

# Design and Performance Study of Rate Staggering Storage for Scalable Video in a Disk-Array-Based Video Server

Xin-Mao Huang, Cheng-Ru Lin, and Ming-Syan Chen, Fellow, IEEE

**Abstract** — *This paper provides a data placement method based on rate staggering to store scalable video data in a disk-array-based video server. Scalable video means a video which is coded in such a way that subsets of the full video bit stream can be decoded to create low quality/resolution videos. Supporting layered multiple resolutions from a video server is very desirable in many applications. Note that in a disk array the video data corresponding to different rates of the same video clip are not required to reside in the same disk. In view of this, we propose and explore in this paper the approach of rate staggering, i.e., staggering video data in the disk array based on data rates. It is shown that the advantages of the proposed rate staggering method include: (1) minimizing the intermediate buffer space required by the server, (2) achieving better load balancing due to finer scheduling granularity, and (3) alleviating the disk bandwidth fragmentation. These advantages enable a video server using the rate staggering method to provide feasible solutions to some video stream requests which cannot be met otherwise. The system throughput can thus be increased. We also conduct several simulations for various applications. These experimental results show that the rate staggering method can significantly improve the performance of a VOD system.*

**Index Terms** — Multi-resolution video, video server

## I. INTRODUCTION

Recently, the promise of multimedia technologies to have a significant impact on both information providing service and entertainment business. These multimedia technologies are used in video applications which have important impact to consumer electronics [7], [12], [13], [19], [23]. Given the extremely large data size, the major challenge to handle multimedia data is to support not only very high disk bandwidth for video retrieval but also very high network bandwidth for data transmission [18], [20]. ATM (Asynchronous Transfer Mode) has been proposed as a solution to meet the demand for high network bandwidth. On the other hand, disk arrays are employed to provide the disk bandwidth required for a video server [10], [11], [15], [16], [21] [24], [25], [26]. For example, to display HDTV quality image, it will require video data at 2 Mbytes per second (even after compression). It is very undesirable to store such a video in a single disk because

of two reasons. First, a 100 minute HDTV movie will require an amount of 12 Gbytes storage. Such a large disk is usually expensive. Second, playing a hot (i.e., frequently requested) movie by a single disk may cause performance bottleneck. In fact, even for playing movies of ordinary quality, the need to support multiple video streams by a video server also calls for the use of disk-arrays. Consequently, it is highly desirable to use disk striping in a disk array to handle the storage and retrieval of video data [2], [10], [21], [22], [3]. Conventionally, disk striping is done by dividing the video data into blocks according to their presentation order (i.e., time sequence) and storing these blocks into different disks. It is noted that with such a disk striping better load balancing can be achieved by staggering the starting times of different video streams [2]. This is referred to as time staggering. Moreover, a dynamic heuristic broadcasting (DHB) protocol has been proposed in [3]. DHB is a slotted protocol and prepares a transmission schedule starting at the next slot according to the user request. On the other hand, a heterogeneous source allocation and scheduling method that supports application-level soft real-time performance guarantee was proposed in [12]. In addition, a different method that uses the proxy server to store the stream media file for reducing bandwidth has been proposed in [1]. The stream media file should be stored at each proxy according to the network interface bandwidth, number of disks, and processing capacities. In addition, for real-time media data transmission, the network service provider needs to dedicate part of network resources for Quality of Service (QoS) in terms of end-to-end delay and throughput. Therefore, it is important to partition the end-to-end QoS requirement of a network flow along the links of a given path such that the deviation in the loads on these links is as small as possible [11].

Multi-resolution coding, referring to the encoding technique which accommodates at least two resolutions in a video stream, is able to provide scalable video [4], [8], [17]. Scalable video means a video which is coded in such a way that subsets of the full video bit stream can be decoded to create low quality/resolution videos [5], [6], [9]. Supporting layered multiple resolutions from a video server is desirable by the broadcasting industry since a video provider (such as a VOD company) may want to

provide different customers with different levels of service. Naturally, the resolution of a video ordered by a customer with an HDTV will be higher than that ordered by a customer with a conventional TV. In addition, multiple resolution encoding is useful for the computer industry for such applications as multiplatform decoding which allows video to be decoded by platforms of different capabilities, and also multiwindow decoding where videos of different resolutions can be independently selected by decoders to produce videos for different window sizes.

A significant research effort has been elaborated upon multi-resolution coding, and led to the development of hierarchical coding techniques such as subband coding. Basically, subband coding is an approach of using a filter bank to decompose the original video into several frequency bands, resulting in a set of multiple resolution videos [8]. For example, by dividing the frequency domain into four regions, one can decompose the original video into four sets, say R1, R2, R3 and R4, where  $R_i$  is called rate  $i$  data. Then, R1 corresponds to the basic video (class 1), R1+R2 corresponds to class 2 video, R1+R2+R3 corresponds to class 3 video, and R1+R2+R3+R4 corresponds to the full resolution video. Figure 1 shows a disk-array-based video server which provides multi-resolution video.

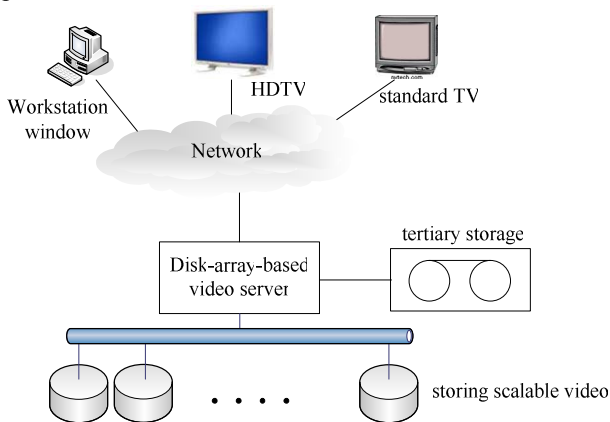


Fig. 1. An illustration for a disk-array-based video server to provide multi-resolution video.

Note that video data of different rates can be separately stored to provide different resolutions of videos. Essentially, it is not required to store those data corresponding to different rates of the same video clip within the same disk. The approach of staggering video blocks in the disk array based on data rates is termed *rate staggering* in this paper. We propose and explore in this paper the rate staggering method to store scalable video data in a disk-array-based video server so as to minimize the buffer space required by the server and to improve the system throughput. The storage unit of video data is a

TABLE I  
SCALABLE VIDEO PLACEMENT WITH DISK STRIPING, BUT WITHOUT RATE STAGGERING.

Disk No.	1	2	3	4	5	6	7	8
Rate1	$B_{1,1}$	$B_{2,1}$	$B_{3,1}$	$B_{4,1}$	$B_{5,1}$	$B_{6,1}$	$B_{7,1}$	$B_{8,1}$
Rate2	$B_{1,2}$	$B_{2,2}$	$B_{3,2}$	$B_{4,2}$	$B_{5,2}$	$B_{6,2}$	$B_{7,2}$	$B_{8,2}$
Rate 3	$B_{1,3}$	$B_{2,3}$	$B_{3,3}$	$B_{4,3}$	$B_{5,3}$	$B_{6,3}$	$B_{7,3}$	$B_{8,3}$
Rate 4	$B_{1,4}$	$B_{2,4}$	$B_{3,4}$	$B_{4,4}$	$B_{5,4}$	$B_{6,4}$	$B_{7,4}$	$B_{8,4}$

block, which is composed of a sequence of frames. An

TABLE II  
SCALABLE VIDEO PLACEMENT WITH RATE STAGGERING.

