Disk Scheduling for
Multimedia Database Applications

Jayant R. Haritsa M. B. Karthikeyan
SERC, Indian Institute of Science, Bangalore

Abstract

A special aspect of multimedia database applications is that they require continuity in their data access. The maximum number of users who can simultaneously use the multimedia database is limited by this continuity requirement. Formulas for computing the maximum number of users and the associated memory requirement have been presented in the literature. However, these formulas do not take into account the disk scheduling algorithm used by the multimedia server. In this paper, we analyze the effect of this factor and show that practical disk scheduling algorithms drastically reduce the amount of memory required to support the continuity constraint. A simulation-based study is used to confirm these effects and to highlight the performance characteristics for various types of media.

Keywords: Multimedia, Database, Scheduling

1 Introduction

The last few years have seen a great deal of interest and research in the area of multimedia database systems [TKDE93, JSAC93, TCJ93]. In particular, the design of multimedia servers for *continuous media*, that is, multimedia requiring continuous real-time flow, has received considerable attention [Vin93, Kenc93a]. A commercial example of this type of environment is **"video-on-demand"**, wherein a database of movies is stored in a server which electronically provides services equivalent to those provided by video rental stores. Here, movies are stored in digitized compressed form on high-capacity disks at the server, and subscribers access the server via a high-speed network. Subscribers can ask for selected movies to be shown on their home monitors. It is the database server's responsibility to ensure satisfactory play-back of these movies, that is, it has to ensure the following two conditions are met [Rang92]:

- Continuity of video and audio playback
- Synchronization of video and audio playback

The first condition requires that, for each subscriber, the playback of the movie should proceed at its real-time rate. This means that the server should admit a new subscriber *only if* this does not result in discontinuity for any of the current subscribers. The second condition requires that, for each subscriber, the playback of the audio and video segments of the requested movie should be temporally coordinated. Research into techniques for guaranteeing continuity is reported in [Vin93, Kenc93a], while schemes for ensuring synchronization are described in [Litt90a, Rang93a].

Two factors affect the movie database server's ability to maintain the continuity requirement: (1) the initial placement of media blocks on disk, and (2) the subsequent scheduling of media block transfers from the disk. Solutions to the media placement problem are inherently limited due to the one time nature of the placement process (reorganization of multimedia data is prohibitively expensive because files are huge) [Kenc93a]. This then means that, in the long run, system performance becomes primarily dependent on the efficiency of the scheduler. In this paper, we focus our attention on the influence of disk scheduling in meeting continuity requirements for multiple subscribers.

Disk Scheduling

The normal mode of operation for a movie database server is to multiplex among all the currently admitted subscribers, transferring a finite number of media blocks for each subscriber before switching to the next subscriber [Vin93]. A full sequence of serving all subscribers is called a **service round**, and the database server repeatedly executes service rounds until all the requests are completed. Note that the rate of transfer of successive blocks for an individual subscriber is governed by the relative placement of disk blocks belonging to the particular movie, while the time to switch from one subscriber request to the next is dependent on the relative placement of the selected movies. The blocks transferred during each service round are made available for playback only at the beginning of the following service round and are therefore buffered in the interim period.

Based on the above framework, we can envisage three levels of scheduling in a multimedia database server: an **admission scheduler**, a **disk scheduler**, and a **quantity scheduler**. The admission scheduler decides whether or not to admit a new subscriber based on the continuity requirement. The disk scheduler decides, in each service round, the *order* in which subscribers are served. Finally, the quantity scheduler decides the *number* of blocks to be transferred for each subscriber in a service round.

Formulas for computing the *maximum* number of subscribers that may be simultaneously supported by the movie database server without losing continuity, and the associated memory requirement, are derived in [Vin93]. The parameters of these formulas include the subscriber playback rates, the "granularity" of video storage (number of frames per media block), the average "scattering parameter" (how far apart successive blocks of a movie are scattered on the disk, that is, the placement), the average size of a video frame in bits, and the disk transfer rate. An additional input is the quantity scheduler, that is, the number of blocks transferred per subscriber in each service round. For example, the number of blocks transferred by the Quality Proportional Multi-Subscriber Servicing algorithm [Vin93] is proportional to the playback rate of the movie selected by the subscriber.

