

Performance Analysis and Optimization of a Distributed Video on Demand Service

Daniela Alvim Seabra dos Santos Alex Borges Vieira
Berthier Ribeiro-Neto Sérgio Vale Aguiar Campos
Department of Computer Science
Federal University of Minas Gerais
Av. Antonio Carlos 6627 - 31270-010
+55 31 3499-5860 - Belo Horizonte MG - Brazil
{daniela, borges, berthier, scampos}@dcc.ufmg.br

Abstract

Video on Demand (VoD) services are very appealing these days. In this work, we discuss four distinct alternatives for the architecture of a VoD server and compare their performances under different conditions. We later used the best configuration found in our analysis to evaluate, using simulations, the performance of a distributed VoD service in an ATM network covering an area the size of a large city neighborhood. We also introduced optimizations to the system, such as anticipated delivery and retrieval of video blocks from neighbor clients. Our results indicate that a server-driven operation mode (i.e., based on cycles) is not the most appropriate choice for a variety of workloads, even when the layout is striped (a result that challenges the conventional wisdom in the field). Also, our optimization strategies increased significantly the number of clients served in the system, which in our study case represented a savings of approximately 33% in the hardware required for the deployment of the service.

1. Introduction

A Video on Demand (VoD) service allows its clients to watch videos through a network, on demand. It is a very appealing service because it provides much more flexibility than cable TV and video stores. Additionally, there is a great number of areas where VoD systems can be successfully employed, such as entertainment, distance learning, and video conferencing. Moreover, thanks to advances in network technology, the deployment of such systems is feasible.

An evaluation of a distributed VoD service is appropriate because of the complexity of its implementation in a real

environment. Analytical validation and the investigation of the best design choices are imperative if the service is to be migrated from an academic to a commercial application. Using simulations, we have been able to investigate the behavior of the system under various scenarios. Furthermore, it is easier to make changes (hardware and software architecture) in a simulation model than in a real system in full operation.

The old and traditional notion that VoD users watch a video, from start to finish, without interruption can no longer be considered the only case. While the majority of users will watch videos in a deterministic, sequential way, there are many users who will frequently make use of VCR-like features, will watch only certain parts of a movie, will watch videos of different length, or will watch small portions of several clips.

In order to determine how to provide the best service under these conditions, we discuss four alternatives for the architecture of a VoD server, compare their performances with a non-sequential access pattern, and suggest what would be the best alternative for the next generation of VoD servers, which will have to handle heterogeneous applications, different access patterns, and user interactivity. The four alternatives arise from a distinction we introduce between the layout and the server operation mode. Conventionally, VoD servers are based on striping techniques with two fundamental implications: (i) the blocks of a same video are stored in the disks of the server in a round-robin fashion and (ii) the server operates in cycles, pushing one video block to each client in the system during a cycle. We say that the layout is striped and the operation mode is server-driven [1, 13, 18]. However, other alternatives are possible. For instance, a random layout, where the video blocks are randomly distributed across the disks of the server, can be used, as proposed in [10, 16, 17]. Also, the operation of the

server can be driven by requests from the clients, which allows eliminating cycles at the server. In this case, the server, upon receiving a block request from a client, directs this request to the disk which holds the copy of the requested block. In a disk queue, the block requests will be served according to some disk scheduling algorithm. We call this operation mode client-driven [16, 17]. Here, we investigate the behavior of a VoD server under four distinct configurations: a striped layout combined with a server operation in cycles (striping), a striped layout combined with a server operation without cycles (striping-no-cycle), a random layout combined with a server operation without cycles (random), and a random layout combined with a server operation in cycles (random-with-cycles). The striping configuration corresponds to what is conventionally referred to as *striping layout*. The random configuration corresponds to what is conventionally referred to as *random layout*. Such a study has not been done before and yields interesting insights.

We have later used the best VoD server configuration found in our analysis to evaluate the performance of a distributed VoD service in an ATM network covering an area the size of a large city neighborhood. We have also investigated the benefits of introducing some optimizations to the system, such as an anticipated delivery scheme and the retrieval of video blocks from neighbor clients. In our anticipated delivery scheme, whenever the system has enough bandwidth available, it would be able to anticipate the delivery of video blocks being served at the moment and, consequently, free bandwidth for future use. To accomplish this, we assume that clients have disks inside their set-top boxes so that an entire film can be stored before being seen. As we have clients storing videos in our system, we can allow, in certain situations, a newly arrived client to obtain blocks from a neighbor, instead of requesting them from a remote server. Through this approach, we can improve the system performance and attend a higher number of clients, without having to add any extra hardware.

This paper is organized as follows. Section 2 presents the related work. Section 3 describes a video on demand service. Section 4 describes how we have modeled the distributed VoD service. In section 5, we present the optimization schemes we have proposed. Section 6 presents our simulation results. Finally, in section 7, we summarize our work and present our conclusions and future work.

2. Related work

In [15], it is described the design, implementation, and performance analysis of the RIO multimedia storage server. They use a client-driven operation mode and a random layout for video block distribution across the disks of the system. In their work, they have not considered the impact

of using real video streams. In addition, they have not addressed, in details, issues involving extending RIO to a distributed environment, have not compared the random and the striping layout with regard to the mean client latency, and have only considered the striping layout with a server-driven operation mode.