Disk No.	1	2	3	4	5	6	7	8
Rate1	$B_{1,1}$	$B_{2,1}$	$B_{3,1}$	$B_{4,1}$	$B_{5,1}$	$B_{6,1}$	$B_{7,1}$	$B_{8,1}$
Rate2	$B_{7,2}$	$B_{8,2}$	$B_{1,2}$	$B_{2,2}$	$B_{3,2}$	$B_{4,2}$	$B_{5,2}$	$B_{6,2}$
Rate 3	$B_{5,3}$	$B_{6,3}$	$B_{7,3}$	$B_{8,3}$	$B_{1,3}$	$B_{2,3}$	$B_{3,3}$	$B_{4,3}$
Rate 4	$B_{3,4}$	$B_{4,4}$	$B_{5,4}$	$B_{6,4}$	$B_{7,4}$	$B_{8,4}$	$B_{1,4}$	$B_{2,4}$

example for the conventional scalable video placement, which has disk striping but not rate staggering, is shown in Table I, where a disk array of 8 disks is used and the block comprising rate  $j$  data of the  $i$ th clip is denoted by  $B_{i,j}$ . An example for the scalable video placement with rate staggering is given in Table II. As it will be shown later, the approach of rate staggering has the following three advantages:

1. The intermediate buffer space required by the server is minimized.
2. Better load balancing is achieved due to finer scheduling granularity.
3. The disk bandwidth fragmentation is alleviated due to better bandwidth allocation.

It will be seen that a video server using the proposed rate staggering method is able to provide feasible solutions to some stream requests which cannot be met otherwise. The system throughput can thus be increased. Note that the proposed rate staggering approach can be used together with time staggering to lead to further performance improvement. More specifically, it is shown that the rate staggering method is statically optimal when it is used with the concept of time staggering<sup>1</sup>. To the best of our knowledge, despite of its importance, there is no prior work on exploring the approach of rate staggering to deal with scalable video in the disk array. The imminent need of scalable video in many increasingly attractive multimedia applications justifies the importance and timeliness of this study. In addition, we also conduct several simulations for various applications. We measure the performance of our simulation model from both the viewpoints of the user end and the server end. These experimental results show that the rate staggering method can significantly improve the performance of a VOD system.

<sup>1</sup>The definition of static optimality is given in Section 3.3.

This paper is organized as follows. The method of using rate staggering to store scalable video data is described in Section 2. Section 3 illustrates in detail the advantages of the proposed rate staggering. This paper concludes with Section 4.

## II. USING RATE STAGGERING TO STORE SCALABLE VIDEO DATA

This section describes the proposed rate staggering method. Let  $r$  be the number of different classes of video the server can provide. Using subband coding, the number  $r$  depends on the number of frequency bands partitioned and is usually flexible. The whole video data is divided into  $r$  partitions, called rate 1 data, rate 2 data, ..., and rate  $r$  data. The lowest quality video, referred to as class 1 video, requires only rate 1 data for playout. The second to the lowest quality video, referred to as class 2 video, requires both rate 1 and rate 2 data for playout. In general, class  $i$  video requires all rate  $j$  data,  $1 \leq j \leq i$ , for playout. Let  $P$  (in byte/second) be the playout speed for the decoder to play out the full resolution video and  $T$  be the one round retrieval time by the disk array. Then, using the double buffering method (i.e., the buffer space is chosen to be twice as that needed to accommodate the data retrieved in one round), the buffer space required by the server for a full resolution stream is equal to  $2TP$ . It is noted that, as pointed out in [14], a larger block size will lead to a higher disk throughput<sup>2</sup>, showing a trade-off between the server buffer space required and the system throughput. The choice of block size is system dependent and will not be discussed in this paper. For ease of exposition, we assume that all blocks have the same block size,  $b$  (in bytes), in what follows<sup>3</sup>.

Let  $k$  be the displacement factor for staggering data blocks of different rates in the disk array. For example, the displacement factor in Table II is two. Clearly,  $\lceil \frac{TP}{b} \rceil$  is the number of data blocks needed by a full resolution video stream within the time duration  $T$ . Hence, in order to achieve load balancing among disks, the displacement factor  $k$  is designed to be  $\lceil \frac{TP}{br} \rceil$ . As can be seen later, such a placement by rate staggering can spread the workload of each stream evenly across disks.

Denote the buffer size of the end decoder as  $B_D$ . It is noted that the maximal amount of data the end decoder can retrieve at a time is half of its total buffer size

<sup>2</sup>The choice of a larger storage unit can amortize the disk arm positioning overheads over a larger read time.

<sup>3</sup>The assumption for blocks containing data of different rates to have the same size is deemed reasonable under subband encoding. However, more provision is needed to justify this assumption under MPEG encoding [8].

(assuming that the other half is being used for playout).  $B_D$  thus has to be greater than or equal to  $2 \times \lceil \frac{TP}{b} \rceil \times b$ . The data placement for scalable video in a disk array of  $n$  disks can be determined by Procedure R below.

Procedure R: Data placement for scalable video in a disk array of  $n$  disks.

Step 1: Determine the displacement factor  $k = \lceil \frac{TP}{br} \rceil$ .

Step 2: Place block  $B_{i,j}$  in disk  $d(i,j)$ , where

$$d(i,j) = [(j-1)k + j]_n$$

It can be verified from Table II that with  $n = 8$  and  $k = 2$ ,  $B_{3,2}$  resides in disk  $d(3,2) = [2+3]_8 = 5$  and  $B_{5,4}$  resides in disk  $d(5,4) = [6+5]_8 = 3$ .

## III. ADVANTAGES OF RATE STAGGERING

In this section, we describe the advantages of using rate staggering to store scalable video data in the disk-array-based video server. The advantages include smaller buffer space required by the server (described in Section 3.1), better load balancing (described in Section 3.2) and less disk bandwidth fragmentation (described in Section 3.3).

### A. Smaller Buffer Space Required by the Server

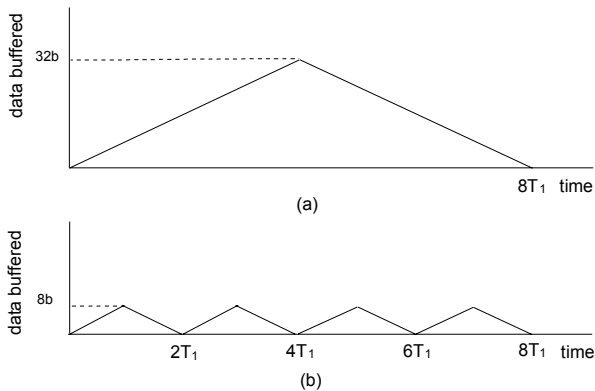
Consider the video data placement given in Table I. For ease of presentation, consider the scenario of serving one video stream and also assume the time required for each disk retrieval is proportional to the amount of data retrieved. Let the time required for one disk to retrieve one data block of size  $b$  be  $T_l$ . Suppose that the previously mentioned double buffering method is used and that the decoder of an end player, with a local buffer size  $16b$ , requires the playout speed  $8b/T_l$  to play the full resolution video. It can be seen that for the data placement in Table I, one has to retrieve all 32 blocks in one round of retrieval (with the duration  $4T_l$ ), which would require the available buffer size of  $2 \times 32b = 64b$  at the server. Note that the server can only send  $8b$  data to the end player at a time due to the limited buffer size of the decoder (i.e.,  $16b$ ). Given the placement in Table I, the server has to retrieve all 32 blocks (4 blocks from each disk) in order to extract 8 blocks to send to the decoder for playout.

