A. García-Martínez[a,*], J. Fernández-Conde[b], A. Viña[c]

[a]*Departmento de Tecnologías de las Communicaciones, Universidad Carlos III, Madrid, Spain*
[b]*Department of Computer Science, University of Massachusetts, Amherst, MA, USA*
[c]*Departmento deElectrónica y Sistemas, Universidade da Coruña. A Coruña, Spain*

Abstract

In this article we present, analyse and evaluate a new memory management technique for video-on-demand servers. Our proposal, Memory Reservation Per Storage Device (MRPSD), relies on the allocation of a fixed, small number of memory buffers per storage device. Selecting adequate scheduling algorithms, information storage strategies and admission control mechanisms, we demonstrate that MRPSD is suited for the deterministic service of variable bit rate streams to intolerant clients. MRPSD allows large memory savings compared to traditional memory management techniques, based on the allocation of a certain amount of memory per client served, without a significant performance penalty. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Memory management; Video on demand server; Variable bit rate streams; Ideal deterministic admission control; Disk scheduling

1. Introduction

In Video on Demand (VoD) servers, the interactive transference of multimedia contents to a large number of users via network requires a significant amount of computing resources. All the hardware and software elements have to work co-ordinately in order to offer the desired quality of service at a minimum cost per client.

We will focus our attention on the efficiency of the process of transferring multimedia information from the VoD server to the users. Due to the inherent characteristics of the service of multimedia contents (temporal restrictions, large volumes of data and high bandwidth requirements), appropriate policies for server resource scheduling, information storage management and client admission control should be carefully selected. These strategies are not independent but they are closely related with each other.

Memory management is an important topic that has to be taken into account when specifying how to schedule server's resources. Most of the current VoD scheduling models assume the existence of an underlying *per-client memory management* scheme, where the number of buffers needed is proportional to the number of clients served simultaneously. This buffering strategy demands a substantially large amount of memory, given the fact that the number of concurrent sessions maintained by a VoD server is usually high.

In this article we propose an innovative memory management policy, *Memory Reservation Per Storage Device* (MRPSD), that dramatically reduces the large amount of memory required for multimedia information transmission in VoD servers when used in combination with any cycle-based scheduling algorithm. Our policy is based on the allocation of a fixed number of buffers per storage device, as opposed to the reservation of memory on a per-client basis.

Compression techniques generate information blocks of different sizes corresponding to the same client playback time. The use of formats that benefit from these techniques in VoD systems is widely established, with the purpose of reducing the required bandwidth and server storage capacity. As a result, a realistic memory management strategy should consider the existence of Variable Bit Rate (VBR) streams in the service of multimedia contents. When proper scheduling algorithms, storage strategies and admission control mechanisms are selected, MRPSD performs similarly to per-client memory management schemes in the deterministic service of VBR contents to intolerant clients.

We present and evaluate the results obtained by detailed simulation of a VoD system devoted to the service of VBR multimedia contents to intolerant clients, incorporating our *per-storage-device* memory management scheme and appropriate policies for scheduling, storage and admission control. Its performance and memory requirements are compared with analogous VoD systems that implement per-stream buffering techniques.

* Corresponding author. Address: Universidad Carlos III. Avda. de la Universidad, 30.28911 Leganés, Madrid, Spain.

E-mail addresses: alberto@it.uc3m.es (A. García-Martínez), jefer@cs.umass.edu (J. Fernández-Conde), avc@des.fi.udc.es (Á. Viña).

The specific distribution of the multimedia contents in storage devices has a major impact in the VoD system performance. Storage systems based in the co-operative use of several disks can increase the number of clients served. We include in our server model the possibility of incorporating several disks. In addition, different information storage alternatives will be studied and evaluated.

The rest of the paper is structured as follows: Section 2 details several major software architectural aspects of a VoD server that are relevant for the exposition of our work. Our memory management policy, MRPSD, is presented in Section 3. Section 4 deals with the experiments carried out on our VoD server simulator and the analysis of the results obtained. Related work is summarised in Section 5. We end with the conclusions.

2. VoD server software architecture

In this section, we provide general information about some major aspects of the VoD server's software architecture involved in the process of transferring continuous multimedia contents: resource scheduling, storage management and client admission control. Other related procedures such as security, client accounting, connection management, fault tolerance support, etc. are out of the scope of our study.

2.1. Scheduling of server resources

The option of sending all the information to the clients prior to its playback is not attractive, because this would increment client latency and decrement concurrency in the server. In addition, it would require large amounts of memory and storage capacity at the client side. It is preferred to send, with a certain frequency, small fragments of data that correspond to the playback of the information for a short time interval. The contents that are served in this fashion are known as *streams* [12].

The vast majority of VoD servers are based on the *server-push* paradigm, that is, once the client requests the service, the server is responsible for automatically sending information fragments to the client with the periodicity required [23]. This behaviour is more efficient than the *client-pull* model, where the client explicitly requests each fragment of information when it is needed in order to maintain continuous playback.

The use of cyclic scheduling algorithms is considered a convenient strategy for the service of multimedia contents. Cyclic scheduling algorithms rely on a common cycle for organising the service of all clients. The order in which clients are attended is decided by the server at the beginning of each cycle. In every cycle, the server sends a fragment of information to each client, according to the order established. The objective of the server is to send enough information in every cycle so that all the clients are able to maintain continuous playback of the information. We will assume that the cycle time is constant.

The scheduling of server resources can be better understood if we divide the process of sending data fragments to the clients into two different tasks. The first task is in charge of transferring the data fragments from disk to memory, whereas the second delivers them from memory to the network. The required synchronisation between both tasks is influenced by the memory management technique used by the scheduler. In the two tasks identified for the scheduling process carried out in the VoD server, the two physical elements potentially slower, as compared to CPU, I/O buses and memory, are the network interface and, specially, the storage device, leading to a *disk-bounded* system. Therefore the optimisation of storage device access is the main objective of the VoD server scheduler.

There is a large number of disk scheduling algorithms. The family of cyclic disk schedulers results particularly suited to the service of multimedia contents to different clients. Depending on the order established inside each cycle, we can distinguish:

- *Round Robin:* all the clients are served in the same order in every cycle. This is usually enforced by the division of a cycle into slots of equal size, assigning a fixed client to the same slot in all cycles. This scheme, known as *Slotted Round Robin*, presents small latency, although it is not bandwidth efficient because the service time for a client is usually smaller than the time reserved for a slot.
- *SCAN:* clients are served according to the position occupied by the streams requested in the storage device, in order to avoid unnecessary disk head movements. Worst-case latency is increased in one cycle as compared to Round Robin [12], but disk access efficiency is highly improved. SCAN is a very popular algorithm, due to its good performance. C-SCAN (Circular-SCAN), is a SCAN variation in which data is only read when the disk head travels from the outward to the inward of the disk.
- *Grouped Sweeping Scheduling* (GSS)[32]*:* it establishes a compromise between Round Robin and SCAN. Cycles are divided into *G* groups, with each client assigned to a group. Each group is served in a fixed order every cycle, and the streams in each group are served in a SCAN basis.

The size of the data fragments read from disk is an essential VoD server design parameter. We should observe that the larger the fragments, the more efficiency is obtained in accesses to the storage device, since the time spent in disk seek and disk rotation is less relevant. However, client latency and memory requirements also increase. Note that, in VBR contents, the size of the fragments vary depending on the cycle and the client.

