論文 97-22-11-17

Modeling of a Storage Subsystem in a Multimedia Information System

CheolSu Lim* *Regular Member*

ABSTRACT

In this paper, we present a video-on-demand (VOD) system design model that addresses and integrates a number of inter-related issues. Then with analysis and performance evaluation, we investigate various aspects of disk and buffer managements in the given model. Based on the analysis results, we suggest that a distributed buffering scheme with intermediate buffers may be useful to transform bursty disk accesses into a continuous stream for glitch-free performance of VOD systems. Also, through simulation, we illustrate that massive multimedia information storage design techniques such as prefetching, clustered striping, and real-time disk scheduling integrated with the distributed buffering mechanism may enhance end-to-end real-time performance of VOD systems under wide-area networks.

I. Introduction

Recent development in computer technologies in the areas of processors, storage, and computer networks and communications make it possible to build a system to provide services that allow customers to access necessary multimedia information from remote archives when they need it. Such a system, typically referred to as *video-on-demand* (VOD) system, promises enormous usages in a variety of areas in human society including entertainment (e.g., movie and game on demand), media broadcasting, (e.g., news on demand), education (e.g., distance learning), and so on. Due to its enormous potential, a number of different approaches to its implementation have been discussed. Especially, VOD service provision has been recently discussed in connection with internet information servers such as World Wide Web as internet technology is booming.

Whether a VOD system is implemented as part of internet information servers or as a stand-alone system with limited connectivity, it presents some unique data access characteristics and requirements unseen in conventional information systems. Such new requirements include massive volume of multimedia data, intra-media continuity, inter-media synchronization, and end-to-end performance guarantee among others [1]. VOD services require end-to-end real-time performance to display successive images for motion to flow smoothly without any pause or interruption.

Despite the recent advance of computer hardware component performance, it is still a challenge to meet the timing constraints on designing VOD systems. To satisfy the end-to-end real-time performance requirement, system software and drivers at various levels including operating systems and network protocols should directly consider timing constraints of requests in job

*서경대학교 컴퓨터공학과
論文番號:97145-0430
接受日字:1997年 4月 30日

scheduling and resource management. However, in this paper, we focus on the design of VOD system's storage subsystem to meet its end-to-end real-time performance.

Technical feasibility of VOD systems has been discussed in [2, 3]. Though there have been active studies for storage subsystem design, few considers the perfomance in terms of satisfying real-time requirements. Most work tackled the problem to improve I/O system throughput [4], and the issues of disk layout [1, 6] and disk striping technology [7, 8] have also been discussed from a conventional perspective. Several real-time disk scheduling algorithms were introduced and assessed in [9]. Study in [10] have given a high-level description of a distributed system for multimedia data processing, but they did not describe in detail how real-time requirements in data retrieval can be met. Another work for buffer management in multimedia information system focused on buffer replacement algorithms for improving real-time performance.

The study in this paper was motivated by the following observations to the previous work in this area. First, we observe that as opposed to high-level components such as OS kernel and network protocols, storage subsystem regarding its real-time performance has not been actively studied despite of its potential impact on the end-to-end performance requirement of VOD systems. Second, we observe that previous work in this area has dealt with one or two subproblems separately in the design, and that few has considered the design with an integrated view. We believe that a design combining various aspects of the subsystem should be studied to investigate their interactions and impct on the performance in a more realistic environment.

Ⅱ. System Model

1. The Model

Our VOD system model provides a general architecture common to most VOD systems for this baseline research. It can be viewed through stepwise refinement as shown in Figure 1. The system stack model at stage 1 shows the entire system components to support VOD applications. Major components of the system to be elaborated at stages 2 and 3 are the VOD main server (MS), store and forward (S&F) buffer of intermediate servers (IS), high-speed networks, and client's display devices referred to customer premises equipment (CPE). MS is a large scale video (multimedia information) repository and is connected to ISs via high-speed network links such as STS-3 (155 Mbps) or STS-12 (622 Mbps).