On the other hand, given the data placement in Table II, one only has to retrieve 8 blocks in one round of retrieval (with the duration  $T_l$ ), as shown in Table III, which requires the buffer size of  $2 \times 8b = 16b$  at the server, only a quarter of that required by the case without rate staggering. Illustration for the amount of data buffered in half of the buffer space is shown in Figure 2. It is noted that the above request from a decoder cannot be satisfied by a server with an available buffer space within the range  $[16b, 64b)$ , unless the technique of rate staggering is

employed. It can be seen that this advantage of using rate staggering still holds when multiple streams are considered.

**TABLE III**  
FOUR ROUNDS OF FULL RESOLUTION DATA RETRIEVAL IN A DISK ARRAY WITH RATE STAGGERING.

Disk No.	1	2	3	4	5	6	7	8
Rate1	B <sub>1,1</sub>	B <sub>2,1</sub>	B <sub>1,2</sub>	B <sub>2,2</sub>	B <sub>1,3</sub>	B <sub>2,3</sub>	B <sub>1,4</sub>	B <sub>2,4</sub>
Rate2	B <sub>3,4</sub>	B <sub>4,4</sub>	B <sub>3,1</sub>	B <sub>4,1</sub>	B <sub>3,2</sub>	B <sub>4,2</sub>	B <sub>3,3</sub>	B <sub>4,3</sub>
Rate 3	B <sub>5,3</sub>	B <sub>6,3</sub>	B <sub>5,4</sub>	B <sub>6,4</sub>	B <sub>5,1</sub>	B <sub>6,1</sub>	B <sub>5,2</sub>	B <sub>6,2</sub>
Rate 4	B <sub>7,2</sub>	B <sub>8,2</sub>	B <sub>7,3</sub>	B <sub>8,3</sub>	B <sub>7,4</sub>	B <sub>8,4</sub>	B <sub>7,1</sub>	B <sub>8,1</sub>



**Fig. 2. Illustration for the amount of data buffered in half of the buffer space: (a) without rate staggering and (b) with rate staggering.**

**TABLE IV**  
DATA RETRIEVAL FOR CLASS 2 SCALABLE VIDEO.

Disk No.	1	2	3	4	5	6	7	8
Rate1	B <sub>1,1</sub>	B <sub>2,1</sub>	B <sub>1,2</sub>	B <sub>2,2</sub>	-	-	-	-
Rate2	-	-	B <sub>3,1</sub>	B <sub>4,1</sub>	B <sub>3,2</sub>	B <sub>4,2</sub>	-	-
Rate 3	-	-	-	-	B <sub>5,1</sub>	B <sub>6,1</sub>	B <sub>5,2</sub>	B <sub>6,2</sub>
Rate 4	B <sub>7,2</sub>	B <sub>8,2</sub>	-	-	-	-	B <sub>7,1</sub>	B <sub>8,1</sub>

**TABLE V**  
DELAYING THE STARTING TIME OF STREAM C TO AVOID BANDWIDTH FRAGMENTATION..

Disk No.	1	2	3	4	5	6	7	8
Rate1	B <sub>1,1</sub>	B <sub>2,1</sub>	B <sub>1,2</sub>	B <sub>2,2</sub>	-	-	-	-
Rate2	C <sub>1,1</sub>	C <sub>2,1</sub>	B <sub>3,1</sub>	B <sub>4,1</sub>	B <sub>3,2</sub>	B <sub>4,2</sub>	-	-
Rate 3	-	-	C <sub>3,1</sub>	C <sub>4,1</sub>	B <sub>5,1</sub>	B <sub>6,1</sub>	B <sub>5,2</sub>	B <sub>6,2</sub>
Rate 4	B <sub>7,2</sub>	B <sub>8,2</sub>	-	-	C <sub>5,1</sub>	C <sub>6,1</sub>	B <sub>7,1</sub>	B <sub>8,1</sub>

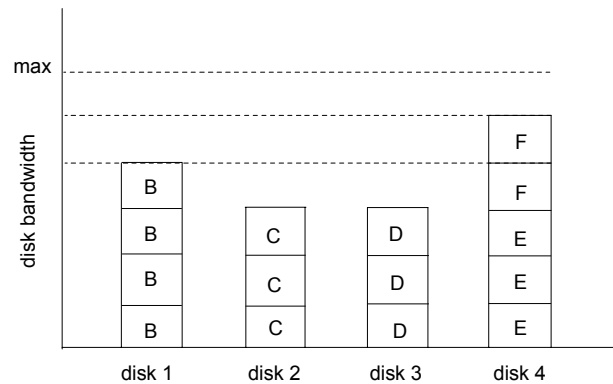
### B. Better Load Balancing

The finer storage granularity provided by rate staggering leads to better load balancing among disks. An illustrative example for data retrieval for class 2 scalable video using the rate staggering method is shown in Table IV, where due to rate staggering, the workload incurred by stream *B* is spread across 4 disks, instead of 2 disks otherwise. More importantly, in a multi-user environment better load balancing can be achieved by proper delaying

the start of a newly requested video so as to avoid bandwidth fragmentation. An illustration for delaying the starting time of stream *C* from Round 1 to Round 2 to avoid bandwidth fragmentation is shown in Table V. It is worth mentioning that such a time staggering method is previously known and can in fact be used without rate staggering. However, the finer storage granularity provided by rate staggering can certainly lead to more benefit from this time staggering method.

### C. Less Disk Bandwidth Fragmentation

A video server using the rate staggering method can provide feasible solutions to some stream requests which cannot be otherwise satisfied, thus increasing the system throughput. In addition to the scenarios described before, consider the problem of disk bandwidth fragmentation in a multi-user environment. Disk bandwidth fragmentation means that after a sequence of bandwidth allocation and deallocation, the available bandwidth in each disk does not suffice to accommodate an incoming request even there is an enough amount of aggregate bandwidth. Figure 3 shows a snapshot for the bandwidth used by each disk, where the disk retrieval is staggered by time but not by rate<sup>4</sup>. Suppose that stream *B* in Figure 3 is a full resolution video (i.e.,  $r = 4$ ). Without rate staggering, a video server with the workload described in Figure 3 has to reject any incoming request requiring a full resolution video. In contrast, Figure 4 shows the disk bandwidth allocation using both time and rate staggering for the same existing streams as in Figure 3. It can be seen that the video server with the configuration in Figure 4 is able to accommodate two more full resolution video streams, leading a higher system throughput than that in Figure 3.



**Fig. 3. Disk bandwidth allocation with time staggering, but without rate staggering.**

Let  $N$  be the number of total blocks that a disk array can retrieve in one round (i.e., within the time duration  $T$ ).

<sup>4</sup>Note that the 4 *B* blocks in Figure 3 denote 4 blocks from video stream *B* for the same video clip but different data rates (e.g.,  $B_{1,1}$ ,  $B_{1,2}$ ,  $B_{1,3}$  and  $B_{1,4}$ ). The same explanation applies to all other streams. Also, as a result of time staggering, the number of blocks retrieved by each disk (i.e., (4, 3, 3, 5) in Figure 3) will rotate as time advances..

