PARALLEL FILE STRIPING ON OPTICAL JUKEBOX SERVERS

J. Tárraga, R.D. Hersch

Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland

joaquin.tarraga@axstech.com, rd.hersch@epfl.ch

ABSTRACT

In the near future, large digital media servers are expected to offer storage capacities in the order of petabytes. Servers made of clusters of PC's connected to jukeboxes may represent an interesting alternative compared with servers made of arrays of magnetic disks. However, due to disk exchange overhead, higher seek times and lower data transfer rates, access to data located on optical disks is significantly slower than access to data located on magnetic disks. In the present contribution, we analyze the benefits and limitations of striping files across multiple optical disks in order to speedup the retrieval of large pieces of information. We also show that server applications requiring both computer power and I/O bandwidth may be distributed over several processing nodes and access files striped over multiple optical disks. For such applications, as long as disk exchanges are not necessary, a predictable speedup can be obtained by ensuring a high enough number of server computers, optical disk drive units, and enough communication bandwidth.

Keywords: Storage server, jukebox server architecture, optical disks, parallel server applications, stripe files

1. INTRODUCTION

A recent report about the evolution of the amount of information produced in the world and the exponential growth rate of digital magnetic storage [1] suggests that in the near future, central server sites will need storage capacities beyond the size of a petabyte (10^{15} bytes).

Building server relying mainly on magnetic disks as is the case today becomes problematic when the storage capacities increase. For example, to store one petabyte of data, 10'000 disks of 100 GB each are needed. Such a large disk array requires costly interfaces, consumes considerable energy and requires an important maintenance effort, since disk failures are expected to occur frequently. Furthermore, tape systems for backup and data exchange are very expensive. Despite high-speed networks, data exchange of huge datasets between servers is extremely slow (try to send one terabyte of data over a one gigabit/s network connection).

An alternative to magnetic disk arrays is the use of optical disk jukeboxes. Jukeboxes may potentially have as many drives and robotic arms needed to satisfy external access requests. The amount of information that can be stored in a jukebox depends only on the physical place present for storing the optical disks. Optical disks are passive devices, they do not consume energy and their expected life time is longer than the one of magnetic disks. Furthermore, duplicate disks are easy to create, both for information distribution and for creating backup copies.

Proc. 2002 IEEE International Conference on Multimedia and Expo, Vol. 2, 173-176

However, optical disks are considerably slower than magnetic disks and for loading a new file or starting a new application, the jukebox robot needs to possibly unload a disk and load the desired optical disk into a drive. Once an optical disk is loaded and in rotation, its content is directly accessible as on a hard disk, but with 10 times longer seek times (typically 70 to 140ms) and a disk throughput reduced by a factor of 5 (typically 2 to 4 MB/s on CD-ROMs).

In the present contribution, we would like to analyze under which conditions file striping across multiple optical disks can bring an increase in data throughput. We also would like to verify that server applications requiring both computer power and I/O bandwidth may be distributed over several processing nodes and access files striped over multiple optical disks. We give a brief overview of the software architecture needed for the parallel access to striped files. Finally, we analyze the performance of a real application making use of parallel accesses to files striped over multiple CD-ROMs.

2. RELATED WORK

Much research has been carried out in respect to the layout of video streams on disk arrays [2],[3]. The use of optical jukeboxes was considered in the context of hierarchical video-on-demand servers [4],[5].

3. SERVER ARCHITECTURE

A server architecture based on optical jukeboxes may comprise several server nodes, each one consisting of a computer and a number of jukeboxes. Optical jukeboxes, both for CD-ROMs and DVDs have the potential of offering terabyte storage capabilities. Fig. 1 shows a jukebox server made of several small jukeboxes (e.g. NSM jukeboxes [6]), whose drives are connected through SCSI or IDE channels to server PCs. Server PCs and client stations may be connected by a Fast or Gigabit Ethernet switch.

