On Configuring Hierarchical Storage Structures

Ali Esmail Dashti and Shahram Ghandeharizadeh
Computer Science Department
University of Southern California
Los Angeles, CA 90089
{dashti,shahram}@cs.usc.edu
tel +1-213-740-4781
fax +1-213-740-7285

Abstract

The rapid progress in mass storage technology is enabling designers to implement large databases for a variety of applications. The possible configurations of the hierarchical storage structures (HSS) are numerous and result in various tradeoffs, e.g., storage cost, throughput, initial latency, etc. A naive design might waste system resources and result in high cost. For example, one naive design decision is to add disk caches for those applications whose bandwidth is already satisfied by the tertiary storage device and can tolerate a high latency.

In this study, we introduce a configuration planner for HSS in support of video-on-demand applications. The input to the planner are the number of simultaneous displays required by the target application (i.e., throughput), database size, and expected pattern of access to the objects. It's output are the choice of mass storage devices at each level of the hierarchy. The resulting configuration supports the application requirements at a minimum cost per display. It is possible to extend this study to consider other application requirements, such as: initial latency and storage reliability. Moreover, the presented concepts can be generalized to scientific applications, such as those described in [3].

1 Introduction

Multimedia information systems, such as video-on-demand applications, are data intensive applications that are benefiting the most from the rapid progress in mass storage technology. One of the main design challenges of hierarchical storage structures (HSS), in support of video-on-demand application, is to guarantee continuous display of video objects at a minimum cost. To support continuous display, it is necessary to retrieve and display data at a pre-specified rate, otherwise, the display will suffer from frequent disruptions and delays, termed hiccups. To minimize the system cost, it is necessary to use the correct mixture of mass storage devices for a given target application requirements. For the past few years, techniques have been developed to support continuous display from disks, see [11, 1, 6, 8] for details. This study is an extension of those studies.

The remainder of the paper is organized as follows. In Section 2, we present a display-caching technique to support continuous display using tape libraries. In Section 3, we describe a configuration space of our planner. Finally, in Section 4, we describe a number of possible future extensions.

2 Continuous Display using Tape Libraries

Similar to magnetic disks, tape juke boxes are mechanical devices. However, the characteristics of a tape juke box is different from that of a disk drive [2, 10, 9]. The typical access time to data from a magnetic disk drive is in the order of milliseconds, whereas with a tape juke box, due to the overhead of mounting and dismounting tapes from the drives and its sequential nature, latency times in the order of seconds (minutes) is not uncommon. Due to the long latency time associated with tape access and due to the mismatch between the tape production rate, D_T, and the display consumption rate, C (i.e., Production Consumption Ratio [7], $PCR = \frac{C}{D_T}$), it is necessary to use caching to simulate continuous display from tapes. Otherwise, the display may suffer from hiccups. This display-caching (DC) should not be confused with object-caching (OC). With OC, complete objects are cached at different levels of the hierarchy to minimize the number of references to the slower devices in the hierarchy. Whereas with DC, the cache is used to bridge the gap between the retrieval behavior of tapes and the display requirement of the objects to guarantee a continuous display. Here, we focus on DC to describe: 1) how much data should be cached (prefetched) to guarantee continuous display from tape libraries, 2) where to maintain the cached data?, and 3) when the display should start? In the rest of this section, we address these three topics. We start with a brief description of the physical characteristics of the tape libraries.

Tape devices may operate in either multiplexing mode or streaming mode (non-multiplexing) [7, 12]. In the multiplexing-mode, data transfer is interrupted in the middle of an object retrieval to retrieve portions of other objects. The tape drive must locate the data by performing a sequential seek (T_{Search}) and possibly dismount the tape cartridge in the tape drive and mount another in its place (T_{Switch}) to access other objects. These actions increase the wasteful work performed by the tape drive (i.e., T_{Search} and T_{Switch}), and increase the wear and tear of the tape device. In the streaming-mode, the tape drive retrieves one object at a time, and, hence, at most one T_{Search} and T_{Switch} are performed per object retrieval. Here, we only consider operating the tape device in streaming-mode. (Note: the cost/stream might be lower when using multiplexing-mode.)