At stage 2, MS holds video programs at on-line storage devices in digital compressed form using MPEG coding scheme. The disk efficiency at this stage is the focus of this paper. Stage 3 addresses the situations in which continuous media (CM) data traverses a linear sequence of resources. It models how multimedia data is retrieved from disks and transferred to internal buffers of CPEs through ISs. It is this stage that represents a situation such as an

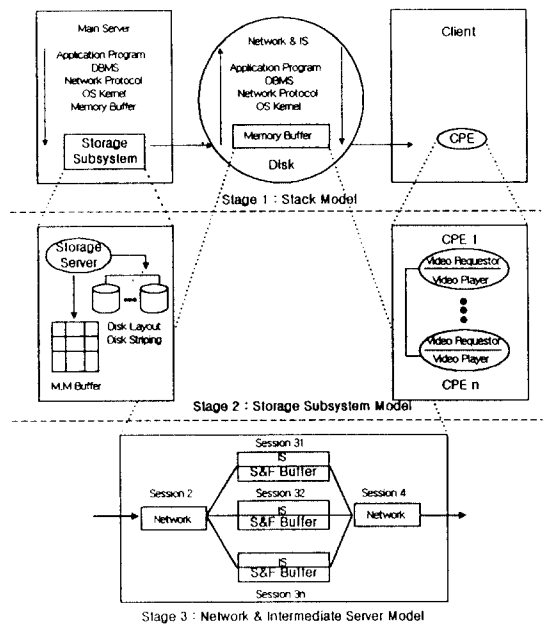


Fig. 1 VOD System Architecture

integrated session comprised of sessions with each of the resources involved. Resources in an integrated session handle a stream of continuous media in a pipeline fashion. Video frames requested from each CPE at 30 frames/sec will be carried to the IS through high-speed channels. The S&F buffers of ISs will store the frames from the main memory buffer and forward it to the client at the real-time rate, i.e., 1.544 Mbps.

2. Preliminary Considerations

In order to have an insight into designing VOD systems and to identify the significant factors in end-to-end performance of the systems, we examine the following massive information storage techniques, and build an experimental framework integrating the techniques in a distributed environment.

Disk Layout Scheme concerns about how to store the data blocks on disks through external mapping. Since multimedia data are intrinsically isochronous and can deliver meaning only when these are played back continuously, optimal disk access time(esp, seek time) depends on data block layout scheme on disk.

Clustered Striping Technique is proposed and analyzed in [11] as an enhanced technique of conventional disk striping for multimedia information system. It enables either a multiple disk or a parallel system to guarantee a continuous retrieval of a media object at the bandwidth required to support user playback rate because it avoids the formation of bottlenecks by striping an object across the disk clusters The conventional disk striping technique can be classified into two classes by the data retrieval scheme, i.e., sequential retrieval and pipelined retrieval. At this moment,

$$\text{Total access time of sequential retrieval} = N*((S+R)_{max}+Tf_{max})$$
$$\text{Total access time of pipelined retrieval} = (S+R)_{max}+N*Tf_{max}$$

where S represents seek time, R rotational delay, Tf transfer time and N the number of clients.

Disk Scheduling Algorithms: Traditional algorithms might not be suitable for real-time VOD applications because they do not account for timing constraints or deadlines in their scheduling decision. In our real-time traffic model, requests are scheduled according to each algorithm that knows the deadlines of the requests in the queue such as EDF (Earliest Deadline First), FD-SCAN (Feasible Deadline SCAN) and SSEDV(Shortest Seek and Earliest Deadline by Value) algorithm.

Buffering Requirements for Continuous Playback: Buffering is required in VOD servers to smooth out playback variations, network delay jitter, and bandwidth differences of various resources across the entire system. In order to guarantee timely availability of media blocks for continuous playback, a read-ahead depending on buffer size may also be considered before initiating playback.