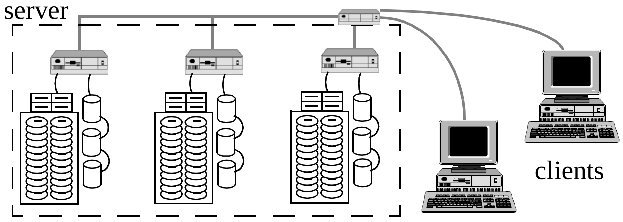

Fig. 1. Terabyte server made of several small jukeboxes

Central parameters for jukebox systems [7] are the time $T_{exchange}$ required by the robotic device to move an optical disk from an optical drive unit to a magazine slot and vice-versa, the physical bandwidth X_{drive} of a single drive unit, the disk spin up time T_{spinup}, the head displacement time T_{seek} and the mean rotation time $T_{rot.}$

4. FILE STRIPING ANALYSIS

Let us analyze to which extent access to striped files [8] allows to improve the response time. Since small jukeboxes have often 4 optical disk drive units (Fig. 1) and large jukeboxes (e.g. the GiantROM from ALP Electronics [9]) may have several dozens of drive units, applications requiring high data throughputs may access simultaneously data striped over multiple optical disks. Due to the limited capacity of a single optical disk, it is highly probable that for each new file access request, an optical disk exchange is required. On an NSM jukebox, for example, an optical disk exchange induces an overhead of 17s. Depending of the file size, the data transfer time may also be a significant fraction of the total response time. Fig. 2 illustrates an optical disk exchange and a data transfer of a 500 MB file on a NSM Satellite jukebox. The entire procedure requires 184s, where 8.7s are spent by the robot arm for performing a disk exchange, 8.6s for the spin-up time, and 166.6s for transferring the file with a data throughput of 3 MB/s.

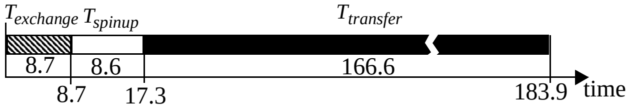

Fig. 2. Time for retrieving a 500 MB file from a jukebox

The file access time can be reduced by striping the file across several optical disks and reading the subfiles simultaneously. The stripe factor (*SF*) defines the number of optical disks storing a striped file. Fig. 3 shows the access time for the same 500 MB file striped across 4 optical disks (i.e. *SF* = 4), 125 MB per disk, on a NSM Satellite jukebox. The total file access time is reduced to 85s. Hence, striping effectively increases the bandwidth of the system and reduces the file transfer time.

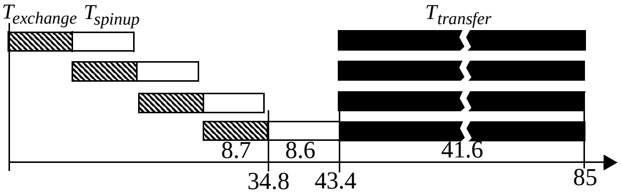

Fig. 3. Time for retrieving a 500 MB file striped across 4 CD-ROMs of a single jukebox

However, striping increases the number of disk exchanges performed per request, which further contributes to the already high disk exchange latency. The latency penalty for a file striped across *SF* optical disks on a single-arm jukebox is of *SF-1* additional disk exchanges (Fig. 3). In the case of Fig. 3, the speedup obtained by parallel file striping is 184 / 85 = 2.16. However, for smaller files, there may be no speedup at all. The following equation expresses the response time as a function of the stripe factor (*SF*) for a given file size.

$$T(SF)_{tot} = (SF \cdot T_{exchange}) + T_{spinup} + T_{rot} + T_{seek} + \left(\frac{F_{size}/X_{drive}}{SF}\right)$$

Fig. 4 plots the total response time for different file sizes as a function of the stripe factor. When increasing the stripe factor, the response time decreases until it reaches the minimal response time that corresponds to the optimal stripe factor ($SF_{optimal}$). When further increasing the stripe factor, the response time increases linearly with slope of $T_{exchange}$, since the term ($SF*T_{exchange}$) becomes the most significant term of the total response time.