A bandwidth allocation scheme is called *statically optimal* if a video server using that scheme can accommodate any incoming request sequence for streams  $\{s_i\}$ ,  $1 \leq i \leq h$ , if and only if  $\left\lceil \frac{TP}{br} \right\rceil \sum_{i=1}^h c_i \leq N$ , where  $c_i$  is the class of video requested by stream  $s_i$ . Then, omitting its straightforward proof, we have the following theorem.

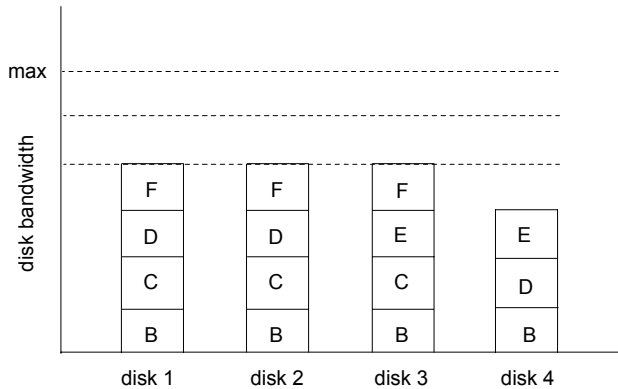


Fig. 4. Disk bandwidth allocation with both time and rate staggering.

**Theorem 1:** The proposed rate staggering is statically optimal when it is used with the concept of time staggering.

Clearly, the static optimality does not hold for the scheme associated with Figure 3, where without rate staggering an incoming stream for a full resolution video (i.e., 4 blocks) cannot be granted despite that  $\left\lceil \frac{TP}{4b} \right\rceil = 1$  and  $N = 24 > 15 + 4$ .

IV. PERFORMANCE RESULTS

The simulation model is described in Section 4.1 and several experiments are conducted in Section 4.2 to

examine the performance of rate staggering and time staggering schemes.

A. System Model

In our empirical studies, only the data placement methods of the two discussed systems, i.e., rate staggering systems and time staggering systems, vary in each experiment so that we can comparatively analyze these two methods. We also propose a new data placement method for comparison purposes, which place the blocks of videos randomly in the disk array. This method is referred to as a non-staggering one in what follows.

The system model used in our simulation is shown in Figure 5. The server model is shown in Figure 5(b) where the server controller gets requests from clients. Note that each request retrieves a segment of video, i.e., a request for a class  $i$  video needs to read  $i$  blocks from the disks. Then, the controller dispatches the read-commands to corresponding disks. Those commands will store in the *command queue* of each disk. If the queue is already full

TABLE VI  
THE DESCRIPTION OF PARAMETERS USED IN THE SIMULATION

Parameter Name	Description
BLOCK_SIZE	The data size of each band in a segment of video
BUFFER_SIZE	The size of buffer in the user side
DISK_NUM	Number of disks in the disk array
DISK_BW	The read bandwidth of each disk in server
PLAY_INT	The playing time of each segment of the video
QUEUE_SIZE	The size of command queue in each disk
VIDEO_CLASS	The class of video that users require

while a new command is being entered, the server will send a "reject" message to the user who issues the request and the user will exist the system. After all blocks needed in the request have been collected, the server controller sends the data back to the user. The simulation model in the user side is shown in Figure 5(a). The controller of a user sends requests to the VOD server. When the data returns, the user controller stores the data in the proper

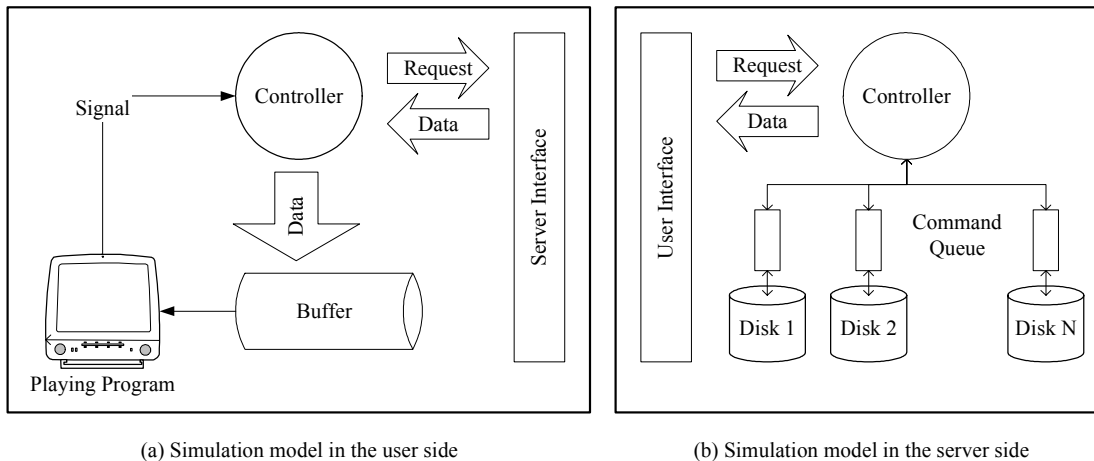


Fig. 5. The VOD system in simulation

buffer slot. After filling up the buffer, the playing program will start to read the data at the first buffer slot and play it out. Then, when playing out the first segment, the controller will send the next request to the server while the playing program grabs the next buffer slot to play. If the next buffer slot is unavailable, a buffer underflow error will be triggered. Then, the playing

**TABLE VII**  
THE PARAMETERS OF THE SIMULATION MODEL USED IN EXPERIMENT 1 AND EXPERIMENT 2.

Parameter Name	Value
BLOCK_SIZE	80 K Byte
BUFFER_SIZE	2 segments of video
DISK_NUM	16
DISK_BW	10000 KBps
PLAY_INT	500 ms
QUEUE_SIZE	Unlimited
VIDEO_CLASS	Random: Uniform Distributed.

program will stop playing and wait until the data of the video segment returns.

A description of the parameters used in the simulation system is given in Table VI.

### B. Simulation Results

We have conducted several experiments to evaluate the performance of system for the three staggering VOD systems.

#### Experiment 1: The Performance Benchmarks from Users' Viewpoint

In the first experiment, several benchmarks from the viewpoint of users are considered while the number of users varies. The parameters used in the first experiment are summarized in Table VII. Note that since we want to trace the performance of the system as the number of users varies in this experiment, we do not limit the size of command queue in each disk.

Due to the limitation of the bandwidth of disks of the server, the theoretical maximum number of users that this system is able to support, however, is 400, as obtained by the following formula.

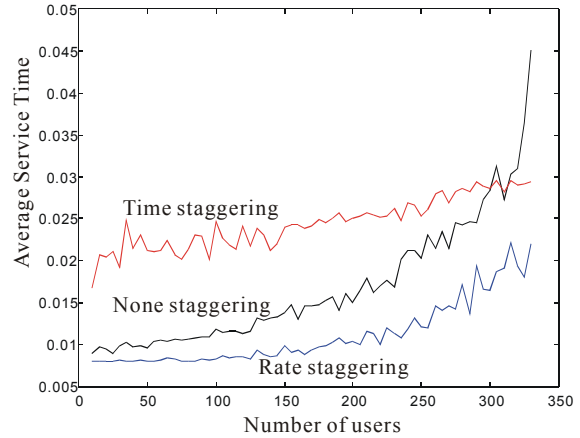
$$\text{Average bandwidth of each user} = \frac{1+2+3+4}{4} \times 80 = 0.4 \text{ KB/ms} = 400 \text{ KB/s}$$

$$\text{Maximum number of users} = \frac{\text{Total disk bandwidth}}{\text{Average bandwidth per user}} = \frac{16 \times 10000}{400} = 400$$

First, the average service times of the three systems are shown in Figure 6. As described in previous sections, by utilizing the rate staggering placement, the server can reduce the minimum service time of a request to only  $\frac{1}{(i-1)}$  times of that by the time staggering placement.

