Efficient Memory Management in VOD Disk Array Servers Using Per-Storage-Device Buffering

Alberto García-Martínez
D. Electrónica y Sistemas
Universidad Alfonso X el Sabio
Madrid (Spain)
alberto@cesat.es

Jesús F. Conde
D. Computer Science
University of Massachusetts
Amherst, MA. (USA)
jefer@cs.umass.edu

Angel Viña
D. Electrónica y Sistemas
Universidade da Coruña
A Coruña (Spain)
avc@des.fi.udc.es

Abstract

In this paper we present a new buffering technique that reduces Video-On-Demand server memory requirements in more than one order of magnitude. This technique, Per-Storage-Device Buffering (PSDB), is based on the allocation of a fixed number of buffers per storage device, as opposed to existing solutions based on per-stream buffering allocation. The combination of this technique with disk array servers is studied in detail, as well as the influence of Variable Bit Streams. We also present an interleaved data placement strategy, Constant Time Length Declustering, that results in optimal performance in the service of VBR streams. PSDB is evaluated by extensive simulation of a disk array server model that incorporates a simulation-based admission test.

1. Introduction[1]

The high rates and large quantities of data required for serving Video on Demand (hereafter VOD) are causing a growing demand for computing resources, especially in large-scale disk-array server systems. Among these increasing requirements, we highlight the significantly large amount of memory required for scheduling and transmission purposes.

Most of the current cycle-based models assume the existence of an underlying per-stream memory management scheme, in which the number of buffers needed is proportional to the number of streams (i.e., clients) that are served simultaneously. In general, per-stream buffering requires a substantially large amount of memory in high bandwidth systems. The most popular per-stream buffering implementation, Double Buffering, is based on the co-operation of two buffers of identical size per stream. While the first buffer is being filled with data retrieved from disk, a network I/O device reads the information previously retrieved from disk and stored in the second buffer. When the retrieval period ends, the two buffers interchange their roles.

In this paper we present Per Storage Device Buffering (PSDB), a buffering policy that allocates a fixed number of buffers per storage device, as opposed to the reservation of memory on a per-stream basis. In the base case, two buffers are allocated per storage device, and the synchronization is established as follows: while the first buffer is receiving data from disk, the second one is delivering data corresponding to the previous client. Buffers interchange roles whenever a fragment of data is read from disk. PSDB reduces the server's memory requirements in more than one order of magnitude, although it forces the computation of a new buffering admission test that may result in small performance degradation. Further adaptation is required due to the constraints imposed by the use of Variable Bit Rate (VBR) streams and disk array storage systems.

The remainder of the paper is structured as follows: in section 2 we analyze several basic design issues for disk array servers. In section 3 we describe and characterize the PSDB buffering technique, discussing the influence of VBR streams. The simulator is described in section 4, and the results obtained are presented and analyzed in section 5. In section 6 we summarize related work. Finally, section 7 is devoted to conclusions.

2. Design parameters for disk array servers

In this section we consider the following relevant issues in the design of a disk array server:

- The scheduling policy.
- The enforcement of quality of service parameters by the use of an admission control algorithm.
- The distribution of the contents across multiple disks.

[1] This research was supported in part by the National R&D Program of Spain, Project Number TIC97-0438

1089-6503/98 $10.00 © 1998 IEEE

- The network delivery policy.

2.1. Scheduling policies

The definition of a global period is considered a convenient strategy for scheduling client requests in the server [11]. Depending on the order established inside each period, the main two approaches are:

- *Round Robin*: All the clients are served in the same fixed order on each round. This restriction is often enforced by the division of a cycle into equally sized slots [14].
- *SCAN*: Clients are served according to the position in the disk of the streams they request, trying to avoid unnecessary disk head movements. Disk efficiency is greatly increased compared to Round Robin (and compared to real-time scheduling algorithms like EDF or RMT), but worst case latency is doubled.

2.2. Admission control

The server can provide several quality-of-service levels in terms of the guarantees offered about the number of frames lost [11]:

- *Deterministic*: The server guarantees that all frames are delivered in time.
- *Statistical*: Deadlines are guaranteed to be met with a certain probability.
- *Best effort*: The system offers no guarantees.