We can determine the optimal stripe factor ($SF_{optimal}$) by minimizing the response time

$$\frac{dT}{dSF}(SF)_{tot} = 0 \Rightarrow SF_{optimal} = round\left(\sqrt{\frac{F_{size}/X_{drive}}{T_{exchange}}}\right)$$

For example, for the 500 MB file size, the optimum stripe factor is 4, which corresponds to a speedup of $T(SF=1)_{tot}$ / $T(SF=4)_{tot}$ = 184 / 85 = 2.16. A higher speedup cannot be attained, since disks are exchanged sequentially by a single robotic arm.

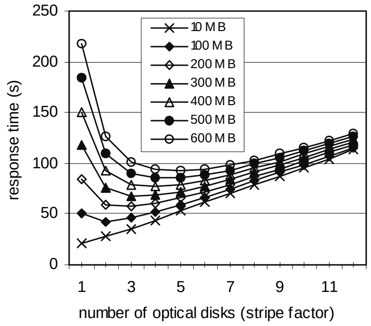

Fig. 4. Response time as function of the number of optical disks for file sizes varying from 10 to 600 MB.

In order to reduce the latency due to sequential disk exchanges, one may build large storage systems with several small jukeboxes incorporating each one its own robotic arm. Files are striped across the jukeboxes in order to perform the disk exchanges in parallel. Fig. 5 shows the access to a 500 MB file striped across 4 disks each one residing on a different jukebox. Striping across different jukeboxes allows to reach a speed-up of $T(SF=1)_{tot}$ / $T(SF=4)_{tot}$ = 184/59 = 3.1. Since the incompressible single disk exchange overhead time remains a significant fraction of the total response time, we do not reach the ideal speedup of 4 .

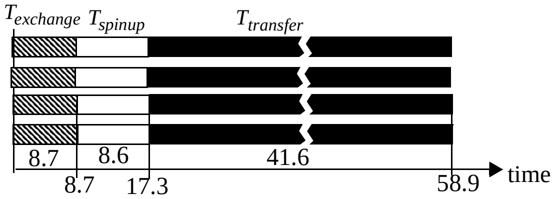

Fig. 5. Time for retrieving a 500 MB file striped across 4 disks located in different NSM Satellite jukeboxes.

Another disadvantage of striping is that it increases the contention for system resources (i.e. for drives as well as for the robot arm). The larger the striping factor, the more contention there is for robot arm and drive resources. Since the system response time increases with contention, under heavy loads it might not be advantageous to stripe even very large files.

5. PARALLEL STRIPED FILE SERVER

Let us describe the software architecture of a parallel striped file server. A *striped file*, i.e. a file whose data is declustered across multiple optical disks, is composed of multiple *extent files*, i.e. subfiles. Each *extent file* comprises a set of *extents* (stripe units) and a table whose entries specify the size and the storage location of extents, i.e. their byte address within the corresponding *extent file*.

The optical jukebox server offers a set of threads providing support for accessing data from multiple optical disks.The *InterfaceServer* thread is responsible for coordinating parallel file operations. It maintains on a magnetic disk a *shadow directory tree* containing the directory trees of all present optical disk files in the server jukeboxes. One or several *ExtentServer* threads run on each slave server node and access the optical disks for reading extent files (subfiles). One *RobotServer* thread runs on each slave server node and controls the jukebox robotic arm.

