interval assigned to the multimedia sessions, 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 of Table 4 we show the percentage of disk bandwidth consumed by the multimedia sessions when the upper bound T_{period}^{max} is chosen.

When we increase the number of supported sessions, both buffer and bandwidth requirements will increase (Fig. 5). If there are five multimedia sessions accessing the file system, the system can perform within these constraints, but it cannot accept additional multimedia sessions. In this case 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.

5 Discussion

From the analysis presented in Sections 3 and 4, it is appropriate to describe considerations for choosing the length of a round-robin scheduling period, and to describe the impact of session consumption rates.

5.1 Consideration for Choosing a Period

Two hard requirements must be met when choosing the length of a period, otherwise the system cannot function for a given workload. 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:

$$T_{period}^{max} > T_{period} > T_{period}^{min} \tag{13}$$

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 4, 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 cannot 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 have derived. However, when the number of sessions increases, our estimates approach the real upper and lower bounds. There are two justifications for our assumption. First, in the lightly loaded case, there are always enough resources for use. We are more concerned about the heavily loaded situation in which 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 bounds 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 read data from the disk for non-real-time jobs during every period T_{period}. For a set of multimedia sessions, A is maximized when $T_{period} = T_{period}^{max}$ [14]. This means if we increase the period T_{period} we can have additional disk bandwidth leftover for non-real-time tasks.

From a memory perspective, a multimedia file system must minimize its buffer utilization to make memory available for other system tasks. From Equ. 11, we see that when period $T_{period} = T_{period}^{min}$, the buffer requirement is minimized. From the above two results, we seek to increase the period for more disk bandwidth for non-real-time traffic but also to reduce the period for more free memory for non-real-time tasks. In the extreme case, if we minimize the T_{period} value, we minimize the buffer requirement and maximize free memory for other non-real-time 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-real-time processes to use but will also result in complete memory consumption. In this case, even if the disk has ample bandwidth available, no non-real-time process can use it. Thus, these two soft requirements are in conflict.

To improve the response time for non-real-time processes, we can change the period

T_{period} dynamically with feedback from the operating system to balance resource allocation. For example, if there are tasks suspended due to disk bandwidth shortages and there is free buffer space available, the file system can extend the period T_{period} in order to have more disk bandwidth to assign to non-real-time processes. If there are non-real-time processes waiting for memory and the disk is idle during the leftover interval, the file system can shrink the period T_{period} in order to free memory for additional non-real-time processes.

Table 5: Refined Model vs. Worst Case

	Refined Model		**Worst Case**	
N	Period (ms)	Buffer Allocation (bytes)	Period (ms)	Buffer Allocation (bytes)
1	104	35,000	86	29,000
2	214	144,000	213	143,000
3	385	386,000	421	423,000
4	706	946,000	823	1,103,000
5	1,577	2,641,000	1,924	3,222,000

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 Equ. 7, we know that the amount of consumed buffer space is determined by the period length T_{period}. By Equ. 8, the period length depends on the sum of random variables $T_{latency}(i)$. We assume the worst case, take the maximum value 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 sessions can be accepted with at least a probability q of not starving. In Table 5, we define $q = 95\%$. In this case, if there are five multimedia sessions in the system we can save 20.8% of available memory.

5.2 Consumption Rate for Multimedia Sessions

There are several factors that effect the consumption rate for a multimedia session. The most important factor is the data compression ratio affecting the multimedia data. For example, for video data, a compression ratio in the range of 1:10 to 1:100 is not uncommon.

In Fig. 6, we show a set of constrained bandwidth-buffering regions for sessions with

