論文95-32B-11-1

연속미디어 저장 서버에서의 실시간 저장 및 검색 기법

(Real-Time Storage and Retrieval Techniques for Continuous Media Storage Server)

林 喆 壽 *

(CheolSu Lim)

요 약

본 논문에서는 멀티미디어 온 디맨드(MOD)저장 서버를 설계시에 고려해야할 연속미디어의 저장 및 검색과 관련된 이슈들을 제기한다. 연속 미디어 데이터의 저장과 검색이라는 MOD서버 설계의 2가지 주요한 요소를 지원하기 위해, 본 논문에서는 고성능 MOD 서브 시스템의 통합 솔루션으로 제안된 디스크 배열, 디스크 스트리핑 및 실시간 디스크 스케쥴링 기법들을 논의하고자 한다. 여기에서 제안된 클러스터 스트리핑 기법은 입/출력 병목현상을 방지함으로써, 복수의 디스크 상에서나 병렬시스템이 사용자의 재생율을 지원하는데 필요한 대역폭으로 연속 미디어 데이타를 연속적으로 검색할 수 있도록 보장한다.

Abstract

In this paper, we address the issues related to storage and retrieval of continuous media (CM)data we face in designing multimedia on-demand (MOD) storage servers. To support the two orthogonal factors of MOD server design, i.e., storage and retrieval of CM data, this paper discusses the techniques of disk layout, disk striping and real-time disk scheduling, which are integrated as a combined solution to the high-performance MOD storage subsystem. The proposed clustered striping technique enables either a multiple-disk or a parallel system to guarantee a continuous retrieval of CM data at the bandwidth required to support user playback rate by avoiding the formation of I/O bottlenecks.

I . Introduction

As the recent improvement in processor,communication and the other system resources continues, the performance of storage subsystem becomes increasingly important. Previous studies have focused on how to improve I/O system throughput [1], however multimedia applications such as movie on-demand, news on-demand, home shopping, banking, and education, and so on, require the design of high-performance storage server that can handle real-time demands. A major challenge when designing multimedia on-demand (MOD) system is to support timeliness and synchronized retrieval of continuous media (CM) such as video frames

* 正會員, 新世紀 通信 企劃室

(Strategic Planning Dept., Shinsegi Telecomm. Inc.)

接受日字:1994年11月18日, 수정완료일:1995年11月7日

and audio samples. CM service requires massive volumes of data, hence it requires high bandwidth. For example, audio data rates are from 8.0 kbps to 1.4 Mbps and the data rates for uncompressed NTSC video is about 45 Mbps. As an alternative, we may use compression techniques depending on application semantics.

To support these kinds of CM sessions concurrently, we might need aggregate bandwidths of multiple disks and efficient real-time disk scheduling techniques. We restrict ourselves to the support provided at the MOD server, with special emphasis on three critical issues: disk layout, disk striping and real-time disk scheduling. This is to find a solution to the MOD service requirement in the design of MOD storage subsystem, the major subproblems relating to disk and buffer designing strategies have to be solved. Hence, the MOD storage server must organize CM data on disk so as to guarantee that their recording and retrieval proceed at real-time rates.

Technical feasibility for MOD application support has been introduced in [2,3]. The disk layout issue has been discussed in [4-6]. Those studies pointed out that conventional layouts are not adequate for MOD services because they were designed to support only text files with small sizes and do not focus on realtime retrieval of data. Disk striping technique to improve I/O bandwidth was discussed in [4,7]. Several real-time disk scheduling algorithms were introduced and assessed in [8-11]. As the major components of MOD storage subsystem, disk striping and real-time disk scheduling issues including disk layout schemes are handled in [12-14] individually or by partial combinations. As an continuing work of [4], our study covers both real-time storage and real-time retrieval issues of MOD system and is different from the previous ones in that it takes an integrated approach and this comprehensive work is presented using experimental results.

The rest of this paper is organized as follows : In section 2, we have discussed preliminary considerations for the performance-critical factors such as disk layout schemes, disk striping and disk scheduling to get an intuition for the design of MOD service system. The clustered striping which exploits the parallelism is proposed in section 3. The results of performance evaluation are shown in section 4. Section 5 concludes with a summary and a plan for further study.