An admission test, computed each time a new client requests a content, is performed to check the availability of the resources required for a given quality-of-service level. The conditions tested refer primarily to disk, network and memory availability.

Variations in stream playback requirements (when VBR streams are used) and disk transfer rate compromise the efficiency of worst-case admission tests. Statistical tests provide better resource utilization, but do not ensure jitter-free operation. Fortunately, service requirements are known in advance when a client demands a movie, as long as the client does not issue VCR-like operations such as rewind or fast forward. In this scenario, *simulation-based* admission tests (e.g. the *Ideal Deterministic* admission test [4]) offer deterministic guarantees and optimal performance by estimating future service demands. The scheduler uses a model of the underlying hardware and software to perform an admission test based on the pre-computed requirements of each cycle. The model can be built over a simple average performance estimation [14], or may involve a time-consuming simulation of the whole multimedia system (disk, buses, operating system, network interface, etc.).

2.3. Data placement

The data placement strategy is crucial in system's performance, since it determines the amount of time spent in disk head movement and the transfer speed in zoned disks. For single disks, it is highly recommended to store contiguously all the sectors that correspond to the same content (or at least, to the amount of data that will be read in a cycle for each content).

Logical consecutive sectors can be distributed over several physical devices in many ways. Scattering the contents into several disks increases system's bandwidth and availability, since popular movies are not restricted to the bandwidth of a single disk. Two options are considered in the distribution of the contents [11]:

- *Data striping*: Several physical sectors belonging to different disks are accessed in parallel in order to obtain a larger logical sector. All the disks are involved in each request. This configuration requires a special disk controller to preserve spindle synchronization.
- *Data interleaving*: Each disk serves its requests in an independent fashion. Fragments corresponding to a single request can be stored entirely in one or more disks, but not necessarily in all disks. Other fragments belonging to the same content can be placed in different disks. Declustering among all the disks has been traditionally performed in fixed-size fragments (*Constant Data Size Declustering*, CDSD). The *striping unit* is the amount of logically contiguous data stored on a single disk.

We propose another data distribution technique for the service of VBR streams, *Constant Time Length Declustering* (CTLD), based on the division of a content in variable size fragments. Each fragment accounts for the playback of a client in a single cycle and is stored in a single disk. Different fragments can be placed in different disks. This strategy guarantees that only one seek is incurred in the service of a request for a given cycle.

2.4. Delivery policy

Bernhardt et al. [2] identify two different data delivery policies.

- *Bursty Transfer* mode: The server sends, in just one transfer, all the data that a client needs in a cycle. Memory buffers at the client store the fragment until its playback.

- *Continuous Transfer* mode: The client receives smaller fragments of data with more frequency. Each fragment should maintain the playback for a small period (for example, a Group of Pictures for MPEG compression).

Continuous Transfer, compared to Bursty Transfer, reduces client's memory requirements and allows a smoother transmission flow over the network. However, processing small size packets is more inefficient than processing large ones, and data delivery scheduling is more complex. Bursty Transfer is an attractive option when enough memory is available at the client side, for example in personal computers interfacing Enterprise Video Servers, video-game stations, etc.

3. PSDB in VBR disk array servers

3.1. Preliminary considerations

It is convenient to state the following assumptions and conditions of applicability of PSDB:

- We define a *storage device* as the minimum storage subsystem that can serve a single disk request; it could be a single disk, each disk in an interleaved disk array or the whole disk array in a spindle-disk configuration.

- A global-period based scheduling algorithm is used. We will choose C-SCAN (a variation of SCAN in which data is read only when the disk head traverses from the outer zone to the inner zone) for our study, although an analogous analysis can be carried out for other cycle-based scheduling algorithms.

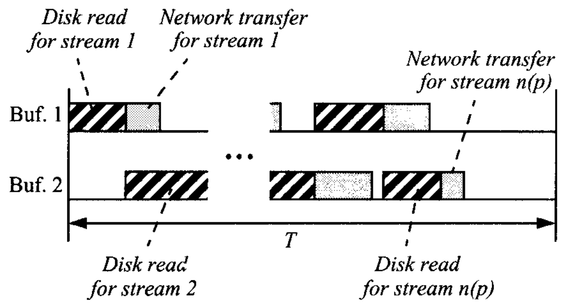
- PSDB requires the use of Bursty Transfer mode.