Network Bandwidth Control: The problem in ensuring seamless end-to-end delivery of multimedia traffic over networks has to do with the bandwidth mismatch between each intervening resources. The B-ISDN and the ATM paradigm that has been developed to handle broadband services over high-speed networks have as their prime purpose the provision of the appropriate QoS(Quality of Service) for a set of different traffic types. The broadband capability using the B-ISDN means that users are capable of receiving the high-resolution images or videos.

Ⅲ. A Buffering Mechanism of Intermediate Servers

Providing VOD services to a large number of subscribers using high-speed backbone often requires an enormous amount of buffer memory at the main server side. Also, the memory sometimes creates a

bottleneck in the system. If buffer occupancy were to ever become negative, then, at that instant, the display device could be starved of the required media block, resulting in a loss of continuity. To avoid this problem, we propose to spread the buffer capacity requirement to several ISs in network.

1. Distributed Buffering Scheme

Our distributed buffering scheme uses pooled S&F video buffers of IS at the edges of the network. With this distributed buffering scheme, the number of users connected to a front-end switch can be reduced by scattering connections to the main server. An IS needs to maintain two types of connections with a CPE: a one-way data connection which transfers video frames from an IS to a CPE and a full-duplex control connection which transfers control information through an application level protocol. By this way, users can have continuous accesses to service, while only a small number of accesses needs to be generated from the server optimizing both network and storage resources of the main VOD server.

At the start of a session, data prefetching is performed for the first few seconds to load part of a selected information from the main server to an IS. This prefetching causes an instant pause. After this time on, the requested information will be continuously feed at video signal rate (e.g., 1.544Mbps) into buffers.

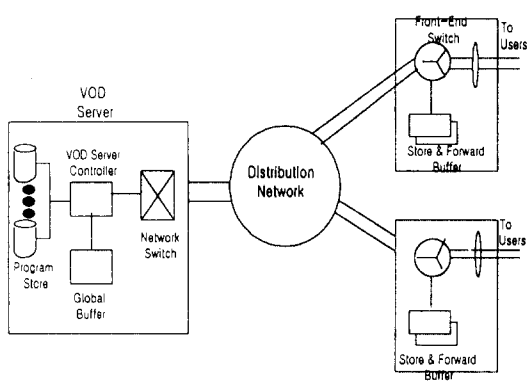


Fig 2. The Configuration of distributed buffering Scheme

2. Buffer Management Policies

Three basic functionalities of buffer management in a multimedia information system are request admission, buffer allocation and buffer replacement. The goal of buffer management is to increase the number of requests meeting their deadlines, at the same time, to maximize the number of concurrent sessions with the given capacity. Considering the performance goal, buffer management in a VOD system requires changes to the traditional buffer management algorithms optimized for average response time and throughput. In this regard, we propose a buffer management scheme that can be integrated with admission control and resource reservation schemes.

We introduced a two-tier admission control scheme under the distributed buffering scheme, which is required due to the limitations of disk bandwidth, buffer size, and network bandwidth. Whenever a customer request comes into the arriving queue monitored by the scheduler of IS, the first level of admission control is performed at the intermediate server, and the second level control done at the main server. Using the following formula, the admission control policy checks for each request separately to see if there is sufficient buffer space, disk bandwidth and network bandwidth available to fetch, store and transfer data for the request. Given (n-1) sessions already in progress, the n-th incoming request is admitted if and only if the following inequalities hold :

1. Disk bandwidth : $\sum_{i=1}^{n} d_{i,t} \le B_d,\ \forall t = 1,...,k$

2. Buffer size : $\sum_{i=1}^{n} l * d_{i,t} \le B_s,\ \forall t = 1,...,k$

3. Network bandwidth : $\sum_{i=1}^{n} d_{i,t} \le B_x,\ \forall t = 1,...,k$

where $d_{i,t}$ represents the continuous media rate for request i at time t, l represents the length of each session (of 1 second duration), k represents the total number of subscribers in each intermediate server, B_d

represents the disk data rate, B_s represents the buffer size, and Bx represents the network transmission rate.