In [8], the authors present a logical model for a distributed media-on-demand (MoD) system. It was presented policies for assigning requests to the nodes so as to minimize the load among the interface nodes and the traffic on the network. Although our work presents similarities to theirs, there are differences in the emphasis given in each analysis. In our work, we have concentrated on improving the system performance by alleviating the load on the server disks, which we consider to be the main bottleneck of our system. We did so through our optimization schemes. This type of analysis was not addressed in their work. Second, we have investigated system performance under different considerations that the authors have not addressed, such as different layout schemes, videos of different length, and clients making use of VCR-like features.

In [19], the authors have examined several performance issues in the design of clustered video servers which support thousands of concurrent streams. Our work draws upon their work but with several key differences. First, the authors have only taken into account a server-driven operation mode (servers operating in cycles or in a push mode). Moreover, they have not considered how videos of different length and clients making use of VCR-like features influence system performance. Finally, in our work we propose optimization schemes in order to increase the number of clients the system can serve.

3. Video on demand

A VoD system is comprised of the following modules: one or more video servers, a storage system which is usually represented by an array of disks, clients, and a network serving as the communication channel. The clients can be represented by a PC, a TV connected to a set-top box, or even a laptop connected to a wireless link. In general, videos are divided into data blocks and distributed across the disks of the system according to some layout policy.

The server is the central part of a VoD system. It contains a CPU, an amount of memory for buffering, and disks which are used to store the encoded videos. It is responsible for receiving and processing client requests. Each client requests one video block from the server at a time. The server will read the blocks from one of its disks and send them to the client through the network. Upon receiving the video block, the client stores it on a buffer and decodes it on the user display device. Before finalizing the block decoding, the client must receive the next block. The time the client

has to wait from the moment it sends a request for a certain video to the moment the video playback actually begins is called **latency**. During the transmission of two adjacent blocks of a certain client, the server is able to attend other clients. This is due to the fact that the rate at which the server reads data from its disks is higher than the rate at which the client decodes and displays the video block.

The **layout** is the mapping of the video blocks onto the disks of a VoD server. There are two main types of layout schemes frequently addressed in the literature: striping and random layout.

The striping layout is discussed in [1, 5, 13, 18]. Through this scheme, the video blocks (called stripes) are distributed across the disks of a server in a round-robin fashion. For instance, if the i -th block of video X is stored on the j -th disk of the server, then the i th + 1 block of this same video is stored on the $((j$ -th + 1) mod (number of disks)) disk of the server. Striping is considered to be a good layout for sequential access patterns such as video and audio because it distributes the workload evenly across the disks of the system. On the other hand, it does not present a good performance when clients use VCR-like features and usually leads to high latencies: when a new client is admitted, it has to wait until the available bandwidth reaches the disk which holds the copy of the first block of the requested video.

The random layout is addressed in [10, 15, 17]. Through this approach, the video blocks are randomly distributed across the disks of a server. This layout has the advantage of being simple to implement and suitable for various access patterns, such as VoD with VCR-like features, interactive and VBR (variable bit rate) applications. Moreover, it usually leads to low latencies to the newly arrived clients in the system. Conversely, as no assumptions about the access pattern are made, a random layout may cause hot spots in some disks: it is statistically possible that a large number of block requests would be directed simultaneously to the same disk.

The **operation mode** is related to the entity in charge of scheduling block requests for a client admitted into the system. The block requests can be generated by VoD servers or clients. In this work, we call these two types of operation mode as **server-driven** and **client-driven**, respectively.

In a server-driven operation mode, the client sends a request for an entire video to the VoD server. After that, the server has to control when the next video block should be read from one of its disks and sent to the client. The server operates in cycles of fixed length and, in each cycle, sends one video block to each client in the system. Despite its popularity, this operation mode has shortcomings. For instance, in a cycle-based operation, disks are synchronized and this may cause them to be idle towards the end of each cycle, reducing system performance. In a client-driven operation mode, the client is responsible for requesting video

blocks from the server and knowing the right moment to do so. The server does not operate in cycles and the task of reading blocks from the disks does not have to wait for synchronization points. The server simply processes the requests inserted in each disk queue.

4. Modeling a distributed VoD service

In our distributed version of the VoD service, there are several servers attending requests from hundreds of clients. Between these two entities there is a computer acting as a dispatcher, which represents an interface between clients and servers. The communication among these three entities is made by an ATM network. Figure 1 illustrates a global view of the system architecture.

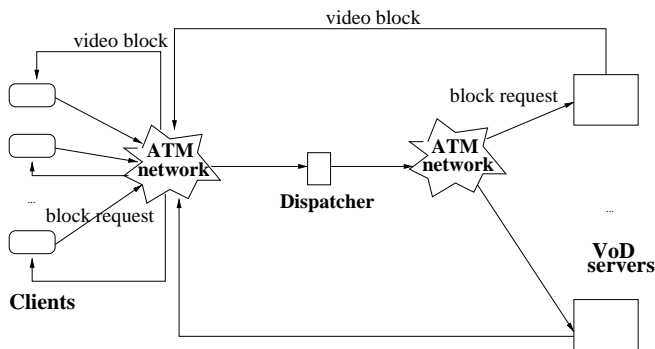


Figure 1. Global view of the system architecture

Clients do not send requests to the servers directly. They send requests to the dispatcher, which has the mapping structure of the video blocks in all servers. Upon receiving a block request (under a client-driven operation mode), the dispatcher sends it to the server which holds the copy of the requested block. The video block requested is read from the server and sent to the client through the ATM network, without passing through the dispatcher. The client of our system is represented by a TV connected to a set-top box. The set-top box is in charge of translating the client signals into requests to the VoD system. The set-top box contains a CPU and an ATM network interface.