- The server is *disk bounded*, i.e., network and bus are not bottlenecks. The network has enough resources, and never discards packets. I/O operations are performed in parallel with bus transmissions. Bus influence is at most a second order factor, as confirmed by some studies based on some studies based on performance analysis of data intensive applications using fast SCSI and ATM networks on high-end UNIX stations ([12]).

- Interaction between client and server is achieved through a *server-push* model, rather than a *client-pull* one. Clients and server are loosely coupled, and network transfers are not driven by acknowledgments from the clients.

3.2. PSDB model description

PSDB replaces a set of buffers allocated on a per-stream fashion by a fixed number of buffers per storage device. Let us analyze the base case first: when two buffers are used (PSDB-2), while one buffer is receiving data from the storage device corresponding to the stream *i+1*, the other buffer is delivering data corresponding to stream contains data, corresponding to the stream *i* to the network. These two buffers interchange their tasks whenever a whole fragment is read from the storage device (see figure 1).



Disk read for stream 1
Network transfer for stream 1
Network transfer for stream n(p)
Buf. 1
Buf. 2
T
Disk read for stream 2
Disk read for stream n(p)

Figure 1. Buffer management in PSDB-2

The admission test for PSDB-2 checks two conditions upon the arrival of a new request:

- The network buffer has to be completely empty before the buffer switch occurs. The switch is caused by an interrupt generated by the disk when then retrieval of a fragment of data.

- All the network transferences have to be performed before the end of the current period. As a result, network transfer cycle and disk scheduling cycle will stay synchronized. Guaranteeing that the information that corresponds to a cycle is delivered in the same cycle reduces the maximum start-up latency from two cycles to one cycle in a SCAN-driven server. As a drawback, this condition is responsible for a short disk idle time on each cycle. Note that this condition is not mandatory for the application of PSDB.

Network transfers, triggered by disk interrupts, are started after a certain delay (L_{net}). This delay accounts for the sum of two components: the time taken to execute the interrupt code and issue the transference (which embodies the part of the network protocol processing corresponding to the first fragment of data to send), and the delay the network interface may suffer when the information is transmitted.

The parameters used in the analysis and evaluation of the PSDB model are listed in table 1.

Table 1. Model parameters

T	Global cycle period
$n(c)$	Number of users served in cycle c
τ	Set of periods in a VOD server's run
$b(s,c)$	Number of bytes requested by stream s in cycle c
$L(s,c)$	Disk latency incurred when switching to stream s in cycle c
$R_{disk}(i,p)$	Disk transfer rate for the data requested by stream i in period p
$L_{net}(i,p)$	Time employed in starting network processing (interrupt service, first part of the protocol processing that corresponds to the first fragment of data to send) plus the maximum delay the network interface may suffer when the information is transmitted
R_{net}	*Network transfer rate, including protocol processing in steady state*
M	*Memory required for scheduling*
B	*Number of buffers used in PSDB*

The two conditions of the admission test can be expressed as

$$\forall (p \in \tau) \qquad \forall i\,(2 \le i < n(p))$$
$$L(i+1,p) + \frac{b(i+1,p)}{R_{disk}(i+1,p)} > L_{net}(i,p) + \frac{b(i,p)}{R_{net}} \qquad (1)$$

$$\forall (p \in \tau)$$
$$\sum_{j=1}^{n(p)} \left(L(j,p) + \frac{b(j,p)}{R_{disk}(j,p)} \right) + L_{net}(n(p),p) + \frac{b(n(p),p)}{R_{net}} \le T \qquad (2)$$

that should hold for all the periods that the new stream is expected to last.

The left side of equation (2) is an upper bound of the time taken in disk transference, so it is sufficient to assure that all disk operations are performed in a cycle, and consequently no additional scheduling tests are needed.

If more buffers are added to the per-storage-device buffer pool, condition (1) should be replaced with the requirement of delivering to the network all the data available in a buffer before the storage device attempts to reuse the buffer for disk operation (figure 2). Condition (2) remains unchanged.