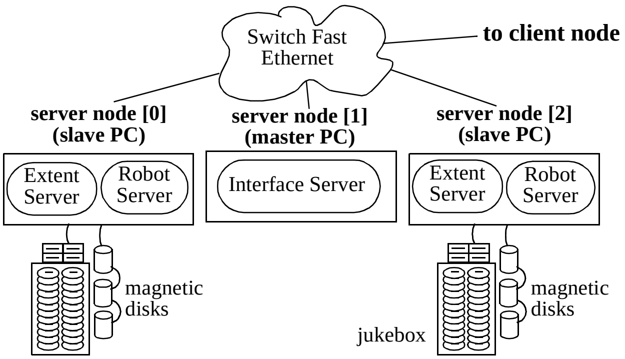

Fig. 6. Server threads mapped onto a jukebox server

Fig. 6 shows how the server threads are mapped onto a jukebox server architecture comprising 3 server nodes. When accessing a *striped file* stored in multiple optical disks, a client thread sends an *open_file* request to the *InterfaceServer* thread. By looking up its *shadow directory tree*, the *InterfaceServer* thread identifies the locations of the contributing optical disks, i.e. their jukebox and magazine slot. If the disks reside in the magazine slots and drive units are available, the *InterfaceServer* thread sends *load_disk* requests to the corresponding *RobotServer* threads who issue commands to the robotic devices to move the optical disks from their magazine slots to the selected drive units. Then, the *InterfaceServer* thread sends *open_disk_file* requests to the corresponding *ExtentServer* threads which open *extent files* from the disks and send back *extent file descriptors*. The *InterfaceServer* thread returns to the client a table of *ExtentServer* thread indices and a table of *extent file descriptors.* For subsequent reading of data, an application thread sends *read_extent* requests directly to the corresponding *ExtentServer* threads which read from the optical disks and send the data back. There is no further communication with the *InterfaceServer* thread, besides the last *close_file* request. In addition to simple parallel data accesses, the jukebox server may execute server applications carrying out parallel data access and processing operations, where processing operations are pipelined with the corresponding data access operations (i.e. disk extent accesses).

6. AN APPLICATION EXAMPLE

To verify the effectiveness of combining parallel access to files striped across optical disks and parallel computations, we run the Visible Human Slice Sequence Animation Server [10] in parallel on the server. It creates MPEG animations through the body by assembling successive slices of the Visible Human data set [11] along a user-defined trajectory.

For enabling parallel access to its data [12], the Visible Human 3D volume is segmented into *sub-volumes* (stored as extents) of size 54x54x19 RGB voxels, i.e. 162.3 KB, which are striped over a number of CD-ROMs. In order to extract an image slice from the 3D image, the extents intersecting the slice are read from the drive units and the slice parts contained in these volumic extents are extracted and resampled (Fig. 7).

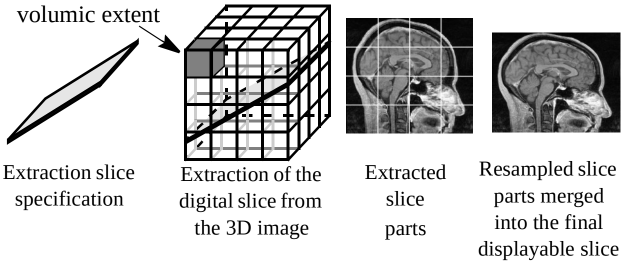
volumic extent
Extraction slice specification
Extraction of the digital slice from the 3D image
Extracted slice parts
Resampled slice parts merged into the final displayable slice

Fig. 7. Extraction of slice parts from volumic file extents

When the interface server thread (Fig. 6) receives the animation request, the request is converted into a list of slices orthogonal to the trajectory. This slice list is split into individual slice requests. Each slice is extracted in parallel by the slave PCs. Once a slice is extracted, the interface server writes the slice into a slice buffer and an MPEG-1 encoder thread running on the master PC gets the extracted slices from the slice buffer, compresses them into MPEG-1 format and sends the resulting animation part to the client (in streaming mode).