Thus, it follows that the average service time of a rate

staggering system will be around 12 ms, i.e.,  $\frac{(0+1+2+3)}{4} \times \frac{80}{10000} = 12 \text{ ms}$ , less than that in a time staggering system, which is consistent with the results shown in Figure 6.



**Fig. 6. Average service time vs. number of users.**

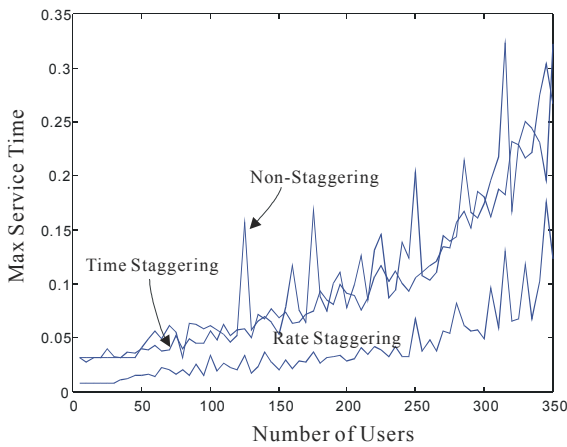
It is interesting to see that as the number of users increases, the average service time of a non-staggering system grows faster than that of a time staggering system. When the number of users is close to the theoretical upper bound, the average service time of a non-staggering system will grow beyond that of a time staggering system. However, as we shall see in the following experiments, the capacity of a time staggering system is still better than that of a non-staggering system. The reason of this phenomenon is that a non-staggering system has the same property as a rate staggering system, i.e., it may allocate different blocks of the same segment of videos into different disks, thereby having a smaller service time of one request. However, a non-staggering system suffers seriously from the unbalance of disk load, which becomes even more severe as the number of users increases.

As shown in Figure 7, the maximum service times of the non-staggering system and time staggering system are similar whereas the curve of a non-staggering system is saw-toothed. The rate staggering system still has the best maximum service time among them.

The average delay jitters of the three systems are shown in Figure 8(a), where it is seen that the delay jitter of a non-staggering system is much larger than those of the two staggering systems. It is because that the blocks of videos are randomly allocated in the disks, and it is possible that one request needs to retrieve data from a very busy disk while the others do not. The non-staggering system suffers from its unbalance of disk load again. As shown in Figure (b), a rate staggering system still incurs the smallest delay jitter among the three systems.

**Experiment 2: The Performance Benchmarks from the Server's Viewpoint**

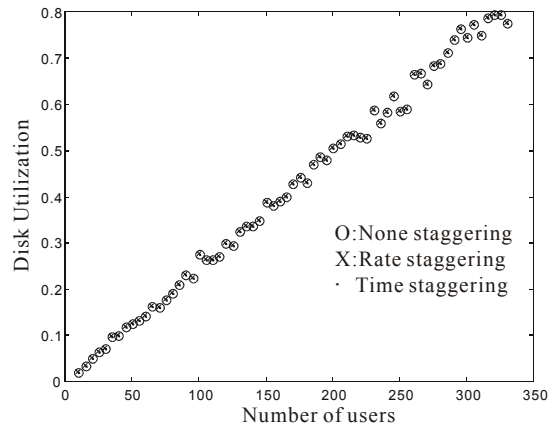
In this experiment, the performance benchmarks of the server side are concerned. The parameters used are same as those in the previous experiment. The relationship between the disk utilization ratio and the number of users is shown in Figure 9. As shown in Figure 9, all of the three systems have the same values for all points. This can be explained by the reason that the access patterns and the number of users are all the same in these three systems. Consequently, they will read the same number of blocks, and the average disk utilization ratios are hence the same.



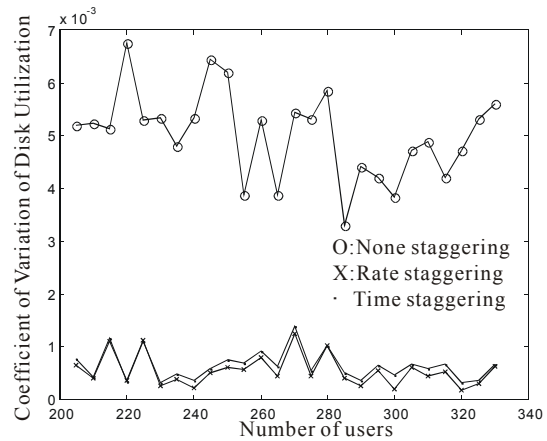
**Fig. 7. The max service time v.s. number of users.**

Another important benchmark related to disk utilization is the balance of the disk usage. The coefficients of utilization variation of all disks are used as the benchmark indexes for load balance. As shown in Figure 10, the rate staggering system incurs a better load balance than the time staggering system. Also note that the two staggering systems have much more balanced loads than the non-staggering system since the two staggering systems will allocate different segments of videos into disks

sequentially. Thus, the workload for the disk array is distributed more uniformly among the disks. The load balance is an important index of the capacity of a system because the capacity in a less balanced system is limited by the bandwidth of the busiest disk.

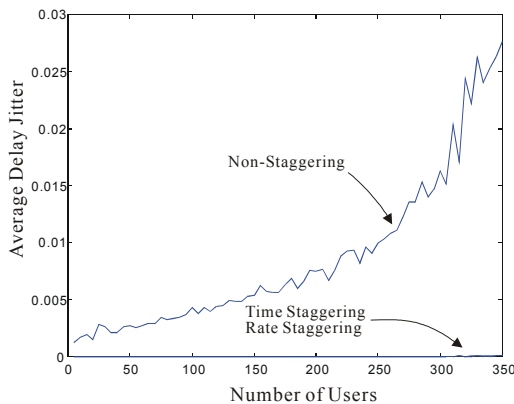


**Fig. 9. Disk utilization vs. number of users**

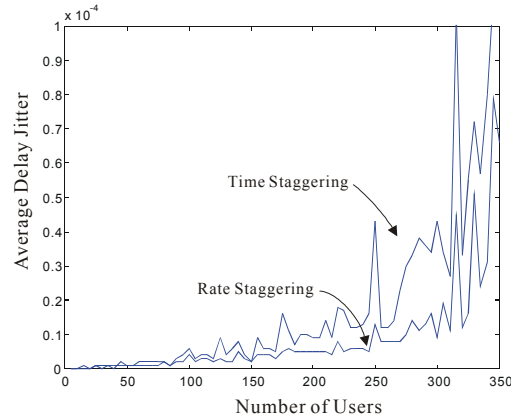


**Fig. 10. The coefficient of variation of disk utilization vs. users number.**

The average number of waiting commands in the command queue of each disk is shown in Figure 11. This



(a) Macrocosm



(b) Microcosm

**Fig. 8. Delay jitter v.s. the number of users**

index will dominate the service time of a request when the number of users increases. As shown in Figure 11, when serving few users, a non-staggering system has a smaller waiting number than a time staggering system. However, as the number of users is close to the theoretical bound, the waiting number of a non-staggering system will grow beyond that of a time staggering system, leading to the same scenario as in the experiment of the average service time.

