Disk Striping Strategies for Large Video-on-Demand Servers

Tat Seng Chua Jiandong Li Beng Chin Ooi Kian-Lee Tan
Department of Information Systems and Computer Science
National University of Singapore
Kent Ridge, Singapore 119260
email: {chuats, lijiando, ooibc, tankl}@iscs.nus.sg

ABSTRACT

The storage structure of videos on disks affects the number of concurrent users a video- on-demand system can support and hence the average waiting time. In this paper, we propose a phase-based striping method which has the desirable characteristics that it guarantees the maximum waiting time. We also develop a data replication scheme to further reduce the average waiting time of the phase-based method. These two schemes, together with the conventional sequential striping scheme, are then employed to optimize the waiting time of videos based on their access pattern. Simulations were conducted using a video server containing 36 videos under different loading and hardware configurations. The results show that such optimization can reduce the waiting time significantly. The use of replication method could further improve the server performance by over 25%. The overall results indicate that with limited resources, the use of phase-based striping method with replication is preferable under heavy loads.

Keywords: Video-on-demand, disk striping, replication.

INTRODUCTION

Recent advances in computer and communication technologies have made it possible for us to store and access large quantity of multimedia data over high-speed networks. Of the different media types, video has received the most attention because of its commercial potential in video-on-demand (VOD) applications. Video data is difficult to handle due to its large size and stringent real-time continuous playback requirements. A 100-minute MPEG-II video with a transfer rate of 4 Mbits/sec requires approximately 3 GBytes of storage space. The playback duration of a video is also very long, usually in excess of over 100 minutes for a full length feature film. Thus, once a video stream has started, sufficient resources must be committed to sustain the continuous playback of the stream over long duration. The main challenge in the design of a VOD system is on how to optimize the use of limited resources to serve a large number of users with low overall waiting time.

The main component of a VOD application is the video server, which is responsible for the storage and retrieval of videos. Most studies consider the use of multiple disks organized in the form of disk-farm or disk array [2]; while others consider the use of tertiary storage [10]. The use of multiple disks gives rise to load-balancing problems of optimizing the use of limited resources to serve a large number of video streams. The sequential disk striping method has been proposed to overcome this problem [5, 9]. It has been found to be superior in performance than storing entire video on a single disk [2, 5]. [8] proposed a phase-based matrix representation to store a video on a single disk to enable different portions of the same video to be served in phases, allowing different groups of users to view the same video at regular intervals.

In this paper, we investigate the use of different disk striping strategies to store videos based on their popularity. In order not to tie up an entire disk to a single phase-based video and to fully utilize the entire bandwidth of multiple disks, we develop a phase-based striping method to stripe each phase of a phase-based video over multiple disks. Furthermore, we develop a data replication scheme to further reduce the fixed waiting time of the phase-based method. This results in a three-tier strategy for storing videos based on their popularity – phase-based method with replication for very popular videos; simple phase-based method for popular videos; and sequential striping method for less popular videos. The combined system aims to serve a large number of users with minimum waiting time.

This paper discusses the design of our system and our video striping strategies. It also presents our simulation results on using different variants of the strategy on a video server containing 36 videos under different loading and hardware configurations.

RELATED STUDIES

At the heart of the VOD system is the video server, which is responsible for the storage and retrieval of video data. Several issues must be considered in designing a cost-effective video server. Among them are: (a) the strategy for the placement of data on disks; (b) the scheduling of disk accesses; (c) the management of buffer on the server; and (d) the admission control policies. This section reviews the issues and the related work done in these areas.

Due to the voluminous nature of video data and the high-cost of random access memory (RAM), a number of early studies investigated the use of hierarchical storage system involving magnetic disk and tertiary storage [7]. Such systems typically organize videos in a three-level hierarchical storage system comprising RAM, magnetic disk drives, and one or more tertiary storage devices (such as CD-ROM, tape libraries) [10]. Videos are stored based on their popularity, with a small number of popular videos stored in their entirety on disks, while the rest on the tertiary storage. [10] demonstrated the feasibility of such systems via simulation using the parameters derived from an actual system. However, the study is quite limited as only eight video terminals are simulated. [2] studied the performance of such hierarchical systems and concluded that it is not cost effective to use tertiary storage based on current technology.

Because of the limitation of the current tertiary storage devices, most of the existing works on VOD servers employ disks as primary storage [1, 5, 7, 9]. However, modern disks have limited storage capacity (1 - 9 GBytes) and relatively low transfer rates (30-60 Mbit/sec). In order to meet the storage and bandwidth requirements of storing and serving multiple videos concurrently, a video server would need to store the videos on multiple disks. Each video can be stored on a single disk or stripped over multiple disks. Study in [2, 5, 9] has shown that video striping improves disk utilization and load-balancing, and hence increases the number of concurrent streams. [9] considered both the coarse-grained disk striping method based on disk farm, and fine-grained method based on disk array, and concluded that the former method can support more concurrent video streams than the later. This is because disk has relatively high latency for data access (10-20 ms), and sufficient amount of the video data must be transferred in each disk access in order to better utilize the effective disk transfer bandwidth. [5] presented a comprehensive simulation study involving 16 disks and 64 one-hour videos, and considered various disk scheduling and buffer management strategies. They concluded that the use of priority disk scheduling algorithm coupled with real-time love prefetching for buffer management offers the best performance. [2] suggested that for a fully utilized system, replicating popular movies and scattering their starting positions throughout the disk farm help in reducing the average waiting time.