The buffers added establish a cushion that protects network data integrity. This protection results in fewer

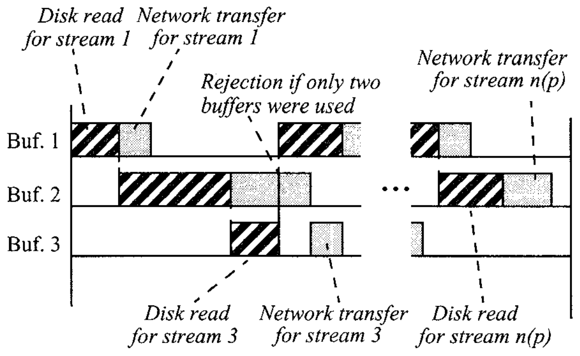

Disk read for stream 1
Network transfer for stream 1
Rejection if only two buffers were used
Network transfer for stream n(p)
Buf. 1
Buf. 2
Buf. 3
Disk read for stream 3
Network transfer for stream 3
Disk read for stream n(p)

Figure 2. Buffer management in PSDB-3

An upper bound for the memory required by PSDB is

$$\forall (p \in \tau) \qquad M = B \operatorname*{Max}_{i=1}^{n(p)} (b(i,p)) \qquad (3)$$

while the amount of memory required by Double Buffering is upper-bounded by

$$\forall (p \in \tau) \qquad M = 2\ Max \sum_{i=1}^{n(p)} b(i,p) \qquad (4)$$

3.3. Per storage-device buffering of VBR streams

The admission test proposed in the previous section is highly dependent on variations in the number of bytes requested in a cycle, especially when disk and network transfer rates are similar. The difference between disk and network transfer speeds in state-of-the-art hardware configurations is usually too small to absorb the rate divergences found in VBR contents. Therefore, pure PSDB-2 would lead to extremely low performance.

Some solutions to overcome performance degradation are:

- Increase the number of buffers allocated per storage device (PSDB-3, PSDB-4,...), at the cost of a slight memory usage increment.
- Use PSDB-2 with *Multiple Cycle Reading* (MCR): If the admission test fails because network delivery for stream i takes longer than disk retrieving for stream $i+1$ (figure 3), we increase disk reading time by adding all the data required for stream $i+1$ corresponding to the next cycle (figure 4). Therefore, client $i+1$ receives data corresponding to the current and the next cycle, (in the next cycle, this stream will be skipped). If the test still fails in the modified schedule, subsequent cycles are

grouped in a single read for client *i+1*. Finally, if the scheduling test fails on the modified schedule, the stream tested for admission is rejected.

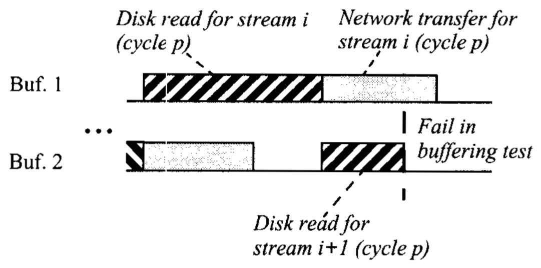

Disk read for stream i (cycle p)
Network transfer for stream i (cycle p)
Buf. 1
Fail in buffering test
Buf. 2
Disk read for stream i+1 (cycle p)

Figure 3. Fail in PSDB-2 buffering test

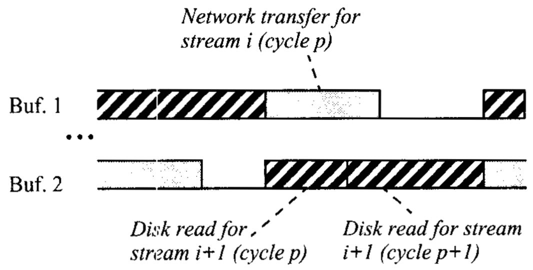

Network transfer for stream i (cycle p)
Buf. 1
Buf. 2
Disk read for stream i+1 (cycle p)
Disk read for stream i+1 (cycle p+1)

Figure 4. PSDB-2 - Multiple Cycle Reading

Notice that, in order to be able to use the pre-computed information that characterizes the requirements of the movie, *b(i,p)*, data corresponding to the same cycle of a client should not be split.

4. PSDB simulation model

We have used a disk array simulator built over Sim++ [7], a discrete event simulation package for the evaluation of PSDB. Disk information is based on a Seagate ST31200W zoned disk [10], including disk head movement, rotation and variable transfer speed, depending on the zone considered (maximum transfer rate: 4.17 MByte/s; minimum transfer rate: 2.33 MByte/s). Neither the bus that connects the disks nor the internal caches have been modelled.

