

DESIGN AND PERFORMANCE STUDY OF SCALABLE VIDEO STORAGE IN A DISK-ARRAY-BASED VIDEO SERVER

Zheng-Ru Lin and Ming-Syan Chen

Electrical Engineering Department
National Taiwan University
Taipei, Taiwan, ROC
{owenlin,mschen}@arbor.ee.ntu.edu.tw

ABSTRACT

This paper explores 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 noted that the advantages of the proposed rate staggering method enable a video server to provide feasible solutions to some video stream requests which cannot be met otherwise. The system throughput can thus be increased. Performance studies on the methods proposed are conducted. It is empirically shown that a VOD system employing rate staggering can significantly outperform a system employing time staggering for the playout of scalable video.

.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 has created several new ventures [4] [10] [16]. 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 [13] [12]. Disk arrays are employed to provide the disk bandwidth required for a video server [8] [14] [17] [18]. In fact, it is highly desirable to use disk striping in a disk array to handle the storage and retrieval of video data [1] [5] [8] [14] [15]. 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 [1]. This is referred to as time staggering.

Multi-resolution coding, referring to the encoding

technique which accommodates at least two resolutions in a video stream, is able to provide scalable video [6] [11]. 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 [2] [3] [7]. 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 multi-window 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 [6]. For example, by dividing the frequency domain into four regions, one can decompose the original video into four sets, say R_1 , R_2 , R_3 and R_4 , where R_i is called rate i data. Then, R_1 corresponds to the basic video (class 1), R_1+R_2 corresponds to class 2 video, $R_1+R_2+R_3$ corresponds to class 3 video, and $R_1+R_2+R_3+R_4$ corresponds to the full 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 block, which is composed of a sequence of frames. An example for the conventional scalable video placement, which has disk striping but not rate staggering, is shown in Table 1, 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 2. Performance studies on the methods proposed are conducted.

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

Table 1: Scalable video with disk striping (no rate staggering).

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

Table 2: Scalable video placement with rate staggering.

It will be seen that a video server using the proposed rate staggering method significantly outperforms the one using the conventional time staggering in that the former 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.

This paper is organized as follows. The method of using rate staggering to store scalable video data is described in Section 2. Performance studies are conducted in Section 3. 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 playback. The second to the lowest quality video, referred to as class 2 video, requires both rate 1 and rate 2 data for playback. In general, class i video requires all rate j data, $1 \leq j \leq i$, for playback. Let P (in byte/second) be the playback 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 $2TP$. Note that a larger block size will lead to a higher disk throughput, since the choice of a larger storage unit can amortize the disk arm positioning overheads over a larger read time, 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.

Let k be the displacement factor for staggering data blocks of different rates in the disk array. For example, the displacement factor in Table 2 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 (assuming that the other half is being used for playback). B_D thus has to be greater than or equal to $2\lceil \frac{TP}{b} \rceil b$. 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

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

It can be verified from Table 2 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 $d(5,4) = [6 + 5]_8 = 3$.

Consider the video data placement given in Table 1. 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_1 . 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 playback speed $8b/T_1$ to play the full resolution video. It can be seen that for the data placement in Table 1, one has to retrieve all 32 blocks in one round of retrieval (with the duration $4T_1$), which would require the available buffer size of $2 * 32b = 64b$ 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 1, 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 playback.

On the other hand, given the data placement in Table 2, one only has to retrieve 8 blocks in one round of retrieval (with the duration T_1), as shown in Table 3, which requires the buffer size of $2 * 8b = 16b$ at the server, only a quarter of that required by the case without rate staggering. 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.

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

Table 3: Four rounds of full resolution with rate staggering.

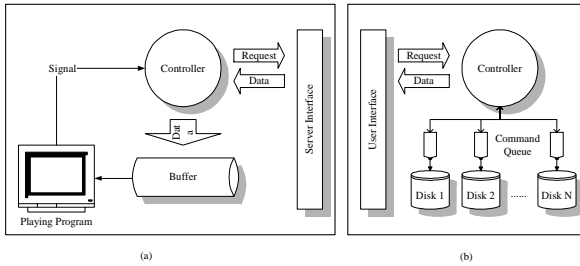


Fig. 1. The VOD system model of the simulation. (a) user side (b) server side

III. PERFORMANCE RESULTS

The simulation model is described in Section III-A and several experiments are conducted in Section III-B to examine the performance of rate staggering and time staggering schemes.

