스트리밍 미디어를 위한 캐쉬 화일 시스템

오재학[O*] 차호정[**]
광운대학교 컴퓨터과학과
**연세대학교 컴퓨터과학과
ojh@cs.kwangwoon.ac.kr hjcha@cs.yonsei.ac.kr

A Cache File System for Streaming Media

Jaehak Oh* Hojung Cha**
*Dept. of Computer Science, Kwangwoon University
**Dept. of Computer Science, Yonsei University

요 약

본 논문에서는 스트리밍 미디어의 효율적인 캐싱과 재전송을 목적으로 범용 화일 시스템에 기반한 사용자 수준의 스트리밍 미디어 캐쉬 화일 시스템(umcFS)을 설계하고 구현하였다. umcFS 구조는 큰 캐쉬 블럭 구조와 제어블럭의 정적 할당을 기본 정책으로 유닉스 화일 시스템과 비교되는 확장된 1차 간접 블럭 참조를 통해서 캐쉬 블럭을 관리한다. 제안된 캐쉬 화일 시스템은 기존 화일 블럭 캐싱 시스템에 비해 512KB, 1MB 블럭의 임의 입출력에서 약 7~13%의 성능향상을 보였다.

1 서론

최근에 미디어 스트리밍 분야는 서비스 유료화에 따른 사용자 QoS 향상을 위해 스트리밍 미디어 캐싱 기술 개발과 캐싱 네트워크 구성에 많은 관심을 보이고 있다. 스트리밍 미디어 캐싱은 지역 네트워크 캐싱 시스템을 기반으로 사용자가 선호하는 콘텐츠의 일부를 저장하고 관리하는 기술로써 스트리밍 서버의 콘텐츠 전송량을 감소시키며 기간 네트워크 트래픽을 지역망으로 이전시키는 효과가 있어 전체 네트워크 트래픽의 균일화를 유도한다[1][2].

스트리밍 미디어 캐싱 시스템의 핵심요소는 캐쉬 입출력 시스템이다. 스트리밍 미디어의 캐쉬 입출력은 서버로부터 콘텐츠 입력과 클라이언트로 출력하는 양방향 구조로 스트리밍 서버와 비교하여 최대 두배의 입출력량을 갖는다. 따라서 효율적인 입출력 시스템의 설계는 스트리밍 미디어 캐싱 시스템의 성능과 밀접한 관계에 있다. 캐쉬 입출력 시스템은 범용 화일 시스템 상의 화일 입출력과 전용 캐쉬 화일 시스템의 구축을 고려 할 수 있다. 범용 화일 시스템 상에서 화일 단위의 입출력 구조는 화일과 디렉토리를 이용해 쉽게 구현할 수 있으나 여러 단점들이 있다. 캐싱된 미디어의 선호도에 따라 캐싱량 조절시 대상 미디어를 재구성하는 입출력 부하가 발생하며 화일 블럭 단위의 캐쉬 구조의 설계시 미디어의 대용량성에 비례해 화일의 수가 증가하므로 화일 시스템의 자체 운영 부하를 가중시키고 화일 시스템 제어를 위한 시스템 콜을 빈번하게 발생시킨다. 한편 전용 캐쉬 화일 시스템의 구성은 커널과 사용자 수준의 개발로 나뉜다. 커널 수준의 개발은 저수준 입출력을 직접 관리하여 효율성을 극대화 할 수 있는 장점이 있지만, 개발기간이 길고 시스템 이식성이 낮은 단점이 있다. 사용자 수준의 개발은 범용 화일 시스템을 기반으로 개발 기간을 단축시킬 수 있고, 이식성이 뛰어난 장점이 있다. 또한 멀티미디어 화일 시스템에 기반할 경우 저수준의 입출력을 제어할 수 있는 장점이 있다.