Instead of striping videos on multiple disks, [8] considered a different approach of organizing video on a single disk in order to serve multiple users with guaranteed waiting time. They proposed a phase-based matrix scheme where a single video is stored and accessed one column at a time. Each column may serve multiple rows (or phases) of users, and the size of a column is designed to fully utilize the disks transfer bandwidth. The scheme guarantees the reliable delivery of video streams at regular intervals and thus is able to support a large number of concurrent users with low average waiting times. The disadvantages of this method are that it requires large buffer, and the entire disk is tied-up to serve a single video. Thus, if the movie is not popular, the disk will be under-utilized. [1] considered the replication of striped video data on a disk-array-based video server with the aim of providing fault tolerance to the system.

Other issues that affect the overall performance of the system include disk scheduling, buffer management and admission control. Various disk scheduling schemes have been proposed to ensure fairness and to minimize seek time [5, 12]. Efficient management of limited buffer space in the form of RAM to support many concurrent users is discussed [5, 9]. Most works focus on providing deterministic admission control that guarantees the quality of service for all admitted requests [12]. However, some recent work attempted to increase the number of supportable requests by exploiting the fact that human perception is tolerant to brief distortions in audio and video [6, 11].

DISK STRIPING METHODS

In this section, we first describe two of the commonly used strategies for storing videos: the sequential disk striping method and the phase-based method. We then discuss our proposed phase-based striping method, the replication method, and the combined disk striping scheme.

Sequential Disk Striping Method(ssm)

From previous discussions, it is clear that the striping of video over multiple disks could achieve higher throughput and ensure better load-balancing as compared with the single-video-per-disk configuration. Assuming that each video is divided into multiple stripes of size d bits. In the sequential disk striping method (ssm), consecutive strips belonging to the same video are stored on a m-disk server in a round robin fashion, as shown in Figure 1 for videos A and B.

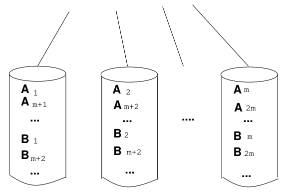


Figure 1: Sequential striping of videos A and B

$$\begin{bmatrix} 1 & 2 & \ldots & n \\ n+1 & n+2 & \ldots & 2n \\ 2n+1 & 2n+2 & \ldots & 3n \\ \vdots & \vdots & & \vdots \\ (m-1)n+1 & (m-1)n+2 & \ldots & mn \end{bmatrix}$$

(a) general case

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 8 & 9 & 10 & 11 & 12 & 13 & 14 \\ 15 & 16 & 17 & 18 & 19 & 20 & 21 \\ 22 & 23 & 24 & 25 & 26 & 27 & 28 \end{bmatrix}$$

(b) for a video with 32 stripes

$$\begin{bmatrix} (1, 8, 15, 22), (2, 9, 16, 23), \ldots (7, 14, 21, 28) \end{bmatrix}$$

(c) storage in disk in columnn major form

Figure 2: The organization of video in matrix format

Videos are stored on disks in compressed form and need to be displayed at a rate r_{disp}. For MPEG-II, $r_{disp} = 4$ Mbit/sec. To access data residing on a particular cylinder and track on the disk, the disk head must first seek the desired cylinder (t_{seek}), rotate to the beginning of the track (t_{rot}), settle down (t_{settle}), before data can be transferred at the rate of r_{disk}. In practice, q stripes are normally retrieved within the half cycle of the elevator algorithm as the disk head is moved from the innermost (or outermost) to the outermost (or innermost) cylinder. The total maximum possible seek time during one such cycle is $2*t_{seek}$. Thus in order to transfer q stripes from a disk on time, we require the following equation to hold: $q*(d/r_{disk}+t_{rot}+t_{settle})+2*t_{seek} \leq (d/r_{disp})$. In general, $r_{disk} \gg r_{disp}$, thus multiple videos can be served simultaneously. Detailed study by [5] has demonstrated the effectiveness of such approach in supporting a large number of concurrent users. In addition to better load balancing and utilization of all disks, this approach has the advantage that at light load, user requests can be served immediately. This is because spare bandwidth is available to start a new stream immediately. However, at high load when the disk bandwidth is fully utilized, a user might have to wait for a long time before being served. This is particularly bad for popular movies because: (a) when spare resource is available, multiple requests for the same popular movies may take up multiple resources by initiating separate video streams; and (b) when the load is saturated, a large number of viewers for popular movies will have to suffer long waiting times.

Original Phase-based Matrix Scheme

Several approaches can be used to overcome the problem of long waiting time for ssm when the system is fully loaded. One way is to use the strategy of [4] in reserving dedicated resources for popular movies. An elegant way to achieving this is to employ the phase-based matrix organization of video as proposed by [8]. In this scheme, a video is organized as a (mxn) matrix of video stripes as shown in Figure 2. Here, the video is divided into m*n strips. Consecutive strips are ordered sequentially from left to right and from top to bottom. For simplicity, Figure 2b shows the organization of a specific case where the movie is divided into 28 stripes.

