Malaysian Journal of Computer Science, Vol. 18 No. 1, June 2005, pp. 50-56

THE DESIGN OF VIDEO-ON-DEMAND SERVERS

Putra Sumari
Multimedia Research Group, Computer Science School
Universiti Sains Malaysia, Minden
11800, Penang, Malaysia
email: putras@cs.usm.my

Hailiza Kamarulhaili
Mathematical School
Universiti Sains Malaysia, Minden
11800, Penang, Malaysia
email: hailiza@cs.usm.my

ABSTRACT

Video on demand (VOD) system is an electronic video-rental service over a computer network. It provides users a service to browse and watch any video at any time with VCR (rewind, forward and pause) operations. One of the requirements for VOD system implementation is to have a VOD server that acts as an engine to deliver videos from the server's disk to users. Designing VOD server, particularly the video layout component on the server's disk, is an interesting issue as it determines the number of streams being delivered to users. Video blocks should be laid intelligently with less latency on disk and hence produce high number of streams. However, due to real time and large volume characteristics possessed by the video, the designing of video layout is a challenging task. Real time constraints the distribution of blocks on disk and hence decrease the number of streams being delivered to users. Another important aspect of designing video layout is the VCR operation service since it provides interactivity. It possesses the same challenge as in streaming videos to users as it consumes bandwidth of the server. In this paper we present a new video layout scheme for a VOD server called Disk-region storage scheme. The scheme partitions the disks into regions and uses a special algorithm to distribute blocks of video data within regions. The scheme is capable of producing a high number of streams as well as providing VCR operations to users. The simulation showed that the scheme increases the number of simultaneous stream by 15%. The scheme is also being designed with a simple one-way data retrieval mechanism and has a good load balancing on an array of disks.

Keywords ***VOD server, Data storage, Simultaneous streams, VCR operations***

1.0 INTRODUCTION

A Video-on-Demand (VOD) is an electronic video rental service over a computer network [2, 3, 4, 6, 9, 10, 11, 12]. It has three main components: VOD server, transport network and user display equipment. The VOD server stores a large number of videos and its role is to distribute videos to users on request. Transport network is a wired medium for video delivery from the server to the users. The user display equipment is a television type device for viewing purposes. VOD application eliminates a specific time schedule characteristic as in traditional television programmes [4] and introduces a scenario in which viewing any video at any time is possible.

The VOD server is an important component in this application as its function is to retrieve as many blocks of video data as possible and send them to users. In other word, VOD server is responsible for retrieving different blocks of different video data and send them to different users simultaneously. This is not an easy task due to the real time factor and the large volume of characteristics possessed by the video. Real time characteristic requires that video blocks have to be retrieved from the server's disk within a deadline for continous delivery to users [6]. Failure to meet the deadline will result in jerkiness during viewing. A large volume characteristic requires large bandwidth consumption. For example, a frame with a color depth of 24-bit-colour with 640 pixels x 480 pixels resolution produces 900 Kbytes of data. At 30-frames-per-second presentation speed, the amount is 27 Mbytes [5]. Even the format of MPEG 1 and 2 standard with 1.5 Mbps and 4 Mbps playback rates respectively are considered large in VOD application.

One of the approaches for obtaining a high number of simultaneous streams is to have a VOD server with a good data layout scheme on the disk surface. Due to real time constraints and disk sector-track architecture, the design of data layout can easily create high latency problem. The disk head has to travel within tracks and sectors during data retrieval process. Latency reduces the number of simultaneous streams that the server can support. Conventional Random and Contiguous data layout [8] are the most higher latency issue during data retrieval in VOD system. Numerous data layout schemes have been proposed [1-4, 7, 10, 12, 13] for VOD server. These schemes are designed to minimise the latency so that the number of streams is increased. However in most of these cases, the overall latency is still significant and reduces the number of simultaneous streams. Furthermore most of these works are designed without VCR operations scheme. VCR operation is also crucial since it needs attention from the disk head to retrieve data for the service.

This paper presents the design of VOD servers. The VOD server consists of four major components. The first component is a new data layout scheme called Disk-region storage scheme for storing video on disk surface. The disk is partition into regions and data are distributed within regions using a special algorithm. Secondly, we outline a data retrieval policy of the disk-region scheme for maximum streams. The third is a buffering component. In this scheme latency is reduced and hence increase the number of simultaneous streams. The scheme has a simple retrieval mechanism with a good load balancing properties on multiple disks environment while supporting high number of simultaneous streams. The final component is the VCR operations (start, pause, forward, rewind and resume) which provides interactive services to users.