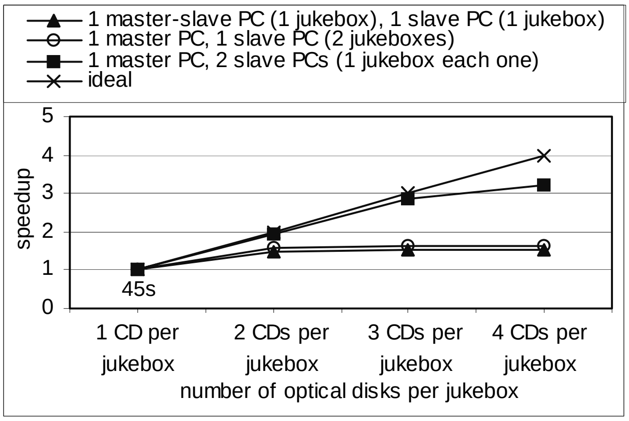

Fig. 8. Speedup of slice sequence animation extraction for three server configurations, as a function of the number of simultaneously accessed CD-ROMs (always two jukeboxes)

Accessing and extracting slices from the 3D Visible Human data set requires high processing power and considerable storage space (13 GB). In order to test parallel accesses to CD-ROMs having a capacity of 650 MB each, we only store the head (1.2 GB) of the Visible Human (on 2, 4, 6 or 8 CD-ROMs). In the experiment, an animation comprising 143 256x256 24-bit/pixel slices is extracted. Extents read from CD-ROMs are cached in the extent memory cache maintained by the extent server. The size of the extent memory cache is of 100 MB. For this animation, 773 extents (123.8 MB) are read from optical disks and 8729 extents are read from the extent cache. Fig. 8 shows for three server configurations the speed-up as function of the number of CD-ROMs on which the Visible Human data set is striped. The first configuration (same as in Fig. 9a,b) comprises a 2 PC server, with one PC acting as master and slave and one PC as slave only. Both PCs are connected each one to one NSM Satellite jukebox. In the second configuration (same as in Fig. 9c) comprising also two server PCs, one PC acts as master only and the second PC, connected to two jukeboxes, acts as slave only. The third configuration comprises 3 server PCs (same as in Fig. 9d), one acting as a master and two PCs acting as slaves, each one connected to one jukebox. The master PC runs at 733 MHz, in contrast to all other PCs which run at 333MHz.

The master PC merges slice parts into full slices and encodes full slices into MPEG-1 format. The slave PCs read extents from optical disks, extract and resample slice parts.

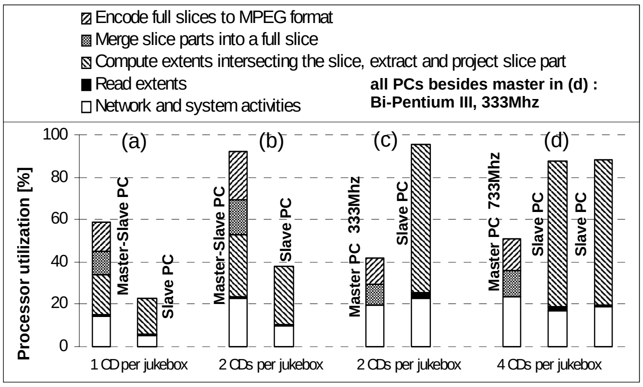

Fig. 9. Server PC processor utilization (mean value of the utilization of the two Bi-Pentium III processors)

For the 2 PC server configuration where one PC is at the same time master and slave (Fig. 9a), when extracting the animation from 2 CD-ROMs (1CD per jukebox), the I/O bandwidth is the bottleneck. From 4 CD-ROMs (2 CDs per jukebox), the bottleneck shifts from the drive unit throughput to the limited processing power available on the server PC acting as master and slave (Fig. 9b). In the case where the slave PC is responsible for both jukeboxes (Fig. 9c), it reaches its maximal utilization rate and represents therefore the bottleneck.