III-A. System Model

The system model used in our simulation is shown in Figure 1 which consists of two parts of our simulation model, i.e., the user side in Figure 1a and the server side in Figure 1b. In Figure 1a, the controller of a user sends its requests to the VOD server. When the data returned from the server, the user controller stores the data in proper slot of its own buffer. The playing program will then start to read and play the data in the slots in sequence. After playing out a segment, the playing program sends a signal to inform the controller to send the next request to the server. Then, the playing program grabs the next slot of the buffer to play out. If the buffer is empty, an error of buffer underflow occurs, and the playing program will stop playing and wait until the data of the video segment returns from the VOD server. In Figure 1b, the server controller gets requests from users. Each request will retrieve a segment of video. Specifically, a request for a class i video will need to read i blocks from disks, and the controller dispatches the read commands to disks. Those commands are stored in a command queue of each disk. When the queue is full, a reject message will be sent to the requesting user. After all blocks needed by the request are obtained, the server controller sends the data back to the requesting user.

III-B. Performance of Rate Staggering

The finer storage granularity provided by rate staggering leads to better load balancing among disks. 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. 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 benefits from this time staggering method.

In addition, 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

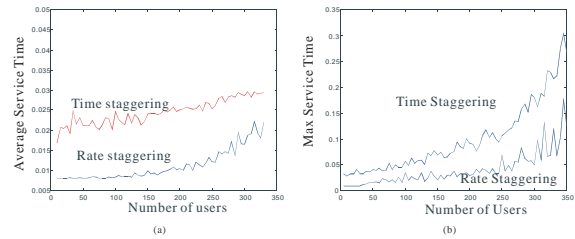


Fig. 2. (a) Average service time vs. number of users. (b) Maximum service time vs number of users

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.

Experiment 1: Service Time of Each User

Parameters	Value
The data size of each block	80 K Byte
The size of buffer	2 segments of video
The size of disk array	16
The read bandwidth of disks	10000 KBps
The playing time of one segment	500 ms
The class of video required	Uniform

Table 4: The parameters of the simulation model.

In the first experiment, an important performance index, service time, is evaluated as a function of the number of users. The parameters used in the first experiment are summarized in Table 4. It is assumed that the size of command queue in each disk is unlimited so that when the number of users increases the system performance can be observed.

Note that the theoretical maximum number of users that the system can support is approximately 400 as estimated by the following formulas.

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

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

It can be seen in Figure 2a that the average service time of the system with rate staggering is much smaller than that of a system with time staggering by a discrepancy of 10 ms. Also, as shown in Figure 2b, the rate staggering system has a better maximum service time than that of the time staggering system.

Experiment 2: Number of Waiting Commands in the Command Queue

The parameters used in this experiment are same as in Table 4. In this experiment, the average number of waiting commands in the command queue of each

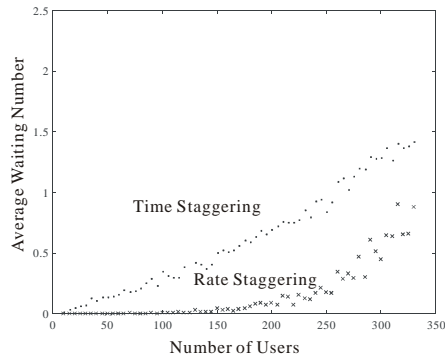


Fig. 3. Average number of waiting commands in the command queue of each disk.

disk is investigated and shown in Figure 3 as a function of the number of users. Note that the number of waiting commands will significantly affect the service time of a request especially when the number of users is large. It is also shown that a VOD system employing rate staggering significantly outperforms a system employing time staggering for the playout of scalable video.