A significant and perhaps surprising omission in the above parameter list is the *disk scheduler* which decides the order of subscriber service in each service round – we address this issue in this paper. The effect of disk scheduling is analyzed in detail and it is shown that disk scheduling has no impact on the subscriber limit since data transfer is the main bottleneck which determines this limit. However, practical disk scheduling algorithms such as FCFS or SCAN [Teor72] significantly reduce the amount of *memory* required to support a given set of subscribers. In fact, the SCAN algorithm is optimal in this regard. A simulation-based study is made to confirm these effects and to highlight the performance characteristics for various types of media. To the best of our knowledge, this is the first research work to consider the effects of specific disk scheduling policies with respect to maintaining multimedia playback continuity.

Organization

The rest of this paper is organized as follows: We present, in Section 2, an analysis of the effects of disk scheduling on continuous multimedia performance. The performance model is described in Section 3, and the results of the simulation experiments are highlighted in Section 4. We conclude, in Section 5, with a summary of the results presented here.

2 Performance Analysis

In this section, we first present the subscriber limit derived in [Vin93] and show that disk scheduling has no impact on this limit. Subsequently, we demonstrate that practical disk scheduling algorithms significantly reduce the amount of memory required to support the continuity constraint.

In the following discussion, we use the notation followed in [Vin93], which is summarized in the table below:

R_{pl}	Video playback rate	frames/sec
η_{vs}	Granularity of video storage	frames/media block
s_{vf}	Size of a video frame	bits
l_{ds}	Scattering parameter	sec
l_{max}^{seek}	Maximum disk seek time	sec
l_{max}^{rot}	Maximum disk rotational latency	sec
R_{dr}	Disk data transfer rate	bits/sec

We assume here (and in the remainder of this paper) that the database server utilizes the Quality Proportional Multi-Subscriber Servicing (QPMS) quantity scheduler mentioned in the Introduction, since this scheduler is optimal with respect to maximizing the number of simultaneous subscribers [Vin93]. With this scheduler, for each subscriber i, the number of blocks transferred (B_i) in a service round is proportional to the playback rate (R_{pl}^i) of the selected movie. That is, $B_i = k * R_{pl}^i$, where k is the proportionality constant. The total memory required to temporarily store the blocks transferred in each service round of n subscribers is therefore $B_{tot} = 2 * k * n * R_{pl}^{avg}$, assuming double buffering. For a given subscriber set, therefore, the value of k directly determines the memory required to maintain continuity.

2.1 Subscriber Limit

A limit on the number of subscribers exists because the retrieval rate for each of the subscribers can be satisfied if and only if the service time per round does not exceed the minimum of the playback durations of all subscribers. Based on this fact, it was shown in [Vin93] that maintaining continuity with multiple subscribers requires the following equation to hold true (for each service round):

$$n * \alpha + n * k * \beta \leq k * \delta \tag{1}$$

where
n is the number of subscribers;
α is the average time taken to switch from one subscriber to the next;
k is the proportionality constant of the QPMS quantity scheduler;
$\beta = R_{pl}^{avg} * (l_{ds}^{avg} + \frac{\eta_{vs} * s_{vf}^{avg}}{R_{dr}})$ is the average transfer time for a single subscriber's blocks; and
$\delta = \eta_{vs}$ is the granularity of video storage.

From the above equation, it is straightforward to see that n_{max}, the subscriber limit, is given by

$$n_{max} = \left\lceil \frac{k\delta}{k\beta + \alpha} - 1 \right\rceil \tag{2}$$

The value of k can become arbitrarily large since an unlimited supply of buffers is assumed to be available. Also, the parameter α which denotes the average subscriber switching time, is bounded above by $(l_{seek}^{max} + l_{rot}^{max})$. Therefore, in the limit, the above expression reduces to

$$n_{max} = \left\lceil \frac{\delta}{\beta} - 1 \right\rceil \tag{3}$$

The effect of disk scheduling is captured in the parameter α. Since this term disappears from the expression for n_{max}, it is clear that in the unlimited buffer case, it is data transfer which becomes the performance bottleneck, not disk scheduling. Therefore, the choice of disk scheduling algorithm has no impact on the subscriber limit in this situation.

2.2 Memory Reduction

Given a set of n subscribers, where $n \leq n_{max}$, the value of k can be determined by the following expression, which is obtained by rearranging Equation (1):

$$k = \frac{n\alpha}{\delta - n\beta} \tag{4}$$

As described earlier, k is directly related to the total memory requirement. Therefore, the above expression indicates the number of buffers required to support the n subscribers.

