1. INTRODUCTION

Multimedia applications require support for the storage and retrieval of multimedia data, which typically consist of video, audio, text, and images. These data can be categorized into continuous media (CM) data (e.g., video, audio) and non-continuous media data (e.g., text, images). CM data, in contrast to nonCM data, have certain timing characteristics associated with them. For example, video clips, which are typically stored in units of frames, must be delivered to viewers at a certain rate (typically 30 frames/sec). For video compressed using the MPEG-I standard, this translates to a data rate of approximately 1.5 Mbps. CM data are also voluminous. For example, a 100-minute video compressed using the MPEG-I compression algorithm requires about 1.25 GB of storage space.

The timing characteristics and the large volumes of CM data make the design of a multimedia storage server a challenging task. Such a server should:

1. provide rate guarantees for the storage and retrieval of CM data;
2. provide support for VCR operations (e.g., fast-forward, rewind, pause);
3. provide support for retrieval of nonCM data concurrently with CM data; and
4. manage critical storage resources such as memory, disks, and tapes so as to maximize throughput and reduce response times.

A multimedia server typically employs several secondary (e.g., disk) and tertiary (e.g., tapes, optical disks) storage devices to store the data permanently. A small amount of RAM is used to stage the data retrieved from disks and tapes before they are transmitted to clients. Cost, latency, and transfer rates of these devices are as described in Table 1. Designing a multimedia server that is cost-wise optimal from the standpoint of minimizing storage cost is an important and challenging research problem. Because tapes have the least storage cost, the cost of a server can be reduced considerably by storing CM clips primarily on tapes. However, because tapes have very high latencies for data access, in order to retrieve data for a large number of CM clips concurrently, frequently accessed CM clips can be stored on magnetic disks (which have lower access latencies but are more expensive). Thus a multimedia server consists of a storage hierarchy with tapes at the bottom, disks in the middle, and RAM at the top. The exact configuration of the server (e.g., the number of disks and tapes, the amount of RAM) must be determined based on the number of clips and the frequency with which they are accessed.

The remainder of this paper discusses some of the major research problems we
are working on as part of the *Fellini* multimedia storage server project at AT&T Bell Labs.

### 2. DISK STORAGE ISSUES

In order to retrieve data for several CM clips at a guaranteed rate $r$, the data for the clips are retrieved into RAM in rounds [Özden et al. 1995a; Chen et al. 1993]. If $d$ is the size of data retrieved for each clip during a round, then the duration of each round must not exceed $d/r$. Thus the number of clips that can be concurrently serviced is bounded by the number of clips for which $d$ bits can be retrieved in time $d/r$. Because data access from disk may potentially incur a seek and rotational delay, reducing these delays will increase the number of clips that can be serviced concurrently. This can be achieved by clever data-retrieval and layout strategies that reduce/eliminate latencies. For example, C-SCAN disk scheduling reduces the overhead of seek latency, whereas retrieving the integral number of tracks eliminates rotational delay. Also, the round duration must be carefully chosen because a large round duration increases the amount of RAM required (because $d$ is larger), whereas a small round duration could decrease the number of clips retrieved.

A number of additional problems arise when retrieving data from disks. In modern disks, outer tracks have higher transfer rates than inner tracks. By taking into account the disk transfer rate of the tracks where CM clips are stored, one could substantially reduce buffer requirements. Also, to service both CM as well as nonCM data, schemes would either have to reserve a portion of a round or use the *slack* time during a round to service nonCM requests. Furthermore, if clips are frequently inserted and deleted, then they must be stored noncontiguously on disks (in fixed-size blocks) to reduce fragmentation. Unlike conventional file systems that use a small block size (4–8KB), the block size for clips must be chosen as close to $d$ (the data retrieved during a round) as possible, in order to reduce the latency overhead. Index structures for accessing random blocks for a clip also need to be designed. Finally, the algorithms used must allow the server to retrieve data for clips with different rate requirements (JPEG video requires a data rate of 7 Mbps as opposed to 1.5 Mbps for MPEG-I).

### 3. DISK-STRIPPING ISSUES

Because a video server utilizes several disks, schemes for laying out the videos on multiple disks are crucial to distributing the load uniformly across the various disks, thereby utilizing the disk bandwidth effectively. For example, certain videos may be more popular than others, and a naive approach that stores every video on an arbitrarily chosen disk could result in certain disks being burdened with more requests than they can support while other disks remain underutilized. Also, unless a video is replicated on several disks, the number of clients that are concurrently accessing portions of the video is bounded above by the bandwidth of the disk storing the video.

When striping continuous media clips across disks, several research issues must be addressed. Consider a scenario in which every stream is paused just before data for the stream is to be retrieved from a disk $D_i$. If all the streams are resumed simultaneously, then the resumption of the last stream for which data are retrieved from $D_i$ may be de-
layed by an unacceptable amount of time. Replicating the continuous media clips across multiple disks could help in balancing the load on disks as well as reducing response times when disks get overloaded. Furthermore, in environments such as movies-on-demand, where there are a large number of requests for a few movies, retrieving data for the movies periodically and having multiple client requests share the retrieved data could result in decreased response times. Schemes for doing this need to be developed.