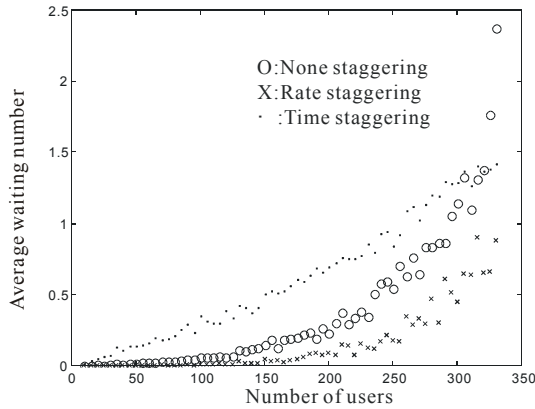


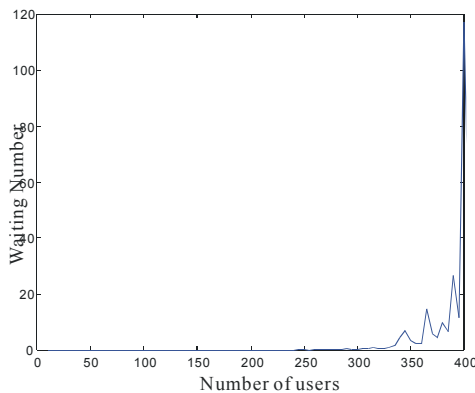
Fig. 11. Average waiting number of the command queue in each disks vs. users number.

As the number of users approaches to the theoretical upper bound, these indexes of performance fall rapidly.

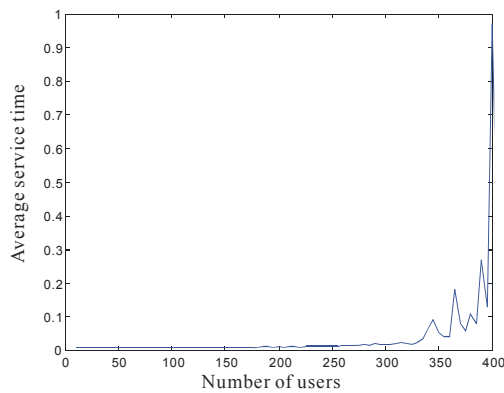
TABLE VIII  
THE PARAMETERS OF SIMULATION MODEL IN EXPERIMENT 2.

Parameter Name	Value
BLOCK_SIZE	80 K Byte
BUFFER_SIZE	2 segments of video
DISK_NUM	16
DISK_BW	10000 KBps
PLAY_INT	500 ms
QUEUE_SIZE	5
VIDEO_CLASS	4
USER_ARRIVAL	Poisson process with mean =5

The systems also become unstable and are out of the normal operation range of a VOD server as shown in



(a). Average waiting number vs. Number of users



(b). Average service time vs. Number of users

Fig. 12. The theoretical upper bound limitation.

Figure 12. It can be seen that both the average service time and the waiting number in the command queue grow rapidly.

**Experiment 3: The Capacity of VOD Servers**

In this experiment, the capacities of the three systems are comparatively analyzed. Here, the users are generated by the Poisson arrival scheme. When the number of users approaches to the upper bound of the system, the server starts to reject some of them due to the limited disk bandwidth and limited size of command queue. We run the simulation long enough for the system to reach a stable state so that we can obtain the maximum number of users that can be supported of each system. The simulation results are shown in Figure 13. The parameters used in this experiment are summarized in Table VIII.

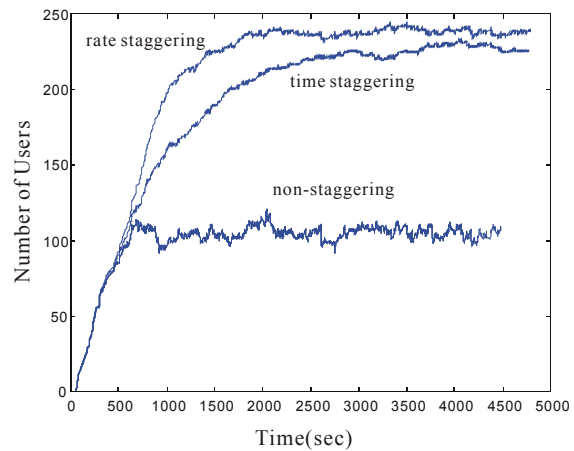


Fig. 13. The number of users vs. time for class 4 videos.

In this experiment, we assume all users require videos of the same class, i.e., class 4, thus preventing the system from only rejecting users requiring videos of higher classes. We can obtain the upper bound of the number of users in this system as 250 by the following formulas.



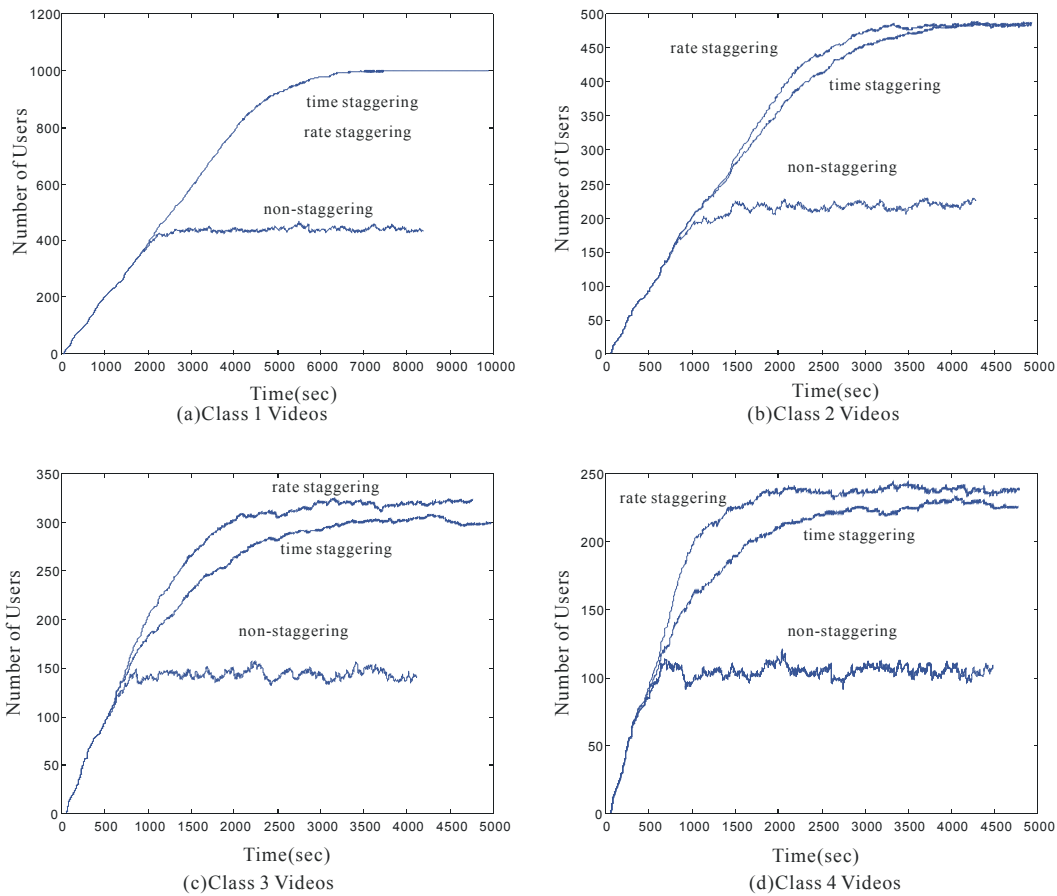


Fig. 14. The number of users vs. time for all classes videos.