Ⅱ. Preliminary Considerations

To get an intuition for the design of MOD service system, identification of the significant factors in the end-to-end performance and to provide a valid experimental frame, we have studied the following storage subsystem issues in advance.

1. Disk Layout Schemes

Disk space utilization and disk access time(esp, seek time) depend on data block layout scheme on disk. If we call strands as a series of sequentially recorded video frames or audio samples [15], a MOD storage server stores and retrieves media strands by data blocks on disk. At this time, the issue how to distribute storage pattern is important in guaranteeing continuity requirement. In this regard, we attempt to derive the condition for continuous retrieval from the two factors, block size(M) and inter-block gap(G). The factors should be arranged so that retrieval time of a media block from disk should not exceed the playback time of the media block.

There are three schemes that can be

considered. First, random allocation simply divides and allocates a video file by data block to disks. So this scheme is easy to implement and is flexible to read or write desired data blocks. However, it is hard to guarantee the continuity requirement because of increased delay time at the time of retrieval. Second, constrained allocation scheme studied in [6,15] allocates requested data blocks out of feasible data block sets. This scheme can keep the disk access time within continuous playback requirements by reducing the switching overhead and by constraining inter-block gap of successive blocks of the strand. However, it requires some techniques like merging to utilize the disk space more efficiently. Finally, contiguous allocation scheme can also guarantee the real-time playback requirements of CM because of its fast access time. However this scheme suffers the overhead for disk reorganization to delete or add the data blocks and in addition, entails fragmentation problem. Our experiments use contiguous allocation scheme for the MOD storage server model, because our target application requires read-only long-run queries and does not suffer from the problems of this scheme.

2. Disk Striping Techniques

Disk striping, also known as disk interleaving, was originally proposed in [7] with the goal of I/O bandwidth improvement. This scheme distributes the consecutive logical data blocks among the disks of striped group in a round-robin fashion, hence provides load balancing and data transfer parallelism. We consider disk striping as a critical CM data distribution, thereby enhance the performance of real-time retrieval.

1) Independent Disk Approach

This approach accesses data without disk striping. This scheme relies on sequential read of video files on disks. This approach suffers little head repositioning overhead, i.e., seek time and rotational latency are required only at the initial stage, and this overhead is almost negligible compared to transfer time. That is,

$$\text{Total retrieval time} = (S + R)_{max} + N * Tf_{max} \quad (1)$$

where S represents seek time, R rotational latency, N the number of clients, Tf transfer time and max means the maximum time required to service the requested video file.

Whereas this scheme provides short retrieval time when supporting multiple clients, biased data-reference patterns (disk skew) may create "hot spots", resulting in starvation of the next client that attempts to access same data. If a disk bandwidth is lower than the required playback rate, this scheme cannot support continuous retrieval of CM data.

2) Conventional Disk Striping

This technique means the sequential retrieval approach in the view of user. At this moment,

$$\text{Total retrieval time} = N * ((S + R)_{max} + Tf_{max}) \quad (2)$$

and the disk access pattern by this technique is shown in figure 1.

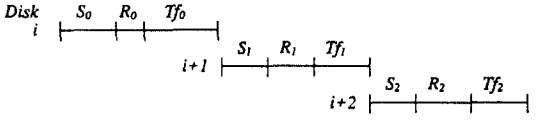

그림 1. 순차 검색의 디스크 엑세스 패턴
Fig. 1. Disk access pattern of sequential retrieval.

The first case, sequential retrieval, can avoid the starvation of the next user on the same object, but disk bandwidth and

utilization may decrease because of the increased overhead for disk head repositioning. When the system switches from one disk to another, the next disk drive incurs head repositioning overhead and hence cause the loss of bandwidth. The second case, pipelined retrieval, can also avoid the starvation problem, but the total bandwidth is the same as the independent disk approach.