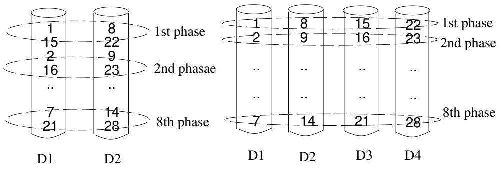
In the original proposal by [8], the entire matrix is stored on a single disk in a column major form, as shown in Figure 2c. As columns are stored and retrieved consecutively from disk, there is no overhead for disk seek and latency times. Thus the main I/O disk overhead is the transfer time. The number of rows m can therefore be set to the largest integer less than or equal to (r_{disk}/r_{disp}). The value m also gives the number of phases in which up to m groups of users can view different phases of the same video simultaneously. The phase delay, t_d, is given by L_{video}/m, where L_{video} is the length of the entire movie. t_d gives the upper bound in waiting time for any new users of this movie. The strip size d (and hence n) is chosen appropriately so that the amount of video transferred in each disk access is reasonable. A good value for d is 512 KBytes.

This scheme requires three buffers of size m*d to operate. The first buffer stores the current column being viewed. The second buffer holds the column being transferred from disk. The third buffer stores the first column in order to give time for the disk head to re-position at the beginning of column 2 after reading column n.

Phase-based Striping Method(psm)

As the original phase-based method stores the entire video on a single disk, it is effective only if the movie is popular. However, for unpopular movies, the disk will be idle and underutilized. We propose to extend the sequential disk striping method to store the data for the phase-based matrix scheme onto multiple disks. Instead of storing the entire column consecutively onto a single disk, stripes in each column are distributed evenly onto multiple disks in a round robin fashion as shown in Figure 3. For clarity, Figure 3 shows only the case where the video is divided into 28 blocks (see Figure 2b), and they are striped into two different configurations with two (Figure 3a) and four disks (Figure 3b). The scheme can be generalized to m disk configuration.

Whenever a new phase of video is required, the combined bandwidth of the disk farm is used to transfer the whole column of videos onto the buffer. Since the



(a) Two–disk configuration (b) Four disk configuration

Figure 3: The striping of phase-based video

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \\ 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 \\ 25 & 26 & 27 & 28 & 1^* & 2^* & 3^* & 4^* \end{bmatrix}$$

(a) replicate at 1/2 position

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 \\ 19 & 20 & 21 & 22 & 23 & 24 & 25 & 26 & 27 \\ 28 & & 1^* & 2^* & 3^* & 4^* & 5^* & 6^* & 7^* \end{bmatrix}$$

(b) replicate at 1/4 position

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 \\ 19 & 20 & 21 & 22 & 23 & 24 & 25 & 26 & 27 \\ 28 & 1^* & 2^* & 3^* & 4^* & 5^* & 6^* & 7^* & 8^* \end{bmatrix}$$

(c) replicate at 1/8 position

Figure 4: Replication schemes for phase-based method

combined bandwidth is much larger than the bandwidth required to transfer a single column, no disk is tied up entirely just to serve a single video. The users viewing requests can still be satisfied at regular intervals of t_d. Thus, such scheme offers the advantages of both disk striping and phase-based matrix organization method. The buffer requirements for the striping method is only two columns – one for the transfer and the other for serving the users. The first column does not need to be stored as there is sufficient bandwidth to transfer this column in time at the start of the new retrieval cycle.

Disk Replication for Phase-based Method(psmr)

Although the phase-based method offers regular starting of videos at fixed intervals and limits the maximum waiting time of new users to t_d, it is not very effective at light load as users still need to wait for an average duration of $0.5 * t_d$. Further reduction of this time can be made by replicating videos. Here, we are interested to devise a replication scheme for the phase-based video organization. We call our new method phase-based striping method with replication (psmr).

One simple observation from the phase-based organization is that a new phase can be started whenever the starting stripe (stripe 1) is encountered. This happens whenever the first column is retrieved at the end of a complete retrieval cycle. Thus if we can insert additional starting stripe within the cycle, additional phases can be started before the end of the cycle. For example, if we position an additional starting stripe in the middle of a cycle, then new phases can be started at mid point. This will cut down the average waiting time by up to half of the usual t_d. Of course, the matrix will have slightly more columns or rows in order to accommodate additional replicated stripes, and more buffer is needed to sustain the additional phases. We shall examine these issues by using the specific example shown in Figure 2b.

If we stripe the phase-based organization onto multiple disks, the transfer bandwidth is no longer a limiting factor in deciding the number of rows required. However, in order to have the same transfer bandwidth for both psm and psmr videos, we keep the number of rows the same in both cases. Thus, the additional replicated stripes will result in more columns for the psmr videos. In this study, we investigate the cases of replicating starting stripe at half-way, quarter-way, and one-eighth positions. The resulting matrix structures are shown in Figures (4a-4c), where replicated data are shown with an asterisk. Note that the number of rows is fixed at 4, while the number of columns is increased from 7 to 9 (for the one-eighth case).