The model of the VoD servers is based on the ALMADEM-VoD video server [2], which was developed in the VoD Laboratory at the Department of Computer Science of the Federal University of Minas Gerais. In our analysis we have considered that all the VoD servers follow the same model. The ALMADEM-VoD video server runs on a PC containing 4 disks SEAGATE BARRACUDA with 17 GB of storage capacity each, and an ATM network interface. The disks are connected to 2 IDE controllers (2 disks

per controller). There is a buffer allocated to each newly arrived client with capacity for 2 video blocks per client.

For configurations using a client-driven operation mode we have used the FIFO (first in, first out) scheduling algorithm in the disks of the servers. We have made this choice based on two reasons [15]. First, for a FIFO service discipline on the disk queues, the system performance can be predicted by a very simple disk model, based on a Gaussian distribution of the I/O service time. Second, although SCAN-based algorithms can reduce the seek overheads and provide lower I/O times, FIFO minimizes request delay variances, which is a very important factor for a performance metric based on stochastic delay bounds. In [3] a set of experiments with the ALMADEM-VoD disks (all 4 disks are identical) has been made in order to measure the I/O service time for several block sizes. In each experiment, a large number of read requests (40,000) was generated, and the I/O service time of each individual request was measured without taking into consideration queue delays. Table 1 shows the results obtained. These data were used to generate a Gaussian distribution for the I/O service time of the ALMADEM-VoD video server disks, which we use in our simulations.

Block Size (KB)	Mean (ms)	Std. Deviation (ms)
1024	76.03	15.38
512	46.29	9.02
256	31.50	8.71
128	24.17	6.16

Table 1. I/O service time according to FIFO - ALMADEM-VoD disk

For a server-driven operation mode we have used the bidirectional SCAN algorithm, which is the most popular algorithm for VoD servers operating in cycles [17]. With SCAN, requests are served in cycles and in the order of their cylinder position on the disk surface. In [3], the authors have conducted another set of experiments in our disks in order to compute the probability density function (pdf) of the I/O time $t_{nr,bs}$ to read nr requests for a block size bs . These nr requests were served according to the bidirectional SCAN. The mean I/O service time for 20 requests served according to the bidirectional SCAN for an ALMADEM-VoD disk was 878.23 ms. The standard deviation was 31.12 ms. The block size considered was 512 KB.

Our ATM network model is comprised of three levels of switches. The first and upper level is comprised of just one large switch. The second and third levels are comprised of 6 and 72 lower-cost switches, respectively. The first-level switch is an ASX-1000 from Marconi [12]. It has 128 ports, 10 Gbps of transfer capacity, and provides point to point connectivity at 622 Mbps. The VoD servers, the dispatcher,

and all the 6 second-level switches are connected to this switch. The second-level switches are ASX-200 also from Marconi with 32 ports and 2.5 Gbps of transfer capacity. Besides being connected to the first-level switch through an OC12 link, each of these second-level switches is connected to 12 third-level switches through optical links at 155 Mbps (SONET - OC3). Finally, the switches at the third level are LE25 from Marconi with 24 ports (plus an additional port expansion for interfaces at 155 Mbps), and 2.5 Gbps of transfer capacity. They provide links at 25 Mbps, are lightweight, and can be installed within the customer premises. Apart from the connection to the second-level switch by the additional port, each of these switches is in charge of 24 client residences through twisted pair category 5 links at 25 Mbps. There is a total of 1728 residences connected. This topology is illustrated in Figure 2.

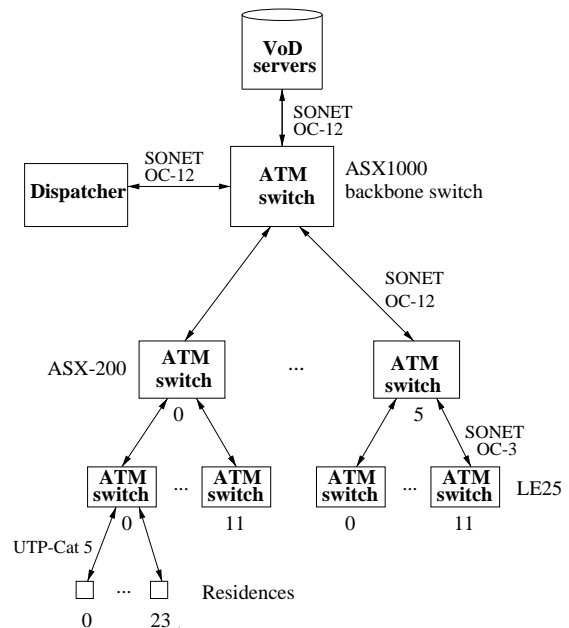


Figure 2. ATM network architecture

While we have based our topology on equipment from Marconi, any other manufacturer could have been used. Also, while we have opted for a three-level topology, most of our conclusions and experimental results are not affected by this choice. Our motivation for adopting a specific topology and a given manufacturer was to place our study on a real-case scenario.