본 논문에서는 스트리밍 미디어의 효율적인 캐싱과 재전송을 목적으로 범용 화일 시스템에 기반한 사용자 수준의 스트리밍 미디어 캐쉬 화일 시스템(umcFS)의 설계와 구현을 기술한다. 캐쉬 화일 시스템은 작은 크기의 패킷 미디어들을 캐쉬 블럭 단위로 묶어 디스크 기반에 전형적인 캐쉬 입출력 시스템을 유형화한다. 화일 디스크는 범용 화일 시스템 위에 물리적인 데이타 블럭의 연속성을 최대한 확보하여 논리적인 캐쉬 저장 공간을 제공하고 확장성을 제공한다. umcFS 구조는 큰 캐쉬 블럭 구조와 제어 블럭의 정적 할당을 기본 정책으로 유닉스 화일 시스템과 비교되는 확장된 1차의 간접 블럭 참조를 통해서 캐쉬 블럭을 관리하고 사용자 수준의 라이브러리로 개발되어 시스템 간의 이식성, 확장성과 단기 개발 등이 고려되었다. 논문의 구성은 다음과 같다. 2장에서 스트리밍 미디어 캐쉬 시스템의 개요를 살펴보고 3장에서 스트리밍 미디어 캐쉬 화일 시스템의 세부사항을 기술한다. 4장에서는 umcFS의 저수준과 API 수준의 입출력 성능을 비교 검증하고 5장에서 결론을 맺는다.

2 스트리밍 미디어 캐싱 시스템

스트리밍 미디어 캐싱 시스템은 그림 1에서 나타낸 바와 같이 캐싱 시스템의 관리자들과 캐쉬 화일 시스템으로 구성된다. 캐싱 시스템의 관리자들은 네트워크 관리자, 세션 관리자, 콘텐츠 관리자, 캐쉬 관리자, 프로토콜 제어기로 구성되며 각각의 역할은 다음과 같다. 네트워크 관리자는 서버와 클라이언트의 입출력을 담당하고 패킷 단위의 효율적인 입출력을 위해 버퍼 관리와 입출력 스케쥴링을 제공한다. 세션 관리자는 클라이언트의 요구를 받아 캐쉬와 서버의 상태 정보에 따라 수용 여부를 결정하고 설정된 세션의 종료 때까지 세션을 유지 관리한다. 또한 기존의 스트리밍 프로토콜을 이용하여 서버 세션과 클라이언트 세션을 독립적으로 유지하고 캐싱될 미디어 정보를 콘텐츠 관리자에게 제공한다. 콘텐츠 관리자는 수용된 세션에 대해서 클라이언트로 전송할 캐쉬 미디어를 제공하고 서버로부

• 본 연구는 정보통신부에서 지원하는 대학기초연구지원사업으로 수행하였슴 (과제번호 : 2001-076-3)

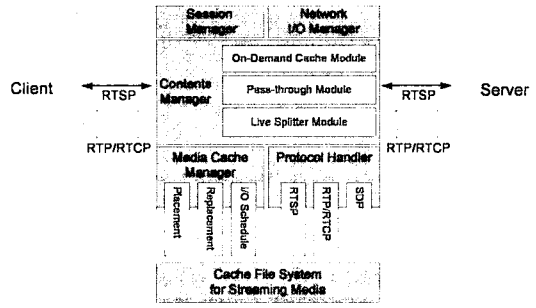


그림 1 캐쉬 서버 구조

터 받은 후미 미디어와 캐쉬 시스템의 선두 미디어에 일관성 있는 관리를 지원한다. 서비스에 따라 On-demand 캐쉬, Pass-through, Live-splitter 모듈로 구성된다. On-demand 캐쉬는 서버에서 화일로 구성된 미디어에 대해 캐쉬 기능과 전송 서비스를 제공한다. Pass-through는 프락시 서버의 역할을 하고, Live-splitter는 실시간으로 방송되는 스트리밍 미디어를 단일 스트림으로 전송 받아 다수의 지역 네트워크 사용자들에게 복사를 통해서 전송하는 서비스다. 프로토콜 제어기는 상위 세션 관리자와 콘텐츠 관리자의 RTSP/RTP 프로토콜 처리를 담당한다. 캐쉬 관리자는 캐쉬과 관련된 정책을 수행하고 세션 관리자와 콘텐츠 관리자에 세션유지를 위한 캐쉬 미디어의 정보를 전달한다. 또한 캐쉬 화일 시스템을 통해 캐쉬된 미디어 입출력을 관리한다. 캐쉬 화일 시스템은 위 관리자들의 상호작용에 의해서 서버의 미디어들을 캐쉬 입력하고 네트워크 관리자 버퍼에 캐쉬 출력을 제공한다.