This paper is organised as follows: Section 2 gives a background and related works. Section 3 outlines the VOD server design and its data layout. Section 4 presents the VCR operations. Section 5 presents the simulation results and finally Section 6 is the conclusion.

2.0 BACKGROUND AND RELATED STUDIES

A VOD server is an important component in video on demand application. It stores a large number of videos and acts as an engine to distribute data from the disks to users. A more specific responsibility is to retrieve different video blocks either belonging to the same or different videos and send them simultaneously to different users. Simultaneous streams refer to the simultaneous transmission channels for the delivery of video from the server to users. This is a challenging task since video holds real time characteristics. Once the stream is initiated, then the streaming should be guaranteed towards the end of the playback. The fact is that video blocks are sat scattered on the disk surface due to the sector-track architecture. This has caused high latency during data retrieval process and has led to the decrement of the number of simultaneous streams that the system can support. One technique to handle this is to design the data layout with an aim to minimise latency and hence increases the number of simultaneous streams.

Numerous researchers have made great contributions to the data storage design. In [7, 8], storage issues in supporting real time data are investigated. The work focused on traditional random and contiguous data allocation. These approaches however are facing to much latency and unsuitable for the read only category like VOD services. In [2], a circular skip cluster scheme is proposed to enhance data retrieval technique in the VOD server. The overall latency is still significant and has decreased the number of simultaneous streams. Furthermore, no VCR operation facility is reported in this work. In [7, 1], disks are partitioned into regions. Data are distributed within regions in a special way. The overall latency however is still significant and decreases the number of simultaneous streams. In [13] scalable scheme is presented and the disks are divided into zones. This scheme however used random placement in intra zones that lead to latency problems. In [10], Phase based storage scheme is proposed. The video is organised into a column-row representation. Disk data are stored in column order and concentric fashion. Latency is eliminated and the number of concurrent streams is maximised. This scheme however has made the inner space of the disk idle and the data cannot be written or read. Therefore, we present another method in devising a new storage architecture with the objective of maximising the number of simultaneous streams as well as providing VCR operations to the users.

3.0 VIDEO-ON-DEMAND SERVER ARCHITECTURE

The architecture of a VOD server is shown in Fig. 1. The video server consists of an array of disks and a buffer. Data blocks are fetched and placed temporarily in the buffer before being sent to the network. Blocks of data are flushed from the buffer in simultaneous fashion to the network (Fig. 1). It shows five simultaneous streams labelled as $s1$, $s2$, … , and $s5$ which are flushed out to the network. The number of blocks that goes into the buffer is determined by the data layout in the disks.

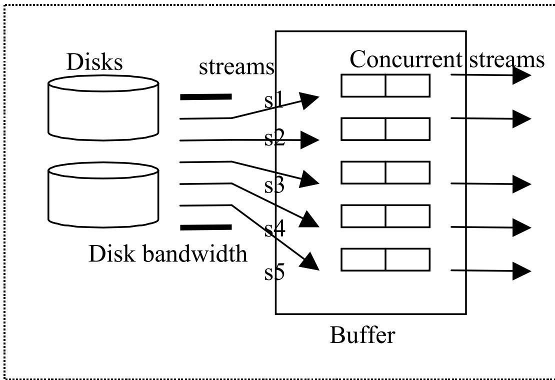

Fig. 1: Video-on-Demand Servers Architecture

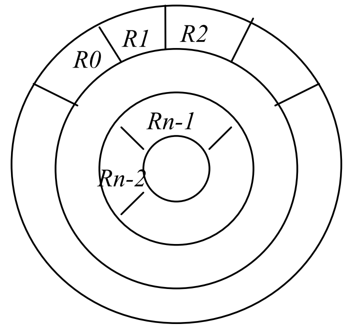

Fig. 2 : Disk_region concept: disk is divided into n regions

3.1 Disk-Region Layout Scheme

First let us consider a single disk environment. The disk surface is divided into regions of equal size (Fig. 2) where *n* regions are labeled as *R0* to *Rn-2*. The first region is labeled as *R0* and the last region is labeled as *Rn-2*. The number of regions (*n*) is derived as follows:

$$n = (disk\_size/disk\_rate)/s \tag{1}$$

The number of blocks (*p*) that sits in a single region is derived as follows:

$$p = disk\_rate/video\_rate \tag{2}$$

Equation (1) states that the size of each region is *(disk_rate)/s* (note here that the size of region is identical to the size of disk bandwidth) and Equation (2) states that each region can occupy a maximum of *p* blocks of size *(video_rate)/s* each. Therefore, Equation (1) and Equation (2) states that *p* are the maximum number of blocks that sits in a single region. *p* also indicates the number of simultaneous blocks to be retrieved and will form maximum streams.

In multiple disks environment, we assume all disk are identical in capacity and hence the number of regions of each disk is identical to each other. Each disk has *n* regions. With *m* disks and *n* regions, we can represent as (*m* x *n*) matrix (Fig. 3). The rows indicate disks and the columns indicate regions. The (*m,n*) notation is the indicator of block location, e.g. position (*1,2*) means block *i* sits on disk *1* region *2*.

Table 1 : Parameter used in the study

Ri	Region *i*th
disk_size	Disk capacity
disk_rate	Disk bandwidth
video_rate	Playback rate
p	Number of simultaneous streams
d	Block size
m	# of disk or row
n	# of regions or column

	region0	region1	region2	region3	region4
Disk 0	(b1,b16,..)	(b7,...)	(b13,..)	(b4,..)	(b10,..)
Disk 1	(b11,0)	(b2,b17,..)	(b8,2)	(b14,..)	(b5,..)
Disk 2	(b6,..)	(b12,..)	(b3,b18,..)	(b9,..)	(b15,)

Fig. 3 : Block b1, b2, …, bk are striped across disks and regions

3.2 Data Layout

Our data placement based on mathematical number theory is known as residual classes. Let us have *m* disks with *n* regions each. We denote the set of all integers such that $m \equiv i \bmod (y)$ and call residual classes *i* modulo *y*. For instance if $y = 3$, then we have 3 residual classes which are 0, 1 and 2. The class of 0 contains all integers which when divided by 3 leaves no residue. The class of 1 contains all the integers which when divided by 3 leaves 1 as residue and so on. In our case, we have two integers *m* and *y* with $(m, y) = 1$. The greatest common factor of *m* and *y* is equal to 1, with an additional condition that $y = [n]_m$, where $[n]_m$ means that the greatest relative less or equal to *n* which is relatively integer to prime *m*. From the following formula, the *i*-th block of video is then stripped across *m* disk and *n* regions. That is, the *i*-th block is placed in:

$$(\text{region number} \equiv i(mod)m, \text{ disk number} \equiv i(mod)n) \qquad (3)$$

For example, let the number of disk (*m*) be 3 and the number of region (*n*) be 5. The video is composed of blocks b1, b2, b3, .. , bk. The data blocks placement is shown in Fig. 3. Equation (3) will produce the layout of the following pattern, block b1 is stored at location (0,0); block b2 is stored at location (1,1) and so on until the end of the block. The sequence of blocks sitting in each region must be in ascending order. For example, Fig. 3 shows all the blocks at location (1,1) are in ascending order that is b1, b16, …. , b-th.

3.3 Data Retrieval and Buffering Scheme

In our scheme, the R/W heads of all disks should be synchronised. Data retrieval is done column by column, that is, region 1 of all *m* disks are first read, followed by region 2 of all *m* disks and continues towards the last region of all *m* disks. On reaching the last region, the R/W head returns back to the first region and repeats the reading process. With this data placement, real time retrieval of blocks is guaranteed. In Fig. 3, block b1 is located at region 0 and the subsequent block b2 is located at region 1. The gap between block b1 and block b2 is bounded by real time constraint (satisfied by equation (1) and (2)). Block b2 in region 1 will be retrieved after block b1 is retrieved from region 0.