5. Optimization strategies

In this section we describe two techniques we have introduced to make the best use of the bandwidth available

and increase the maximum number of clients in the system. These approaches have been introduced only when the operation mode was client-driven. The reason is the lack of flexibility of the server-driven operation mode. In this work, we refer to these two techniques as **anticipated delivery and retrieval of video blocks from neighbors**.

Assuming that a client has a disk inside its set-top box with capacity to store a video, we can introduce an anticipated delivery scheme which works as follows: whenever the system has enough bandwidth available, the server would anticipate the delivery of the video blocks being served at the moment and, consequently, free bandwidth for future use. The video blocks delivered according to this scheme would be stored on the set-top box disk of the respective clients. Consequently, we can allow, in certain cases, a newly arrived client to retrieve video blocks from a neighbor, instead of requesting them from a remote server.

In our approach, the system keeps track of the bandwidth available. The total load of the system is represented by the maximum number of clients the servers can attend. A client always sends the first block request to the dispatcher. The first block request is always served by the VoD servers. The following block requests, depending on the decision taken by the dispatcher, can be served by the VoD servers (standard service), or the client set-top box server (anticipated delivery service), or a neighbor set-top box server (neighbor service).

The standard service is the default alternative chosen by the dispatcher and it is how the system usually works without optimization. In this case, until all the video blocks of a certain client are retrieved, this client will consume one unit of system bandwidth.

In the anticipated delivery service, if there is enough bandwidth available in the system, the dispatcher can allow a client to request video blocks at a higher rate. Instead of requesting one video block at a time, the client requests several. The number of blocks requested each time (`MAX_MULTIPLE_BLOCKS`) is a parameter of the system and while a client keeps requesting them, it consumes `MAX_MULTIPLE_BLOCKS` units of system bandwidth. The client will continue sending these multiple block requests to the system until all the blocks of the associated video are retrieved. When the client stops sending these multiple block requests, it frees `MAX_MULTIPLE_BLOCKS` units of system bandwidth. The block requests will be directed to the VoD servers as if they were independent block requests from distinct clients. The only difference is that the retrieved video blocks, instead of being stored on the client buffer, will be stored on the client set-top box disk. Thus, the set-top box will act as a small VoD server to the client.

The dispatcher will provide neighbor service to a newly arrived client only if (i) there is a neighbor storing a video

which is the same as the one requested by the client and (ii) the neighbor is able to attend this client and (iii) there is enough network bandwidth available at the neighbor link. If all these conditions are true, the dispatcher will provide the neighbor's identification to this new client. After that, the new client will send block requests to its neighbor's set-top box server, instead of sending them to the dispatcher. As the block requests from this client will never be sent to the VoD servers, there is no consumption of system bandwidth, while at the same time, a new client is being served.

Our set-top box contains a MPEG decoder in hardware, a small SEAGATE BARRACUDA disk with 4 GB of storage capacity, a buffer with capacity for the storage of 2 video blocks, and an ATM network interface. It also contains a CPU that can act as a small video server.

6. Simulation results

Our analysis has been made using the simulation package SES/workbench [6]. It is a general-purpose simulation package that allows the modeling of real-life problems from various areas of Computer Science. The models have been implemented using the SES/sim language, which is a superset of the C++ language. All of our simulations have been executed in an *Ultra-SPARC-II* computer with 512 MB RAM and two 248 MHz processors under the Sun OS v5.5.1 operating system. First, we have compared our simulator with the simulator used to study the RIO multimedia storage server [15, 17]. This is an important step in illustrating that our simulator yields the expected results. We have later configured our system with the same parameters as those of the ALMADEM-VoD video server [2]. We have used this simulator to study the behavior of the ALMADEM-VoD video server under the four configurations considered. Our next step was to analyze the performance of our distributed VoD system varying the number of servers in the system. Finally, we have analyzed our system introducing the optimization schemes proposed and compared the system performance with and without the optimization strategies.

6.1. Performance metrics

The main goal of a VoD system is to serve, simultaneously, as many clients as possible. At the same time, the system should provide low latencies and good quality of service to its clients. Thus, the performance metrics of our analysis are the maximum number of clients the VoD system can serve simultaneously and the mean client latency.

As for admission control, we have only considered statistical admission control in our analysis due to its superior performance compared to its deterministic counterparts [9, 11]. We have adopted the same strategy used in [15]: a

statistical delay guarantee approach. The system guarantees that requests are served within a given delay bound with a very high probability. The probability of exceeding the delay bound is equal to 10^{-6} . In other words, we assume that the VoD system performs satisfactorily if only one in every one million requests suffers a request delay higher than the delay bound. Here, the request delay is the time elapsed from the moment a block request is generated until the associated video block is stored on the client buffer. With regard to disk scheduling, as we have mentioned in section 4, for the configurations using a client-driven operation mode, we have used the FIFO scheduling algorithm in our simulations. For a server-driven operation mode, we have used the bidirectional SCAN algorithm.

Our videos are encoded by the MPEG1 technique [7], divided into blocks of 512 KB, and stored across the disks of the system according to a chosen layout strategy. While the block size can be varied, we have concentrated our attention on blocks of 512 KB because the optimum block size for common videos varies between 500-600 KB [15]. We have considered three types of videos in our analysis: news clips (length between 2 and 4 min), sitcoms (length between 20 and 30 min), and full-length feature movies (length between 1:40 and 2:40 h). We have studied how the video type affects the system performance. Based on video store rental patterns, we have assumed, as in [4], that the choice of videos by the clients is highly localized, with a small number of videos receiving most of the accesses. We have used the Zipf's Law to characterize this locality. This law states that the probability of choosing the n -th most popular of M movies is $\frac{C}{n}$ where $C = \frac{1}{1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{M}}$.