3. Real-Time Disk Scheduling

While the preceeding two major issues, disk layout and disk striping, were associated with how to store data blocks on disks, disk scheduling algorithms are concerned with how to read the stored data block from disks. Hence the disk scheduling algorithms are closely tied up with data storing techniques. In this regard, there may exist some optimal disk scheduling algorithms paired with disk layout and disk striping techniques.

1) Traditional Disk Scheduling Algorithms

We have surveyed the following non-real-time disk scheduling algorithms: FCFS (First Come First Served), SSTF (Shortest Seek Time First), SCAN, C-SCAN (Circular SCAN). Traditional seek optimization algorithms such as SSTF or SCAN improve disk-arm utilization by serving requests close to the current arm position. The C-SCAN algorithm works in the same way as SCAN except that it always scans in one direction. In these algorithms, the service queue of incoming queries is ordered by the requests' relative positions on the disk.

However, such algorithms might not be suitable for real-time MOD applications such as movie on-demand, news on-demand, shopping, and so on because they do not account for timing constraints or deadlines in their scheduling decision. This induction can also be observed by the experimental results from [9,10].

2) Real-Time Disk Scheduling Algorithms

Our real-time disk scheduling policies focus on minimizing the number of requests that miss their deadlines Any requests that have missed their deadlines would not be serviced. We have implemented three algorithms after considering the performance results from [9].

EDF (Earliest Deadline First) schedules a request with the earliest deadline. The original EDF algorithm [11] assumed that tasks were preemptible with zero preemption cost. However, current disks are not preemptible [13]. EDF gives each real-time request a deadline and serves requests in that order.

FD-SCAN (Feasible Deadline SCAN) algorithm is based on SCAN algorithm, but introduced the notion of deadline feasibility [10]. At the point of scheduling decision, all requests are checked to determine which has the earlist feasible deadline. The track location of the request with the earliest feasible deadline is used to determine the scan direction.

SSEDV (Shortest Seek and Earliest Deadline by Value) algorithm was introduced in [9]. The basic idea behind this is to give the disk I/O request with the earliest deadline a high priority, but if a request with a larger deadline is very close to the current arm position, then it may be assigned the high priority.

Ⅲ. Clustered Striping

This technique enables either a multiple-disk or a parallel system to guarantee a continuous retrieval of an object at the bandwidth required to support user playback

rate because it avoids the formation of bottlenecks by striping an object across the clusters.

1. Protocol and Algorithm

To describe the protocol we use the following terms :

- object (V) - a media type that should be presented continuously such as movie file.
- subobject (V_i) - a contiguous portion of the object multiplexed into multiple disks. We assume that the size of each subobject is the same.
- cluster (C_i) - a group or subgroup of striped disks.
- datablock - in a cluster, a subobject is declusterd into several data blocks.
- degree of declustering (M) - the number of disks in a striped group.
- stride (k) - the distance (i.e.,number of disk drives) between the first data block of subobject (V_i) and the first of subobject (V_{i-1}).

1) Each disk repositions its head within Tswitch(C).
2) Each disk starts fetching requested data block with Tblock(Vi).
3) Once a datablock of each disk is read into memory, start the synchronized transmission of datablock into subscribers' workstation.
4) Each disk performs datablock fetching and transmisssion concurrently.

In the protocol, upon activation of the disk drives in a cluster, each disk drive performs the first two steps and repeats all steps until all the subobjects of object (V) are played back. Let B_{disk} denote a disk bandwidth and B_{play} denote a required bandwidth for continuous playback. Also let $T_{switch}(C)$ denote the worst case delay time within cluster C to reposition each disk head, which is the sum of the maximum seek time and maximum rotational delay of each disk. Let $T_{block}(V_i)$ be the time required to fetch a data block into main memory buffer.