In this scheme we use double buffering approach for buffering mechanism [3] and it suits our data retrieval process. The scheme consumes buffer of size (*2mdisk_rate*) which is the sum of two regions of all *m* disks. In order to visualise the buffering operation, the buffer is divided into two rooms of size (*mdisk_rate*) each as shown in Fig. 5. While the first room is filled with data of size (*mdisk_rate*), the second room flushed out data of size (*mdisk_rate*). The role switch, that is the second room, is now filled with data of size (*mdisk_rate*) and at the same time the first room flushed data out with size (*mdisk_rate*). This buffering policy preserves the real time constraint of video data. When the R/W head reaches the last region, it returns to the first region and repeats the process.

4.0 VCR OPERATIONS

This section describes how VCR operation (rewind, fast rewind, forward, fast forward and pause) are supported by our Disk-region scheme as mention in Sections 3.1, 3.2 and 3.3.

The videos are stored in the disk subsystem in the form of MPEG standard. The MPEG standard contains three types of frames namely, Intraframe (I), Predicted frame (P) and Bidirectional frame (B). I frames are stand-alone frames and can be decoded independently of other frames. P frames are dependent (reference) to the previous I/P frame and can be decoded only if the previous I/P frame is available. B frames are dependent (reference) to the previous I/P and the following I/P frames and can be decoded only if the previous I/P and the following I/P frames are available. A block of video that contains a sequence of frames beginning with an I-frame and ending with a P frame is called an independent sequence of frames (Fig. 4). Independent sequence of frames contains references for every B-frame in it, therefore it can be decoded by MPEG decoder.

I	B	B	P	….	B	B	P	I

Fig. 4: Independent sequence of frames

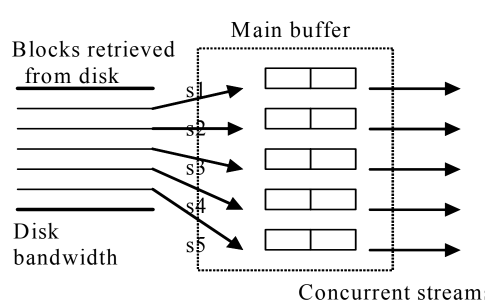

Fig. 5 : The system buffer architecture

Using the description in Section 3.3, we can summarise the buffer architecture of our system (Fig. 5). The buffer contains the main buffer. The main buffer stores blocks received from the disk. Double buffering technique is used for the buffering operation. While the first half of the main buffer sends blocks to users, the second half of the main buffer receives blocks from disks. Then the roles switch. We now describe the VCR operation (start, pause, forward, rewind and resume) in our system. *Starts*: the playback of the video starts once the main buffer contains the first block of the video. Block of size *d* is transmitted to the users at *video_rate*. *Pause*: once the block in the main buffer that contains an I-frame is transmitted to the users, the I-frame is repeatedly sent to the users. *Forward*: in the main buffer, the following steps are executed:

1. Continue transmitting data normally until P-frame is transmitted from one of the room at main buffer and the next room of the main buffer contains I-frame
2. Transmit block beginning with I-frame in the room of main buffer that contains I-frame
3. Go to step 1.

Thus during *forward* independent sequence of frames are transmitted to the users. *Rewind*: this operation is implemented in a similar fashion to the *forward* except that instead of jumping to the next room of the main buffer, it jumps to the previous room of the main buffer for the previous block. The following steps are executed:

1. Continue transmitting data normally until P-frame is transmitted from one of the room at main buffer and the previous room of the main buffer contains I-frame
2. Transmit block beginning with I-frame in the room of main buffer that contains I-frame
3. Go to step 1.

Resume: in case the previously issued command is either forward or rewind, blocks will continue to be transmitted normally from the current main buffer of the playback video. If the previous command was *pause,* then once when one of the rooms in the main buffer which contains I-frame following the last P-frame is transmitted, normal transmissions is resumed from that room beginning with the I-frame.

5.0 SIMULATION STUDY

In this section we conduct a simulation study for measuring the latency and throughput of the system. We consider four Seagate Baracuda disks with the following characteristics:

Disk size = 2.25 GB capacity
Transfer rate = 75 Mbps
Min. seek time (Smin) is 0.6 milliseconds
Max. seek time (Smax) is 17 milliseconds
Rotational latency time is 8.33 milliseconds.