Our VoD servers provide CBR traffic (i.e., the server sends data to the clients at a constant bit rate) and the video display rate was approximately 1.5 Mbps. We have considered 2 buffers at the client side (i.e., $nb = 2$). The first buffer stores the active block being read by the MPEG decoder and the other stores the next video block being retrieved by the server. The arrival process was similar to the one described in [15]: a bounded arrival process which had a fixed number of client playout requests in consecutive intervals of constant duration T . The arrival time of the client playout requests were randomly distributed over each of these intervals. The duration of the interval T was set to 10 times the mean I/O service time and we generated 20 block requests in each interval.

We have taken into account the following VCR-like features: fast-forward, rewind, pause, and stop. In addition, when the system is configured to allow for the use of VCR, we assume that 40% of the clients use VCR-like features and that each client uses only one of the four VCR-like features provided. We have chosen this percentage of 40% arbitrarily because we have not found any information in the literature about the actual behavior of a client when using

VCR-like features. The choice of a particular VCR-like feature is made randomly. We have also considered that these VCR-like feature requests lasted no more than 5 minutes (i.e., at most 5 minutes of pause or 5 minutes of a movie skipped backward or forward), and that the duration of such requests was uniformly distributed.

Our system would allow for the use of our anticipated delivery scheme provided that at least two thirds of system bandwidth were available. The number of requests sent simultaneously by a client receiving anticipated delivery service (constant MAX_MULTIPLE_BLOCKS) was set to 10 in each set of experiments. In each simulation, the system underwent three distinct phases in a 24-hour period: one hour under light load, one hour under mild load, and the rest of the period under heavy load (system at its saturation level). These parameters were chosen arbitrarily because we could not find any information in the literature regarding this problem (such a study is beyond the scope of this work). Moreover, our intention was to illustrate the effectiveness of our optimization schemes from a study case. As for the retrieval of video blocks from neighbors, we set the limit of clients per neighbor to 16. The reason is that there is a bandwidth limitation of 25 Mbps in each link between a client and a third-level switch. As each client requires approximately 1.5 Mbps of bandwidth (MPEG1 encoding scheme), more than 16 clients retrieving blocks from the same neighbor would generate network delays.

6.2. Comparison with the simulator for the RIO multimedia storage server

In [17], the authors describe a performance evaluation of RIO using a simulator that they have developed and validated. We have acquired a degree of confidence in our simulator by comparing its results with those generated by their simulator.

The RIO multimedia storage server uses a random layout and a client-driven operation mode. The disk scheduling algorithm used was FIFO and the admission control policy was the same as the one described in section 6.1. The system was configured with 14 disks connected to 7 fast-wide SCSI buses (2 disks per SCSI bus). In their simulations, they have not considered videos. There was only one multimedia object, and the clients sent block requests to randomly chosen blocks of this object. All the streams had approximately the same playout rate (1.5 Mbps - MPEG1). The client arrival rate was the same as the one described in section 6.1. In each simulation, $2 * 10^7$ block requests were generated. We have conducted the same experiments in our simulator configured in the same way as RIO. Table 2 compares the results found for both simulators regarding the number of clients in the system, for different block sizes. Our results are quite close to those generated by the

RIO simulator and the behavior at overloading conditions (maximum throughput) is similar for both simulators.

Block Size (KB)	RIO	Our simulator
128	193	187
512	294	290
1024	308	306

Table 2. Maximum number of clients for the RIO and our simulator

6.3. The ALMADEM-VoD video server

In this section, we present the simulation results of the ALMADEM-VoD video server under four distinct configurations: striping, striping-no-cycle, random, and random-with-cycle. For this, we have adapted the simulator described above, changing the parameters of the RIO multimedia storage server to those of the ALMADEM-VoD video server. In each simulation, we have generated more than $2 * 10^7$ block requests.

Table 3 details the maximum number of clients the system could serve simultaneously for our four configurations, and varying the type of video. We have observed that the system can attend a higher number of clients when the videos being accessed are news clips. This can be explained by the nature of the Zipf Law and the size of the videos. For instance, in this configuration, the system can store or 47 full-length feature movies, or 245 sitcoms, or 2055 news clips. News clips provide a better distribution of the video popularity when compared with full-length features and sitcoms, because the number of news clips the system can store is significantly higher than the system capacity for full-length features and sitcoms. Moreover, the client rotativity is higher when the system stores news clips.

The configuration with best performance, in all cases, is striping-no-cycle, followed closely by random. The striping configuration was always inferior to the configurations operating in client-driven mode. The random-with-cycles configuration performed the worst, with performance figures far below the other three configurations. Thus, it is not a competitive alternative. While one can argue that the differences in performance between striping-no-cycle, random, and striping are not large, one should keep in mind that an operation without cycles (i.e., client-driven) is simpler to design, implement, and maintain. It, therefore, seems preferable to adopt either striping-no-cycle or random.

