멀티미디어 파일 시스템을 위한 효율적 버퍼 관리

홍철의*

The Efficient Buffer Management for a Multimedia File System

Chuleui Hong*

이 논문은 2011년도 상명대학교 연구비를 지원받았음

요 약

멀티미디어 데이터는 연속적이고 대용량이며 실시간 특성을 가지고 있으므로 주문형 비디오(VOD)를 서비스하는 멀티미디어 서버에서는 잦은 디스크 입출력 및 프로세서와 디스크 사이의 속도 차이로 인한 시스템 성능 저하가 발생한다. 따라서 멀티미디어 데이터에 대한 디스크 접근을 줄이는 효율적인 버퍼 관리가 필요하다. 본 논문에서는 멀티미디어 서버에서 연속적이고 실시간 특성이 있는 비디오 데이터를 대상으로 버퍼의 적중률을 높여 디스크 접근을 줄이고, 동시에 서비스 받는 사용자수를 늘리며 시스템 전반에 걸친 버퍼 이용도를 높이는 버퍼 관리 방법에 대하여 연구한다. 이를 위해 여러 가지 자원 관리 알고리즘 및 정책변화에 대한 시뮬레이션을 통하여 다양한 환경하에서의 각 알고리즘의 성능을 비교 분석한다.

ABSTRACT

The multimedia data for video-on-demand(VOD) service has large, continuous and real time characteristics. The frequent disk I/O operations takes much time and decrease the system performance in multimedia services. Therefore the efficient buffer management is needed in order to reduce the disk accesses to multimedia data. This paper addresses how to increase the buffer hit ratio and the number of users in a multimedia service like VOD by increasing the utility of buffer. This paper also simulated various resource management algorithms and strategies and evaluated, compared and analyzed their performances.

키워드

멀티미디어 파일 시스템, VOD 서버, 버퍼 관리, 디스크 스케줄링, 시뮬레이션

Key word

Multimedia File System, VOD Server, Buffer Management, Disk Scheduling, Simulation

* 정회원 : 상명대학교(교신저자,hongch@smu.ac.kr)

접수일자 : 2011. 03. 14
심사완료일자 : 2011. 05. 03

Ⅰ. 서 론

멀티미디어 데이터는 숫자나 문자 등 텍스트뿐만 아니라 비디오, 오디오 등으로 구성된 복합 미디어 데이터를 말한다. 멀티미디어 데이터는 연속적으로 표현되어야 하고, 대용량이며, 강한 실시간 특성을 가진다. 또한 멀티미디어 데이터는 단일 미디어 데이터들 사이에 시간 및 공간적인 동기를 유지해야 한다는 특성을 가지고 있다. 일반적으로 멀티미디어 시스템은 멀티미디어 데이터를 저장, 접근하는 서버와 데이터를 요구하는 사용자 시스템 그리고 서버와 사용자 시스템을 연결시켜 주는 통신망으로 구성된다[1].

멀티미디어 데이터 중에서 VOD(주문형 비디오, Video On Demand) 시스템에서 다루는 오디오와 비디오 데이터는 정해진 시간 안에 연속적 처리가 요구되는 특성을 가지고 있다. 멀티미디어 데이터의 처리 요구가 급증되면서 대용량의 멀티미디어 데이터의 효율적 관리를 위한 저장 관리자에 대한 많은 연구가 진행되어 왔다.

광디스크와 같은 고속 대용량 저장장치의 개발과 고속 통신망 및 처리기, 압축 기법 등을 사용하여 대용량 멀티미디어 데이터를 실시간 처리하고 있다. 그러나 이러한 기술의 발달에도 불구하고 멀티미디어 데이터의 디스크 입출력은 처리기와 디스크간의 속도차이로 인해 입출력 요구 때마다 디스크에 접근하게 되면, 디스크의 느린 데이터 전송 속도 때문에 시스템의 응답시간과 처리능력이 나빠지게 된다[2]. 따라서 서버와 클라이언트 사이에서 디스크 접근을 줄이기 위한 효율적인 버퍼 관리가 필요하다.