We propose *Fetch and Discard* (*FAD*) buffer management scheme which replaces data on a fragment basis (of the amount of 1 second playback). A FIFO policy is used to decide which of the fragments from the free-pool are to be allocated to a new request, i.e., when a new request is to be allocated a fragment from the free-pool, the oldest fragment in the free-pool is allocated first. FAD reuses fragments that may already exist in S&F buffer, i.e., if any of the required fragments happen to exist in the buffer then they are grabbed from the buffer, thereby avoiding fetching those fragments again from the disk. FAD does not consider sharing explicitly, i.e., when the fragments of a topic have been played back, they are not preserved in buffer for a future user who might request the same information.

```
do while there are requests in the queue
   select next request;
   admitted = false;
   for each segment in the video,
        check if the segment is already in the free-pool
              and grab it if it is available;
   for each segment grabbed from the free-pool,
        check for buffer bandwidth violation;
   for segments that have to be fetched from the disk,
        modify the total profile for disk, buffer and network accordingly
        and check for disk, buffer and network bandwidth violation;
   if no violation at any stage
        admitted = true;
   else
        restore old total profile;
        reject request;
   endif
end do
```

Fig 3. Admission Control Scheme

```
do forever
   do for all active sessions
      if buffers are not shared with another session
        when a new segment is encountered
            return the used buffers to the free pool;
            acquire new buffers from the free pool;
            fetch segment from disk;
      else
        if leading user
            preserve the used buffers for lagging users;
            acquire new buffers from the free pool;
            fetch data from disk into the buffers;
        else if lagging user
            return the used buffers to the free pool;
            read from preserved buffers;
        end if
      end if
   end do
end do
```

Fig 4. Buffer Management policy

3. Buffer Capacity Analysis and Prefetching

If the buffer capacity were to ever become negative, then, at that instant, the display device could be starved of the required media block, resulting in a loss of continuity. Such a discontinuity can be alleviated by prefetching. Thus, in order to guarantee timely availability of media blocks for continuous playback, a read ahead depending on buffer size may be necessary before initiating playback.

In the stage 3 model in Figure 1, a video frame arrival processes at the interface I is represented with a *linear bounded arrival process* (LBAP) as described in [12] with the following three fixed parameters: M (maximum frame size in bytes), R (maximum frame rate in frames/sec), and P (prefetching limit in the unit of frames). In this model, a distributed system is decomposed into a chain of resources traversed by the messages on their end-to-end path. Examples of such resources are single schedulable devices (such as CPU) or combined entities (such as networks). In processing CM data, unlike other random accesses to disks, the next data block that will be accessed can always be anticipated, and thereby the data block prefetching can be easily used to increase buffer utilization, and to alleviate the differences in bandwidths of disks and networks. These properties may be helpful to improve end-to-end performance in CM data delivery. Using the above three parameters, the prefetching P(t) of an LBAP can be defined as follows:

$$P(t) = \max_{t0<t} \{0, N_I(t_0, t) - R^* |t - t_0|\}$$

where $N_I(t_0, t)$ denotes the number of frame requests

arriving at the interface I during $[t_0, t)$, hence it represents the amounts of frames that have to be buffered.

To explore the operational behavior of VOD services and derive the buffering and prefetching requirements, we can describe the following formulas: Let R(t) denote the number of media blocks read from the disk upto time t and let

$$R(t) = \begin{cases} 0, & \text{if } t \le 0 \\ R(t_{max}), & \text{if } t > t_{max} \end{cases}$$

where t_{max} is the time that reading is completed. Let t_0 be the playback start time and also let the number of frames played back by user's device at time t be $P(t, t_0)$. If frames are played back at rp frames per second, then

$$R(t, t_0) = \begin{cases} 0, & \text{if } t < t_0 \\ r_p(t - t_0), & \text{if } t \ge t_0 \end{cases}$$

If the number of frames read is greater than the number of frames played back at any time, then the frames that have been read but not yet played back must be buffered according to the following formula:

$$B(t, t_0) = R(t) - P(t, t_0)$$

If we analyze the buffer capacity with prefetching, then

$$B(t, t_i) = \begin{cases} R(t), & \text{if } t < t_i \\ R(t) - r_p(t - t_i), & \text{if } t \ge t_i \end{cases}$$

where t_i is the minimum playback start time that can prevent the starvation of the required media block. Now using an example, we show how we can compute the amount of data that need to be prefetched at the main server and intermediate servers[13].