$$\text{Average bandwidth of each user} = \frac{4 \times 80}{500} = 0.64$$

$$\text{KB/ms} = 640 \text{ KB/s}$$

$$\begin{aligned} \text{Maximum number of users} &= \frac{\text{Total disk bandwidth}}{\text{Average bandwidth per user}} \\ &= \frac{16 \times 10000}{640} = 250 \end{aligned}$$

The simulation results are shown in Figure 13 where it can be seen that the capacity of a rate staggering system is much higher than that of a time staggering system since a rate staggering system has less disk bandwidth fragmentation. In this simulation, from 200 seconds to 700 seconds, the average number of users of the three systems are 239, 228 and 105 respectively, i.e., a rate staggering system can provide another 4.8% capacity over a time staggering system for class 4 videos. The capacities of users for different classes videos are shown in Figure 14 where the rate staggering system outperforms other systems especially for higher classes videos. The reason behind is that in a rate staggering system, the server can reduce the minimal service time of a request for a class  $i$  video to  $\frac{1}{(i-1)}$  times of that in a time staggering system.

Therefore, a rate staggering system performs better

especially for a higher classes video. As a special case, for class 1 videos, the data placement method of a rate staggering system is reduced to that of a time staggering system, accounting for the results shown in Figure 14(a).

In Figure 15, the capacities of the three VOD systems are shown as a function of the size of command queue of each disk. When the size of each command queue increases, the system will have a higher capacity. Thus, this is a trade-off between the performance and the memory space of a VOD system.

#### Experiment 4: The Size of Buffer Needed in the VOD Server

In some systems, the device at the user ends may not be as powerful as personal computers. For example, the size of buffer may be limited due to the device cost. However, using a low bandwidth devices, e.g., a CD-ROM array, as its storage system, a system will cause a quite longer service time because of a longer time needed to read data. Consequently, we need a larger buffer in clients in order to play the video fluently.

The system discussed in this experiment has a CD-ROM based storage system, and is designed to provide several HDTV channels. The model is shown in Figure 16. In the new model, each channel is connected to a virtual

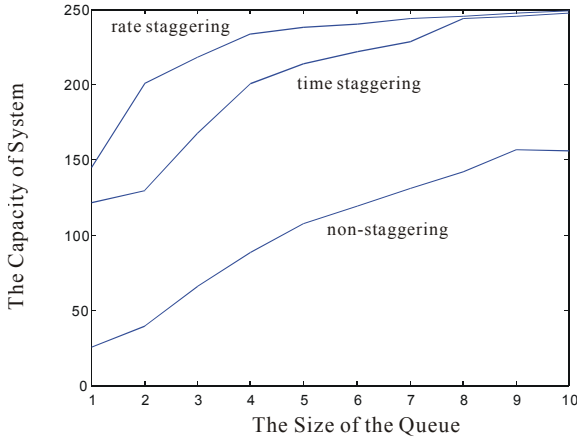


Fig. 15. The capacity vs. the size of the queue in each disk.

user who receives the video streams from the server. As such, we can model the HDTV system as the VOD system described in previous experiments. In the new system, after receiving the video data, the virtual users send the data to the corresponding channels and then broadcast them to the real users. The parameters used in these experiments are summarized in Table IX. The theoretical upper bound of the number of channels, limited by the bandwidth of the system, is 20, which is obtained by the following formula.

TABLE IX  
THE PARAMETERS OF SIMULATION MODEL IN EXPERIMENT 3..

Parameter Name	Value
BLOCK_SIZE	500 K Byte
DISK_NUM	16
DISK_BW	5000 Kbps
PLAY_INT	500 ms
QUEUE_SIZE	Unlimited
VIDEO_CLASS	4

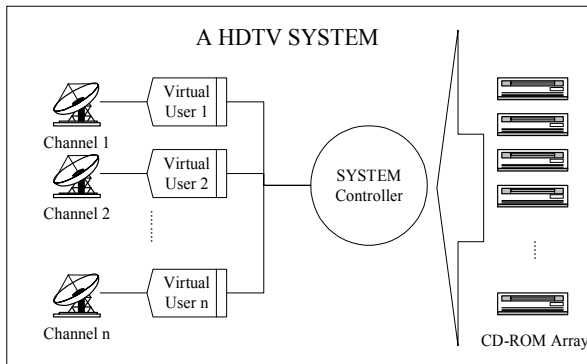


Fig. 16. A modified simulation model.

$$\begin{aligned} \text{Number of Channels} &\leq \frac{\text{Total Bandwidth of the System}}{\text{Average Bandwidth of Each Channel}} \\ &= \frac{16 \times 5000}{\frac{500 \times 4}{0.5}} = 20 \end{aligned}$$

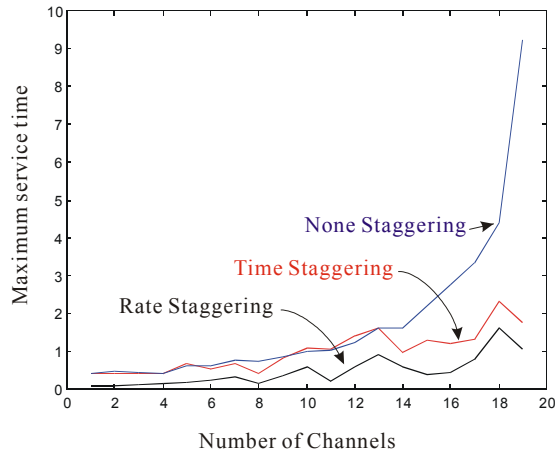


Fig. 17. The maximum service time vs. the number of channels.

The buffer size needed in the system is decided by the maximum service time of all requests as shown in the following formula.

$$\text{BUFFER\_SIZE} \geq \left\lceil \frac{\text{Maximum Service Time}}{\text{PLAY\_INT}} \right\rceil + 1$$

The simulation results are shown in Figure 17. The maximum service time of a rate staggering system is less than that of a time staggering system or a non-staggering system. The buffer size of each channels is shown in Table X where the buffer size in a rate staggering system is always smaller than or equal to that in a time staggering system, thus showing another advantage of a rate staggering system. The merits of a rate staggering system indeed stem from its load balance by placing different bands of videos at different disks, thus reducing the access time for retrieving data from disks.

V. CONCLUSION

In this paper, a data placement method based on rate staggering to store scalable video data in a disk-array-based video server is proposed and explored. It has been shown that the advantages of the proposed rate staggering method include: (1) minimizing the intermediate buffer space required by the server, (2) achieving better load balancing due to finer scheduling granularity, and (3) alleviating the disk bandwidth fragmentation. These advantages enable a video server using the rate staggering method to provide feasible solutions to some video stream requests which cannot be met otherwise. The system throughput is thus increased. We also conduct several simulations for various applications. These experimental results show that the rate staggering method can significantly improve the performance of a VOD system.