Take the case of Figure 4a, a new user may start a new stream (known as the replicated stream) in the middle of the row. Once the replicated stream is started, it must be sustained with continuous data. Figure 4a shows that the replicated stream will be fed with first four stripes of data from the usual retrieval cycle. After stripe 4 is displayed, it must be supplied with stripe 5 onwards. This can be achieved if we buffer the next 4 stripes (stripes 5 to 8) using additional buffer. Thus if we continue to buffer four additional stripes ahead of time, the replicated stream can be sustained. From this observation, it can be seen that to sustain one replicated stream, additional buffer equivalent to half the size of a row of video stripes must be made available. If we start m replicated streams (one for each row), then we will need a buffer size equivalent to half the size of the entire video. The advantages of this technique are that it reduces the average waiting time by close to half, and additional replicated streams can be started as and when sufficient buffer is available.

For the case of replication at the quarter position, additional buffer equivalent to a quarter of the size of the row is needed to sustain one replicated stream. However, the users will have uneven waiting times depending on the times of their arrivals. Similarly, for the case of replication at one-eighth position, the corresponding buffer required per stream is only 1/8 of the size per row. However, the disparity in waiting time will be even greater.

The replication schemes will result in lowering overall average waiting time for the phase-based scheme. However, the relative strength of each scheme needs to be

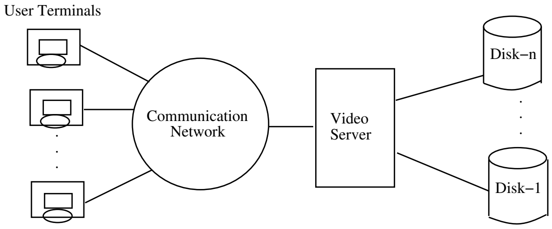


Figure 5: The Structure of the VOD system

assessed.

Combined Video Striping Strategy

Each storage structure is suitable for certain access patterns. To handle different access patterns, we propose a three-tier strategy, where the phase-based striping method with replication (psmr) is used to store very popular videos; the phase-based striping method (psm) for popular videos; and the sequential striping method (ssm) for unpopular videos. The main objective of such a design is minimum overall waiting time.

Such scheme has the advantages that: (a) users of popular videos stored using psm or psmr are guaranteed a finite maximum waiting time. The psm is equivalent to batching users and reserving of resources for popular video. This has been shown to be effective for popular movies at heavy load [4]; (b) as more spare buffer is available, we can start more replicated streams using psmr to further reduce the guaranteed maximum waiting time; (c) at light load, users of unpopular video stored using ssm can be served immediately; and, (d) since the basic resources required for both psm and psmr are fixed, and that of the replicated scheme can be dynamically adjusted, resources can be managed to balance the needs of users of both very popular and unpopular videos.

THE DESIGN OF A VOD SYSTEM

Figure 5 shows the design of our VOD system. The system consists of a video server with a m-disk disk farm. The server is connected to a set of end user terminals by a high speed communication network such as the ATM. As with most studies, we assume that the network is not a bottleneck.

The end user terminals are used by the subscribers to view movies. Each terminal is equipped with a CPU and two MBytes of RAM for buffering data. This configuration is quite standard for a low-end PC. Two MBytes of RAM could buffer an equivalent of 4 seconds of MPEG-II video. The terminal fills the buffer with data before starts showing the video to the user.

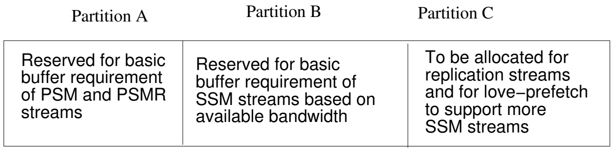


Figure 6: Partitioning of buffer memory

The movies are stored in the disk farm by using the combined strategy as discussed in Section 3.5. Popular videos are organized using psm or psmr; the rest of videos are organized sequentially. All video blocks are striped into the m-disk in the disk farm.

The server contains a large RAM for use as buffer for all users and video streams. Users requirements are translated into requests to transfer appropriate data blocks into the buffer. Three methods of data organizations are involved and each requires a varying amount of basic memory to operate. In order to strike a balance between the requirements of all videos with different organization, the buffer is partitioned as shown in Figure 6. In the following discussions, we assume that there are sufficient buffer to support the basic memory requirements of all psm and psmr videos, and q streams of ssm videos. Also, as the basic buffer requirements of psmr is the same as that of psm, we will treat them as the same in subsequent discussions.

The basic buffer requirement of each psm video is fixed and is equal to the size of two columns in the corresponding matrix organization. We can pre-compute and reserve the required buffer for all psm videos before the start of the application. This is shown in partition A in Figure 6. The number of ssm streams is constrained by both the available bandwidth and buffer size. As there is sufficient buffer in our VOD system to support a large number of ssm streams, the number of basic ssm streams that can be supported is determined by the available disk bandwidth. This number is used to work out the buffer size of partition B. The remaining buffer in partition C, which constitutes the main bulk of buffer, can be used to activate more replicated streams for very popular video or be used to start up more ssm streams without the additional bandwidth overhead. The balance between the two is quite complicated.