(***Example 1***) In some sense, the objective of buffering is to provide an equilibrium state to the system by controlling the input and output of the system, and the objective of prefetching can be viewed as an initiation of this equilibrium state. Let M = 6.25 KB/frame and R = 30 frames/sec per subscriber. Also let the number of the totally admitted subscribers A be $A = \sum_{j=1}^{n} A_j$ where n is the number of intermediate servers and A_j is the number of admitted subscribers in the j-th intermediate server. Now assume that disk bandwidth $B_d = 10$ MB/sec and network bandwidth $B_x = 20$ MB/sec. For prefetching, B_d will be (10/A) MB/sec per subscriber and B_x ($20/A_j$) MB/sec per subscriber.

First, the amount of data prefetched for the main server per subscriber during first 1 second will be

$$\begin{aligned} P_s(t) &= \max\{0, N_I(t_0, t) - R * |1 - 0|\} \\ &= \max\{0, (B_d/A)MB/s - (30 frames/s * 6.25\ KB/frame * 1\ \text{sec})\} \\ &= \max\{0, ((10/A)MB - 187.5KB)/s\} \end{aligned}$$

Now if we assume that the total number of subscribers A is 27, then $P_s(t)$ will be approximately 187. 5 KB/sec per subscriber, so the main server can prefetch the amount of 1 second playback for each client. And if A is equal to or greater than 54, then no prefetching is possible.

Second, the amount of data prefetched for each intermediate server per subscriber during first 1 second will be

$$\begin{aligned} P_{is}^{j}(t) &= \max\{0, N_{is}(0, 1) - R * |1 - 0|\} \\ &= \max\{0, (B_x/A_j)MB/s - (30 frames/s * 6.25\ KB/frame * 1\ \text{sec})\} \\ &= \max\{0, ((20/A_j)MB - 187.5KB)/s\} \end{aligned}$$

If the number of subscribers of an intermediate server (A_j) is 27, then $P_{is}^{j}(t)$ will be approximately 533 KB/sec and if Aj is 54, then 183 KB/sec, hence the intermediate server can prefetch the amount of 1 second playback for its client. And if A_j is equal to or greater than 107, then no prefetching is possible in this intermediate server(IS_j). We will calculate the required buffer size by considering the effects of the

system resource parameters. Among the three factors for determining the buffer size, i.e., prefetching, flow of data, and synchronization, we deal with only the first two, and bypass the issue of synchronization for simplicity. This is possible because we already assumed MPEG coding scheme for digital video. We compute an upper bound of the buffer size either when there are intermediate servers or not.

(***Example 2***) We compute the buffer size of the main server and intermediate servers in the following two cases: with and without prefetching. First, we assume that we do not prefetch. Let M be 6.25 KB/frame and R be 30 frames/sec per subscriber. Then the buffer size of the main server per subscriber, B_1, will be

$$B_1 = P_s(t) + R*(D-U)$$
$$= 0 + (30 frames/s * 6.25 KB/frame) * (19.43 - 1.43) ms$$
$$= 3{,}375\ Byte/frame$$

where D the total delay time of the integrated sessions, and U the minimum unbuffered time. The delay bound of the integrated session D (D_d below) can be computed as follows:

$$D_d(frame) = T_s + T_r + T_x$$
$$= 10.5\ ms/frame + 8.3\ ms/frame + (6.25\ KB/frame)/(10 MB/s)$$
$$= 10.5\ ms/frame + 8.3\ ms/frame + 0.625\ ms/frame$$
$$= 19.43\ ms/frame$$

where T_s is seek time and assumed to be 10.5 ms/frame, T_r denotes rotational delay and is assumed to be 8.3 ms/frame, Tx is determined from disk bandwidth by Sm/Bd [14].