VOD와 같은 멀티미디어 서비스의 특징은 데이터에 대한 접근이 순차적이고 접근 속도가 일정하므로 데이터 접근을 예측할 수 있다. 따라서 사용자가 탐색 속도를 변경하지 않는 한 저장된 데이터의 접근 속도가 일정하므로 다수의 사용자 경우 버퍼에 존재하는 데이터를 다른 사용자가 접근할 시점을 예측할 수 있으며[3] 사용된 인기 비디오의 데이터 블록이 재사용될 확률이 높다는 특징을 가지고 있다[4]. 따라서 VOD와 같은 멀티미디어 서비스에서의 효과적인 버퍼 관리는 디스크 입출력을 줄여 시스템 성능을 향상시킬 수 있다.

본 논문에서는 VOD 서비스 전용의 멀티미디어 서버에서 연속적이고 실시간 특성을 가지는 비디오 데이터를 대상으로 버퍼의 적중률을 높여 디스크 접근을 줄이고 데이터 접근시간에 대한 실시간을 만족시키며 동시에 접근하는 사용자 수를 높일 수 있는 시스템 전반에 걸쳐서 버퍼 이용도를 높이는 버퍼 관리 방법에 대하여 연구한다. 즉, buffering, prefetching, disk I/O scheduling 등 다양한 자원 관리 정책들이 시스템 성능에 미치는 영향을 시뮬레이션을 통하여 알아본다. 또한 여러 자원관리 정책에 따라 데이터 블록 크기 및 서버 메모리 크기의 변화가 최대 사용자 수에 미치는 영향을 비교 분석한다.

Ⅱ. 관련 연구

R. Kumar[5]는 동일 비디오에 대해 서로 다른 시점에 요청이 들어오면 효율적으로 복사본을 만들어 제공하기 위하여 버퍼를 사용하였다. R. Tewari[6]은 구간 캐시 방식을 제안하였다. 이 방법은 비디오 상영 시간을 일정한 구간으로 나누어 구간별로 버퍼링하는 방법으로 서비스 요청이 들어올 경우 해당 비디오 블록이 속하는 구간을 디스크로부터 읽어서 캐싱한다. 그러면 일정 시간 안에 같은 비디오를 요청하는 사용자들은 이미 캐시된 데이터를 읽게 된다. 그러나 이러한 방법들은 같은 비디오를 요청하는 사용자가 많지 않거나 서비스되는 비디오 종류가 많을 경우 메모리를 많이 요구하게 되고 비효율적이다.

K. Gopala[7]은 비디오의 인기도에 따라 할당되는 버퍼의 크기를 정하였으며 비디오의 초기 부분을 버퍼에 미리 할당하여 연결 초기에 발생하는 지연을 방지하고자 하였다. 반면에 W. Shi[8]은 버퍼 관리를 위해 멀티미디어 데이터의 재 참조 확률을 이용하지 않았다. 대신 다수의 서비스 요청들 간에 버퍼를 적절히 할당하여 입출력 시에 발생하는 병목현상을 해결함으로써 전체 시스템 성능을 향상시키고자 하였다.

C. Chen[9]는 멀티캐스트의 라우팅 패스상의 노드에서 버퍼를 사용하여 같은 비디오 클립을 서로 다른 시간에 서로 다른 사용자에게 제공하는 기법을 사용하였다. 이러한 방법은 최적의 라우팅 패스를 선정하는 문제가 NP로서 실시간에 최적의 패스를 구성하기 어렵다.

C. Freedman[10]은 기존 버퍼 관리 알고리즘보다 개선된 prefetching, disk scheduling 등을 제안하여 디스크 요구를 최소화하도록 하였다. 또한 시스템 설계 시 영향을 미치는 변수를 다양하게 변화시킴으로써 지원 가능한 최대 사용자를 늘리고자 하였다.

Ⅲ. 멀티미디어 데이터 버퍼 관리 기법

본 연구에서는 관련 연구들에서 제안한 다양한 멀티미디어 데이터 버퍼 관리 기법들을 시뮬레이션한다. Global LRU(Least Recently Used) algorithm[2]는 가장 최근에 참조된 페이지를 하나의 큐의 끝에 넣고 새로운 블록이 요구되면 큐의 처음부터 검색하여 적합한 것을 할당한다. 그러나 이 알고리즘은 참조된 페이지와 선취(prefetch)된 페이지의 구별이 불가능한 단점이 있다.