We now discuss the buffer management, admission control, and the disk scheduling strategies for our server. The buffer management for psm videos is straight-forward, The fixed set of buffer is simply re-used in a cyclical manner as columns are retrieved repetitively. The buffer size needed is also fixed. The management of buffer for ssm streams is more complicated. In order to take advantage of video stripes remaining in RAM and to perform ore-fetching, we adopt the love-prefetch strategy [5]. The love pre-fetch strategy uses two LRU lists, one for referenced pages and one for pre-fetched pages. If the memory runs out, the system tries to re-use pages in the referenced LRU list before considering those in the prefetched list. In order to fully use the additional buffer in partition C, we must balance the allocation of buffer between replication for psmr videos and love-prefetch for ssm videos. If there are many clients request for ssm videos, more buffer should be allocated to love-prefetch for starting up more ssm streams. If however the ssm videos are unpopular with only few clients, and most of them can be responded immediately, then more buffer should be allocated to start up more replicated streams. Hence the allocation of buffer in partition C needs careful study. A simulation to study the effect different strategies of allocating the buffer under different operating conditions can be found in [13].

For admission control, since the buffer required for psm videos has already been reserved, users of such videos are simply batched to start viewing at fixed starting times. For users of very popular videos stored using psmr, a new replicated streams can start as long as there are available buffer space in partition C. The replicated stream is started on the first come first serve (FCFS) basis based on the availability of reserved buffer. The FCFS policy has been found to be fair and can be more effective than the admission policy based on maximum queue length [4]. Each replicated stream reuses the fixed set of buffer to pre-store data ahead of time until the end of the stream. For users of videos stored using ssm, a new user is admitted if the available disk bandwidth is sufficient or the first block of the requested video is found in the love pre-fetch buffer pool.

Finally, we adopt the real-time disk scheduling algorithm to schedule disk accesses. It has been demonstrated in [5] to be one of the best strategies for striped disk farm. As with one of their strategies, we use two priority classes and the priority cutoffs of four seconds. That is, we group requests that are required within the next four seconds under class 1, and the remaining requests in class 2. Only requests in class 1 are serviced during the current elevator retrieval cycle. The remaining requests are re-classified at the end of each elevator cycle.

SIMULATION STUDY

To demonstrate the effectiveness of our design, we conduct a simulation study on our proposed system. Our system consists of four CPUs and a large common buffer with capacity ranging from one to four GBytes. The server includes a 24-disk disk farm containing 36 different video titles. We use the Quantum Atlas II hard-disk, which is representative of the type of high performance disks available commercially. The parameters of each disk are given in Table 1. We assume all our videos are 90-minute in length. The video is based on MPEG-II with a transfer rate of 4 Mbit/sec. For 36 videos, the total disk capacity required to store all videos is 97.2 GBytes. The popularity of each video is determined based on Zipfian distribution.

Capacity	4300 MBytes/sec
Transfer Rate	7.3 MBytes/sec
Transfer Rate	8.5 msec
Settle Time	0.75 msec
Rotation Time	8.33 msec

Table 1: Parameters of a disk

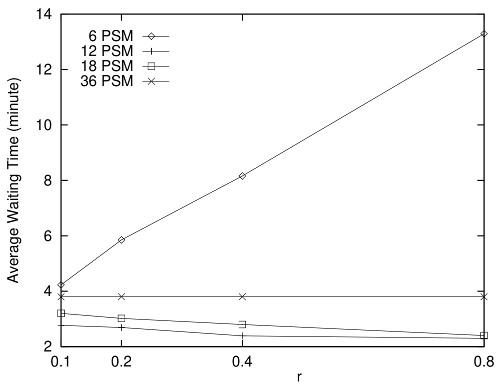
We fix the size of the video stripe at 512 KBytes, which has been shown to be optimal for striped disk farm servers [5]. The playback time of each stripe is one second. The entire video is divided into 5,400 equal sized stripes. For psm videos, they are organized as 12-row by 450-column matrices. The duration of each phase is therefore 450 seconds. Hence the maximum waiting time for the psm video is 450 seconds, or 7.5 minutes. The basic buffer required for a psm video is twice the size of the column, which amounts to 12 MBytes. The total amount of buffer required for partition A is s*12 MBytes, where s is the number of psm or psmr videos. Thus, even if all videos are stored under psm, the total amount of fixed buffer required is only 432 MBytes. A basic server buffer size of 1 GBytes is therefore more than sufficient to support most combinations of our proposed disk organization scheme.

The simulation period is fixed at 12 hours. We vary the total number of user requests during this period in order to test the effects of changing loads on the performance of the system. We use Poisson distribution to characterize the users' arrival rates. The distribution can be varied by changing the Poisson distribution factor, p. Finally, we adopt the strategies discussed in Section 4 for managing the main buffer, scheduling disk accesses, and controlling user admissions.