In interleaved disk arrays, each storage device operates as an independent disk. Before the beginning of the cycle, the disk manager thread splits the requests into subrequests and assigns them to their corresponding storage device. Data reordering is performed at the client side, so no synchronization is required among the storage devices.

The network interface has an initial transfer latency of one millisecond, that accounts for the worst case service time of the disk interrupt, the beginning of network protocol processing and the delay in accessing the network. Unless otherwise stated, a constant transfer rate of 60 Mbit/s (based on measures of UDP/IP transmission over ATM OC-3 [12]) is assumed, embodying steady state protocol processing and network transfer. One network interface (with a bandwidth proportional to the number of disks) is assigned to each storage device.

We are using VBR-encoded MPEG-1 traces from Bellcore [9] (`StarWars`) and from Wuerzburg University [16] (`Asterix`, `MTV`, `Simpsons` and `Video`). Streams are placed contiguously in the disk, extended along the whole surface, with equal spare zones among them. GOP (Group of Pictures) units, 12 frames for the traces considered, are never broken.

The C-SCAN scheduler retrieves data in fragments corresponding to the playback of a ten-seconds cycle unless MCR is used.

Client arrivals have been modelled using a time-homogeneous Poisson process, where minimum interarrival values have been established. The parameters are chosen to obtain system overload conditions. When a client is rejected, the same request is performed in subsequent cycles until it is accepted. The selection of movies is driven by a Zipfian distribution with a skew of 0.271, value observed in video rental requests [17]. Movies that are closer to the outer edge of the disk (with higher transfer rates) are considered to be more popular.

It is important to note that the primary parameter for performance evaluation should be the *Effective Transfer Rate* (ETR), i.e., the number of bytes transferred per time unit (disk utilization is skewed for zoned disks and the number of clients is not appropriate for heterogeneous VBR streams). We also compute the *Memory Saving Factor*, obtained dividing the amount of memory used by Double Buffering into the equivalent value for SPB.

5. Experimental results and analysis

5.1. Data placement policies performance analysis

A preliminary analysis is required to focus on the data placement configurations that provide higher performance. We use Double Buffering to test:

- Spindle synchronized disk arrays.
- Constant Time Length Declustered (CTLD) disk arrays.
- Constant Data Size Declustered (CDSD) disk arrays, with a striping unit larger than the request of any stream

for a cycle. Note there is no guarantee that data corresponding to a cycle of a stream is placed in a single storage device. If the striping unit used is small, results are similar to the spindle-synchronized case, so we will not consider this case.

In figure 5 we can see that spindle-synchronized disk performance degrades as the number of disks grows, due to the larger number of disk head movements. Commercial products featuring more than eight disks are infrequent.

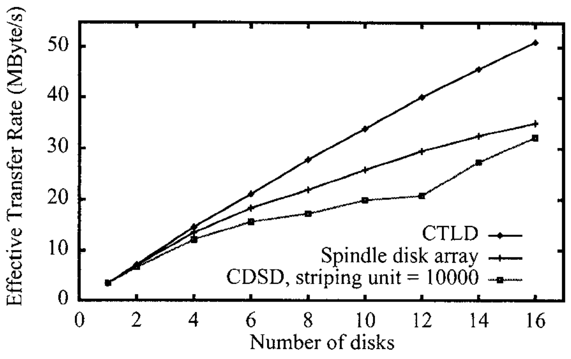


Figure 5. Effective Transfer Rate vs. number of disks for different data placement strategies (with double buffering)

Large striping unit CDSD disk arrays present low performance in the service of VBR streams. This can be explained by the variable size of the requests for different clients, generating a skewed demand distribution across the disks. Some disks can be heavily loaded in some cycles, forcing client rejections, while others may be idle in the same cycle.

CTLD disk arrays serving VBR contents result in a more even distribution of the load across all disks and a minimum number of disk head movements, offering the best performance. In our experiments, figures representing the load of each disk also confirm that load is more evenly distributed in CTLD disk arrays than in CDSD disk arrays.