Table 4 demonstrates VCR operations. It presents the maximum number of clients the system could serve simultaneously for our four configurations and VCR operations allowed. We have also varied the type of the videos. As we

can observe, the only combination sensible to VCR operation was the striping configuration (a striping layout combined with server operating in cycles). The striping-no-cycle and the random configurations deal well with VCR-like operations and are not affected by them. This is a great advantage that, by itself, justifies adopting a configuration based on a client-driven operation mode.

The striping configuration sensibility to VCR operations is due to the use of cycles coupled with the deterministic access pattern assumed by a striping allocation. For the striping configuration, when clients used VCR, a great number of blocks arrived late at the client side. This happened because of the pattern followed by the block requests for striping. If this pattern changes, then it is necessary to wait for spare bandwidth on a new disk. For instance, when a client asks for a pause and after some time resumes, it has to wait until there is spare bandwidth on the disk which holds the next video block in the sequence. The maximum number of clients in the system has to decrease with VCR use in order to adhere to the quality of service to be provided. Regarding the operation mode, we can conclude that the client-driven operation mode provides much more flexibility to the system because, regardless of the layout scheme, VCR operation has not influenced the maximum number of clients the system could serve.

Figure 3 illustrates how the mean client latency varied with the increase in the number of clients in the system for our four configurations. In this case, the system was storing news clips. By observing the curves, we can conclude that the client-driven operation mode provides significantly lower latencies when compared to the server-driven operation mode. The use of cycles leads to high startup latencies for the newly arrived clients in the system. The request for a first block of a certain video will only be served if there is enough bandwidth in the disk which holds the copy of this block. Moreover, if this request arrives after the beginning of a cycle, it has to wait until the beginning of the next cycle to be served, even if the disk being accessed has enough bandwidth to serve a new request.

With regard to the layout scheme, a striping layout leads to higher latencies when compared to a random layout. Considering a server-driven operation mode, when a newly arrived client in the system sends a request for the first block of a certain video, this request will only be served in a cycle if there is enough bandwidth in the disk which holds a copy of the requested block. If the available bandwidth in the system is not in this disk, this request will have to wait for the available bandwidth to cycle from disk to disk, in a carousel-like movement, until it reaches the disk which stores the requested block. For a system with N disks and close to its saturation level, this process can take N cycles in the worst case. With a random layout, the spare bandwidth does not move from disk to disk in a deterministic

Video Type	Striping	Striping-no-cycle	Random	Random-with-cycle
Full-length feature	212	225	215	105
Sitcom	212	225	215	115
News clips	216	230	225	120

Table 3. Maximum number of clients - 4 configurations - No VCR operations

Video Type	Striping	Striping-no-cycle	Random	Random-with-cycle
Full-length feature	150	225	215	110
Sitcom	150	225	215	115
News clips	175	230	225	120

Table 4. Maximum number of clients - 4 configurations - VCR operations

way, thus leading to lower latencies. The same is valid for a client-driven operation mode. With a striping layout, if there is a great number of requests accessing the same disk (hot spot), this hot spot will keep moving from disk to disk in a carousel-like movement until the end of the video exhibition. If a newly arrived client tries to retrieve the first block of a video in this disk, this client may have to wait for a long time in this disk queue. With a random layout, a hot spot can occur, but, contrary to striping, it will soon disappear, because the next block requests will not access the next adjacent disk. From the results above, we conclude that latency is another parameter of the system that favors a random configuration.

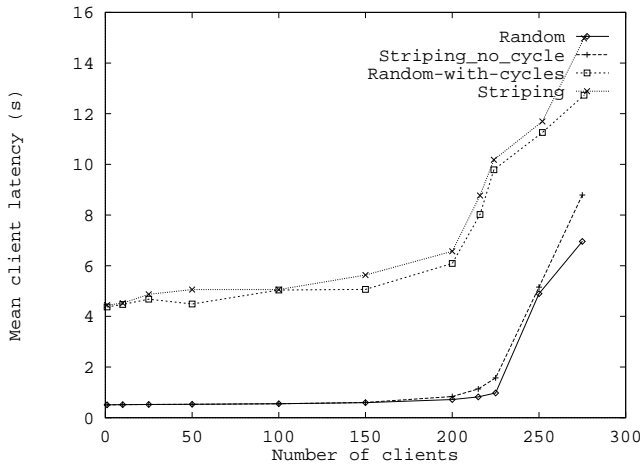


Figure 3. Number of clients x Mean client latency (s)

6.4. The distributed VoD system and the optimization strategies

As for our distributed VoD system, we have only considered a client-driven operation mode and a random layout

because, as we have shown in the last section, it is the configuration which provides the best overall performance. We varied the number of servers and the video length and then, presented how our optimization schemes resulted in an increase in the system capacity for serving clients.

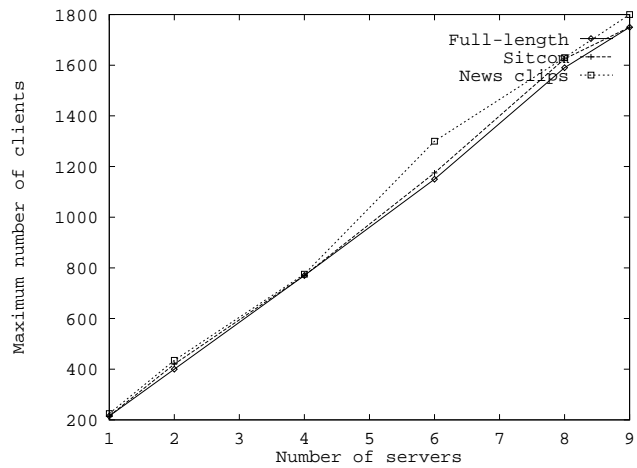


Figure 4. Number of servers X Maximum number of clients

Figure 4 illustrates the maximum number of clients our distributed system is able to serve simultaneously with the increase in the number of servers. Each curve represents the system storing only one of the three types of videos available. We conclude that 9 servers are required to attend the 1728 residences of our topology. Figure 5 shows how the mean client latency varied with the increase in the number of clients in the system. The clients were accessing sitcoms, and each line represents the system configured with a certain number of servers and at its saturation level. For any given configuration in this graphic and, up to its saturation level, the system is able to serve its clients providing reasonable low latencies (i.e., latencies lower than one second).