Considering that the size of the fragments used for delivering data to the network can be different from the size used for reading data from disk, we can identify two ways of delivering information to the network [1]: *Continuous Mode* and *Bursty Mode*. In continuous mode, data is divided in very small fragments that maintain playback for a very short period of time (for example, a Group of Pictures for MPEG compression, or a single frame for JPEG). In bursty mode, larger fragments are sent; these fragments usually correspond to the amount of data required for a client in a cycle. Continuous mode, compared to bursty mode, 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 can be more complex.

Memory management in VoD servers has been traditionally associated with per-client reservation policies (e.g. in Refs. [1,3,6,7,11,15,17,19,21,22,30]). The most popular per-stream buffering implementation, *Double Buffering*, is based on the co-operation of two buffers of identical size allocated to *each* client. In a given cycle, while one buffer is being filled with data retrieved from the storage device, a network I/O device reads from the other buffer the information that was retrieved from the storage device in the previous cycle. In the following cycle, the two buffers interchange their roles. Its implementation requires large amounts of memory.

In addition to double buffering, other per-client techniques have been proposed in order to reduce the total amount of memory required. For example, in *Single Buffering*, a single buffer is reserved per client. The information read from disk in a cycle has to be sent to the network before the information reading corresponding to the following cycle. This technique is typically combined with Round Robin schedulers. The buffering technique proposed with Group Sweeping Scheduling establishes a compromise between double and single buffering in the number of buffers needed. Single buffering and the buffering technique associated to Group Sweeping Scheduling are limited by their per-stream nature to a 50% reduction of the amount of memory required with respect to double buffering. Despite the memory savings obtained with these two techniques, double buffering is still very popular due to its high grade of de-coupling between the read-from-disk and sent-to-network operations, that allows SCAN disk request ordering.

2.2. Information storage

Disk efficiency is determined by three major components in the process of data reading: seek time, rotation time and transfer time [25]. *Seek time* accounts for the time needed to place the disk head over the destination track, and depends on the number of tracks traversed. Typical values range from 1 to 20 ms. Once in the proper track, some time is spent while disk rotates to reach the first sector of the data. This time is called *rotation time*, and it depends on the rotational speed of the platters. Maximum values are usually around 10 ms. Disk transfer rates vary according to the disk zone considered: they are higher in outer tracks, that contain a larger number of sectors. Data layout, along with disk scheduling, are responsible for the actual values of seek time, rotation time and transfer time.

In a VoD server with a single disk, there exist several models for storing the data disk blocks of an individual content:

- *Scattered Placement:* each block can be allocated everywhere in the disk. The sequential access to data corresponding to a client in a cycle will incur in a large number of intra-file seeks and rotations, resulting in high disk read times.
- *Contiguous Placement:* all data blocks belonging to the same content are stored contiguously. This allocation strategy allows higher performance, compared to scattered placement, but it may suffer from external fragmentation. We can also split a content into fragments, storing contiguously the information of each fragment. These fragments may store the data required in order to maintain the playback for a client during a cycle. This allocation policy is called *Locally Contiguous Placement*. In cyclic scheduled servers, locally contiguous placement offers the same performance as contiguous placement, suffering less external fragmentation.
- *Constrained placement:* Vin and Rangan [30] studies the bound of the average distance, measured in tracks, between a finite sequence of blocks. This policy tries to establish a compromise in performance and fragmentation between scattered placement and contiguous placement. Constrained placement requires elaborated algorithms in order to guarantee that the constraints imposed are met. In addition, it does not consider rotation times, that usually are only slightly lower than seek times.

In systems containing several disks, we can distinguish two options for the distribution of contents among the disks [12]:

- *Data striping:* several physical sectors from all the disks are accessed in parallel, in order to obtain a larger logical sector. This configuration requires a special disk controller to achieve spindle synchronisation among all disks.
- *Data interleaving*: each disk serves its requests in an independent fashion. Fragments corresponding to a single request can be stored in one or more disks. The *striping unit* is defined as the amount of logically contiguous data stored in a single disk [14]. De-clustering among all the disks is generally performed in fixed size fragments.

The factors that present a stronger influence in the performance of a multi-disk server are:

- Efficiency in the use of each disk. It is important to reduce the relative importance of seek and rotation

times, in order to increase the relative amount of time devoted to data transference.
- Fairness in load distribution among all disks.

These two factors largely depend on the data distribution policy chosen, on the scheduling algorithm employed (access ordering, cycle time, etc.), and on the particular characteristics of the contents served (playback speed, variation pattern for a VBR content in the amount of data to be served on each cycle, etc.).

2.3. Admission control

VoD systems incorporate admission control mechanisms in order to limit the access of new clients to the system and thus preserve the quality of service of the clients previously admitted. A new client should specify the desired quality of service, so that the server can calculate the resources needed for that client and decide whether to admit the client or not. Whenever a new client asks for service, the VoD server has to perform an admission test, checking the availability of resources like storage device bandwidth, network bandwidth and memory.

Different approaches for specifying admission control tests can be followed [18], depending on the type of clients being served (*tolerant* or *intolerant* [31]). For *intolerant* clients, not accepting any discontinuity in the service, the VoD server should incorporate an admission control policy offering deterministic guarantees, that is, the quality of service must be assured. In the case of tolerant clients, statistic guarantees (a certain probability of discontinuity is given) or even best effort policies may be employed.

We can identify two kinds of admission control algorithms suitable for intolerant clients:

1. *Worst-case:* a new client is admitted only if there are enough resources to serve all the clients in a worst-case scenario, in which worst-case bounds for all parameters are used. In disk bounded systems disk seek, rotation and transfer times are the most restrictive parameters.
2. *Simulation-based:* simulation-based admission control algorithms rely on the prediction of the future behaviour of the system. Admission of a client depends on the estimation of the service time needed to satisfy all the client demands in subsequent cycles. The model used to make the prediction can be defined with different levels of detail. Worst-case admission control algorithms can be viewed as a particular case of simulation-based algorithms with an extremely simple simulation model. However, we will reserve the denomination of *simulation-based* for algorithms that require an accurate estimation of the data needed by all clients on each cycle. As worst-case algorithms perform poorly with VBR streams, the only reasonable choice to serve these contents to intolerant clients is simulation-based admission control.

In simulation-based admission control, the more precise the estimation is the higher the number of clients admitted and consequently the performance of the system. Note that admission control mechanisms, in conjunction with the scheduling policy, determine the final performance of the system. Accurate predictions can be obtained if the admission control mechanism incorporates a detailed hardware model of the server. This model, in disk-bounded systems, should involve precise estimations of disk seek times, rotation times, and transfer speeds. An admission control algorithm that employs this highly detailed information model is called *ideal deterministic* [4].

3. Memory reservation per storage device

Per-client memory management schemes suffer from a linear increase of the amount of memory required with the number of clients served. In this section we propose an innovative strategy that largely decreases the memory consumption in VoD servers, namely MRPSD, in which memory is allocated on a per-storage-device basis. Performance penalty is negligible when compared with VoD servers that rely on per-client buffering schemes.

In the following paragraphs, we first specify the conditions of applicability of MRPSD. After this specification, we describe the fundamentals of MRPSD, by means of the simplest case, MRPSD-2, in which only two buffers are required for memory management. We will continue by analysing the memory requirements of MRPSD-2, as compared to double buffering. Then, we will propose some variations in MRPSD-2 that perform effectively in the service of VBR streams. We finish this section discussing the implementation of MRPSD in VoD servers with multiple disks.