The seek model is an approximation based on the models proposed by [14].

$$\text{Latency} = \begin{cases} 0.4 + (0.2 \times \sqrt{\text{Smax}}) + 8.33, & \text{if Smax} < 400 \\ 2.3 + (0.0052 \times \text{Smax}) + 8.33, & \text{if Smax} \geq 400 \end{cases}$$

We use five MPEG 1 standard videos with 60 minutes length each. Block size in our scheme is set to 1.5 Mbits. With four disks, we derive 240 regions to form (4 x 240) matrix. The placement of blocks satisfy Equations (1), (2) and (3). The simulation was developed using the CSIM/C++ discrete event simulation language [11]. With videos and users access pattern, we set the popularity of each video based on Zifpian (*z*) [3] distribution and the characteristic of users arrival rates is based on Poisson (*p*) distribution. We adopt the default value of both Zipfian and Poisson parameter by $z = 0.2$ and $p = 8$ respectively [3].

Fig. 6 shows the effect of different size of blocks against latency. We compare the performance of our scheme for three different sizes of blocks. Fig. 6 shows that 1.5 Mb size block is the lowest latency. It satisfies Equation (1) and (2) of our scheme. Block with size 1.5Mb satisfies the real time constraint for our scheme. They fit into the region area and left no gap between blocks. Making the block size smaller has led to an increment of latency because smaller block in size lead to a gap between blocks. The number of blocks may sit in region is fixed and adding more blocks will affect the real time data retrieval.

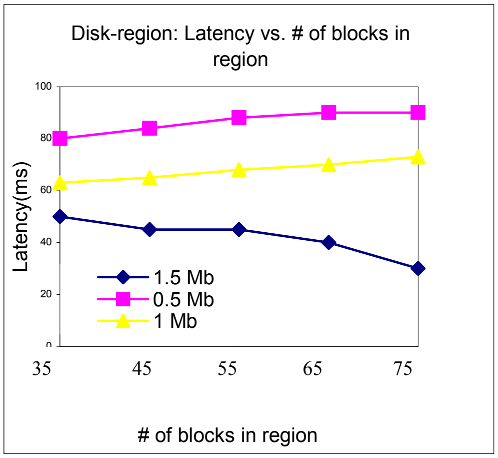

Fig. 6: Latency performance

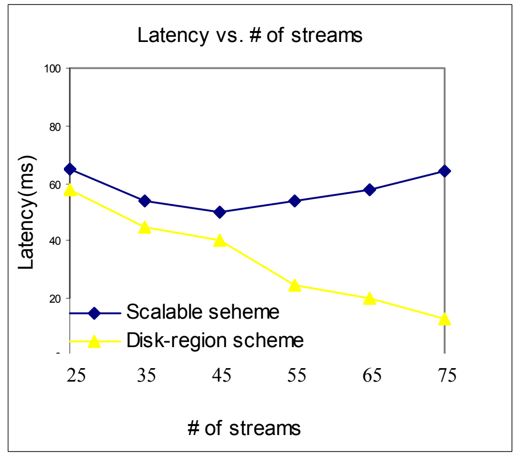

Fig. 7: Throughput

Fig. 7 shows the effect on the number of streams against the latency. Disk-region scheme produces the highest streams simply because of less latency in our scheme when compared to other schemes. The figure shows 15% increment in number of streams compared to existing scalable schemes.

6.0 CONCLUSION

The Video-on-Demand (VOD) system provides an environment of viewing videos through computer network. The viewing is supported by VCR operations such as forward, fast forward, rewind, fast rewind and pause. VCR is an important element for interactivity. One important element in this system is the design of VOD server which acts as an engine to deliver videos stream from the server's disk to users. The number of streams deliver from the server to users is determined by a video layout scheme implemented in the server's disk. Video blocks should be laid intelligently with less latency on disk and hence produce high number of streams. However, real time and the large size of videos are the main factors contributing to the difficulty of designing video layout. Blocks of video data to be laid on the disk surface are constrained by real time issues hence increase latency. This paper presented a new video layout scheme for VOD server called Disk-region storage scheme. Disks are divided into regions and data are distributed within regions using a special algorithm. The data placement within regions is designed so that subsequent blocks of the same video are placed at different region bounded by real time characteristics. The blocks are arranged so that there are no gaps between blocks in a single region. A simple one-way retrieval mechanism is achieved and latency is tremendously reduced. The scheme has a good load balancing on an array of disks. Our scheme has increased the number of streams by 15%. The scheme supports VCR operations for interactivity as a requirement of VOD servers.