To show that total running time for this protocol can be completed within linear order of time, we express this protocol in

[Algorithm: Clustered_Striping]

```
begin
  for i := 1 to Nc, in parallel
     if (Busy[i] != 1)
     then for j :=1 to Nd, in parallel
       begin /* both head repositioning & datablock fetching concurrently */
               strip[i][j].head := strip[i][j].track;
               strip[i][j].mbuffer_block := strip[i][j].datablock;
       end
     else
            enqueue the request
  for i := 0 to (Nc-1)  /* datablock transmission from MM buffer to IS */
     for j := 0 to (Nd-1)
         strip[i][j].ibuffer_block := strip[i][j].mbuffer_block;
     end
  end
end
```

[Protocol: Clustered Striping]

algorithmic convention. In this algorithm, it requires that CM data be retrieved from striped disks in a round-robin fashion within each cluster and placed into the main memory buffer(mbuffer_block). Then they are transferred to the client's workstation through the buffers of intermediate server(ibuffer_block) at the real-time rate.

2. Data Retrieval Time

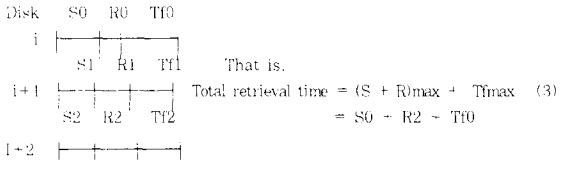

그림 2. 동시 검색 패턴
Fig 2. Concurrent retrieval pattern.

This technique can utilize the aggregate bandwidth of at least $\lceil B_{play} / B_{disk} \rceil$ disk drives to support the continuous playback of an object, which requires larger bandwidth than the given disk bandwidth. As Figure 3 shows, this technique greatly reduced the loss of disk bandwidth by continuous data transfer from disk clusters to memory buffer. Note that to bring this benefit, this approach requires larger buffer capacity whose minimum buffer requirement is defined as:

$$B_{disk} * (T_{switch} + T_{fetch}) \quad (4)$$

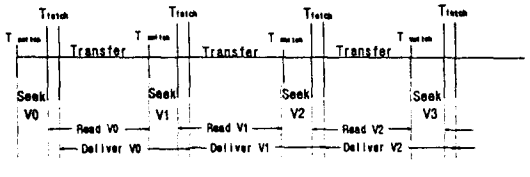

그림 3. 디스크 클러스터에서의 연속적인 데이터 전송
Fig. 3. Continuous data transfer from disk clusters.

where B_{disk} represents bandwidth of a single disk, Tswitch the maximum duration of the first step of protocol, T_{fetch} is the time required to read a sector into memeory.

Ⅳ. Performance Evaluation

1. Simulation Parameter Settings

To implement our simulation system, we used SMPL which is an event-oriented simulation language. To gather statistics, 200,000 queries are generated for each experiment and each result is calculated with 95% confidence interval. The arrival process is assumed to be Poisson with rate λ, where λ varies from 22 to 40 to represent different workloads. For workload control, we varied the number of incoming clients and slack times are selected uniformly within [10,30] ms which is tolerable lagging time in real-time systems [9-11] and other parameter values are from Sun Micro DSCSI disk model.

The formulae used for deadline setting is as follows:

$$\text{Deadline} = \text{Arrival\_time} + \text{Average\_access\_time}(n) + \text{Slack\_time} \quad (5)$$

where an access time for a request is expressed by

$$\text{Average access time}(n) = \text{seek}(n) + \text{rotational delay} + \text{transfer time} \quad (6)$$

where n is track distance, rotational latency is uniform in [0,16.7], transfer time is constant as 1.25 ms/frame because we assume that disk bandwidth is 5 MB/sec and each video frame size is 6.25 KB/frame.

2. Performance Results

And all the results are presented in Figure 4.1 through 4.6. The effectiveness of disk striping technique itself is already shown in [1-4], and so we do not discuss this point in this

paper further. Figure 4.1 and Figure 4.2 show the effectiveness of EDF algorithm under clustered striping. As figures show, real-time performance is greatly improved especially when the traffic load is not heavy. For example, when the number of concurrent user is 100, approximately 100% of performance gain is possible.