3.1. Conditions of applicability: a VoD system model

The following assumptions are stated to model the system used to evaluate MRPSD:

- A server-push model is considered.
- The VoD server scheduler must be cycle-based. MRPSD is designed to be combined with algorithms like SCAN, Round Robin, etc. The cycle time is constant, and the amount of information transmitted to a certain client may vary in each cycle when VBR streams are served.
- Information can be sent to the clients using bursty transfer mode. It is assumed that enough memory to store the information corresponding to a cycle is available at the client side.
- The network offers enough bandwidth, and can deliver traffic with real-time constraints. The time required to perform all disk operations in a cycle is larger than the time required to perform all operations in the network interface with the same amount of data (in other words, our system is *disk bounded*). Influence of buses and other VoD server elements is a second-order factor, as compared to storage device and network. We should

Table 1
MRPSD model parameters

T	Global cycle period
$n(c)$	Number of users served in cycle c
η	Set of clients for which all the contents of the server are requested
τ	Set of different cycle 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
$R_{\text{disk}}(i,p)$	Disk transfer rate for the data requested by stream i in period p
$L_{\text{net}}(i,p)$	Sum of the time needed for in starting network processing (interrupt service, first part of the protocol processing that corresponds to the first fragment of data to send) and 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

point out that our technique is also valid for network-bounded systems, by interchanging disk and network in the exposition that follows.
- Locally contiguous or contiguous information storage policies are employed. These strategies allow a higher effective transfer rate from the storage device.
- Clients are intolerant, and consequently worst-case and simulation-based admission control mechanisms are evaluated. Tolerant clients can also benefit from MRPSD, but in this study we consider the more restrictive case of intolerant clients.

The following parameters are used in the description and evaluation of the MRPSD model: Table 1.

3.2. MRPSD description

MRPSD is based on the reservation of a fixed amount of memory per storage device, as opposed to traditional memory management schemes that reserve memory on a per-client basis. We will begin describing a simple model MRPSD-2 that reserves only two buffers of identical size per storage device. For simplicity in the analysis, we assume that only one storage device is present in the VoD server.

In the MRPSD-2 model, the two identical size buffers work-co-operatively, repeating the following sequence of actions in each cycle (see Fig. 1):

1. Client service is ordered according to the scheduling algorithm selected. In some scheduling algorithms as SCAN or C-SCAN, the order in which clients are served may vary from cycle to cycle.
2. At the beginning of the cycle, data corresponding to the first client is read from disk and stored in the first buffer.
3. The following is repeated for all clients being served: one buffer receives data from the storage device, corresponding to client $i + 1$ (client $i + 1$ refers to the client that is served immediately after client i), when data previously stored in the other buffer, corresponding to client i, is delivered to the network. The buffers interchange their tasks whenever the fragment of information corresponding to client $i + 1$ is retrieved from the storage device. The beginning of each transference of information via network is activated by hardware interrupts issued by the storage device. An interrupt is raised every time a fragment of information is read from disk.

The server admission control mechanism should check the following two conditions every time a new client asks for service, to avoid any discontinuity in the service:

1. Data should never be lost when buffers are switched. As a sufficient condition to accomplish this requirement, network operation for client i must be completed before disk operation for client $i + 1$; in other words, the network buffer must be empty before the buffer switch occurs.
2. In each cycle, enough data should be read from disk and sent to the network in order to maintain the playback of all clients during the whole period. We should observe that, in a disk-bounded system, only disk operation has to be considered. Note that, by sending the data to the clients in the same cycle it is read, we reduce by one cycle the maximum client start-up latency, as compared to a equivalent SCAN based system with double buffering.

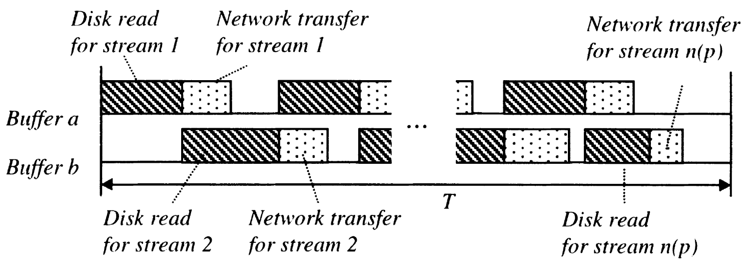

Disk read for stream 1
Network transfer for stream 1
Network transfer for stream n(p)
Buffer a
Buffer b
...
T
Disk read for stream 2
Network transfer for stream 2
Disk read for stream n(p)

Fig. 1. Buffer management in MRPSD-2.

The admission test can be mathematically expressed as:

$$\forall p \in \tau \ \ \forall i, 2 \leq i < n(p)$$

$$L(i+1,p) + \frac{b(i+1,p)}{R_{\text{disk}}(i+1,p)} > L_{\text{net}}(i,p) + \frac{b(i,p)}{R_{\text{net}}} \tag{1}$$

$$\forall p \in \tau$$

$$\sum_{i=1}^{n(p)} \left(L(i,p) + \frac{b(i,p)}{R_{\text{disk}}(i,p)} \right) + L_{\text{net}}(n(p),p) + \frac{b(n(p),p)}{R_{\text{net}}} \leq T \tag{2}$$

Before accepting a new client into the system, these conditions should hold for all the periods that the new stream is expected to last.

An analysis of the equations indicates that, if long cycles are used, the influence of disk latency in the admission of new clients is small. The key parameters to guarantee both conditions are the relationship between disk and network speeds and the difference in the size of the data fragments read for consecutive streams in the same cycle. Locally contiguous or contiguous information storage policies define a worst-case scenario for the application of MRPSD, as they reduce the value of the left side of Eq. (1). The effect of zoned disks in MRPSD behaviour is taken into account in both conditions.

All the parameters in Eqs. (1) and (2) (disk latency, amount of information to transfer in each cycle per client, disk transfer speed) must be evaluated accurately, in order to be able to obtain the maximum system performance. Therefore, the most appropriate admission control mechanism for MRPSD is ideal deterministic. Nevertheless, we can bound some parameters by worst-case analysis if a thorough simulation of the system is not feasible.

3.3. Memory requirements analysis

Memory consumption is the primary figure of merit when comparing MRPSD with traditional per-stream buffering strategies. In the evaluation of this magnitude, we have to differentiate between *static* and *dynamic* memory reservation. In *static* reservation, all the memory is allocated when the server starts its execution, whereas in *dynamic* reservation memory is allocated immediately before the service of each fragment of information. Between these two options we find an interesting midpoint, *dynamic reservation per client*, where memory is reserved in the instant of client acceptance.

Dynamic reservation allows allocating only the minimum amount of memory required by the system. As a drawback, the continuous allocation of contiguous memory fragments may result in an excessive overhead. In a system with cyclic scheduling and dynamic reservation, the allocation of buffers can be performed at the beginning of the cycle, since the size of all fragments that are to be processed in the cycle is known at this stage. This also holds for servers that implement MRPSD.

The amount of memory needed in MRPSD-2 with static reservation is:

$$M = 2 \operatorname*{Max}_{\forall p \in \tau, \forall i \in \eta} (b(i,p)) \tag{3}$$

This expression takes into account the worst-case situation: two clients demanding at the same time the content that requires the maximum amount of data for maintaining playback in one cycle.