REFERENCES

[1] E. Chang and H. Garcia-Molina, "Reducing Initial Latency in Media Servers". *IEEE Multimedia*, Vol. pp. 50-61, 1997.

[2] C. K. Chang and C.-C. Shih, "A Circular Skip-Cluster Scheme to Support Video-on-Demand Services". *Multimedia System*, Vol. 7, No. 2, 1999, pp. 107-118.

[3] T. S. Chua, J. Li, B. C. Ooi and K. L. Tan, "Disk Stripping Strategies for Large Video-on-Demand Servers", in *The 4th ACM International Multimedia Conference*, Boston, MA, 1996, pp. 297-306.

[4] A. Srivastava, A. Kumar and A. Singru, "Design and Analysis of a Video-on-Demand Server". *Multimedia Systems* , Vol. 5, No. 4, 1997, pp. 238-254.

[5] B. Furht, D. Kalra, F. L. Kitson, A. A. Rodriguez and W. E. Wall, "Design Issues for Interactive Television Systems". *Computer*, 1995, Vol. 28, No. 5, pp. 25-39.

[6] D. J. Gemmell, H. M. Vin, D. D. Kandlur, P. V. Rangan, and L. A. Rowe, “Multimedia Storage Server: A Tutorial”. *Computer,* 1995, Vol. 28, No. 5, pp. 40-49.

[7] S. Ghandeharizadeh, S. H. Kim and C. Shahabi, “On Configuring a Single Disk Continuous Media Server”, in *SIGMETRICS Performance Evaluation, Ottawa Ontario, Canada*, Vol. 23, No. 1, 1995, pp. 37-46.

[8] J. C. l. Liu, D. H. C. Du and J. A. Schnepf, “Supporting Random Access on Real-Time Retrieval of Digital Continuous Media”. *Computer Communications*, Vol. 18, No. 3, 1995, pp. 145-159.

[9] V. O. K. Li and W. Liao, “Distributed Multimedia System”. *Proceeding of the IEEE*, Vol. 85, No. 7, 1997, pp. 1063-1108.

[10] B. Ozden, R. Rastogi and A. Silberschatz, “On the Design of a Low-Cost Video-on-Demand Storage System”. *Multimedia System*, Vol. 4, No. 1, 1996, pp. 40-54.

[11] V. S. Subrahmanium, and S. Jajoda, *Multimedia Database Systems: Issues and Research Directions*: Springer, New York, 1996, pp. 237-261.

[12] P. Sumari, M. Merabti and R. Pereira, “Video-on-Demand Server: Strategies for Improving Performance”. *IEE Proceedings Software*, Vol. 146, No. 1, 1999, pp. 33-37.

[13] C.-S. Wu, G.-K. Ma and M.-C. Liu, “A Scalable Storage Supporting Multistream Real-Time Data Retrieval”. *Multimedia System*, Vol. 7, 1999, pp. 458-466.

[14] C. Ruemmler and J. Wilkes, “An Introduction to Disk Drive Modelling”. *IEEE Computer*, March 1994.

BIOGRAPHY

Putra Sumari obtained his MSc and PhD in 1997 and 2000 from Liverpool University, England. Currently, he is a lecturer at the School of Computer Science, Universiti Sains Malaysia, Penang. His research areas include video on demand system, multimedia storage server, MPEG standard, image/video compression and retrieval, image Cryptography. He has published more than 50 papers related to these areas. He is the head of the Multimedia Computing Research Group, School of Computer Science, USM. Member of ACM and IEEE, Program Committee and reviewer of several International Conference on Information and Communication Technology (ICT), Committee of Malaysian ISO Standard Working Group on Software Engineering Practice, Chairman of Industrial Training Program School of Computer Science USM, Advisor of Master in Multimedia Education Program, UPSI, Perak.

Hailiza Kamarulhaili obtained his MSc and PhD from Liverpool University, England in 1996 and 1999 respectively. Currently, she is a lecturer at the School of Mathematics, Universiti Sains Malaysia, Penang. Her research areas include analytical number theory, analysis, number theory, uniform distribution mod one, diophantine approximation, and cryptography. She has published more than 40 papers related to these areas. She is a member of ACM, Member of School of Mathematic USM Penang, Program Committee of International Conference of Geometric Modelling and Graphic (CGIV).