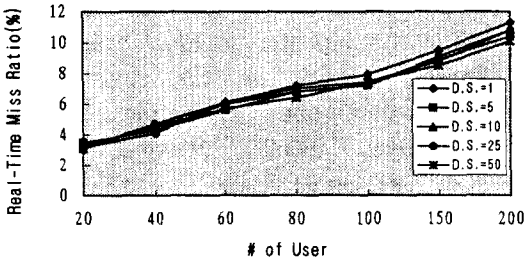

그림 4.1. 일반적인 스토리핑하에서의 EDF 성능
Fig. 4.1. EDF Performance under Conventional Striping.

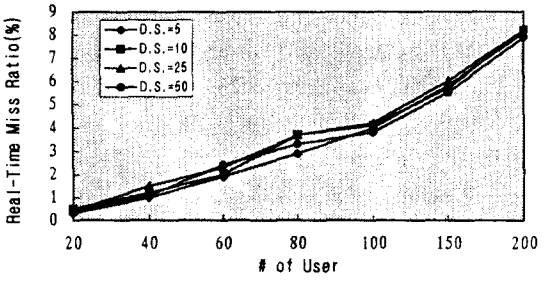

그림 4.2. 클러스터 스트리핑하에서의 EDF 성능
Fig. 4.2. EDF Performance under Clustered Striping.

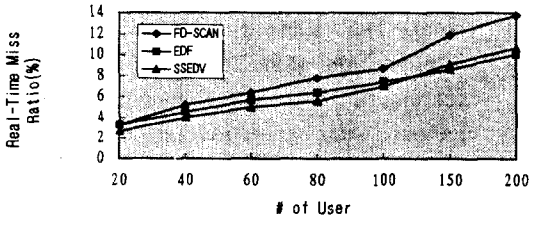

그림 4.3. 일반적인 스트리핑하에서 각 알고리즘의 성능
Fig. 4.3. Performance of Each Algorithm under Conventional Striping.

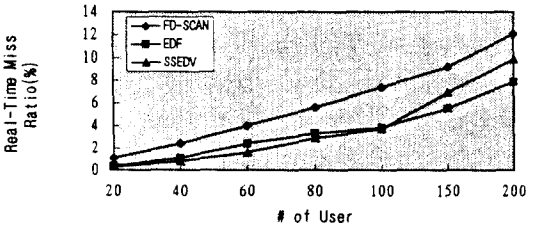

그림 4.4. 클러스터 스트리핑하에서 각 알고리즘의 성능
Fig. 4.4. Performance of Each Algorithm under Clustered Striping.

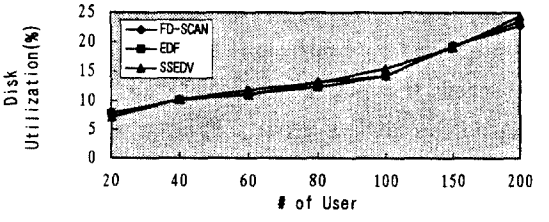

그림 4.5. 일반적인 스트리핑하에서 디스크 이용율(%)
Fig. 4.5. Disk Utilization under Conventional Striping(%), DS=50.

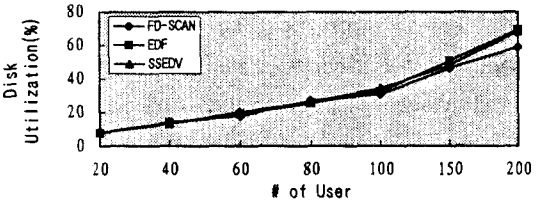

그림 4.6. 클러스터 스트리핑하에서 디스크 이용율(%)
Fig. 4.6. Disk Utilization under Clustered Striping(%), DS=50 & M=5.

The next two figures, Figure 4.3 and Figure 4.4, show the performance of each algorithm under clustered striping and EDF performs best in most situations because it imbues deadline concept most. As the traffic loads are increased, the effectiveness of clustered striping is decreased because of queueing delay. However, if we increase the number of disks in the experimental frame, more than

50, this degenerated effect can be overcome.

The last two figures, Figure 4.5 and Figure 4.6, show the disk utilization ratio. For example, more than 100 users, the disk utilization under clustered striping becomes approximately double. Hence this means better real-time performance if the other conditions are the same.