5.2. PSDB performance and memory usage analysis

In this subsection, we compare performance and memory use for Double Buffering and PSDB. The use of pure PSDB would lead to an unacceptable number of rejections, so we have incorporated Multiple Cycle Reading (MCR) to all the tests. We concentrate on spindle-synchronized and CTLD disk arrays, since they provide the best performance in the service of VBR streams.

Figure 6 shows that the Effective Transfer Rate achieved for PSDB is close to the Double Buffering case. The maximum performance degradation observed in the series (compared to Double Buffering) is less than 6.7% for PSDB-2, and less than 1.3% for PSDB-3.

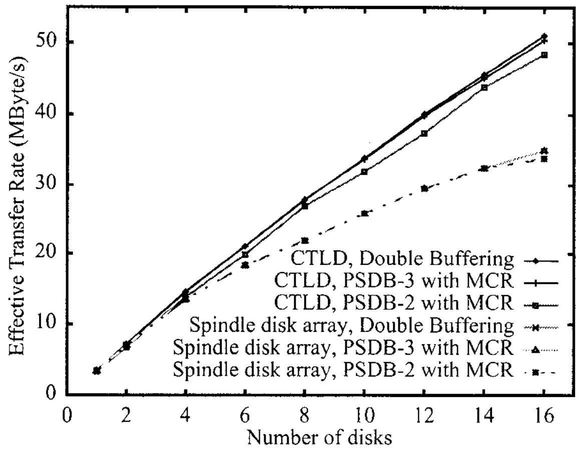


Figure 6. Effective Transfer Rate vs. number of disks for different buffering policies

In figure 7 we can see a logarithmic representation of the amount of memory consumed by each buffering policy. Double Buffering memory usage grows with the number of disks, ranging from 150 MBytes to 2.5 GBytes. CTLD is particularly appropriate to PSDB because short disk services are avoided by the placement in a single storage object of the data corresponding to a client's cycle, reducing the need for MCR when compared to CDSD.

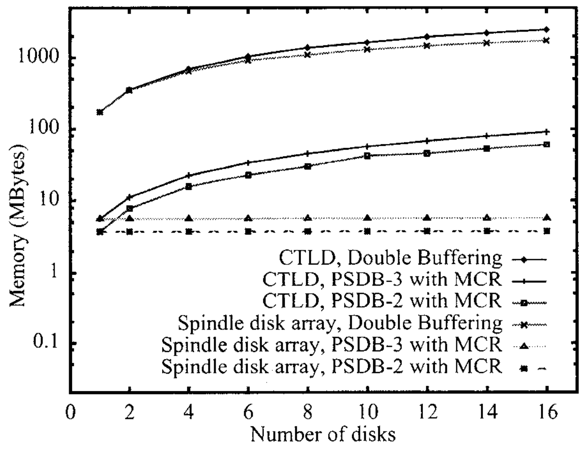


Figure 7. Memory usage vs. number of disks for different buffering policies

Both PSDB-2 and PSDB-3 solutions lead to Memory Saving Factors greater than 27. For spindle disk arrays, the amount of buffers needed does not depend on the number of disks used, resulting in Memory Saving Factors of about two orders of magnitude. On the contrary, for interleaved disk arrays we have to allocate a new set of buffers per additional storage device, so memory requirements are increased. Nevertheless, the total amount of memory is still below 100 MBytes.

5.3. Network transfer rate influence on PSDB

While Double Buffering systems are almost independent of network transfer speed, this factor affects PSDB-2 servers. On one hand, if the difference between network and disk speeds were high enough, the admission test would always succeed. For high margins, PSDB-2 and PSDB-3 perform similarly, resulting the first option in lower memory consumption. On the other hand, when network speed is close to disk transfer rate, buffer conflicts increase the number of cycles grouped due to MCR. Requests are accumulated in some cycles and performance decreases (see figure 8 - note that the maximum disk transfer speed is 33.4 Mbit/s). In this case, PSDB-3 is the only means to achieve performance figures similar to those of Double Buffering. Increasing the number of buffers of PSDB beyond three is only advisable for unbalanced systems presenting extremely low network transfer rates.