The value assigned to α, the average time for the disk to go from serving one subscriber to serving the next subscriber, in [Vin93] is $\alpha = l_{seek}^{max} + l_{rot}^{max}$, that is, the disk head may have to do a full-stroke seek and a full rotation (the full-stroke is the distance from the innermost cylinder to the outermost cylinder). The reason for taking this extreme view is driven by the need to provide *deterministic* guarantees of service to admitted subscribers. However, as we will show here, practical disk scheduling algorithms significantly reduce the value of α and thereby considerably reduce the memory required to maintain continuity for a given set of subscribers, since k is directly proportional to α (Equation (4)).

The two disk scheduling algorithms that we consider are the well-known FCFS and SCAN scheduling algorithms [Teor72]. The FCFS scheduler services subscribers in arrival order, that is, in each round, subscribers are serviced according to their original movie request time. The SCAN scheduler, on the other hand, operates quite differently: In each round, it sweeps the disk head from one end of the disk to the other servicing all subscriber requests that it encounters along the way. The direction of the sweep (inwards or outwards) toggles from one round to the next. In practice, the span of the end points of the disk head sweep are restricted to the extreme cylinders for which requests exist, and do not have to cover the entire physical disk.

In the remainder of this paper, we will denote the scheduler of [Vin93] by QUAD (QUantity–ADmission) in order to simplify the discussion and to highlight the absence of disk scheduling.

2.2.1 Performance of SCAN

As mentioned earlier, the SCAN algorithm services all subscribers in a given round with one sweep of the disk. Therefore, the upper bound on the total seek distance is merely the full-stroke. This bound implies that unlike the QUAD algorithm where a seek time overhead of $n * l_{max}^{seek}$ may be incurred for a service round of n subscribers, the SCAN algorithm *always* incurs a seek time overhead of at most l_{max}^{seek}, independent of the number of subscribers. Therefore, the value of k using SCAN is given by

$$k_{SCAN} = \frac{l_{max}^{seek} + n * l_{max}^{rot}}{\delta - n\beta} \tag{5}$$

Taking the ratio of Equations (4) and (5), we obtain the <u>memory usage factor</u> of SCAN with respect to QUAD to be

$$\begin{aligned} muf_{SCAN} &= \frac{k_{SCAN}}{k_{QUAD}} \\ &= \frac{l_{max}^{seek} + n * l_{max}^{rot}}{n(l_{max}^{seek} + l_{max}^{rot})} \\ &= \frac{(l_{max}^{seek}/n) + l_{max}^{rot}}{l_{max}^{seek} + l_{max}^{rot}} \end{aligned} \tag{6}$$

With increasing n, the above ratio reduces to

$$muf_{SCAN} = \frac{l_{max}^{rot}}{(l_{max}^{seek} + l_{max}^{rot})} \tag{7}$$

In practical disks, the rotation time is typically considerably smaller than the maximum seek time. Therefore, if we assume that rotation time is bounded in $(0, l_{max}^{seek})$, it means that muf_{SCAN} is in the interval $(0, 0.5)$. This means that SCAN will typically require less than half the memory needed by the QUAD scheduler to support the same set of subscribers. An important point to note here is that no assumption was made about the subscriber movie request pattern in deriving the memory usage factor. This means that the SCAN scheduler is robust with respect to its performance improvement.

The above analysis was based on the worst case assumption that a full rotation is made during every subscriber switch. On the average, however, we can reasonably expect that the rotation time to access the first block is only $(l^{rot}_{max}/2)$ [Tane87]. Using this value instead of l^{rot}_{max} in the numerator of Equation (6) means that, in the typical case, the memory usage factor of SCAN is in the interval $(0, 0.25)$ relative to the QUAD scheduler.

In a general computer system environment, where user requests may arrive in random fashion, algorithms such as CVSCAN [Gei87] and others have been observed to outperform SCAN with respect to the average response time metric. However, in the video-on-demand environment, subscriber requests are periodic, localized, and service is made in rounds. For such workloads, the SCAN algorithm incurs the least seeking distance as compared to any other algorithm. This is because any algorithm will atleast have to seek, in every round, the distance between the two extreme (cylinder-wise) subscribers. Since SCAN serves *all* subscribers within this seek distance, it can do no worse than any other algorithm. Therefore, the SCAN algorithm is *optimal* with respect to memory reduction since it minimizes the value of α.