V. Conclusion

Through this work, we have motivated the significance of real-time storage and retrieval techniques for continuous media data and presented the effectiveness of our combined solution for MOD system. This on-going work can also be extended to in a number of directions for future research. First, we will elaborate the MOD storage subsystem to accommodate VCR-like features such as rewind, fast-forward, pause, and so on, which requires variable playback bandwidth. Second, we will analyze how the buffering capacity of MOD server affects the performance of scheduling algorithms. Finally, this can also be extended to the study of distributed buffering scheme to guarantee end-to-end performance of MOD services on Wide Area Networks.

References

[1] D.A. Patterson, G. Gibson and R.H. Katz, "A Case for Redundant Arrays of Inexpensive Disks (RAID)", ACM SIGMOD Conference, 1988, pp. 109-116.

[2] N.G. Davis and J.R. Nicol, "A technological perspective on Multimedia computing", Computer Comm., 14(5), 1991, pp. 260-272.

[3] W.D. Sincoskie, "System Architecture for a large scale video on demand service", Computer Networks and ISDN systems 22, North-Holland, 1991, pp.155-162.

[4] CheolSu Lim and SungChun Kim, "Efficient Storage Subsystem Configuration for Video On-Demand Server", Int'l conference on Distributed Multimedia Systems and Applications, August 15-17 1994, Honolulu, Hawaii, pp.22-26.

[5] P.V. Rangan and H. M. Vin, "Designing File Systems for Digital Video and Audio", Proc. of the 13th ACM Symposium on Operating System Principles (SOSP '91).vol.25, no.5, Oct.1991, pp.69-79.

[6] P.V. Rangan and H.M.Vin, "Efficient Storage Techniques for Digital Continuous Multimedia", IEEE Trans. on Knowledge and Data Engineering, August 1993.

[7] K. Salem and H. Garcia-Molina, "Disk Striping", Fourth Int'l conference on Data Engineering, 1986, pp. 336-342.

[8] M.J. Carey and Jauhari, "Priority in DBMS Resource Scheduling", Proc. of the 15th VLDB Conf., 1989.

[9] S. Chen and J.A. Stankovic, "Performance Evaluation of Two new disk scheduling algorithms for Real-Time systems", Real-Time System Journal, Sep. 1991.

[10] R. Abbott and H. Garcia-Molina, "Scheduling I/O requests with deadlines : A Performance evaluation", Proc. of the 11th IEEE Real-Time Systems Symp., Dec. 1990, pp 113-124.

[11] S. chen and Don Towsley, "Scheduling Customers in a Non-Removal Realtime System with an Application to Disk scheduling", COINS Tech. Reports, 92-58, Univ of Massachusetts, 1992.

[12] S. Berson, R. Muntz, S. Ghandeharizadeh, Xiangyu Ju, "Staggered Striping in Multimedia Information Systems", ACM SIGMOD Conf., May 1994, pp. 79-90.

[13] A.L.N. Reddy and J.C. Wyllie, "I/O Issues in a Multimedia System", Computer Journal, March 1994, pp 69-74.

[14] G.R. Ganger, B.L. Worthington, R.Y. Hou and Y.N. Patt, "Disk Arrays : High-Performance, High-Reliability Storage Subsystems", Computer Journal, Mar 1994, pp. 30-36.

[15] P.V. Rangan, H.M. Vin and S. Ramanathan, "Designing an On-Demand Multimedia Service", IEEE Comm. Magazine, vol.30, no.7, July 1992, pp. 56-65.

저 자 소 개

林 詰 壽(正會員)

1981년 3~1985년 2월 서울대학교 계산통계학과 학사. 1985년 6월 ~ 1986년 7월 데이콤 정보통신 연구소 근무. 1986년 9월 ~ 1988년 5월 미국 Indiana Univ. 전산학 석사. 1988년 6월 ~ 1994년 8월 (주)아시아나항공 시스템사업부. 1991년 9월 ~ 1995년 8월 서강대학교 전산공학 박사. 1994년 8월 ~ 현재 (주)신세기통신 기획실 과장.