Now the end-to-end network delay time U, (D_x below) can be computed as follows:

$$D_x(frame) = D_p + D_t + D_v$$
$$= 1 ms + 6.25 KB/(20 MB/s) + 100 \mu s = 1.43\ ms/frame$$

In this equation, D_p denotes propagation delay, and is assumed to be 1 ms (per 200Km). Dt is determined from network bandwidth as (S_m/B_x), where S_m is the frame size for medium m. Finally, D_v is a variable delay determined by traffic and assumed to be 100 μ sec. Hence, if the number of admitted subscribers is 100, then the size of the buffer required by this scheme is 337.5 KBytes. Also, the buffer size of an intermediate buffer per subscriber, B_3, can be computed as follows:

$$B_3 = P_s(t) + R*(D-U) = 0 + 0 = 0\ byte$$

This result seems strange but is reasonable, because (D-U) equals zero. Note that here we only consider the producer-consumer relation of buffer flow and do not include the buffering for synchronization.

(***Example 3***) Now let us consider the buffer sizes when we do prefetch. Suppose M be 6.25 KB/frame and R be 30 frames/sec per subscriber. Then the buffer size of the main server per subscriber, B_2, is computed as follows:

$$B_2 = P_s(t) + R*(D-U) = |(10MB)/A - 187.5KB| + 3{,}375 Byte$$

The buffer size of an intermediate server, B_4, will be:

$$B_4 = P_s(t) + R*(D-U) = (B_X/A_j - 187.5KB) + 0\ byte$$
$$= (20 MB/s)/A_j - 187.5KB$$

Hence, if the subscribers of IS_j is 100, then the buffer size of IS_j required by this scheme is 1.25 MByte.

Ⅳ. Performance Evaluation

Our simulation system has been written in SMPL (Simulation/Modeling Programming Language), an event-oriented simulation language. We assume that customers watch videos in only normal playback mode because it is unlikely that two clients play the

same position simultaneously. The arrival process and departure process are to be modeled using poisson distribution and exponential distribution respectively.

The simulation parameters we used are listed below in table 1 and the deadline of each request is obtained as follows:

Deadline = "*average disk_access_time*"
+*round_trip network delay time from host to IS*
+*round_trip network delay time from IS to user's CPE*
+(*intentional delay*)

Intentional delay time can be used if we can assume prefetching. For example, if the prefetching time per user is set to 1 second, then 30 frames are to be prefetched for each user in advance and quite large deadlines are set correspondingly.

Table 1. Simulation Parameters

Disk	Storage Capacity	1.2 GB
	No. of tracks	1000
	Track Capacity	1.2 MB
	Frames per track	192
	Average seek time	10.5 ms
	Rotational delay	U~[0, 16, 7]
	Disk bandwidth	80 Mbps
	No. of disks	50
Data-base	No. of Objects(movies)	50
	No. of subobjects/objects	50
	Degree of declustering	5
	$B_{display}$	1.5 Mbps
Network	Network bandwidth	150 Mbps
	Propagation delay	1ms/200km

The first two figures, Figure 5 and 6 present the result that distributed buffering scheme provides better performance than centralized one approximately 30% under the same storage subsystem option of no prefetch. It is important to note that we have already

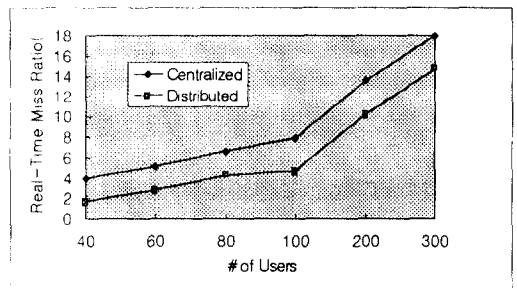


Fig 5. Prefetching = 0ms & EDF (Under Conventional Striping)