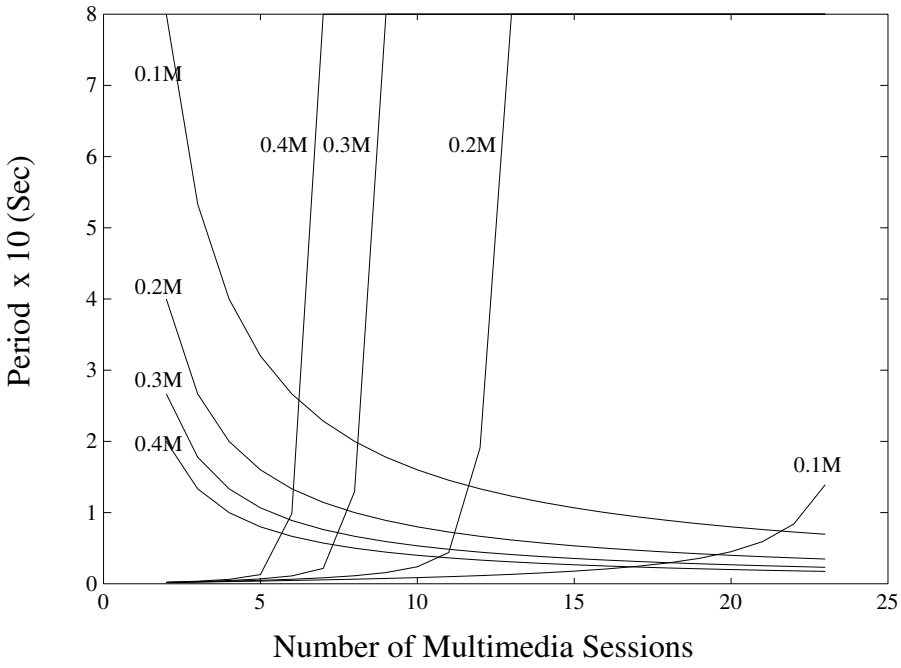

Figure 6: Number of Sessions vs. Period Length

differing data rates due to a range of compression rates. Parameters are otherwise identical to that of Example 1. This figure illustrates the safe region for various consumption rates and allows the selection of period length T_{period} and buffer use for a given number of sessions.

By varying the compression rate we can reduce the bandwidth required for any (video) session and increase the number of multimedia sessions supported per device. Assuming a uniform bandwidth requirement for each session, Fig. 7 shows the number of sessions supported for a range of consumption ratios (bandwidth).

5.3 Variable Video Encoding Rates

In our analysis we have assumed constant-bit-rate (CBR) video encoding. This assumption greatly simplifies analysis and is reasonable based on the MPEG-I ISO 11172 CBR option. However, we recognize that CBR video is not ubiquitous. Our model can be modified to accommodate variable-bit-rate (VBR) compression schemes by aggregating several VBR streams together [3]. For this situation, not only is the disk production rate unpredictable but the display consumption can be unpredictable as well, particularly if software-only decompression of video is used. We view disk seek latencies and the transfer time of VBR

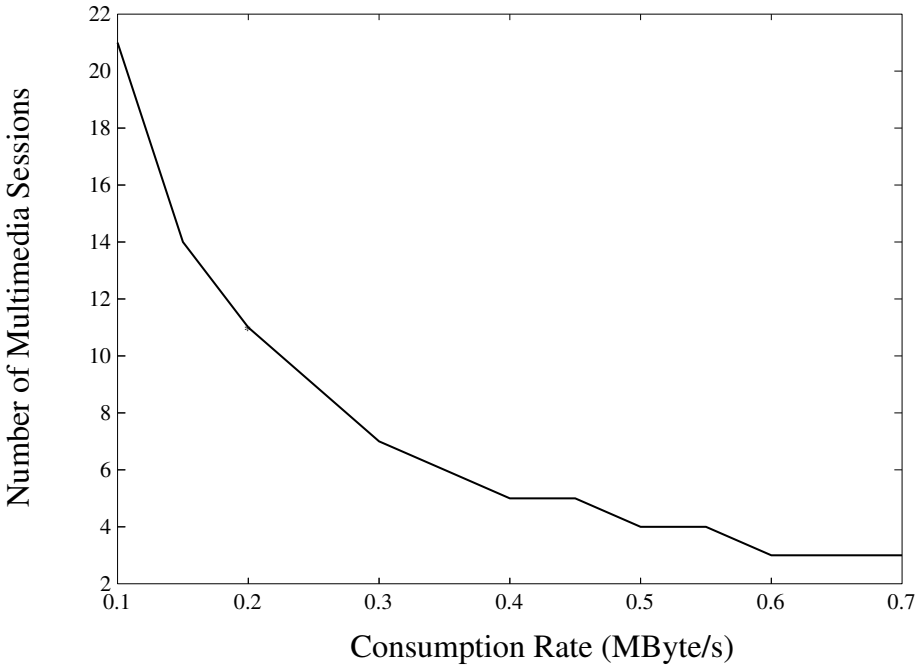

Figure 7: Consumption Rate vs. Number of Sessions

streams as random variables and use a similar probabilistic model to guarantee that the frame loss ratio will be under a given threshold. Moreover, in a related work, we describe an algorithm to reduce the impact of frame losses due to disk starvation [3].

6 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., synchronization among media). However, disk seek latency can be very significant and is 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. Furthermore, we introduced a probabilistic model as a refinement of the round-round scheduling discipline that supports concurrent multimedia sessions. It was found to reduce the amount of re-

quired buffering during data transfer from storage. We showed the acceptance conditions for additional multimedia sessions including 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., and G. Homsy, "A Continuous Media I/O Server and Its Synchronization Mechanism," *Computer*, Vol. 24, No. 10, October 1991, pp. 51-57.