We describe display-caching requirements ($DC\_requirement$) for continuous display using tape libraries for two possible cases: the transfer rate of the tape drive is either 1) lower ($PCR < 1$), or 2) higher than the bandwidth required to display an object ($PCR > 1$). For each case, we consider the use of memory display-caching (DC_memory) and hierarchical display-caching ($DC\_hierarchical$), where $DC\_hierarchical$ consists of disk cache ($DC\_hierarchical_D$) and main memory cache ($DC\_hierarchical_M$). In our evaluation, we make the following assumptions:

- Data layout is contiguous on the tape, to facilitate long transfers.
- A tape juke box consists of one read/write drive, one tape switching mechanism (e.g., one robot arm), and a number of tape cartridges.

- Tape access time, T_{Access}, to an object is the sum of average T_{Search} and T_{Search}.
- All objects belong to a single media type, where the display requirement of an object is C and its size is O.

When $PCR \leq 1$, it is necessary to cache $(1 - PCR)$ of the object before starting its display. This is the minimum amount of caching required to guarantee hiccup free displays, as specified by the pipelining technique [7], i.e., $DC\_requirement = (1 - PCR) \times O$. With main-memory caching, $DC\_requirement$ is maintained in the main memory ($DC\_memory = DC\_requirement$); whereas with hierarchical-caching, $DC\_requirement$ is maintained on disk ($DC\_hierarchy_D = DC\_requirement$). The amount of main memory required for hierarchical caching is 4 blocks: 2 blocks for the retrieval from tape onto disk, and 2 blocks for display from disk ($DC\_hierarchy_M = 4$ blocks), where a block size is equal to a disk block.

When $PCR > 1$, the production rate is faster than the consumption rate and the display of the object may start as soon as the first portion of the object is retrieved (the size of the first portion is a disk block size). The climax of the caching requirement for one object retrieval is immediately after the completion of its retrieval. The size of this cache space is the difference between the object size and the portion of the object that has been displayed in parallel to its retrieval, $O - (\frac{O}{D_T} \times C)$.

It might be possible to retrieve other objects from the tape while displaying the first object from cache. Similar to the first case, the climax of the cache requirement for the second object is $O - (\frac{O}{D_T} \times C)$. This is reached after performing a tape access (T_{Access}) and an object transfer ($\frac{O}{D_T}$). Therefore, the total cache space requirement increases by $(O - (\frac{O}{D_T} \times C)) - (T_{Access} + \frac{O}{D_T}) \times C$. The minimum cache space required to guarantee continuous display for some N_T streams is:

$$DC\_requirement = \sum_{k=0}^{N_T-1} ((1 - \frac{1}{PCR}) \times O - k(T_{Access} + \frac{O}{D_T}) \times C). \quad (1)$$

The maximum number of simultaneous displays from the cache space, N_T, is: $N_T = Max(1, \lfloor \frac{B_T}{C} \rfloor)$, where B_T is the effective bandwidth of the tape drive, $B_T = \frac{O}{T_{Access} + \frac{O}{D_T}}$.

With main-memory caching $DC\_memory = DC\_requirement$, and with hierarchical caching $DC\_hierarcical_D = DC\_requirement$. The main memory requirement for hierarchical caching is equal to $N_T + 1$ blocks (for the display of the N_T streams from a disk cache), and some memory for the retrieval of the data from tape onto disk. Assuming that the main memory can flush data to the disk at the same speed as the transfer rate of the tape (or faster), then the amount of main memory for object retrieval from tape onto disk is 2 blocks per disk. Therefore, the total amount of main memory required for hierarchical caching is $N_T + 3$ blocks, for a single disk cache.

3 Configuration Space

For the purpose of this evaluation, we make the following assumptions:

N	50 Gbyte	100 Gbyte	500 Gbyte	1 Tbyte	5 Tbyte	10 Tbyte
10	$n_M = 1$ $n_D = 6$	$n_M = 1$ $n_D = 3$ $n_T = 1$	$n_M = 1$ $n_D = 3$ $n_T = 1$	$n_M = 1$ $n_D = 4$ $n_T = 2$	$n_M = 1$ $n_D = 5$ $n_T = 6$	$n_M = 1$ $n_D = 4$ $n_T = 11$
50	$n_M = 2$ $n_D = 6$	$n_M = 3$ $n_D = 11$	$n_M = 3$ $n_D = 24$ $n_T = 1$	$n_M = 2$ $n_D = 10$ $n_T = 2$	$n_M = 3$ $n_D = 16$ $n_T = 6$	$n_M = 3$ $n_D = 19$ $n_T = 11$
100	$n_M = 4$ $n_D = 6$	$n_M = 4$ $n_D = 11$	$n_M = 5$ $n_D = 38$ $n_T = 1$	$n_M = 5$ $n_D = 44$ $n_T = 2$	$n_M = 3$ $n_D = 20$ $n_T = 6$	$n_M = 6$ $n_D = 30$ $n_T = 11$
250	$n_M = 9$ $n_D = 9$	$n_M = 9$ $n_D = 11$	$n_M = 10$ $n_D = 50$ $n_T = 1$	$n_M = 11$ $n_D = 81$ $n_T = 2$	$n_M = 12$ $n_D = 51$ $n_T = 16$	$n_M = 11$ $n_D = 63$ $n_T = 11$
500	$n_M = 17$ $n_D = 18$	$n_M = 17$ $n_D = 18$	$n_M = 18$ $n_D = 56$	$n_M = 19$ $n_D = 99$ $n_T = 2$	$n_M = 22$ $n_D = 101$ $n_T = 32$	$n_M = 22$ $n_D = 101$ $n_T = 32$
750	$n_M = 25$ $n_D = 27$	$n_M = 25$ $n_D = 27$	$n_M = 27$ $n_D = 56$	$n_M = 28$ $n_D = 111$	$n_M = 33$ $n_D = 148$ $n_T = 47$	$n_M = 33$ $n_D = 148$ $n_T = 47$
1000	$n_M = 33$ $n_D = 36$	$n_M = 33$ $n_D = 36$	$n_M = 34$ $n_D = 56$	$n_M = 39$ $n_D = 111$	$n_M = 45$ $n_D = 420$ $n_T = 6$	$n_M = 44$ $n_D = 198$ $n_T = 63$
2500	$n_M = 82$ $n_D = 90$	$n_M = 82$ $n_D = 90$	$n_M = 82$ $n_D = 90$	$n_M = 84$ $n_D = 111$	$n_M = 95$ $n_D = 509$ $n_T = 6$	$n_M = 108$ $n_D = 908$ $n_T = 11$
5000	$n_M = 732$ $n_D = 0$	$n_M = 163$ $n_D = 179$	$n_M = 163$ $n_D = 179$	$n_M = 163$ $n_D = 179$	$n_M = 174$ $n_D = 556$	$n_M = 190$ $n_D = 1022$ $n_T = 11$
7500	$n_M = 732$ $n_D = 0$	$n_M = 243$ $n_D = 268$	$n_M = 243$ $n_D = 268$	$n_M = 243$ $n_D = 268$	$n_M = 261$ $n_D = 556$	$n_M = 278$ $n_D = 1111$
10000	$n_M = 732$ $n_D = 0$	$n_M = 1473$ $n_D = 11$	$n_M = 325$ $n_D = 358$	$n_M = 325$ $n_D = 358$	$n_M = 331$ $n_D = 556$	$n_M = 348$ $n_D = 1111$

Table 1: Configurations space, using Zipf distribution with d=0.271.

- The database consists of single media type, where $C = 0.5$ Mbyte/sec and $O = 3.6$ Gbyte.
- The database adheres to a strict memory inclusion policy; such that objects on main memory are a subset of the objects on secondary memory, and objects on secondary memory are a subset of objects on tertiary memory (i.e., OC adhere to strict inclusion policy).
- When available on multiple memory levels, objects are displayed from the fastest memory level only, even when the other memory levels are idle.
- Main memory consists of 64 Mbyte DRAM modules, where the cost of a module is ≈ 400.
- Secondary memory consists of 9 Gbyte disk drives, where the cost of a drive is ≈ 1500. To support continuous display, we assume that disk drives are configured with 2 Mbyte blocks and each disk supports 28 simultaneous displays.
- Tertiary memory consists of 980 Gbyte tape juke boxes, where the cost of each juke box with its cartridges is $\approx 10,000$. Tape access time is 82 seconds, $T_{Access} = 82$, and transfer rate is 10 Mbyte/sec, $D_T = 10$. We assume hierarchical caching, and the maximum number of streams that can be supported by the cache space is 16 displays.

In Table 1, we show HSS configurations for a range of database sizes (from 50 Gbyte up to 10 Tbyte), and a range of desired throughput (from 10 simultaneous displays up to 10,000 displays). For every given database size and desired throughput, the number of storage modules required to hold the database and support continuous display are shown,

where n_M, n_D, and n_T are the number of memory modules, disks, and tape juke boxes. We use Zipf distribution to model the access pattern to the HSS, where the frequency of access to an object j is defined to be $F(j) = \frac{c}{j^{1-d}}$ (c is the normalization constant and d controls access frequency drop off). We set $d = 0.271$ to closely match movie rental access pattern [4].