2.2.2 Performance of FCFS

Moving on to the FCFS scheduler, we observe that there are situations where its performance may be as poor as that of the QUAD scheduler. For example, if the subscriber request sequence alternates between movies on the innermost cylinder and the outermost cylinder, then the FCFS scheduler will pay a switching cost of $l^{seek}_{max} + l^{rot}_{max}$ during each subscriber switch.

If we consider the typical case, however, where subscriber requests are uniformly distributed over the disk, the average disk seek time is $l^{seek}_{max}/3$ for an FCFS scheduler [Coff73]. If we also assume that the rotation time, on average, is $(l^{rot}_{max}/2)$, the memory usage factor of FCFS with regard to QUAD is

$$muf_{FCFS} = \frac{(l^{seek}_{max}/3) + (l^{rot}_{max}/2)}{l^{seek}_{max} + l^{rot}_{max}} \tag{8}$$

Again, assuming that l^{rot}_{max} is bounded in the interval $(0, l^{seek}_{max})$, it is easy to see that muf_{FCFS} is bounded in the interval $(\frac{1/3}{1} = 0.33, \frac{1/3+1/2}{1+1} = 0.42)$.

The above analysis indicates that the SCAN scheduler provides good performance under all conditions, while the FCFS scheduler is reasonable in the average case. We conducted a simulation study to confirm the above derivations and to explore the performance improvements on a variety of media types and storage disks. The performance model and the results of our experiments are described in the following two sections.

3 Performance Model

A model of a video/audio on demand server was used to study the effect of practical disk scheduling policies (SCAN and FCFS) in reducing the memory required to support multiple simultaneous user requests. The model consists of a disk on which movies are stored, and the three schedulers (admission, disk, and quantity) for handling subscriber movie requests. The disk has a fixed number of cylinders, and each movie is stored in a contiguous group of cylinders. A scattering parameter is associated with the storage of successive media blocks of a movie, as explained in the Introduction.

At the beginning of each service round, a new subscriber arrives and requests access to the movie database. A subscriber movie request is characterized by a user-identifier (which uniquely identifies the subscriber), a movie-identifier (which identifies the requested movie), and the playback rate (which could range from normal playback to fast-forward or slow-motion). We assume that the movie requests are uniformly distributed across the disk. Each admitted user generates a stream of media block requests, with each new request being generated when the blocks transferred in the previous service round have all been displayed. Since we are interested in finding the maximum number of simultaneous subscribers, only new user arrivals are modeled and admitted users never depart from the server. When new subscribers cannot be accomodated without losing continuity, the system is said to be saturated.

The resources in the model are a disk and a pool of buffers. The disk is characterized by the set of parameters listed in the table below. Associated with each cylinder of the disk is a queue in which the periodic requests of subscribers are entered. We assume that it takes half a rotation of the disk to access the first block of the set of blocks to be transferred.

Parameter	Meaning
NumCylinders	Number of cylinders
MinSeekTime	Neighboring Cylinder seek time
MaxSeekTime	Full stroke time
RotSpeed	Disk rotational speed
DiskBW	Disk Bandwidth

We consider three types of media: HDTV-Video, NTSC-Video and CD-Audio. These media types represent a broad range of data rate requirements ranging from the most demanding (HDTV) to the least demanding (CD-Audio). These rate requirements are shown in the following table:

Media type	Data Rate Requirement
HDTV	2 MBytes/sec
NTSC	512 KBytes/sec
CD-Audio	180 KBytes/sec

4 Experiments and Results

In this section, we present the performance results from our experiments comparing SCAN and FCFS with respect to QUAD (the [Vin93] scheduler) in a video/audio-on-demand environment. The simulator used to obtain the results is implemented in the SIMSCRIPT II.5 simulation language. As discussed before, the requests of all admitted subscribers are serviced in rounds. For simplicity, we assume that all subscribers have the same playback rate. In all the experiments, an unlimited pool of buffers is available to the server. The total number of buffers required in each service round is computed according to the scheme described in Section 2. This number and the associated number of subscribers is continously logged in each experiment.