Figure 6 and table 5 compare the maximum number of

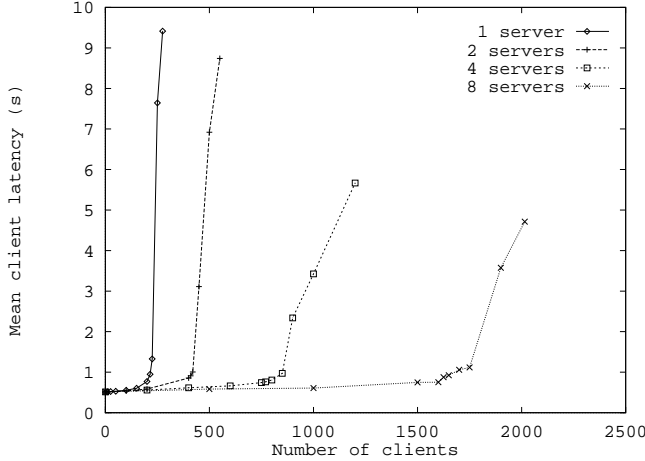


Figure 5. Number of clients X Mean client latency (s)

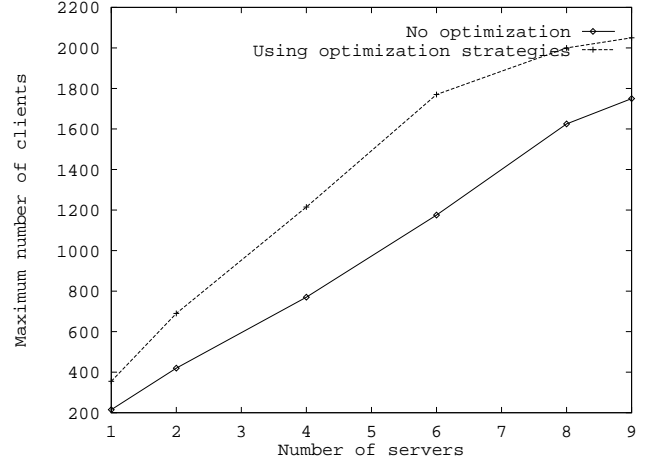


Figure 6. Number of servers X Maximum number of clients - Optimization strategies

Servers	No Optimization	Optimization	Increase
1	215	355	65.11%
2	420	690	64.28%
4	770	1215	57.79%
6	1175	1770	50.63%
8	1625	2000	23.07%
9	1750	2050	17.14%

Table 5. Increase in the number of clients because of optimization strategies

clients our system is able to attend with the increase in the number of servers, for the system configured with and without the optimization schemes proposed in section 5. The clients accessed sitcoms. The introduction of our optimization schemes increased significantly the number of clients the system could serve. Regarding the network topology considered in this work, 6 servers are required to attend the 1728 residences. Figure 7 compares the evolution of the mean client latency with the increase in the number of clients in the system. In this case, the system operated with 2 servers, storing sitcoms, and configured with and without the optimization schemes. The introduction of the optimization strategies does not result in any increase in the mean client latency. For each curve, the latency was less than one second until the system reached its saturation level. The difference is that, with the optimization strategies, the system reaches its saturation level attending a considerable higher number of clients.

7. Conclusions

In this work, we have investigated, using simulations, the behavior of a VoD server under four distinct configurations: a striped layout combined with server operation in cycles (striping), a striped layout combined with server operation without cycles (striping-no-cycle), a random layout combined with server operation without cycles (random), and a random layout combined with server operation in cycles (random-with-cycles). We have also studied the system performance under different conditions such as videos of different length and client interactivity.

First, we have compared our simulator with the simulator used to study the RIO multimedia storage server [15, 17]. Second, we have configured our simulator with the same parameters as those of the ALMADEM-VoD video server [2] and studied its behavior. We have later used the best VoD server configuration found in our analysis to evaluate the performance of a distributed VoD service in an ATM network covering an area the size of a large city neighborhood. We have analyzed the benefits of introducing optimizations to the system, such as anticipated delivery and retrieval of video blocks from neighbor clients.