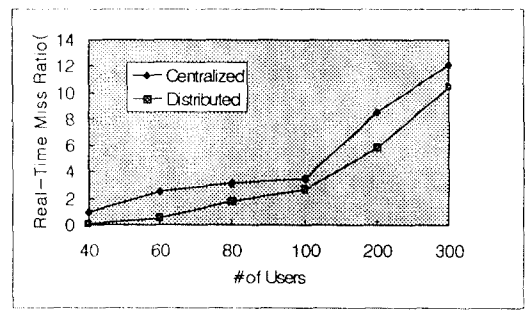


Fig 6. Prefetching = 0ms & EDF (Under Clustered Striping)

tested the performance of each scheduling algorithm under different striping techniques. EDF performs best in most situations. However, as the traffic loads are increased, the effectiveness of clustered striping is getting degenerated because of queueing delay. Nonetheless, in all the cases, real-time performance has been enhanced with clustered striping.

Figure 6 shows the performance enhancement of clustered striping compared with conventional striping technique. The real-time miss ratio is significantly reduced by parallelism especially when the system load is not heavy.

Figure 7 and 8 show the real-time performance depends on the prefetching size. Since larger prefetching size allows later deadline, increased prefetching size guarantees better performance and this gain becomes significant as traffic load becomes heavy. Especially,

Figure 7 shows the significant result that 500 users can be serviced with 95% real-time performance by employing all the proposed techniques under distributed buffering scheme. Our results show that EDF performs better than SSEDV unlike the results reported in [10]. This comes from the differences in the database size and traffic load between their experimental environment and ours.

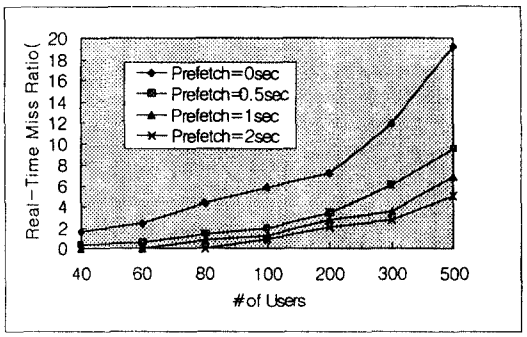


Fig 7. Prefetching 0ms & EDF (Under Clustered Striping)

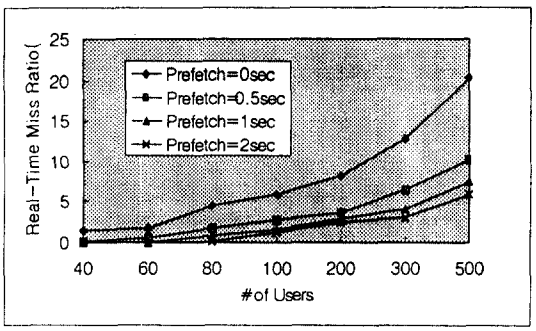


Fig 8. Prefetching 0ms & SSEDV (Under Clustered Striping)

V. Concluding Remarks

In this paper, with the goal of providing better end-to-end real-time performance in a VOD system and thereby increasing the number of satisfied requests, we have proposed and analyzed several design schemes of multimedia information storage system.

The basic ideas of these proposed schemes are suggested in the OS and DBMS areas. The major problem in striping technique is how to store the datablocks on the disks including no striping case depending on application semantics. Since most of VOD requests are I/O-bound, clustered striping introduced the notion of declustering to the disk striping to grant the parallelism at the time of disk access under multi-disk I/O subsystem. Buffering is generally required to solve the flow-control problem because of the bandwidth differences of CPU and network and prefetching is used to increase the hit ratio in virtual memory management and cache management.

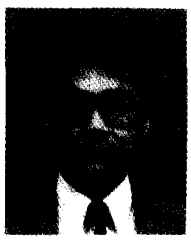
While these schemes are mapped into VOD application domain, the problems raised are not much, i.e., they are feasible solutions in building a VOD storage subsystem. The condition for the clustered striping and prefetching to be successful is the predictability of data to be accessed next. Because of the continuity requirement of VOD data, the next accessed data by prefetching can easily be anticipated. Furthermore, the goal of disk striping to increase I/O bandwidth is most likely fitted for VOD application which requires large data.