If dynamic reservation per client is used, the amount of memory can be reduced, assuming that a simulation-based admission test is employed and information about the requirements in the following cycles is available. The next expression shows the amount of memory required in this case when a new client is accepted in cycle p:

$$M(p) = \operatorname*{Max}_{\forall c \geq p, 1 \leq i < n(c)} (b(i,c) + b(i+1,c)) \tag{4}$$

If the information needed to compute this expression is not available, worst-case bounds can be used instead.

For dynamic reservation, we obtain the following expression:

$$M(p) = \operatorname*{Max}_{1 \leq i < n(p)} (b(i,p) + b(i+1,p)) \tag{5}$$

In the last two cases, we can estimate the maximum amount of memory required in the system as

$$M = \operatorname*{Max}_{\forall p \in \tau} M(p) \tag{6}$$

We can compare these expressions with their analogous for double buffering. For static reservation, the memory required is:

$$M = \operatorname*{Max}_{\forall c \in \tau} n(c) \operatorname*{Max}_{\forall p \in \tau. \forall i \in \eta} (b(i,p) + b(i,p+1)) \tag{7}$$

If dynamic reservation per client is used, we obtain:

$$M(p) = \operatorname*{Max}_{\forall c \geq p} \left(\sum_{i=1}^{n(c)} b(i,c) + \sum_{i=1}^{n(c+1)} b(i,c+1)\right) \tag{8}$$

And finally, for dynamic reservation, the expression is:

$$M(P) = \operatorname{Max}\left(\sum_{i=1}^{n(p)} b(i,p) + \sum_{i=1}^{n(p+1)} b(i,p+1)\right) \tag{9}$$

We can conclude that memory requirement for MRPSD-2 is significantly decreased compared to double buffering. In general, the memory saving factor is roughly proportional to the number of clients served, although its value depends largely on the particular pattern of contents served.

3.4. VBR streams service using MRPSD

If network and disk transfer speeds are similar, the MRPSD-2 admission test may fail if client i demands a large fragment of data and client $i+1$ a small one, because disk operation for client $i+1$ will begin before finishing

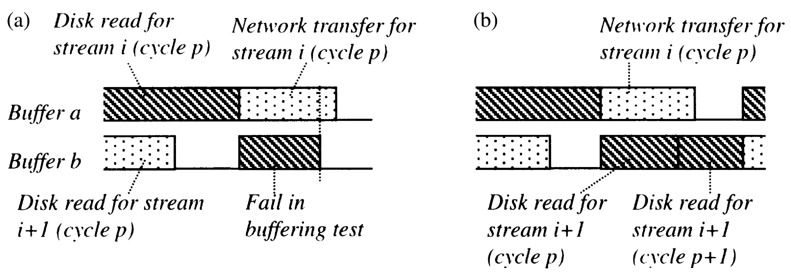

(a)
Disk read for stream i (cycle p)
Network transfer for stream i (cycle p)
Buffer a
Buffer b
Disk read for stream i+1 (cycle p)
Fail in buffering test
(b)
Network transfer for stream i (cycle p)
Disk read for stream i+1 (cycle p)
Disk read for stream i+1 (cycle p+1)

Fig. 2. (a) Fail in MRPSD-2 Buffering Test; (b) MRPSD-2 with multiple cycle reading.

network operation for client *i*, violating condition 1. This is very likely to happen when VBR streams are served. We propose two solutions to overcome this problem: *Multiple Cycle Reading,* and the use of a larger number of buffers per storage device, MRPSD-b. These two solutions may also be used simultaneously.

3.4.1. Multiple cycle reading

Multiple Cycle Reading (MCR) is an adaptation of MRPSD that allows its application to VBR streams in VoD servers. If the test fails because the network transfer for stream *i* takes longer than disk retrieval for stream $i + 1$ (Fig. 2a), we increase disk reading time by adding all the data required for stream $i + 1$ corresponding to the next cycle (Fig. 2b). Client $i + 1$ receives data for current and next cycle (in the next cycle, the service of stream $i + 1$ will be skipped). If the test still fails, more subsequent cycles are grouped in a single read for client $i + 1$. If the scheduling test corresponding to Eq. (2) fails on the final modified schedule, the client tested for admission is rejected.

An important observation is that data corresponding to a cycle for a given client should not be split, in order to (a) use the pre-computed information that characterises the requirements of the amount of data needed for a client in each cycle, and (b) keep the number of disk head movements as small as possible.

The use of MCR may increase the amount of memory required in the server if the accumulation of reads in a cycle establishes a new maximum in the amount of data that is to be sent to a client. However, since MCR is only used when the initial amount of data to be read from disk is small, this increase is not expected to be significant.

3.4.2. MRPSD-b

Another possibility for adapting MRPSD-2 to the service of VBR streams is the utilization of b buffers per storage device (MRPSD-b) The b buffers can be assigned in a circular fashion to the disk read operations issued by the scheduler. As an example, Fig. 3 shows buffer management in MRPSD-3.

Network transmission for client $i + 1$ will be activated by the end of disk operation for the same client, if the operation for network transmission corresponding to client *i* has finished; otherwise, it will be issued at the end of the network transfer for client *i*.

The new condition to verify in the admission test is the following: in every cycle, the information for client *i* must be sent to the network before the information for client $i + b - 1$ is read from disk. This condition is less restrictive than the equivalent for MRPSD-2.

The finishing time for the disk read operation corresponding to client *i* in cycle $p, t_{\text{disk}}(i,p)$, can be computed as:

$$t_{\text{disk}}(i,p) = pT + \sum_{j=1}^{i} L(j,p) + \frac{b(j,p)}{R_{\text{disk}}(j,p)} \tag{10}$$

The finishing time for the network transfer operation corresponding to client *i* in cycle $p, t_{\text{net}}(i,p)$, can be obtained

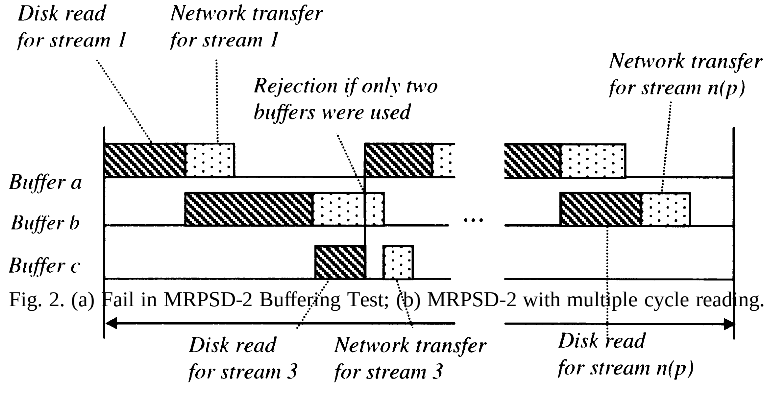

Disk read for stream 1
Network transfer for stream 1
Rejection if only two buffers were used
Network transfer for stream n(p)
Buffer a
Buffer b
Buffer c
...
Disk read for stream 3
Network transfer for stream 3
Disk read for stream n(p)

Fig. 3. Buffer management in MRPSD-3.