The performance evaluation was carried out for all the three media types described in Section 3 under both FCFS and SCAN scheduling policies. The experiments were conducted for two disk configurations: (1) a custom-made multimedia disk with a very high bandwidth of 128 MBps, and (2) a standard disk with a bandwidth of 4 MBps. Several runs were made of each experiment until sufficient confidence was gained in the results. Only statistically significant differences are discussed here.

The performance metric in all our experiments is the amount of memory required to support a given number of subscribers. We also plot the memory usage factor of SCAN and FCFS, that is, the ratio of the number of buffers used by these algorithms relative to that of QUAD.

The simulator was instrumented to generate other statistical information such as the the average seek time during each service round and the total service round time. These secondary measures were used to verify the correctness of the implementation and to lend insight into the observed performance behavior.

In all the experiments, the number of videos/audios stored on disk was 100 and it was assumed that all videos/audios require the same amount of disk space. The recording rate for both video types (HDTV and NTSC) was 30 blocks per second and for CD-Audio was 10 blocks per second (each block stores 18 KBytes of audio data). The storage granularity, that is, the number of frames per media block, is 1 for the video data and 10 for the audio data.

4.1 Experiment 1: Multimedia Disk

The objective of our first experiment was to study the effect of SCAN and FCFS scheduling policies on the buffer space required to guarantee continuity of simultaneous requests for movies stored on a high-bandwith multimedia disk. The parameter settings for the multimedia disk are listed in Table 4a. The block size and the scattering parameter for the three media types are listed in Table 4b.

Disk parameter	Setting
NumCylinders	1000
MinSeekTime	0.5 ms
MaxSeekTime	15.0 ms
RotSpeed	5400 rpm
DiskBW	128MBytes/sec

Table 4a: Multimedia Disk Specifications

Media Type	Media Block Size	Scattering parameter
HDTV	0.5 Mbits	1.0 ms
NTSC	0.13 Mbits	0.25 ms
CD-Audio	0.144 Mbits	0.1 ms

Table 4b

The media type that we studied first was HDTV. The simulator was operated under SCAN and FCFS policies, and statistics were collected for the total buffer requirement and the seek times for each service round till the system reached saturation (the point beyond which no new users can be admitted without compromising the continuity of admitted users). These values are shown in Figures 1a through 1c.

In Figure 1a, we observe that all the algorithms have the same saturation point of $n_{max} = 22$. This confirms our analysis of Section 2 which predicted that disk scheduling does not affect the subscriber limit. We also observe, in Figure 1a, a significant reduction in the total memory requirement for both SCAN and FCFS in comparison with the QUAD scheduler. For example, for 15 subscribers, SCAN and FCFS require 300 buffers and 450 buffers, respectively, whereas QUAD requires 1067 buffers.

The memory usage factor of SCAN and FCFS relative to QUAD are presented in Figure 1b. This figure shows that when the number of subscribers goes beyond 5, SCAN consistently requires only about 23% and FCFS requires only about 41% of the buffer space required by QUAD. The source of this improved performance is the seek time reduction obtained by the disk scheduling policies. This factor is quantitatively shown in Figure 1c where we observe that the average seek time incurred per user request approaches to 1/3 of MaxSeekTime for FCFS and to $1/n_{max} = 1/22$ of MaxSeekTime for SCAN.

The disk specifications listed in Table 4a correspond to $l_{max}^{seek} = 15ms$ and $l_{max}^{rot} = 11.11ms$. Substituting these values in the average case equations of Section 2 we obtain $muf_{SCAN} = 0.21$ and $muf_{FCFS} = 0.40$. These predictions agree closely with the above observed values.

The same experiment was carried out for NTSC and CD-Audio media types, and the performance results are shown in Figures 2a through 2c for NTSC and in Figures 3a through 3c for CD-Audio. Comparing Figures 1b, 2b and 3b, it is evident that the reduction in the memory usage is approximately the same for all three media types. This means that the memory usage reduction obtained by SCAN and FCFS remains nearly invariant across a wide range of data rate requirements.

4.2 Experiment 2: Standard Disk

The goal of our next experiment was to study the effect of disk scheduling policies (SCAN and FCFS) in serving multiple movie subscribers on a standard disk. The parameter settings were taken from disk data sheets supplied with Silicon Graphics Power Challenge Series, and these are listed in Table 5a. The media block size and the scattering parameter for the three media types are listed in Table 5b.