Our simulation results indicate that a server-driven operation mode (i.e., based on cycles) is not the most appropriate choice for a variety of workloads, even when the layout is striped. In fact, the configurations that operated in a client-driven operation mode (i.e., without cycles) performed better in terms of the maximum number of clients in the system and the mean client latency. Furthermore, their performance were not affected by VCR-like operations (as happens with conventional striping). As for the layout scheme, although the striping layout performed slightly better than

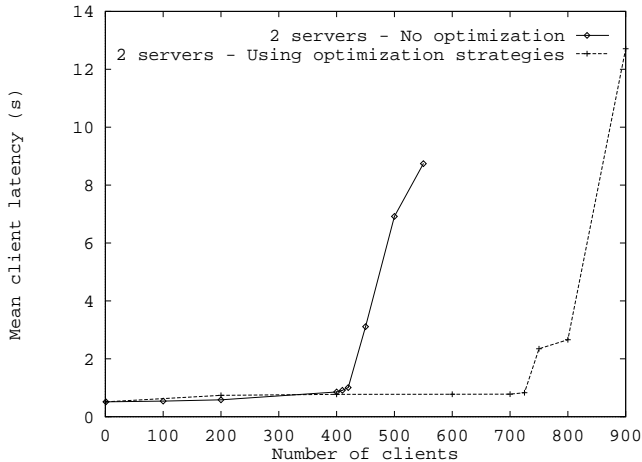


Figure 7. Number of clients X Mean client latency (s) - Optimization strategies

the random layout, when we factor in other issues such as admission control, fault tolerance, and adaptability to disk heterogeneity, the random layout offers greater flexibility coupled with less complexity. Therefore, we believe that a random configuration (i.e., a random layout combined with a client-driven operation mode) seems to be the most appropriate choice for the next generation of VoD servers which will handle different access patterns and applications, and will deal with client interactivity.

As for our distributed system, we conclude that 9 servers are required to attend the 1728 residences of the network topology under consideration. For any number of VoD servers given, and up to the system saturation level, the latencies were lower than one second, which can be considered satisfactory for the final user. The introduction of the optimization schemes increased significantly the number of clients the system was able to attend, without affecting the mean client latency, and without requiring any extra hardware to the system. Regarding our network topology, 6 servers were required to attend the 1728 residences, representing a savings of approximately 33% in the hardware required for the deployment of the service.

As future work, we intend to analyze the system performance considering video blocks of different sizes, heterogeneous VBR applications, and heterogeneous disks. Another interesting study would be to analyze the system performance using other technologies instead of ATM. Finally, an important extension would be to introduce a cache policy like that implemented in [14] which would first check if a certain video block resides on the server memory before allowing a disk access.

References

- [1] S. Berson, R. Muntz, S. Ghandeharizadeh, and X. Ju. Staggered striping in multimedia information systems. *ACM Sigmod*, 1994.
- [2] S. Campos, B. Ribeiro-Neto, L. Bertini, and A. Macêdo. Verification and analysis of multimedia systems. *ACM Multimedia 99*, pages 131–140, November 1999.
- [3] J. Caram. Comparative analysis of random and striped layouts in multimedia servers (in portuguese). Master's thesis, Department of Computer Science, Federal University of Minas Gerais, 2001.
- [4] A. Chervenak, D. Patterson, and R. Katz. Storage systems for movies-on-demand video servers. *14th IEEE Symposium on Mass Storage Systems*, September 1995.
- [5] T. S. Chua, J. Li, B. C. Ooi, and K. L. Tan. Disk striping strategies for large video on demand servers. *ACM Multimedia 96*, pages 297–306, June 1996.
- [6] HyPerformix. <http://www.hyperformix.com>.
- [7] K. Jack. *Video Demystified*. HighText Publications, 1996.
- [8] D. Jadav, A. N. Choudhary, and P. B. Berra. Techniques for increasing the stream capacity of a high-performance multimedia server. *IEEE Transactions on Knowledge and Data Engineering*, 11(2):284–301, April 1999.
- [9] S. Kang and H. Y. Yeom. Statistical admission control for soft real-time vod servers. *ACM Multimedia 2000*, 2000.
- [10] J. Korst. Random duplicated assignment: An alternative to striping in video servers. *ACM Multimedia 97*, pages 219–26, 1997.
- [11] D. Makaroff, G. Neufeld, and N. Hutchinson. An evaluation of vbr disk admission algorithms for continuous media file servers. *ACM Multimedia*, 1997.
- [12] Marconi. <http://www.marconi.com>.
- [13] B. Ozden, R. Rastogi, and A. Silberschatz. Disk striping in video server environments. *IEEE International Conference on Multimedia Computing and Systems*, pages 172–180, June 1996.
- [14] H. Pang, B. Jose, and M. S. Krishnan. Resource scheduling in a high-performance multimedia server. *IEEE Transactions on Knowledge and Data Engineering*, 11(2):303–319, March/April 1999.
- [15] J. R. Santos. *RIO: A Universal Multimedia Storage System Based on Random Data Allocation and Block Replication*. PhD thesis, Department of Computer Science, University of California, Los Angeles, 1998.
- [16] J. R. Santos and R. Muntz. Performance analysis of the rio multimedia storage system with heterogeneous disk configurations. *ACM Multimedia 98*, pages 303–308, September 1998.
- [17] J. R. Santos, R. Muntz, and B. Ribeiro-Neto. Comparing random data allocation and data striping in multimedia servers. *ACM SIGMETRICS 2000*, pages 44–55, June 2000.
- [18] P. Shenoy and H. Vin. Efficient striping techniques for multimedia file servers. *7th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV 97)*, pages 25–36, May 1997.
- [19] R. Tewari, R. Mukherjee, D. M. Dias, and H. M. Vin. Design and performance tradeoffs in clustered video servers. *International Conference on Multimedia Computing and Systems*, pages 144–150, 1996.