Table 2
Seagate ST31200W parameters

Parameter	Value
Number of cylinders	2700
Sector size (bytes)	512
Number of surfaces	9
Rotation speed (rpm)	5411
Seek time for changing to another track of the same cylinder (ms)	1.2
Seek time for changing to a track of an adjacent cylinder (ms)	1.7
Maximum seek time (ms)	19.4
Number of zones	23

recursively by means of the following expression:

$$i = 1 \qquad t_{\mathrm{net}}(i,p) = t_{\mathrm{disk}}(1,p) + \frac{b(1,p)}{R_{\mathrm{net}}} \tag{11}$$

$i > 1$

$$t_{\mathrm{net}}(i,p) = \mathrm{Max}(t_{\mathrm{disk}}(i,p), t_{\mathrm{net}}(i-1,p)) + \frac{b(i-1,p)}{R_{\mathrm{net}}}$$

3.5. MRPSD in multi-disk servers

MRPSD can be incorporated to VoD systems containing more than one storage device. In a data-striped disk array, we can think of the disk array as a larger logical storage device with higher bandwidth and capacity. In this case, MRPSD will be applied in the same way as if we were considering a single disk, supposing that the network interface is capable of handling all the traffic. If more network interfaces are required for the system, we can use MRPSD-b. For k network I/O adapters, allocating $k + 1$buffers allows the parallel network delivery to k clients and the disk reading for one.

If data interleaving is used, it may occur that the data corresponding to a client for a cycle is distributed among several disks. The different parts of the fragment can be read in parallel and disordered. In this case, it is necessary a proxy module to reorder the different parts using the information about the placement of the data in the different disks [16]. This reordering of contents can be done at the client side. In this case, each disk constitutes an independent storage device, and memory reservation must be performed on a per-disk basis (for example, in a MRPSD-2 strategy we would reserve 2 buffers per disk in the array). The admission test must be performed individually on every storage device.

4. Experimental results

In Section 4.1 we describe the simulation model developed in order to study the behaviour of MRPSD quantitatively Section 4.2 is devoted to the experiments conducted on our simulator and analysis of results.

4.1. Simulator description

We have implemented a disk array simulator built over sim^{++} [9], a discrete event simulation package. Disk information is based on a Seagate ST3 1200W zoned disk [13], including disk head movement, rotation and variable transfer speed depending on the zone considered. Some parameters and their values are listed in Table 2. Disk cache and bus transference overhead have not been simulated. Disks can be grouped into interleaved or spindle-synchronised disk arrays.

In our model, the network interface initial transfer latency equals one millisecond, that accounts for the worst-case service time of the disk interrupt, the start up of the network protocol processing and the delay in access to the network. A constant transfer rate of 60 Mbits/s is assumed. This rate embodies steady state protocol processing and real network transfer speed. A network interface is assigned to each storage device, with a bandwidth proportional to the number of disks that it holds.

The multimedia information contents are VBR MPEG-1 encoded traces from Bellcore [10] (`Starwars`) and Wuerzburg University [24] (`Asterix`, `MTV`, `Simpsons` and `Video`). `Starwars` is a monochrome video about two hours long. `Asterix`, `MTV` and `Simpsons` have a duration of 28 min, while `Video` lasts for 4 min. The streams are placed contiguously in the storage devices.

The scheduler implements C-SCAN. The supported memory management strategies are double buffering and MRPSD-b. Admission control can be implemented using simulation-based or worst-case models. In the experiments presented below, an ideal deterministic admission control test has been performed. Disk and network bandwidth availability is checked, and when MRPSD is used, conditions 1 and 2.

Client arrivals have been modelled using a time-homogenous Poisson process, where minimum inter-arrival values have been established. The parameters are chosen to obtain system overload conditions. Whenever a client request is rejected, the same request is issued in subsequent cycles until it is finally accepted. The selection of movies is driven by a Zipftian distribution with a skew of 0.271, value observed in video rental requests [28]. Movies closer to the outer edge of the disk (with higher transfer rates) are considered to be more popular.

There are two primary parameters for performance evaluation: *effective transfer rate* and *memory saving factor*. The *effective transfer rate* is the number of bytes transferred per time unit. Note that disk utilization is skewed for zoned disks and the number of clients is not an appropriate parameter when heterogeneous VBR streams are present. The *memory saving factor* is obtained dividing the amount of memory used in double buffering into the equivalent value

Table 3
Comparison of double buffering and MRPSD-b MCR

Buffering policy	Memory usage (MBytes)	Memory saving factor	Effective transfer rate (MBytes/s)
Double buffering	171	–	3.48
MRPSD-2	3.73	45.8	0.14
MRPSD-2 MCR	3.73	45.8	3.42
MRPSD-3 MCR	5.60	30.5	3.45
MRPSD-4 MCR	7.47	22.9	3.45

for MRPSD. Memory is reserved statically for MRPSD (the most demanding case). In double buffering, the memory required is computed by multiplying the maximum requirements of each content by the number of clients that are requiring that content. This measure is a pessimistic estimation for dynamic reservation per client, very similar to static reservation, except that in this case not all the clients are supposed to choose the most demanding content. For comparison purposes, an average estimation of the memory required by double buffering with dynamic reservation can be obtained multiplying the effective transfer rate by the cycle time, and by two (the number of buffers per client).

4.2. Experiments and analysis of results

4.2.1. Single disk servers

Under the conditions presented above, the results obtained are shown in Table 3.

MRPSD-2 without multiple cycle reading performs poorly, due to the rejection of an excessive number of clients. However, when MCR is added, a very large memory saving factor is obtained, without incurring insignificant performance penalty. As we can see, increasing the number of buffers beyond three does not result in further performance improvement.

The percentage of grouped sectors due to Multiple Cycle Reading is 5.08% for MRPSD-2 MCR, 0.002% for MRPSD-3 MCR, and 0 for MRPSD-4 MCR. These numbers show that MCR is not needed if the number of buffers is sufficiently high.

The mean amount of memory consumed by double buffering with dynamic memory reservation is 69.3 MBytes. Even in this unfair case (comparing static reservation MRPSD with dynamic reservation double buffering), a memory saving factor of 18.6 confirms the benefits of MRPSD.

In the next three sections we compare MRPSD with double buffering, studying the influence of several parameters: cycle length, network transfer speed and heterogeneity of contents.

4.2.1.1. Cycle length influence. Performance and latency in VoD systems depend on the cycle length. When short cycles are used, the relative importance of disk seek and rotation times increases, degrading the effective transfer rate. Conversely, long periods imply high client latency.

Effective transfer rates are very similar for double buffering and MRPSD, regardless of the cycle length value (Fig. 4). On the other hand, memory consumption is affected by cycle length in all cases (Fig. 5). In double buffering systems with large cycle times, very large amounts of memory are consumed, because there are more clients being served simultaneously and more memory is required for each client. For MRPSD, only the second reason is relevant.

4.2.1.2. Network speed influence. Network transfer speed is a key factor in the VoD server admission test when MRPSD is used. If the difference between network and disk speeds is sufficiently high, the admission test holds and no performance penalty is incurred. But if the

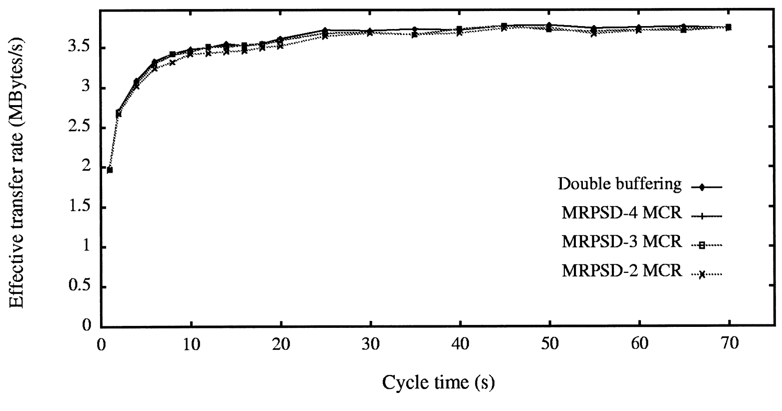


Fig. 4. Effective transfer rate vs. cycle time.