Disk parameter	Setting
NumCylinders	2870
MinSeekTime	0.6 ms
MaxSeekTime	17.4 ms
RotSpeed	5400 rpm
DiskBW	4 MBytes/sec

Table 5a: Standard Disk Specifications

Media Type	Media Block size	Scattering Parameter
HDTV	0.5 Mbits	0.1 ms
NTSC	0.13 Mbits	0.01 ms
CD-Audio	0.14 Mbits	0.001 ms

Table 5b

For HDTV and NTSC, the maximum number of users that can be supported by the above disk is very small. Therefore, we present results only for CD-Audio here, which are shown in Figures 4a through 4c. These graphs indicate that the performance improvements derived here are very similar to those seen for the multimedia disk of Experiment 1. Therefore, it appears that policies such as SCAN and FCFS are robust with regard to providing consistent performance across a range of media types and disk characteristics.

5 Conclusions

In this paper, we have made a preliminary study of the importance of disk scheduling in providing continuous access to a multimedia database. We analyzed the performance characteristics of FCFS and SCAN and predicted that the memory required to support the continuity constraint for servers using these scheduling disciplines relative to that of QUAD, the [Vin93] scheduler, depended on the maximum seek time and maximum rotational latency of the disk. For the worst case, we predicted that SCAN would always require less than half the memory required by QUAD. In the average case, we predicted that SCAN would require less than a quarter that of QUAD, while FCFS would require between 0.33 and 0.42. We also showed that SCAN is optimal in minimizing the memory usage factor.

Using a simulation model of a multimedia-on-demand server, we studied the effects of disk scheduling for HDTV and CD-Audio on both a multimedia disk and a standard disk. The simulation results showed that considerable reductions of memory requirements were realized by SCAN and FCFS due to their reducing the average seek time in switching from one subscriber to the next. In general, SCAN required only about 20 percent, and FCFS only about 40 percent, the memory required by QUAD. These reductions matched closely with those predicted by the analysis.

In summary, we have shown that using SCAN for disk scheduling in multimedia databases results in considerable savings on the memory required to support subscribers.

References

[Coff73] Coffman, E., and Denning, J., "Operating Systems Theory", Prentice-Hall, 1973.

[Gei87] Geist, R., Reynolds, R., and Pittard, E., "Disk Scheduling in System V", *Proc. of ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems*, May 1987.

[JSAC93] *IEEE Journal on Selected Areas in Communication, Special Issue on High Definition Television and Digital Video Communication*, Vol. 11, No. 1, Jan 1993.

[Kenc93a] Kenchammana-Hosekote, D. and Srivastava, J., "Scheduling Continuous Media in a Video-on-Demand Server", Tech. Report, Dept. of Computer Science, University of Minnesota, Minneapolis, Dec 1993.

[Litt90a] Little, T.D.C., and Ghafoor, A., "Synchronization and Storage Models for Multimedia Objects", *IEEE Journal on Selected areas in Communication*, pp 413-427, April 1990.

[Rang92] Rangan, P.V., Vin, H.M., and Ramanathan, S., "Designing an On-Demand Multimedia Service", *IEEE Communication*, pp 56-64, July 1992.

[Rang93a] Rangan, P.V., and Ramanathan, S., "Adaptive Feedback Techniques for Synchronized Multimedia Retrieval over Integrated Networks", *IEEE/ACM Trans. on Networking*, Vol. 1, No. 2, pp 246-260, April 1993.

[Tane87] Tanenbaum, A.S., "Operating Systems: Design and Implementation", Prentice-Hall, 1987.

[TCJ93] *The Computer Journal, Special Issue on Distributed Multimedia Systems*, March 1993.

[Teor72] Teorey, T.T., and Pinkerton, T.B., "A Comparative Analysis of Disk Scheduling Policies", *Communications of the ACM*, Vol. 15, No. 3, pp 177-184, March 1972.

[TKDE93] *IEEE Trans. on Knowledge and Data Engineering, Special Issue on Multimedia Systems*, Vol.5, No.4, pp 551-563, Aug 1993.

[Vin93] Vin, H.M., and Rangan, P.V., "Designing a Multi-User HDTV Storage Server", *IEEE Journal on Selected Areas in Communication*, Vol 11, No 1, Jan 1993.