[2] Chen, H.J., and T.D.C. Little, "Physical Storage Organizations for Time-Dependent Multimedia Data," *Proc. 4th Intl. Conf. on Foundations of Data Organization and Algorithms,* Evanston, IL, October 1993, pp. 19-34.

[3] 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," *Proc. 2nd IEEE Intl. Conf. on Multimedia Computing and Systems,* Washington D.C., May 1995, pp. 65-72.

[4] Chiueh, T.C. and R.H. Katz, "Multi-Resolution Video Representation for Parallel Disk Array," *Proc. 1st ACM Intl. Conf. on Multimedia,* Anaheim, CA, August 1993, pp. 401-409.

[5] Christodoulakis, S., and C. Faloutsos, "Design and Performance Considerations for an Optical Disk-based, Multimedia Object Server," *Computer*, December 1986, pp. 45-56.

[6] Bitton, D., "Disk Shadowing," *Proc. 14th Intl. VLDB Conf.,* Los Angeles, CA, 1988, pp. 331-338.

[7] Ford, D.A., and S. Christodoulakis, "Optimal Placement of High-Probability Randomly Retrieved Blocks on CLV Optical Disks," *ACM Trans. on Information Systems*, Vol. 9, No. 1, January 1991, pp. 1-30.

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

[9] Gray, J., B. Horst, and M. Walker, "Parity Striping of Disk Arrays: Low Cost Reliable Storage with Acceptable Throughput," *Proc. 16th Intl. VLDB Conf.,* 1990, pp. 152.

[10] Kiessling, W., "Access Path Selection in Databases with Intelligent Disc Subsystems," *The Computer Journal,* Vol. 31, No. 1, February 1988, pp. 41-50.

[11] Little, T.D.C., and A. Ghafoor, "Interval-Based Conceptual Models for Time-Dependent Multimedia Data," *IEEE Trans. on Knowledge and Data Engineering*, Vol. 5, No. 4, August 1993, pp. 551-563.

[12] Lougher, P., and D. Shepherd, "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, November 1992, pp. 63-74.

[13] Nakajima, J., M. Yazaki, and H. Matsumoto, "Multimedia/Realtime Extensions for the Mach Operating System," *Proc. Summer 1991 Usenix Conf.*, Nashville, Tennessee, June 1991, pp. 183-198.

[14] Polimenis, V.G., "The Design of a File System that Supports Multimedia," ICSI Tech. Rept. No. TR-91-020, March, 1991.

[15] Rangan, P. V., H. M. Vin, and S. Ramanathan, "Designing an On-Demand Multimedia Service," *IEEE Communications Magazine,* July 1992 pp. 56-64.

[16] Rangan, P.V., and H.M. Vin "Efficient Storage Techniques for Digital Continuous Multimedia," *IEEE Trans. on Knowledge and Data Engineering*, Vol. 5, No. 4, August 1993, pp. 564-573.

[17] Rangan, P.V., and H.M. Vin, "Designing File Systems for Digital Video and Audio," *Proc. of the 13th Symp. on Operating Systems Principles (SOSP'91)* and Operating Systems Review, Vol. 25, No. 5, October 1991, pp. 81-94.

[18] Ruemmler, C., and J. Wilkes, "An Introduction to Disk Drive Modeling," *Computer,* Vol. 27, No. 3, March 1994, pp. 17-28.

[19] *Seagate Wren 8 ST41650N Product Manual (Volume 1)*, Publication No. 7765470-A, Seagate Technology, June 1991.

[20] Tobagi, F.A., J. Pang, R. Baird, and M. Gang, "Streaming RAID – A Disk Array Management System for Video Files," *Proc. 1st ACM Intl. Conf. on Multimedia,* Anaheim, California, August 1993, pp. 393-400.

[21] Wells, J., Q. Yang, and C. Yu, "Placement of Audio Data on Optical Disk," *Proc. Intl. Conf. on Multimedia Information Systems*, Singapore, January 1991, pp. 123-134.

[22] 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, November 1991.

[23] Yu, C., W. Sun, D. Bitton, Q. Yang, R. Bruno, and J. Tullis, "Efficient Placement of Audio Data Optical Disks for Real-Time Applications," *Communications of the ACM,* Vol. 32, No. 7. July 1989, pp. 862-871.

[24] Yu, P.S., M.-S. Chen, and D.D. Kandlur "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, November 1992, pp. 38-49.