4. FAULT-TOLERANCE ISSUES

For a single disk, the mean time to failure (MTTF) is about 300,000 hours. Hence a server with say 200 disks has an MTTF of 1500 hours or about 60 days, and is thus highly susceptible to disk failure. Because data on a failed disk are inaccessible until the disk is repaired, a single disk failure could result in interruption of service, which may be unacceptable. In order to provide continuous reliable service to clients, it is imperative that it be possible to reconstruct data residing on a failed disk in a timely fashion so that data for CM clips can be transmitted at the required rate even if a disk fails.

Fault tolerance in traditional storage servers can be achieved either by mirroring or by parity encoding [Patterson et al. 1988]. These techniques can also be used in a multimedia server [Berson et al. 1995]. However, because a CM server must also provide rate guarantees for CM clips, it must retrieve from the surviving disks the additional blocks needed to reconstruct the required data blocks on the failed disk in a timely fashion. This may not be possible unless for every additional block either (1) it has been prefetched and is already contained in the server's buffer, or (2) bandwidth required for retrieving it has been reserved a priori on the disk containing it. Prefetching blocks has the advantage of reducing the additional load on disks in case of a disk failure; however, additional buffer space is required to store the prefetched blocks. On the other hand, buffer space overheads can be reduced by not prefetching blocks; a drawback is that bandwidth for retrieving the additional blocks needs to be reserved on each disk (this bandwidth goes unused in the absence of failures). The preceding trade-offs must be taken into account when designing fault-tolerant schemes for multimedia servers.

5. BUFFER MANAGEMENT ISSUES

By having requests share a global pool of buffer pages, the number of I/O requests can be considerably reduced, thereby enabling a larger number of requests to be serviced [ Özden et al. 1995b]. For example, if two requests for a clip arrive at an interval of 5 or 10 seconds, then by caching the pages accessed by the first request, disk accesses for the second requests can be totally eliminated.

An important research issue is that of buffer-page-replacement policy. Existing page-replacement policies may be unsuitable inasmuch as they exploit neither the knowledge of outstanding requests nor the fact that CM clip data are accessed predominantly sequentially. For example, consider a server with 100 buffer pages of size $d$ that follows the least recently used (LRU) policy (as in most conventional storage servers). Let data retrieval for clips $C_1$ and $C_2$ be initiated at some round $k$. Thus at round $k + 51$, the first page of $C_1$ and $C_2$ are replaced to make room for the 51st page. If a request for $C_1$ were to arrive in round $k + 52$, then with LRU, pages for the second request for $C_1$ would need to be accessed separately from the disks (because the page to be retrieved for the request during a round would have been paged out during the previous round). However, if, after the second request for $C_1$ arrives, instead of replacing $C_1$'s pages we replace only pages accessed by $C_2$, then only the first page for $C_1$ would need to be accessed from the disk and subsequent pages would already be present in the buffer when needed, thus
eliminating the need for subsequent disk accesses for the second request for $C_1$. Thus a buffer-page-replacement policy that takes into account requests being serviced would result in better performance.

6. TERTIARY STORAGE ISSUES

Because CM data tend to be voluminous, utilizing tertiary storage could result in reduced costs. However, tertiary storage devices have radically different characteristics from magnetic disks. Tapes have a very high latency for data access (in the order of minutes), and optical disks have low transfer rates.

If tapes are to be used for storing video clips, then it is crucial that schemes for reducing latency be devised. In certain tapes (e.g., DLTs), there are multiple parallel tracks along the length of the tape. Furthermore, consecutive bits of data are laid out in a single track in one direction and then the next track in the other direction; before such a tape is replaced, it is rewound to a fixed end and this results in a large latency. By retrieving data in multiples of track lengths, the need to rewind the tape can be completely eliminated, thus reducing latency. For tape drives that do not rewind every time a tape is ejected, latency can be reduced by replicating the CM clip on multiple tapes and storing the position of each tape. When data for a clip are to be retrieved, then the tape that is positioned closest to a replica of the clip on the tape is used. Response times can also be reduced by using disks to cache initial portions of clips.

Finally, the mode of retrieval of CM clips from tapes needs to be explored further. Are data from tapes transferred directly to RAM or are disks used as intermediate storage? Is an entire CM clip transferred in a single access, or are multiple accesses performed and a chunk of the clip transferred in each access?

7. CONCLUDING REMARKS

Multimedia storage managers must ensure that CM data can be stored and retrieved at specific rates. Consequently, algorithms that provide rate guarantees for data retrieved from secondary storage devices (e.g., disks) and tertiary storage devices (e.g., tapes) need to be developed. Schemes for retrieving data from disks must take into account seek and rotational latencies, varying track transfer rates, bad sector remapping, and recalibration of tables due to thermal expansion, whereas schemes for retrieving data from tapes must attempt to reduce the latency for data access. Furthermore, assuming that clips are striped across multiple disks, schemes for implementing VCR operations and retrieving data for CM clips periodically (to be shared by multiple client requests) must be devised. Also, because most of the previously proposed schemes for dealing with disk failures do not address the issue of continuously retrieving data at specific rates, algorithms for reconstructing data on a failed disk in a timely fashion need to be developed. Finally, because accesses to CM data are typically sequential in nature, improved buffer-page-replacement policies that exploit the sequentiality of access need to be devised.

REFERENCES


ACM Computing Surveys, Vol 27, No. 4, December 1995