IV. 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 rate staggering method enable a video server to provide feasible solutions to some video stream requests which cannot be met otherwise. The system throughput is thus increased. Performance studies on the methods proposed have been conducted. It was empirically shown that a VOD system employing rate staggering can significantly outperform a system employing time staggering for the playout of scalable video.

V. ACKNOWLEDGEMENT

The authors are supported in part by the Ministry of Education Project No. 89-E-FA06-2-4-7 and the National Science Council, Project No. NSC 89-2219-E-002-007 and NSC 89-2213-E-002-032, Taiwan, Republic of China.

VI. REFERENCES

- [1] S. Berson, S. Ghandeharizadeh, R. Muntz, and X. Ju, Staggered Striping in Multimedia Information Systems, *Proceedings of ACM SIGMOD*, Minneapolis, MN, pp. 79-90, May, 1994.
- [2] E. Chang and A. Zakhor, Scalable Video Data Placement on Parallel Disk Arrays, *IS&T/SPIE International Symp. on Electronic Imaging: Science and Technology*, Vol 2185: Image and Video Database II, February, 1994.
- [3] E. Chang and A. Zakhor, Variable Bit Rate MPEG Video Storage on Parallel Disk Arrays, the 1st International Workshop on Community Networking Integrated Multimedia Services to the Home, July, 1994.

- [4] M.-S. Chen and D. D. Kandlur, Stream Conversion to Support Interaction Video Playout, *IEEE Multimedia*, Vol. 3, No. 2, pp. 55-58, Summer 1996.
- [5] 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*, Vol. 5, No. 6, pp. 371-379, December 1997.
- [6] T. Chiang and D. Anastassiou, Hierarchical Coding of Digital Television, *IEEE Communication*, vol. 32, pp. 38-45, May 1994.
- [7] T.-C. Chiueh and R. H. Katz, Multi-Resolution Video Representation for Parallel Disk Arrays, *Proceedings of ACM Multimedia*, pp. 401-409, August, 1993.
- [8] G. R. Ganger, B. L. Worthington, R. Y. Hou, and Y. N. Patt, Disk Arrays: High-Performance, High-Reliability Storage Subsystems, *IEEE Computer*, pp. 30-37, March, 1994.
- [9] D.J. Gemmel, H. M. Vin, D.D. Kandlur, P. V. Rangan, and L. A. Rowe, Multimedia storage servers: A tutorial, *IEEE MultiMedia* vol.28, no.5, pp. 40-49, May, 1995.
- [10] W. I. Grosky, Multimedia Information Systems, *IEEE Multimedia*, pp. 12-24, Spring, 1994.
- [11] W. Li, Scalable Video Coding with Fine Granularity Scalability, *ICCE. International Conference on Consumer Electronics*, pp. 306-307, 1999.
- [12] 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*, pp. 406-424, May-June 1999.
- [13] A. L. N. Reddy and J. Wyllie, I/O Issues in a Multimedia System, *IEEE Computer*, pp. 69-74, March, 1994.
- [14] M. Reisslein and K. a. S. K. W. Ross, Striping for Interactive Video: Is It Worth It?, *IEEE International Conference on Multimedia Computing and Systems*, pp. 635-640, 1999.
- [15] P. J. Shenoy and H. M. Vin, Efficient Striping Techniques for Multimedia File Servers, *Proceedings of the IEEE 7th International Workshop on Network and Operating System Support for Digital Audio and Video*, pp. 25-36, 1997.
- [16] S.-L. Tsao and Y.-M. Huang, An Efficient Storage Server in Near Video-On-Demand Systems, *IEEE Transactions on Consumer Electronics*, pp. 27-32, February 1998.
- [17] H. M. Vin and P. V. Rangan, Designing a Multi-User HDTV Storage Server, *IEEE Journal on Selected Areas in Communication*, vol. 11, pp. 153-164, January 1993.
- [18] Y. Wang, J. C. L. Liu, D. H. C. Du, and J. Hsieh, Video File Allocation Over Disk Arrays for Video-On-Demand, *Proceedings of the Third IEEE International Conference on Multimedia Computing and Systems*, pp. 160-163, 1996.