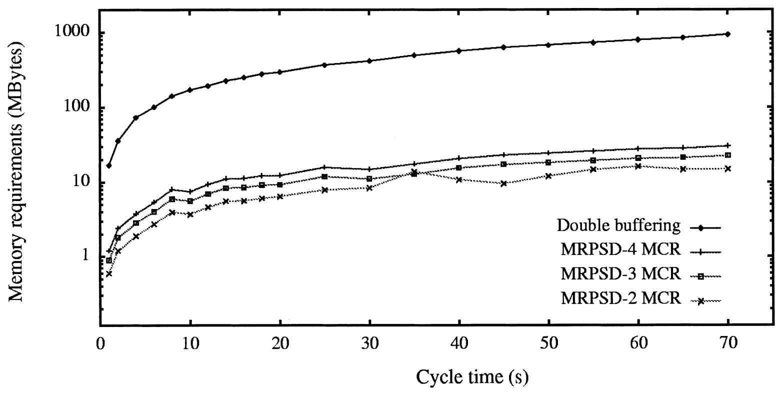


Fig. 5. Memory required vs. cycle time.

difference is small, buffer conflicts are more frequent and the number of cycles grouped is larger. MCR cannot compensate for the degradation in performance when network and disk speeds are very similar (see Fig. 6). The higher memory requirements for low network speeds in MRPSD with MCR can be explained by the increase in the number of cycles that have to be grouped to satisfy condition 1 (Fig. 7). On the contrary, double buffering is not affected by a decrease in network transfer speed until the system is no longer disk-bounded, and clients start to suffer rejections due to network bandwidth shortage.

The difference required between maximum disk and network speeds for acceptable performance is fairly small, and therefore the use of MRPSD does not add stringent network requirements to existing systems.

4.2.2. Heterogeneity of contents

MRPSD performance is heavily affected by the pattern of requests issued by the clients. The variation in successive requests has been identified as an important parameter in MRPSD success. In this experiment we want to check if MRPSD is suitable for the service of contents with significant pattern variations.

For this experiment, we define six different sets of contents:

A: `starwars`, `asterix`, `mtv`, `simpsons` and `video`;
B: `asterix`, `mtv`, `simpsons` and `video`;
C: `starwars`;
D: Constant Bit Rate (CBR) video streams, with a playback rate of 0.5 Mbits/s, lasting 1 h;
E: Several CBR streams, representing audio contents with a playback rate of 64 KBits/s, and different lengths, ranging from 2 min to 5 h; and
F: An heterogeneous set comprised by `starwars`, `mtv`, CBR audio streams and two CBR 800 bits/min streams, low playback rate streams (that could represent, for example, text for subtitles).

In the following table we compare the performance and memory consumption of double buffering and MRPSD-2 MCR when these sets of contents are served.

The variations observed in performance can be explained by two factors:

- *The variability of contents*. If the difference in the amount of data that can be requested by the clients in a cycle is high, the load is not distributed uniformly, and heavily loaded cycles are responsible for client rejections.
- *The efficiency in disk accesses*. Streams with low playback rates require small disk fragments in each cycle, and disk access becomes inefficient.62

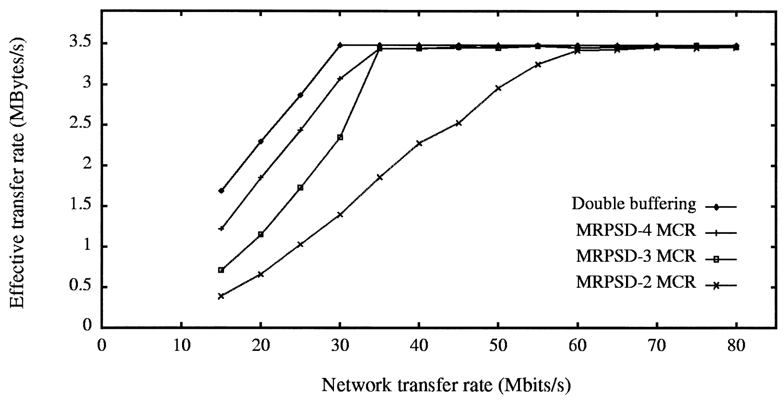


Fig. 6. Effective transfer rate vs. network transfer rate.

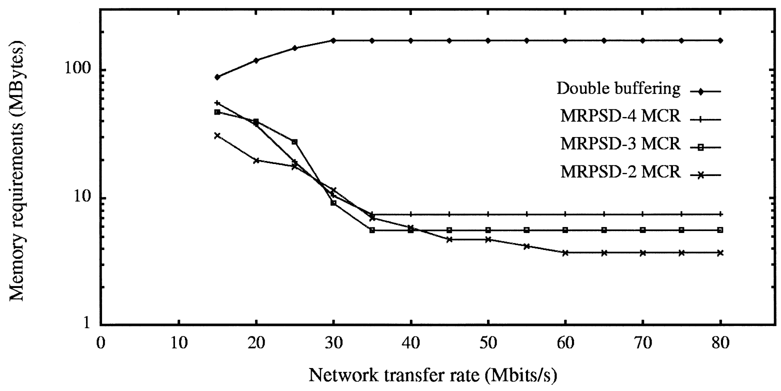


Fig. 7. Memory requirements vs. network transfer rate

Table 4 shows that performance in double buffering and MRPSD-2 MCR is very similar for all cases. The memory saving factor is very large in all circumstances, including the service of very heterogeneous contents (set F). Its particular value is related to the number of clients served in the system (see set E). Pure CBR contents (sets D and E) need less memory than VBR sets because in this case the estimation made for static reservation is closer to the real requirements.

4.2.3. Disk array servers

Interleaved disk arrays with large striping units (compared to the amount of data corresponding to a content that is assigned to a single disk) present low performance in the service of VBR streams. The number of disk head movements is small (one, or at most, two disks are involved in serving one request), but the load is not uniformly distributed among all disks, because the difference in the amount of data stored in different disks can be considerable. We will concentrate in configurations that distribute more uniformly the data among all disks.

The following figures compare spindle-synchronised disk arrays and interleaved disk arrays with different striping units (Figs. 8 and 9).

In spindle-synchronised disk arrays, data is distributed uniformly among all disks. However, the number of different requests served by each disk increases with the size of the array, leading to a sub-linear trend in overall performance due to the excessive number of disk head movements incurred. For interleaved disk arrays with small striping units, similar results are obtained. Comparing MRPSD with double buffering in VoD servers that incorporate several storage devices, we observe that, again, performance is very close.

The following figures detail the amount of memory required on each case (Figs. 10 and 11).

Double buffering memory usage grows steadily with the number of disks, ranging from hundreds of MBytes to 2 GBytes. The trend is similar for interleaved distributions with MRPSD, although the total magnitude is much smaller. We can achieve memory saving factors greater than 30 for MRPSD-2, and 20 for MRPSD-3. If a single network interface (with higher capacity) is used with a spindle-synchronised disk array, only one buffer is required, and memory usage remains almost unchanged.