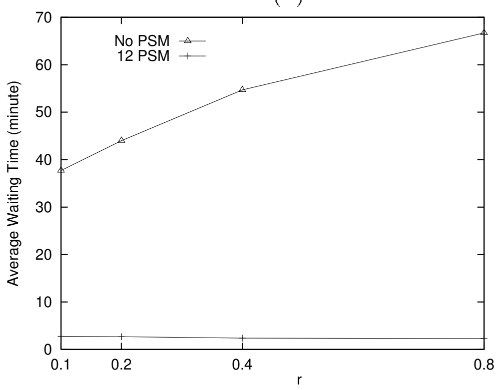
The simulation system is developed using CSIM/C++ on Sun Sparc-2 workstation. The metric of the simulation is the average waiting time.

Simulation Parameters

The variables used in our simulation include: (a) the Zipfian parameter r for varying video access pattern; (b) the Poisson distribution factor p for varying user request distributions; (c) the total number of user requests during the entire simulation period; (d) the total buffer size; and (e) the number of most popular videos stored



(a)



(b)

(Poisson parameter = 8; Total number of requests = 3000; Buffer size = 4 GBytes)

Figure 7: Effects of video access patterns (Zipfian parameter r)

using psm and psmr.

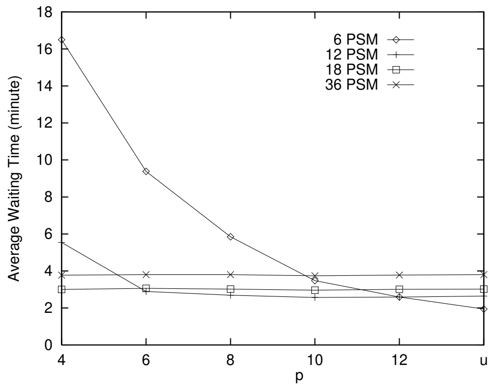
In order to reduce the complexity of the simulation, we vary only one variable during each simulation while keeping the rest fixed. We adopt the default values of [5] as follow:

- Zipfian parameter r = 0.2;
- Poisson distribution factor p = 8;
- Total number of incoming clients = 3,000;
- Total buffer size = 4 GBytes

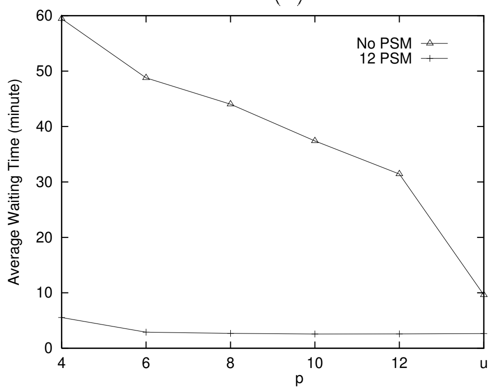
Simulation 1: The Effect of Video Access Patterns

Figure 7 shows the effect of video access patterns on the average waiting time. As the Zipfian parameter r becomes smaller, the access pattern becomes more localized with a small number of videos receiving most of the accesses. From the figure, it can be seen that if no psm video is used, the users need to wait more 37 minutes, and the waiting time gets longer when the access pattern becomes more uniform. By placing the top 6 popular video under psm, the corresponding waiting time is greatly reduced. This is because most users' requests to popular movies only need to wait a fixed duration of less than 7.5 minutes (maximum waiting time). If we place the top 12 most popular videos under psm, the average waiting time is lower to less than 3 minutes. However, as the number of psm videos increases to more than 12, the performance degrades due to the fact that users of psm videos must now wait a finite duration for a new phase to start, while users of ssm videos can start immediately so long as there are resources. Thus, there is no advantage in putting unpopular videos under psm. When all videos are striped based on psm, the performance is not sensitive to access pattern.

Simulation 2: The Effect of User Arrival Rates



(a)



(b)

(r = 0.2; # of requests = 3000; Buffer size = 4 GBytes)

Figure 8: Effect of the arrival of user requests

Figure 8 summarizes the effect of the arrival of requests on the performance of the system. Here, we vary the Poisson distribution parameter p from 4 (very skewed) to 12 (almost uniform). For comparison purposes, we include the result when user requests are arriving at uniform rate.

For the case when no psm video is used (Figure 8b), we observe that when many requests arrive within a short time (for small p), the overall average waiting time is very long. This is because resources are used to start up new ssm video streams and new arrivals have to wait for a long time for free resources. As the user arrival pattern becomes more uniform, its performance also improves steadily to an average waiting time of around 10 minutes for uniform arrival pattern.

A marked improvement in performance is observed if we place the top 6 or 12 most popular videos under psm. Since there are still a big number of ssm videos, the performance is still not as good if the user requests arrive in great concentration (p=4). As the arrival pattern becomes less skewed, the performance of system with 12 psm videos becomes consistently better than the other combinations. If all videos are placed under psm, the performance is independent of the users' arrival pattern. Its performance is also not as good as a system with 12 psm videos as users must wait for a finite amount of time independent of resource availability. This simulation confirms that it is not advantageous to store unpopular video under psm.