REFERENCE

[1] J. M. Almeida, D. L. Eager, M. Ferris, and M. K. Vernon, Provisioning content distribution networks for streaming media,

- In *Proceedings of IEEE Infocom'02*, pages 1746--1755, June 2002.
- [2] S. Berson, S. Ghandeharizadeh, R. Muntz, and X. Ju. Staggered striping in multimedia information systems. In *Proceedings of ACM SIGMOD*, pages 79-90, May 1994.
- [3] S. W. Carter, S. R., J.-F. Pis, S. Mohan, and D. D. E Long. A dynamic heuristic broadcasting protocol for video-on-demand, *Proc 21 st Int. Conf on Distributed Computing Systems*, pages 657-664, Apr. 2001.
- [4] S.-H. G. Chan and S.-H. I. Yeung. Client buffering techniques for scalable video broadcasting over broadband networks with low user delay. *IEEE Transactions on Broadcasting*, 48:19-26, March 2002.
- [5] E. Chang and A. Zakhor. Scalable video data placement on parallel disk arrays. *IS&T/SPIE International Symp. on Electronic Imaging: Science and Technology*, 2185: Image and Video Database II, February 1994.
- [6] E. Chang and A. Zakhor. Variable bit rate MPEG video storage on parallel disk arrays. In *The 1st International Workshop on Community Networking Integrated Multimedia Services to the Home*, July 1994.
- [7] M.-S. Chen, C.-S. Li, H. Hsiao, and P. S. Yu. Using rotational mirrored declustering for replica placement in a disk-array-based video server. *ACM Multimedia System*, 5(6):371-379, December 1997.
- [8] T. Chiang and D. Anastassiou. Hierarchical coding of digital television. *IEEE Communication*, 32:38-45, May 1994.
- [9] T.-C. Chiueh and R. H. Katz. Multi-resolution video representation for parallel disk arrays. In *Proceedings of ACM Multimedia*, pages 401-409, August 1993.
- [10] G. R. Ganger, B. L. Worthington, R. Y. Hou, and Y. N. Patt. Disk arrays: High-performance, high-reliability storage subsystems. *IEEE Computer*, pages 30-37, March 1994.
- [11] K. Gopalan, T.-C. Chiueh, and Y.-J. Lin. Delay Budget Partitioning to Maximize Network Resource Usage Efficiency, In *Proc. of INFOCOM 2004*, Hong Kong, China, March 2004.
- [12] K. Gopalan and T.-C. Chiueh. Multi-Resource Allocation and Scheduling for Periodic Soft Real-Time Applications, In *Proc. of ACM/SPIE Multimedia Computing and Networking (MMC/N2002)*, San Jose, CA, January 2002.
- [13] W. I. Grosky. Multimedia information systems. *IEEE Multimedia*, 1:12-24, 1994.
- [14] D. D. Kandlur, M. Chen, and Z.-Y. Shae. Design of a multimedia storage server. In *Proc. IS&T/SPIE Symposium on Electronic Imaging - Conference on High Speed Networking and Multimedia Applications*, February 1994.
- [15] R. Katz, G. Gibson, and D. Patterson. Disk system architectures for high performance computing. In *Proceedings of the IEEE*, volume 77, pages 1842-1858, December 1989.
- [16] J. Y. B. Lee. On a unified architecture for video-on-demand services. *IEEE Transactions on Multimedia*, 4(1):38-47, March 2002.
- [17] W. Li. Scalable video coding with fine granularity scalability. In *Proc. Of International Conference on Consumer Electronics*, pages 306-307, 1999.
- [18] C.-L. Liu, D. H. C. Du, S. S. Y. Shim, J. Hsieh, and M. Lin. Design and evaluation of a generic software architecture for on-demand video servers. *IEEE Transactions on Knowledge and Data Engineering*, pages 406-424, May-June 1999.
- [19] S. Ramanathan and P. V. Rangan. Architectures for personalized multimedia. *IEEE Multimedia*, 1:37-46, 1994.
- [20] A. L. N. Reddy and J. Wyllie. I/O issues in a multimedia system. *IEEE Computer*, pages 69-74, March 1994.
- [21] M. Reisslein, K. W. Ross, and S. Shrestha. Striping for interactive video: Is it worth it? In *IEEE International Conference on Multimedia Computing and Systems*, pages 635-640, 1999.
- [22] P. Shenoy and H. M. Vin. Efficient striping techniques for multimedia file servers. In *Proceedings of the IEEE 7th*

*International Workshop on Network and Operating System Support for Digital Audio and Video*, pages 25-36, 1997.

- [23] D. A. Tran, K. A. Hua, and S. Sheu. A new caching architecture for efficient video services on the internet, In *IEEE Symposium on Applications and the Internet*, Orlando, FL, USA, 2003. 29.
- [24] S.-L. Tsao and Y.-M. Huang. An efficient storage server in near video-on-demand systems. *IEEE Transactions on Consumer Electronics*, pages 27-32, February 1998.
- [25] H. M. Vin and P. V. Rangan. Designing a multi-user HDTV storage server. *IEEE Journal on Selected Areas in Communication*, 11:153-164, January 1993.
- [26] Y. Wang, J. C. L. Liu, D. H. C. Du, and J. Hsieh. Video file allocation over disk arrays for video-on-demand. In *International Conference on Multimedia Computing and Systems*, pages 160-163, 1996.

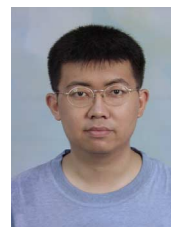


**Xin-Mao Huang** received his MS. degree from the electrical engineering department of National Taiwan University, Taipei, Taiwan, in 1998. He is currently a Ph.D. student in the same department. His research interests include data mining, distributed system, and multimedia applications.



**Ming-Syan Chen** received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, and the M.S. and Ph.D. degrees in Computer, Information and Control Engineering from The University of Michigan, Ann Arbor, MI, USA, in 1985 and 1988, respectively. Dr. Chen is currently a professor and the chairman of Graduate Institute of Communication Engineering, a professor in EE Department, and also a professor in CSIE Department, National Taiwan University, Taipei, Taiwan. He was a research staff member at IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA from 1988 to 1996. His research interests include database systems, mobile computing systems, and multimedia networking, and he has published more than 170 papers in his research areas.

In addition to serving as program committee members in many conferences, Dr. Chen served as an associate editor of *IEEE Transactions on Knowledge and Data Engineering (TKDE)* from 1997 to 2001, is currently on the editorial board of *Very Large Data Base (VLDB) Journal*, *Knowledge and Information Systems (KAIS) Journal*, *Journal of Information Science and Engineering (JISE)*, and *Journal of the Chinese Institute of Electrical Engineering*, and is a Distinguished Visitor of *IEEE Computer Society for Asia-Pacific* from 1998 to 2000. He served as program chairs/co-chairs and tutorial speakers in many conferences. He holds, or has applied for, eighteen U.S. patents and seven ROC patents in his research areas. He is a recipient of the NSC (National Science Council) Distinguished Research Award and K.-T. Li Research Penetration Award for his research work, and also the Outstanding Innovation Award from IBM Corporate for his contribution to a major database product. Dr. Chen is a Fellow of IEEE and a member of ACM.



**Cheng-Ru Lin** received his B.S. and Ph.D. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1999 and 2003 respectively. His research interests include data mining, distributed system, network, and multimedia applications. He is currently a researcher at Arcadyan Advanced Technology Center.