Table 4
Comparison of double buffering and MRPSD-2 MCR for different sets of contents

Set of contents	Memory management	Memory consumption (MBytes)	Memory saving factor	Effective transfer rate (MBytes/s)
A	Double buffering	170		3.48
	MRPSD-2 MCR	3.74	45	3.42
B	Double buffering	176		3.58
	MRPSD-2 MCR	4.19	42	3.39
C	Double buffering	148		2.80
	MRPSD-2 MCR	2.39	62	2.77
D	Double buffering	70.4		3.42
	MRPSD-2 MCR	1.19	59	3.39
E	Double buffering	58.0		2.95
	MRPSD-2 MCR	0.15	387	2.93
F	Double buffering	144		3.10
	MRPSD-2 MCR	2.85	50	3.04

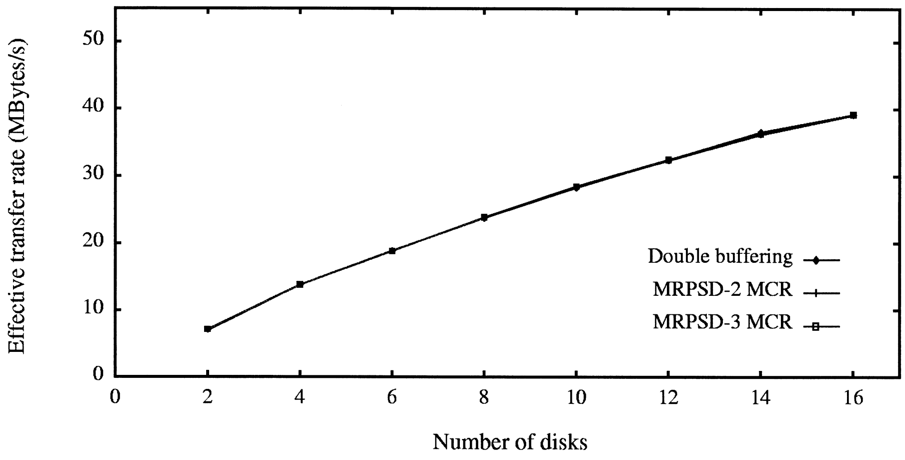


Fig.8. Effective transferratevs. number of disks for data-striped disk arrays.

5. Related work

Some variations to cyclic schedulers have been proposed in order to overcome excessive memory consumption of per-stream buffering strategies, being the most important GSS [32]. However, the total amount of memory needed still depends on the number of users served, and it is always larger than in MRPSD–MCR. To establish a comparison with the results provided in our article, note that the amount of memory demanded never falls below 50% of the memory required by double buffering.

The *subgrouping and subcycling* scheme [29] focuses on buffer reutilization for continuous delivery mode in 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 and does not consider zoned disks. Furthermore, subcycling leads to performance degradation. Mourad [19] develops a technique similar to subgrouping and subcycling, based on GSS, but it also restricted to the service of CBR streams.

Ng [21] combines buffer reuse with single buffering, to free up to 50% of the memory, but the scheme is difficult to implement, requiring non-contiguous buffer management. Chang and Garcia-Molina [5] proposes to space I/O operations in order to approach to the 50% saving limit. Their proposal is also based on per-stream buffering, and considers only CBR streams.

Reutilization of buffers is a known strategy in operating system's practice. However, its application to the VoD environment, in which timely delivery of huge amounts of data is mandatory, has not been properly addressed.

Just-in-time scheduling [2] includes a per-device buffering technique for continuous mode delivery in large arrays composed of independent disks. All disks are involved in the service of a client. Clients are serviced in a SCAN basis, keeping the same order due to stringent data placement strategy that forces all contents' to be placed in the same zone on all the disks. There is a gap between the start of their cycles that 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 contents are equally demanding CBR streams. Disk and network behaviour is not considered to evaluate the

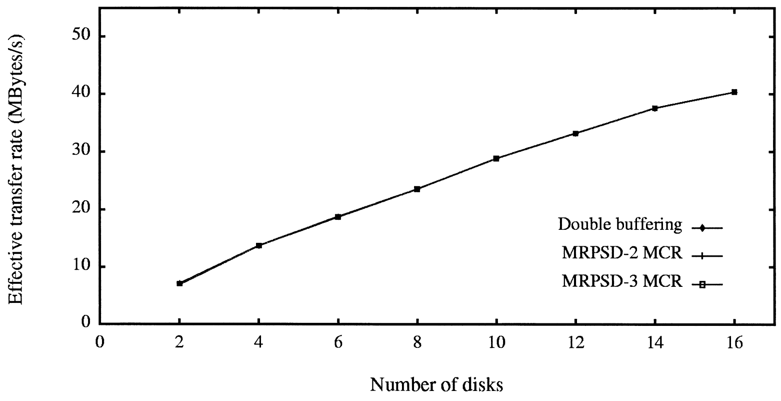
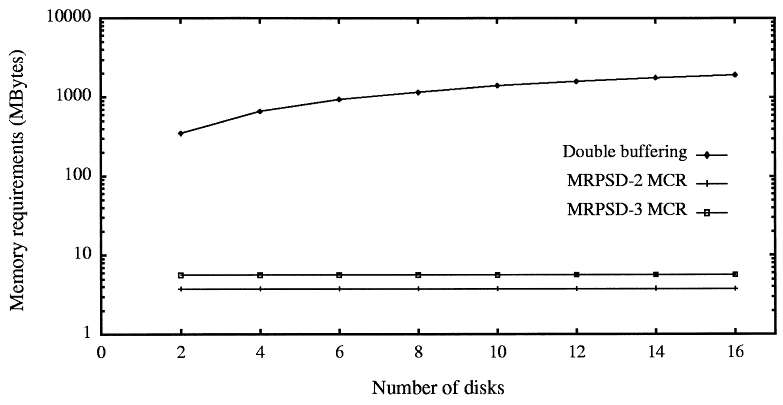


Fig. 9. Effective transfer rate vs. number of disks for interleaved disk arrays with a stripe unit of 1 sector.

Fig. 10. Memory requirements vs. number of disks for data-striped disk arrays.



feasibility of the solution. In addition, the mechanism is only suited for large disk array servers.

6. Conclusions

In this article we have presented, evaluated and validated MRPSD, an innovative buffering technique to be used in combination with cycle-based VoD server schedulers. MRPSD leads to a major improvement in VoD server memory requirements as compared to existing per-stream buffering allocation schemes. Although MRPSD slightly increases network bandwidth demands, enough resources are generally available in off-the-shelf systems.

To implement MRPSD, the possibility of sending data to the clients in bursty transfers is required. However, there are no further noticeable requirements; in particular, it can be combined with servers that implement different admission control strategies. To ensure that the proposed memory management technique proceeds without any information loss in the service of intolerant clients, it is necessary to perform some checks before new admissions. The conditions for MRPSD may result in the rejection of some clients and therefore be responsible for a performance reduction. We have shown with experimental data that this reduction can be very small if the parameters that determine MRPSD's behaviour are properly selected.