With a 3 PC server configuration, one 733 MHz Bi-Pentium III PC is dedicated to the master functions and two 333 MHz Bi-Pentium III PCs are dedicated to the slave operations, each one connected to one jukebox. With up to 3 CD-ROMs per slave PC, I/O bandwidth is always the bottleneck (Fig. 8). From 4 CD-ROMs per slave PC, the network bandwidth and the slave PC's become the bottleneck. Extracting the animation from 8 CD-ROMs takes about 13.9 s. In this time, 9502 slices parts are sent from the slave PCs to the master PC. The mean size of a slice part is of 14336 bytes. In addition to the slice part, the overhead of each message (i.e. control information) sent to the master PC is of 60 bytes. Therefore, 9502 * (14336 + 60) = 136790792 bytes = 130.45 MB are sent to the master PC in 13.9s. We obtain a network throughput of 9.38 MB/s close to the maximum throughput sustained by FastEthernet. While the master PC has a utilization of 51%, the most loaded processor within the slave Bi-Pentium PCs has an utilization of 94% and starts to saturate (Fig. 9d). An additional increase in performance would therefore require both a faster network connection and additional slave PCs.

7. CONCLUSIONS

Large digital libraries, image and multimedia repositories require servers offering terabyte to petabyte storage capacities and high-bandwidth I/O throughput. We propose a jukebox server architecture comprising one master server PC running the server interface receiving client access requests and additional slave server PCs connected each one to one or several jukeboxes.

The jukebox server prototype allows to access large files striped across multiple optical disks. The parallel access experiments show that for large files the available throughput can be considerably increased by accessing in parallel multiple optical disks loaded into multiple optical disk drives.

Server applications requiring both computer power and I/O bandwidth may be distributed over several server processing nodes and access files striped over multiple optical disks. As long as disk exchanges are not necessary, a predictable speedup can be obtained, limited only by the number of available server PCs, the number of optical disk drive units and the communication bandwidth within the server.

Parallel access to files striped across multiple optical disks may offer new opportunities for creating powerful servers offering huge storage capacities and high-throughput access to large files.

REFERENCES

[1] P. Lyman and H. R. Varian, “How much information”, UC Berkeley, retrieved from http://www.sims.berkeley.edu/how-much-info, Feb 2002.

[2] F. A. Tobagi, J. P and, R. Baird and M. Gang, Streaming RAID: A Disk Array Management System for Video Files, In *First ACM Conference on Multimedia*, August 1993, 393-399

[3] J-Y. B. Lee, Parallel Video Server: A Tutorial, *IEEE Multimedia*, Vol. 5, No. 2, April-June 1998, pp. 20-28

[4] A.K. Chervenak, D.A. Patterson, R.H. Katz, Storage Systems for Movies-on-Demand Video Servers, *Proc. 14th IEEE Symposium on Mass Storage Systems*, September 1995, IEEE Press, 246-256

[5] S-H. G. Chan and F. A. Tobagi, Hierarchical Storage Systems for On-Demand Video Servers, *SPIE*, Vol. 2604, 1996, pp. 103-120

[6] NSM Storage Inc., http://www.nsmstorage.com

[7] K. Byrne, Evaluating Jukebox Performance, *IEEE Spectrum*, July 1997, pp. 70-73

[8] K. Salem, H. Garcia-Molina, Disk Striping, *Proc. of the Second International Conference on Data Engineering*, Washington, D.C., February 1986, pp. 336-342

[9] ALP Electronics SA, http://www.swisstorage.com

[10] J-C. Bessaud and R. D. Hersch, The Visible Human Slice Sequence Animation Web server, Proc. IS&T/SPIE Conf. on Internet Imaging, San Jose, California, Jan. 2001, *SPIE* Vol. 4311, pp. 341-348, see *http://visiblehuman.epfl.ch*

[11] M. Ackerman, “The Visible Human Project,” *Proc. IEEE*, Vol. 86, No. 3, Mar. 1998, pp. 504-511

[12] V. Messerli, O. Figueiredo, B. A. Gennart, R. D. Hersch, Parallelizing I/O Intensive Image Access and Processing Applications, *IEEE Concurrency*, Vol. 7, No. 2, 1999, pp. 28-37