이러한 단점을 개선한 선취 알고리즘은 2개의 독립된 LRU 큐를 두고 하나는 선취된 블록들만 넣고, 다른 하나는 참조된 블록들을 넣는다. 이때 선취된 블록 중에 참조된 블록들은 참조 큐에 옮긴다. 그리고 주기적으로 선취 큐 안에 있는 블록 중에 오랫동안 참조되지 않은 블록들은 참조 큐로 옮긴다. 새로운 블록이 요구될 때는 참조 큐에서 먼저 적당한 페이지를 찾고 없으면 선취 큐에서 찾는다.

엘리베이터 디스크 스케줄 알고리즘(elevator disk scheduling algorithm)은 가장 안쪽 실린더에서 시작해서 바깥쪽으로 헤드가 이동을 하면서 이동방향에 있는 요청들만을 처리하고 제일 바깥쪽에 도달하면 다시 역 방향으로 마찬가지 방식으로 요청을 처리한다.

엘리베이터 알고리즘을 개선한 실시간 디스크 스케줄 알고리즘(real time disk scheduling algorithm)은 각각의 디스크 요청을 예정된 상영 시간에 따른 우선순위 클래스로 나누고 각 우선순위 안에서 엘리베이터 방식을 적용해서 그 클래스 안에 있는 요청들을 처리한다. 우선순위 결정 시 적용되는 기준은 현재 시간과 예정된 상영시간과의 차이이다. 각각의 우선순위 안의 요청들이 끝날 때마다 마찬가지의 우선순위 결정 규칙을 적용해서 아직 처리가 안 된 요청들의 우선순위를 조정한다.

파일 striping은 비디오 파일을 일정한 크기의 블록으로 나누어 각각의 블록을 각 서버에 속한 디스크별로 배분을 해서 저장 시킨다. 파일 striping의 장점은 첫 번째, 순차적으로 요청되는 블록을 서로 다른 디스크들로부터 동시에 읽어올 수 있으므로 사용자의 요청이 많은 비디오들에 대해서도 각 요구에 대해 비동기적으로 처리를 할 수 있다. 만약 완전히 모든 비디오들을 striping했을 경우 시간에 따라 변하는 비디오의 접근 빈도(인기 순위)를 위하여 디스크를 재구성하지 않아도 사용자 요구의 부하 변화를 수용할 수 있으며 시스템 상의 모든 디스크들의 입출력에 대한 부하균형(I/O load balance)을 유지할 수 있다.

본 연구에서는 파일 striping기법을 이용하여 디스크 블록의 크기를 256Kbytes단위로 증가 시켜 디스크 블록 크기가 각 자원 관리 알고리즘과 시스템 성능에 미치는 영향에 대해 알아보고 디스크 블록의 최적화 크기를 정의한다.

서버 메모리는 클수록 좋은 성능을 발휘하겠지만 비용문제로 무한정 늘릴 수 없으므로 이러한 문제를 해결하기 위해서는 한정된 용량의 메모리에서 적중률 및 이용도를 높여야 한다.

Ⅳ. 시뮬레이션

본 논문에서는 시뮬레이션을 수행할 멀티미디어 서버로 많은 논문에서 인용되고 있으며 구조가 명확하게 정의되어 있는 SPIFFI VOD System[10] 구조를 선택하였다. SPIFFI VOD System은 각각의 서버당 1개의 CPU, Video data의 prefetching(선취)과 buffering을 위한 메모리, 1개의 Network interface card, 일련의 디스크의 집합으로 구성된 멀티미디어 시스템이다. 이 시스템의 주요 특징으로는 file striping, decentralized file access, buffering, prefetching을 들 수 있다.

본 연구에서는 SMPL/C언어를 사용하여 시뮬레이션을 수행하였다. 비디오의 개수는 64개로 하였고 서비스를 요청한 각 터미널은 초기에 임의의 위치에서 비디오를 시작하게 된다. 비디오의 시청이 끝나면 해당 터미널은 임의로 새로운 비디오를 선택하여 전송을 요

청하게 된다.