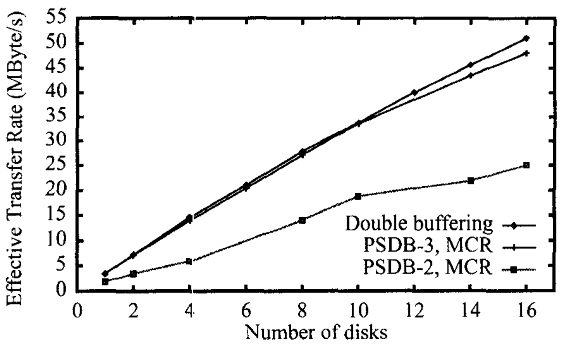


Figure 8. Effective Transfer Rate vs. number of disks for a network transfer speed of 35 Mbit/s (CTLD disk array)

6. Related work

Reutilization of buffers is a common strategy in Operating System's practice, but its application to the VOD environment, in which timely delivery of huge amounts of data is mandatory, has not been carefully studied.

Just-in-time scheduling [3] includes a per-device buffering technique for Continuous Transfer delivery in large arrays composed of independent disks. Clients are serviced in a SCAN basis, keeping the same order from one disk to another due to a stringent data placement strategy that forces all contents to be placed in the same zone on all the storage devices. There is a gap between the start of the cycles from disk to disk, which stands for the playback time of the data sent for a client. The server delivers the data just after being read from disk, requiring three buffers per disk. However, the model considered is too simple: the time taken for reading a track is constant; seek times are accumulated in a single outermost-to-innermost sweep; and the most important restriction, contents are equally demanding CBR streams. Disk and network issues are not considered to evaluate the feasibility of the solution.

The Tiger Filesystem [1] defines time slots that can accommodate the largest disk transfer that may occur. In the design of the system, performance is not considered a capital factor. Data is sent immediately after the end of the slot, but no mention is made to buffer reuse. Finally, the applicability conditions are not clearly stated.

Some adaptations have been proposed to per-stream buffering strategies to overcome excessive memory consumption, being the most relevant *Grouped Sweeping Scheduling,* GSS [6]. GSS establishes a compromise between Round Robin and SCAN by dividing of a cycle into G groups. Each client is assigned to a group, and each group is served in a previously established order. The streams in each group are served in a SCAN basis. Buffers can be reused from group to group, and the total number of buffers required is $n(p) + \left\lceil \frac{n(p)}{G} \right\rceil$. However, the total amount of memory needed still depends on the number of users served. In order to establish a comparison with data provided in our article, note that this amount of memory never falls below 50% of the memory required for Double Buffering. In general, all per-stream buffering techniques suffer from requiring an amount of memory proportional to the number of users being served.

Subgrouping and subcycling [18] focuses on buffer reutilization for Continuous Delivery disk array servers. G groups are formed, and the starting cycle time is shifted from group to group. Disk accesses on each cycle are also divided into G subcycles. This technique, similar to GSS, requires equally demanding CBR streams.

Mourad [13] develops a similar technique, based on GSS and offset schedules. It presents the same problems than GSS and Subgrouping and Subcycling, being the most important the requirement of CBR streams.

Ng [15] combines buffer reuse with Double Buffering, to free up to 50% of the memory, but the scheme is difficult to implement, requiring non-contiguous buffer management.

Chang [5] considers the use of SCAN scheduling with spaced I/O operations to be able to share buffers efficiently. Memory saving can be enforced by the use of scheduling policies that allow free movement of the disk arm (like Round Robin or GSS). Again, their proposal is based on per-stream buffering and considers only CBR streams.

7. Conclusions

The main contribution of this paper is the presentation and validation of PSDB, a new buffering technique for cycle-based schedulers in disk array VOD servers. PSDB is based on the analysis of the disk/network interface interaction to improve buffering memory requirements. This technique is applicable to VBR streams if Multiple Cycle Reading is used, or if three or more buffers are allocated per storage device. Compared to existing per-stream buffering allocation schemes, we report important memory saving factors proportional to the number of streams served. The required admission test slightly decreases system's performance, although experimental results show that this penalty is negligible. There are also extra network bandwidth and computing demands, but enough resources are generally available in off-the-shelf systems.

For testing purposes, we have adapted a simulation-based scheduler to disk array servers, allowing different data placement configurations. Simulation is a must to obtain both high performance and deterministic guarantees in VBR stream servers.