In Table 1, for a range of databases (i.e., 500 Gbyte to 10 Tbyte) and low throughput requirements (i.e., 10-100 displays, depending on the database size), it is not necessary to use disk caches to store complete objects, i.e, no OC is required. The basic tape bandwidth is sufficient for the application requirement. As mentioned earlier, some DC is required to guarantee the continuous display from the tape libraries. However, for the given databases, as the throughput requirement increases, the references to the working set increases and the tapes might become a bottleneck. In this case, it is possible to either: 1) add more disks and/or memory to cache the working set (i.e., OC), or 2) add more tapes and tape drives (i.e., tape replication). For large databases, the later provides the most cost effective solution for low throughput requirements (up to 1000 displays). However, displaying directly from tapes has the following disadvantages: 1) high initial latency, 2) support for video-on-demand functionality becomes complex. Therefore, the utility of displaying directly from tapes is limited by the type of application being considered. For higher throughput requirements, OC on disks provides the best solution for a wide range of requirements, whereas OC on main memory provides the best solution for small databases and very high throughput requirements (e.g., 100 Gbyte and 10,000 display).

4 Future Work

As part of our future research, we will consider the following extensions. First, our current focus was on optimizing for cost per display. It is possible to extend our configuration planner to consider other application requirements, such as initial latency and storage reliability. For example, we expect that storage reliability to be inversely proportional to the data access from tertiary devices [5], and, hence, tape devices should be accessed less frequently. Second, we plan on studying the effects of a changing data access pattern and database size on the optimal configuration. Finally, we plan on studying the performance effects of data placement on tapes.

Acknowledgments

We thank the referees and, specially, Rodney Van Meter for his insightful comments. This research was supported in part by the National Science Foundation under grants IRI-9258362 (NYI award) and ERC grant EEC-9529152, and a Hewlett-Packard unrestricted cash/equipment gift.

References

[1] S. Berson, S. Ghandeharizadeh, R. Muntz, and X. Ju. Staggered Striping in Multimedia Information Systems. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 79–90, May 1994.

[2] M. Carey, L. Haas, and M. Livny. Tapes Hold Data, Too: Challenges of Tuples on Tertiary Storage. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 413–417, May 1993.

[3] R. Chambers and M. Davis. Petabyte Class Storage at Jefferson Lab. In *Proceedings of the 5th NASA GSFC Mass Storage Systems and Technology Conference*, September 1996.

[4] A. Dan, D. Sitaram, and P. Shahabuddin. Scheduling Policies for an On-Demand Video Server with Batching. In *Proceedings of the ACM Multimedia Conference*, pages 15–23, October 1994.

[5] A. L. Drapeau and R. H. Katz. Striped Tape Arrays. In *Proceedings of the Twelfth IEEE Symposium on Mass Storage Systems*, April 1993.

[6] D. J. Gemmell, H. M. Vin, D. D. Kandlur, P. V. Rangan, and L. A. Rowe. Multimedia Storage Servers: A Tutorial. *IEEE Computer*, 28(5), May 1995.

[7] S. Ghandeharizadeh, A. E. Dashti, and C. Shahabi. A Pipelining Mechanism to Minimize the Latency Time in Hierarchical Multimedia Storage Managers. *Computer Communications*, 18(3), March 1995.

[8] S. Ghandeharizadeh, R. Zimmermann, W. Shi, R. Rejaie, D. Ierardi, and T.-W. Li. Mitra: A Scalable Continuous Media Server. *Multimedia Tools and Applications Journal*, 5(1):79–108, July 1997.

[9] B. Hillyer and A. Silberschatz. On the Modeling and Performance Characteristics of a Serpentine Tape Drive. In *Proceedings of the ACM Sigmetrics International Conference on Measurement and Modeling of Computer Systems*, pages 170–179, May 1996.

[10] S. Sarawagi and M. Stonebraker. Reordering Query Execution in Tertiary Memory Databases. In *Proceedings of the 22nd VLDB Conference*, pages 156–167, September 1996.

[11] F. Tobagi, J. Pang, R. Baird, and M. Gang. Streaming RAID-A Disk Array Management System for Video Files. In *Proceedings of the First ACM Multimedia Conference*, August 1993.

[12] P. Triantafillou and T. Papadakis. On Demand Data Elevation in Hierarchical Multimedia Storage Server. In *Proceedings of the 23rd VLDB Conference*, pages 226–235, August 1997.