본 시뮬레이션에서는 데이터 탐색시간을 줄이기 위하여 Hash queue[2]를 사용하였으며 전체 비디오 접근 확률은 Uniform분포와 Zipfian분포 두 가지로 구분하였다. Zipfian분포란 비디오 데이터의 인기도에 따른 접근 확률을 나타내는 분포이다. 시뮬레이션 결과를 통하여 시스템의 자원 관리(buffering, disk I/O scheduling)알고리즘과 I/O Job size(buffer/block size) 및 다양한 정책 변화가 시스템 성능에 미치는 영향을 알아본다.

시뮬레이션 설계변수는 다음과 같다.

1. 서버 수 : 4개
2. 서버 당 CPU 수: 1개
3. 서버 당 디스크 수 : 4~16개
4. 비디오 비트 속도 : 4 Mbits/second
5. 비디오 길이 : 60분/편
6. 해시 큐 수
   - 선취 큐 : 256개
   - 참조 큐 : 256개
   - 디스크 캐시 큐 : 64개
7. 디스크 용량 : 7 Gbytes
8. 디스크 당 비디오 수 : 4편
9. 서버 버퍼 크기 : 128~1,024Kbytes
10. 터미널 버퍼 크기 : 2 Mbytes
11. 디스크 캐시 크기 : 40 Mbytes
12. 페이지 크기 : 8 Kbytes
13. CPU 속도 : 40 MIPS
14. 메모리 접근 시간 : 120 ηseconds
15. I/O 요구 명령 수 : 20,000 instrs
16. 메시지 송신 명령 수 : 6,800 instrs
17. 메시지 수신 명령 수 : 2,200 instrs
18. 8Kbytes 페이지 복사 : 8,192 instrs
19. 최대 디스크 회전 시간 : 16.66 mseconds
20. 최대 디스크 탐색 시간 : 20.0 mseconds
    - 탐색 시간 분포 : triangular distr.
    - 탐색 계수 : 0.283
21. 디스크 전송 시간 : 7.4 Mbytes/sec
22. 디스크 settle 시간 : 0.75 mseconds
23. 네트워크 전송 시간
    - 블록 요구 크기 : 64 Bytes
    - 잠재 시간 : 5.0 μseconds
    - 전송률 : 0.04 μsec/byte
24. 시뮬레이션 시간 : 3시간(Hours)
    - 초기 편향 시간 : 30분(Minutes)

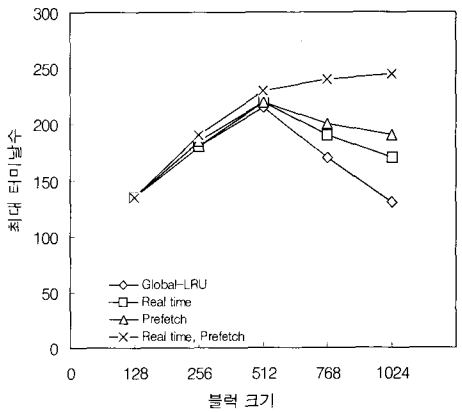

그림 1. 데이터 블록 크기에 따른 최대 터미널 수
Fig. 1 Maximum Terminals as Data Block Size Increases

그림 1에서는 비디오 데이터 블록의 크기를 128Kbytes~1,024Kbytes 범위에서 증가 시키면서 각 알고리즘의 성능을 비교하였다. 비디오 접근 확률은 Zipfian분포로 가정하였다. Global-LRU, Real time disk scheduling, Prefetching algorithm은 블록 크기가 512Kbytes일 때 최대 성능을 발휘하였고 블록 크기가 그 이상 증가하면 서비스 터미널 수가 감소하였다. 그러나 Real time disk scheduling, Prefetching algorithm을 병행하여 사용한 경우에는 블록 크기를 512Kbytes이상으로 증가 시켜도 성능이 계속 향상되었다.

데이터 블록 크기 1,024Kbytes일 때 Global-LRU algorithm에 비해 Real time disk scheduling algorithm은 약 30%, Prefetching algorithm은 약 46%, Real time disk scheduling & Prefetching algorithm은 약 90%의 성능 향상을 보였다.