It is worth to recall that PSDB-3 offers Double Buffering performance for typical VOD systems with very low memory consumption. We have obtained useful conclusions about data placement strategies, being the most relevant the appropriateness of Constant Time Length Declustering in the service of VBR streams.

Further analysis about the margin required between disk transfer speed and network transfer speed, and about the influence of cycle length on single disk operation can be found in [Garc98].

8. References

[1] W. J. Bolosky et al. The Tiger Video Fileserver. *Proceedings of the 6th NOSSDAV.* Zushi, Japan, April 1995.

[2] C. Bernhardt, E. Biersack. The Server Array: A Scalable Video Server Architecture. Chapter 5 in *High-Speed Networking for Multimedia Applications*. O. Spaniol, W. Effelsberg, A. Danthine, D. Ferrari. Eds. Kluwer, March 1996.

[3] S. Berson, R. R. Muntz. Just-in-time Scheduling for Video-on-Demand Storage Servers. *Technical Report, UCLA* Computer Science Department, April 1995.

[4] E. Chang, A. Zakhor. Cost Analyses for VBR Video Servers. *IEEE Multimedia*, Vol. 3, nº 4, Winter 1996, pp. 56 - 71.

[5] E. Chang, H. Garcia-Molina. Effective Memory Use in a Media Server. *Proceedings of the 23rd Very Large Data Base Conference*, Athens, Greece, 1997.

[6] M.-S. Chen, D. D. Kandlur, P. S. Yu. Optimization of the Grouped Sweeping Scheduling (GSS) with Heterogeneous Multimedia Streams. *Proceedings of the ACM Multimedia Conference*, Anaheim, August 1993.

[7] R. M. Cubert, P. Fishwick. Sim++, Version 1.0. *Technical Report.* Department of Computer and Information Science and Engineering, University of Florida, July 1995.

[8] A. García-Martínez, J. F. Conde, A. Viña. Single Pair of Buffers: Reducing Memory Requirements in VBR Media Servers. To appear in *Proceedings of the European Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services (IDMS'98)*, September 1998.

[9] M. Garrett and W. Willinger. Analysis, Modeling and Generation of Self-Similar VBR Video Traffic. *Proceedings ACM SIGCOMM*, August 1994, pp. 269-280.

[10] S. Ghandeharizadeh, J. Stone, R. Zimmermann, Techniques to Quantify SCSI-2 Disk Subsystem Specifications for Multimedia. *Technical Report TR 95-610.* University of Southern California, 1995.

[11] J. Gemmell, H. M. Vin, D. D. Kandlur, P. V. Rangan. Multimedia Storage Servers: A Tutorial. *IEEE Computer Magazine*, vol. 28, issue 5, May, 1995, pp. 40-49.

[12] W. Johnston, B. Tierney, J. Lee, G. Hoo, M. Thompson. Distributed Large Data-Object Environments: End-to-End Performance Analysis of High Speed Distributed Storage Systems in Wide Area ATM Networks. *Proceedings of the NASA / Goddard Conference on Mass Storage Systems and Technologies*, 1996.

[13] A. N. Mourad. Issues in the Design of a Storage Server for Video-On-Demand. *Multimedia Systems*. Springer Verlag, April 1996, pp. 70 - 86.

[14] G. Neufeld, D. Makaroff, N. Hutchinson. Design of a Variable Bit Rate Continuous Media File Server for an ATM Network. *Proceedings of the Multimedia Computing and Networking Conference*, San Jose, CA, January 1996.

[15] R. T. Ng, J. Yang. An Analysis of Buffer Sharing and Prefetching Techniques for Multimedia Systems. *Multimedia Systems*, Springer Verlag, April 1996, pp. 55 - 69.

[16] O. Rose. Statistical Properties of MPEG Video Traffic and Their Impact on Traffic Modelling in ATM Systems. *Technical Report 101*, Institute of Computer Science, University of Wuerzburg, February 1995.

[17] W. Tetzlaff, R. Flynn. Block Allocation in Video Servers for Availability and Throughput. *Proceedings of the Multimedia Computing and Networking Conference*, San Jose, CA, January 1996.

[18] F. Tobagi, J. Pang, R. Baird and M. Gang. Streaming RAID - A Disk Array Management System for Video Files. *Proceedigns ACM International Conference on Multimedia*. Anaheim, CA, August 1993.