We have considered in detail the heterogeneous nature of the movies found in real-world servers. The utilization of ideal deterministic admission control mechanisms has a major relevance in the deterministic service of VBR contents to intolerant clients. The availability of ideal deterministic admission control mechanisms is not mandatory for the application of MRPSD in the service of intolerant clients; worst-case admission tests may be employed at a cost of some performance degradation. Tolerant clients can also benefit from MRPSD, by using statistical admission control. We have developed two proposals that solve the performance problems arising when MRPSD is used in the service of VBR streams: multiple cycle reading, and the increase of the number of zones allocated per storage device. Multi-disk servers have also been considered in the development of MRPSD. Our technique can be easily adapted to servers implementing different data distribution strategies.

Besides the theoretical study of MRPSD, some experiments have been performed with a simulator that models the service of heterogeneous multimedia contents, with cyclic scheduling, several admission control strategies, and a variable number of disks. The results obtained show that

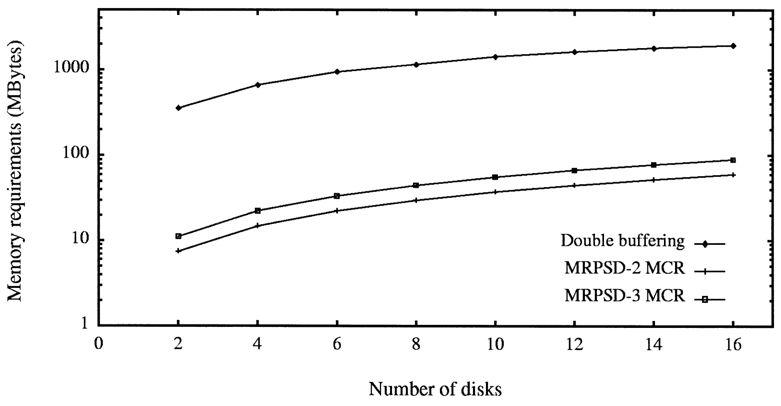


Fig. 11. Memory requirements vs. number of disks for interleaved disk arrays with a stripe unit of 1 sector.

MRPSD allows a reduction in the memory required between one and two orders of magnitude, with a similar performance to per-client memory management strategies.

Bursty transfer represents an obstacle for the application of the technique described in wide-area environments, where information is buffered from one network segment to the next. The provision of network delivery guarantees to the transmission of bursty data can be unfeasible with current network scheduling practices. MRPSD is more suited to local area network environments, like universities requiring the dissemination of lectures, companies distributing news, training courses, etc. In this scenario, data does not need to be buffered when traversing from server to client. Moreover, VoD clients are typically general purpose computing devices that include enough memory to maintain playback with the bursty operation considered. In such conditions, MRPSD offers a cheap and powerful technique to be integrated in VoD servers.

Acknowledgements

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

References

[1] C. Bernhardt, E. Biersack, The server array: a scalable video server architecture, in: O. Spaniol, W. Effelsberg, A. Danthine, D. Ferrari (Eds.), High-Speed Networking for Multimedia Applications, Kluwer, Dordrecht, 1996 Chap. 5.

[2] S. Berson, R.R. Muntz, Just-in-time scheduling for video-on-demand storage servers, Technical Report, UCLA Computer Science Department, April, 1995.

[3] M.M. Buddhikot, G.M. Parulkar, Efficient data layout, scheduling and playout control in MARS, ACM Multimedia Systems 5 (5) (1997) 199–212.

[4] E. Chang, A. Zakhor, Cost analysis for VBR video servers, IEEE Multimedia 3 (4) (1996) 56–71.

[5] E. Chang, H. Garcia-Molina, Effective memory use in a media server, Proceedings of Very Large Data Base Conference, Athens, Greece, 1997.

[6] E. Chang, H. Garcia-Molina, Bubble up: low latency fast-scan for media servers, Proceedings of ACM Multimedia Conference, Seattle, USA, November, 1997.

[7] T. Chiueh, C. Venkatramani, M. Vernick, Design and implementation of the stony brook video server, Software Practice and Experience 27 (2) (1997) 139–154.

[9] R.M. Cubert, P. Fishwick, sim^{++}, Version 1.0, Department of Computer and Information Science and Engineering, University of Florida, July, 1995.

[10] M. Garrett, W. Willinger, Analysis, modelling and generation of self-similar VBR video traffic, Proceedings of ACM SIGCOMM, August, 1994, pp. 269–280.

[11] J. Gemmell, J. Han, S. Christodoulakis, Delay-sensitive multimedia on disks, IEEE Multimedia 1 (3) (1994) 56–67.

[12] J. Gemmell, H.M. Vin, D.D. Kandlur, P.V. Rangan, Multimedia storage servers: a tutorial and survey, IEEE Computer 28 (5) (1995) 40–49.

[13] 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.

[14] K. Keeton, R.H. Katz, Evaluating video layout strategies for a high-performance storage server, ACM Multimedia Systems 3 (2) (1995) 43–52.

[15] D.R. Kenchammana-Hosekote, J. Srivastava, I/O scheduling for digital continuous media, ACM Multimedia Systems 5 (4) (1997) 213–237.

[16] J.Y.B. Lee, Parallel video servers: a tutorial, IEEE Multimedia 5 (2) (1998) 20–28.

[17] J.C.L. Liu, J. Hsieh, D.H.C. Du, M. Lin, Performance of a Storage System for Supporting Different Video Types and Qualities, IEEE Journal on Selected Areas in Communications 14 (7) (1996) 1314–1431.

[18] D. Makaroff, N. Hutchinson, G. Neufeld, An evaluation of VBR disk admission algorithms for continuous media file servers, Proceedings of ACM Multimedia Conference, Seattle, USA, November 1997.

[19] A.N. Mourad, Issues in the design of a storage server for video-on-demand, ACM Multimedia Systems 4 (2) (1996) 70–86.

[21] R.T. Ng, J. Yang, An analysis of buffer sharing and prefetching techniques for multimedia systems, ACM Multimedia Systems 4 (2) (1996) 55–69.

[22] B. Özden, R. Rastogi, A. Silberschatz, On the design of a low-cost video-on-demand storage system, ACM Multimedia Systems 4 (1) (1996) 40–54.

[23] S. Rao, H.M. Vin, A. Tarafdar, Comparative evaluation of server-push and client-pull architectures for multimedia servers, Proceedings of International Workshop on Network and Operating System Support for Digital and Audio Video (NOSSDAV), Zushi, Japan, April, 1996, pp. 57–60.

[24] 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.

[25] C. Ruemmler, J. Wilkes, An introduction to disk drive modelling, IEEE Computer 27 (3) (1994) 47–57.

[28] W. Tetzlaff, R. Flynn, Block allocation in video servers for availability and throughput, Proceedings of SPIE Multimedia Computing and Networking (MMCN), San José, USA. January 1996.

[29] F. Tobagi, J. Pang, R. Baird, M. Gang, Streaming RAID—A disk array management system for video files, Proceedings of ACM Multimedia Conference, Anaheim, USA, August, 1993, pp. 393–400.

[30] H.M. Vin, P.V. Rangan, Designing a multi-user HDTV storage server, IEEE Journal on Selected Areas in Communications 11 (1) (1993) 153–164.

[31] H.M. Vin, A. Goyal, P. Goyal, Algorithms for designing multimedia servers, Computer Communications 18 (3) (1995) 192–203.

[32] P. Yu, M.S. Chen, D.D. Kandlur, Grouped sweeping scheduling for DASD-based multimedia storage management, Multimedia Systems Journal 1 (1993) 99–109.