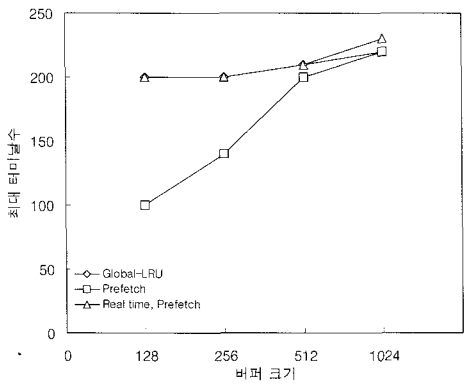

그림 2. 데이터 버퍼 크기에 따른 최대 터미널 수
Fig. 2 Maximum Terminals as Data Buffer Size Increases

그림 2에서는 그림 1과 같은 환경 하에서 버퍼의 크기를 128Mbytes~1,024Mbytes 범위에서 변화 시키며 시스템 성능을 측정하였다. 실험결과 버퍼 크기는 512Mbytes이상 되어야 각 알고리즘과 시스템 성능이 향상됨을 알 수 있었다. Real time disk scheduling과 Prefetching algorithm을 병행하는 것이 가장 효과가 컸으나 Prefetching algorithm의 경우 버퍼 크기가 512Mbytes일 경우 Global-LRU algorithm에 비해 오히려 성능이 50%정도 저하되게 된다. 이것은 버퍼 크기가 512Mbytes 이하일 경우 Prefetch에 필요한 적당한 버퍼 메모리 공간을 확보하기 어렵기 때문이다.

[표 1]은 버퍼를 Global-LRU algorithm과 Zipfian분포로 할당했을 때의 실험 결과를 비교하고 있다. 비디오 분포가 Zipfian분포일 경우 Global-LRU 버퍼를 사용했을 때와 Zipfian버퍼를 사용했을 때 시스템의 성능에 차이가 없음을 알 수 있다.

표 1. 버퍼 할당 정책과 비디오 선택 분포에 따른 최대 터미널 수
Table. 1 The Maximum Number of Terminals according to the Strategy of Buffer Allocation and the Distribution of Video Access

비디오 선택 / 버퍼정책	Uniform	Zipfian
Global	190	210
Zipfian	NA	210

이것은 비디오 접근 분포가 Zipfian 분포일 경우 버퍼에서 각 비디오가 접근 확률만큼 영역을 차지하게 되기 때문이다.

[표 2]는 Global LRU buffer, Zipfian 비디오 분포에서 Real time disk scheduling algorithm과 데이터 블록 선취 개수에 대한 시스템 성능 측정 결과를 보여주고 있다. Prefetching algorithm은 Real time disk scheduling algorithm과 함께 사용할 경우 더 효과적임을 알 수 있다. 또한 1개 이상의 데이터 블록 Prefetch는 데이터 블록끼리의 경쟁을 발생시켜서 성능 향상에 비효율적이다.

표 2. 디스크 스케줄의 실시간 유무 및 선취 개수에 따른 최대 터미널 수
Table. 2 The Maximum Number of Terminals according to the Real-Time Scheduling and the Prefeching number

선취 개수 / 실시간	0	1	2
Non-Real Time	210	210	200
Real Time	190	220	220

[표 3]은 디스크 캐시의 효용을 실험한 결과이다. 비디오 데이터의 대용량 특성상 40Mbytes 용량의 디스크 캐시는 캐시 적중률이 극히 낮을 수밖에 없으므로 디스크 캐시의 효율은 거의 제로에 가깝다고 할 수 있다.

표 3. 디스크 캐시 유무에 따른 최대 터미널 수
Table. 3 The Maximum Number of Terminals according to the Disk Cache

알고리즘 / 디스크 캐시	G-LRU	G-LRU P1	G-LRU RT, P1
Cache	210	210	220
No Cache	190	210	220

[표 4]는 비디오가 Zipfian분포를 이룬다고 가정하고 Global-LRU 할당 버퍼와 Zipfian 할당 버퍼의 시스템 성능을 측정한 것이다. Global-LRU 버퍼의 경우 Real time disk scheduling과 Prefetching으로 성능이 약 5%정도 향상되었다.

그러나 Zipfian 버퍼는 Prefetching을 사용하였을 때 성능이 약 10% 저하되고 Real time disk scheduling과 Prefetching 두 알고리즘을 사용했을 때에도 성능이 향상되지 않는다. 이것은 [표 1]에서와 같이 비디오 분포가 Zipfian분포일 경우 인기도가 높은 비디오가 버퍼 영역을 많이 차지하게 되므로 버퍼를 Zipfian방법으로 재할당할 필요가 없기 때문이다.