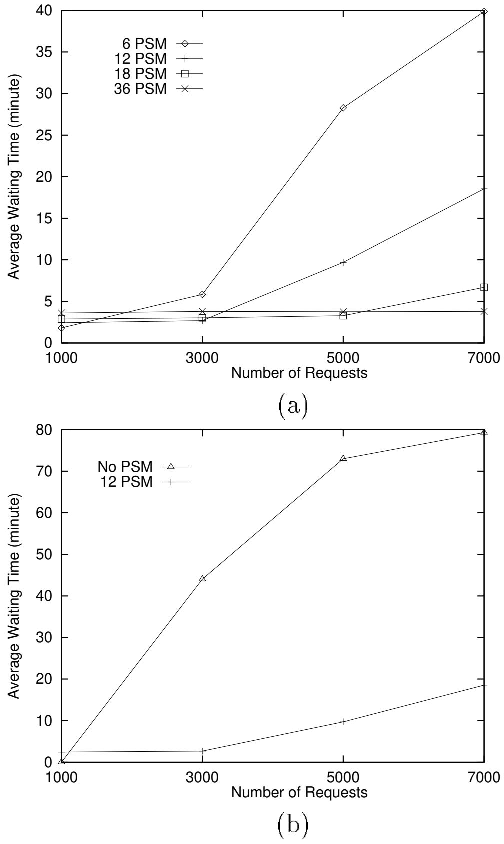
Simulation 3: The Effect of the Total Number of Requests
The number of user requests affects the waiting time as there is a limit on the number of concurrent users that a system can support. Figure 9 shows that when the number of requests is small (1,000), the ssm scheme is efficient as most videos can be initiated almost immediately. However, as the number of user requests increases, the performance of ssm scheme degrades.

The same pattern is observed for schemes with varying number of psm videos. At very high load of 7,000 user requests, the scheme where all videos are placed under psm performs the best. The same is true if requests arrive at uniform rate. This simulation shows that at light load, ssm scheme tends to perform well. At the other extreme of very heavy load, it is preferable for all videos to be placed under psm where the maximum waiting time is fixed.

Simulation 4: The Effect of Buffer Size
This simulation studies the effect of changing buffer size on the performance of the system. As discussed earlier, a psm video requires only the basic buffer to operate, its performance is independent of the availability of additional buffers. With the use of Love-Prefetch technique, additional buffer only plays a part for ssm videos. As the available buffer pool for ssm videos becomes larger, more pages will stay in memory longer and hence there is an increased chance of re-using data on buffer. This will improve the performance of the overall system.

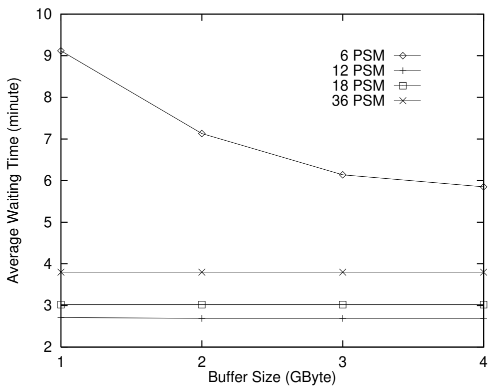
As more popular videos are placed under psm, the effects of extra buffer becomes less important. In fact, optimal performance is again obtained when there are 12 psm videos.

Simulation 5: The Effects of Data Replication
To test the effect of replication, we replicate the top three videos out of the 12 videos. As discussed before,

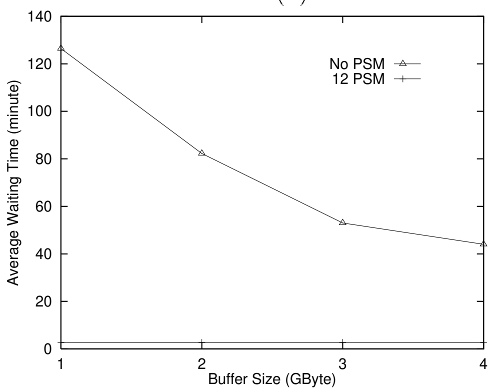


(r = 0.2; p = 8; Buffer size = 4 GBytes)

Figure 9: Effect of total number of user requests



(a)



(b)

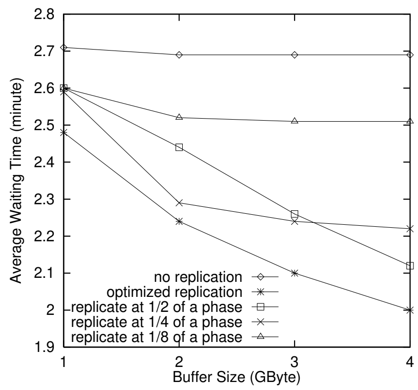
(r = 0.2; p = 8; # of requests = 3000)

Figure 10: Effect of buffer size

we divide the memory into three partitions. We reserve sufficient memory in partitions A and B to ensure that all 12 psm or psmr videos and all basic ssm streams can start. The remaining buffer in partition C is used for replication purpose. Since the memory in partition C is limited, not all possible replicated streams can start. Different replication scheme requires different amount of buffer per stream. For example, videos replicated at 1/8 positions require less buffer per stream than those replicated near the middle of the row. Because of this variation, our strategy is to start as many replicated streams as possible based on available memory and replication scheme.