In line with our research goal, our experimental results show that with combination of a distributed buffering mechanism, data prefetching, a real-time disk scheduling policy and clustered striping technique, VOD server can support up to 500 concurrent clients with only 5% of real-time miss ratio.

This study can be extended into a number of directions. First, we plan to explore a continuous media sharing scheme using multicast facilities. Second, we are interested in an implementation of VCR-like operations with a stochastic traffic modeling on top of buffering scheme and prefetching policy.

참 고 문 헌

1. P.V. Rangan and H. M. Vin, "Designing File Systems for Digital Video and Audio", Proceedings of the 13th ACM Symposium on Operating System Principles, vol. 25, no. 5, pp. 69-79, Oct. 1991.
2. N.G. Davis and J.R. Nicol, "A technological perspective on Multimedia Computing", Computer Comm., vol. 14, no. 5, pp. 260-272, 1991.
3. W.D. Sincoskie, "System Architecture for a large scale video on demand service", Computer Networks and ISDN systems, vol. 22, pp. 155-162, 1991.
4. A.L.N. Reddy and P. Banerjee, "An Evaluation of Multiple-Disk I/O systems", IEEE Trans. Computers, vol. 38, no. 12, pp. 1,680-1,690, Dec. 1989.
5. K. Salem and H. Garcia-Molina, "Disk Striping", Fourth Int'l conference on Data Engineering, pp. 336-342, 1986.
6. G.R. Ganger, B.L. Worthington, R.Y. Hou and Y.N. Patt, "Disk Arrays: High-Performance, High-Reliability Storage Subsystems", Computer Journal, pp. 30-36, March 1994.
7. R. Abbott and H. Garcia-Molina, "Scheduling I/O requests with deadlines: A Performance evaluation", Proc. of the 11th IEEE Real-Time Systems Symposium, pp. 113-124, Dec. 1990.
8. S. Chen and J.A.Stankovic, "Performance Evaluation of Two new disk scheduling algorithms for Real-Time systems", Real-Time System Journal, Sep. 1991.
9. D.O. Anderson, R. Govindan and G.Homsy, "Abstractions for continuous media in a network window system", Proceedings of the International Conference on Multimedia Information Systems '91, pp. 273-298, Jan. 1991.
10. C.S. Lim, "Real-Time Storage and Retrieval Techniques for Continuous Media Storage server", KITE Journal, vol. 32-B, no. 11, pp. 1365-1373, Nov. 1995.
11. D.P. Anderson "Metascheduling for Continuous Media", ACM Trans. on Computer Systems, vol. 11, no. 3, pp. 226-252, August 1993.
12. C.S. Lim, S.C. Kim and J.Y. Lee, "A Distributed Buffering Scheme for Continuous Media in Wide Area Networks", The 1st Int'l workshop on IEEE Real-Time Computing Systems and Applications, pp. 168-172, Dec. 1994.
13. T.D.C.Little and A.Ghafoor, "Scheduling of bandwidth-constrained multimedia traffic", Computer Communication, vol. 15, no. 6, pp. 381-387, Aug. 1992.



임 철 수(CheolSu Lim) 정회원

1985년 2월 : 서울대학교 계산통계학과 졸업

1985년 6월~1986년 7월 : (주)데이콤 정보통신 연구소 연구원

1986년 9월~1988년 6월 : 미국 인디애나 주립대 전산과 석사

1988년 6월~1994년 6월 : (주) 아시아나항공 정보통신부 과장

1991년 9월~1995년 8월 : 서강대학교 전산과 박사

1994년 8월~1997년 2월 : (주) 신세기통신 기획실 과장

1997년 3~현재 : 서경대학교 컴퓨터공학과 전임강사

※주관심분야 : 이동 밀티미디어통신, ATM switch 구조