표 4. 버퍼 할당 정책에 따른 최대 터미널 수
Table. 4 The Maximum Number of Terminals according to the Strategy of Buffer Allocation

알고리즘 / 버퍼정책	G-LRU	G-LRU P1	G-LRU RT, P1
Global LRU	210	210	220
Zipfian	210	190	210

V. 결 론

본 논문에서는 멀티미디어 서버에서 많은 사용자에게 glitch없는 VOD서비스를 동시에 지원하기 위해 필요한 자원 관리 알고리즘들의 정책 변화 효과 및 최적 데이터 블록/버퍼 크기에 관해 연구하였다. VOD시스템이 4개의 서버로 구성되어 있고, 각 서버는 4개의 디스크와 1Gbytes 용량의 버퍼를 가지고 있다는 가정 하에 자원 관리 정책들이 시스템 성능에 미치는 영향에 대해 알아보았다. 또한 데이터 블록 크기 및 서버 메모리 크기의 변화가 서비스 가능한 최대 사용자수에 미치는 영향에 대해서 연구하였다.

시뮬레이션 결과 블록 크기 1Mbytes, 버퍼 크기 1Gbytes에서 prefetching algorithm과 real time disk scheduling algorithm을 병행하여 사용하였을 때 시스템 성능이 최대가 됨을 알 수 있었다. 또한 데이터 블록의 선취 개수 및 디스크 캐시 유무는 시스템 성능에 별 영향을 미치지 않는 다는 결과를 보였다. VOD와 같은 멀티미디어 서버에서는 사용자가 요구하는 다양한 기능의 서비스를 제공하는 동시에 지원 가능한 최대 사용자 수를 늘리는 것이 가장 중요하다.

이러한 VOD서버를 효율적으로 구현하기 위해서는 본 논문에서 고려하지 않은 고속 전 후진 등의 특수 기능을 사용자가 요구할 때에도 QoS를 보장하는 서버 자원 관리 시스템에 대한 총체적인 연구가 필요하다.

참고문헌

[1] Renu Tewari, Harrick M. Vin, Asit Dan & Dinkar Sitaram, " Resource Based Caching for Web Servers", Tech report of CS of U. of Texas at Austin, 1997

[2] Maurice J. Bach, "The Design of The UNIX Operating System", Prentice-Hall, 1986, pp.38-59

[3] D. Rotem and J. L. Zhao, " Buffer Management for Video Database Systems", Proceeding of Data Engineering, 1995, pp.439-448

[4] D. Gemmell, H. Vin, D. Kandlur and P. Rangan, " Multimedia Storage Server: A Tutorial and Survey", IEEE Computer, 1995

[5] R. Kumar and K. Ganesan, "A Reliable Replication Strategy for VoD System using Markov Chain", International Journal of Computer Science and Information Security, Vol. 6, No. 2, 2009

[6] Renu Tewari, Asit Dan, "Buffering and Caching in Large-Scale Video Servers", Tech-Reports, Texas Austin, 1995

[7] K. Gopala and P. Jayarekha, "A Strategy to enable Prefix of Multicast VoD through Dynamic Buffer Allocation", International Journal of Computer Science Issues, Vol. 7, No. 2, 2010

[8] Weifeng Shi etal, "Trading memory for disk bandwidth in Video on Demand", Tech report, Dept. of CS at USC, 1997

[9] C. Chan, S. Huang, and T. Su, "Buffer-assisted on-demand multicast for VOD applications", Multimedia Systems, Vol. 12, No. 2, 2006

[10] C. S. Freedman, "The SPIFFI Scalable Video-n-Demand System", SIGMOD, 1995

저자소개

홍철의(Chuleui Hong)

1989년 미국 New Jersey Institute of Technology 전산학(석사)
1992년 미국 Univ. of Missouri - Rolla 전산학(박사)

1992년~1997년 한국전자통신연구원 선임연구원
1997년~현재 상명대학교 컴퓨터과학부 교수
※관심분야 : 분산시스템 및 알고리즘, 최적화 기법, 멀티미디어 응용