Figure 11 shows the effect of replicating the top three most popular psmr videos at 1/8, 1/4 and 1/2 positions. The general results demonstrate that the use of replication, in addition to phase-based organization, leads to further improvement in the performance of the system. The gain is even more significant for larger buffer sizes.

As expected, when the buffer size is small (= 1 GBytes), the 1/8 replication scheme performs much better than the other two schemes due to the simple fact that more



(r = 0.2; p = 8; # of requests = 3000; # of psm videos =9; # of psmr = 3)

Figure 11: Effect of replications and buffer sizes

1/8 replicated streams which require less memory can be supported. However, as the buffer size increases, more streams of the other schemes, which have small waiting time, can be initiated. Hence, the 1/2 replicated psmr scheme outperforms the other two when the buffer size is large.

These results lead us to observe that: (a) it is better to replicate data as near to the middle as possible if there are enough memory to start all replicated streams; and (b) if the memory is limited, it is better to choose a replication scheme that requires less memory so that more replicated streams can start. To confirm our observations, we devise an optimal replication scheme, where the position of replication is chosen to ensure that all possible replicated streams can start and the position of replication is as near the middle as possible. From Figure 11, it can be seen that our optimal replication scheme performs the best under different buffer sizes. As compared with using only psm for the top 12 most popular videos, the use of optimal replication further results in up to 25% improvement in performance when the buffer size is 4 GBytes.

CONCLUSIONS

In this paper, we proposed a phase-based disk striping method (psm) which has the following advantages: (a) guaranteed maximum waiting time; and (b) ensured better load-balancing and utilization of all disks in the disk farm. We also proposed a data replication scheme to further reduce the finite average waiting time of the psm scheme. To handle different access patterns, we proposed to store videos using different storage strategy based on their popularity.

We conducted extensive simulation study to evaluate the effectiveness of our proposed system under a variety of operating conditions, such as: (a) varying video access patterns; (b) different requests distributions; (c) different system loads in terms of total number of user requests; and (d) different buffer sizes. From the study, we can draw three conclusions. Firstly, the use of psm for popular videos leads to significant improvement in the performance of the system under most operating conditions. In some cases, the improvement is up to 400% compared to using ssm alone. Secondly, it is effective to use a combination of psm for popular videos and ssm for less popular videos. Finally, the use of replication in conjunction with phase-based scheme results in further improvement in the performance of the system, especially when there is sufficient buffer.

Further research will be conducted in the following directions: (a) to support VCR operations such as pause, rewind, and fast forward, etc.; (b) to support non-sequential access of video required in content-based video retrieval and virtual editing applications [3]; (c) to devise better strategies for the utilization of buffer to support different video striping strategies; and (d) to implement our proposed method in a distributed environment.

Acknowledgments

The work of J. Li was supported by a research scholarship from the National University of Singapore. The work was in part supported by the NUS research grant number RP-950694 (VISA).

REFERENCES

1. M.S. Chen, H.I. Hsiao, C.S. Lie and P.S. Yu. Using rotational mirrored declustering for replica placement in a disk array-based video server. Proceedings of the ACM Multimedia'95, 121-130, 1995.

2. A.L. Chervenak and D.A. Patterson. Choosing the best storage system for video server. Proc. of the ACM Multimedia'95, 109-119, 1995.

3. T.S. Chua and L.Q. Ruan. A video retrieval and sequencing system. ACM Trans of Information Systems, 13(4): 373-407, 1995.

4. A. Dan, D. Sitaram and P. Shahabuddin. Scheduling policies for an on-demand video server with batching. Proc. of ACM Multimedia'94, 15-23, 1994.

5. C.S. Freedman and D.J. DeWitt. The SPIFFI scalable video-on-demand system. Proceedings of the 1995 ACM-SIGMOD International Conference on Management of Data, 352-363, 1995.

6. P. Lougher and D. Shepherd. The design of a storage server for continuous media. The Computer Journal, February 1993, Vol. 36, No. 1, 32-42.

7. B.C. Ooi, K.L. Tan, T.S. Chua and J. Li. Storage management of continuous media data. IEICE Data Engineering, December 1995, Vol. 95, No. 407, 1-6.

8. B. Ozden, A. Biliris, R. Rastogi and A. Siberschatz. A low-cost storage server for movie on demand databases. Proc. of the 20th VLDB Conference, 594-605, 1994.

9. B. Ozden, R. Rastogi and A. Siberschatz. Disk striping in video server environments. Data Engineering, 18(4): 4-16, 1995.

10. H. Suzuki, K. Nishimura, A. Uemori and H. Sakamoto. Storage hierarchy for video-on-demand systems. Proceedings of SPIE, 198-206, 1994.

11. H.M. Vin, P. Goyal, and A. Goyal. A statistical admission control algorithm for multimedia server. Proc. ACM Multimedia'94, 33-40, 1994.

12. P. Yu, M.S. Chen and D.D. Kandlur. Design and analysis of a grouped sweeping scheme for multimedia storage management. Proc. of the 3rd International Workshop on Network and Operating System Support for Digital Audio and Video, 38-49, 1992.

13. T.S. Chua, B.C. Ooi and K.L. Tan. Design and analysis of a high performance VOD server. working paper, 1996.