3 캐쉬 화일 시스템

캐쉬 화일 시스템은 스트리밍 미디어 캐쉬 시스템을 위한 스트리밍 미디어의 효율적인 저장과 재전송 구조로 설계되었으며 범용 화일 시스템에 기반한 사용자 수준의 입출력 제어를 지원한다. 그림 2는 캐쉬 화일 시스템 개요를 보여준다.

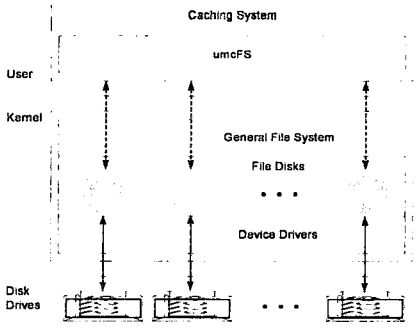


그림 2 캐쉬 화일 시스템 개요

캐쉬 화일 시스템의 주요 특징은 다음과 같다. 캐쉬 화일 시스템은 스트리밍 미디어에 기반한 캐쉬 블럭 입출력 구조이다. 네트워크 입출력을 위한 스트리밍 미디어의 작은 패킷이 캐쉬 입출력 처리 단위이면 빈번한 시스템 콜 수행으로 캐쉬 입출력 처리 부하가 발생한다. 이와 같은 시스템 부하를 줄이기 위해 캐쉬 블럭과 네트워크 입출력 버퍼의 크기를 동일하게 하여 캐쉬 입출력 단위를 구성하였다. 캐쉬 화일 시스템은 범용 화일 시스템에 기반한 사용자 수준의 입출력 제어 구조이다. 캐쉬 화일 시스템은 사용자 수준의 라이브러리 구조로 설계되어 이식성과 범용성이 좋으며 범용 화일 시스템이 지원하는 64-bit 주소 범위와 Direct I/O 등의 유용한 기능을 이용한 효율적인 입출력 체계를 지원한다. 또한 사용자 라이브러리 구조의 API를 구성함으로써 캐쉬 시스템의 독립적인 개발을 지원한다. 캐쉬 화일 시스템은 입출력 제어를 위해 가상 디스크인 화일 디스크를 구성한다. 화일 디스크는 디스크 드라이브에 화일 시스템을 구성하듯이 범용 화일 시스템을 기반으로 선형적으로 구성된 가상 디스크이며 캐쉬 화일 시스템의 제어 구조를 통해 캐쉬 입출력을 수행한다. 미디어 블럭 관리 구조는 유닉스 화일 시스템과 유사한 블럭 관리 구조이며 1차 간접 블럭 참조 구조이다. 유닉스 화일 시스템은 다단계 참조에 따른 화일 시스템의 자체 부하를 가지고 있어 화일의 연속성 확보에 취약한 구조이다. 캐쉬 화일 시스템은 1차 간접 참조 구조와 큰 캐쉬 블럭 구조를 지원하여 시스템의 처리량을 줄이고 블럭 단위의 데이타 연속성을 확보한다. 또한, 운영체제 내에 버퍼 캐쉬의 임의 입출력 구조에 착안하여 디스크 기반에 효율적인 임의 블럭 입출력과 제어를 제공한다. 캐쉬 블럭은 스트리밍 미디어의 전송 패킷을 관리하기 위한 캐쉬 단위이다. 미디어 블럭 구조는 블럭 내에 패킷 정보를 담는 블럭 헤더와 패킷들로 구성된다. 블럭 헤더는 패킷의 시간과 순서 값에 기반한 제어 구조를 구성하여 프로토콜 정보의 검색과 패킷의 입출력을 용이하게 하며 전송 패킷의 캐쉬을 위한 일관성있고 효율적인 구조를 제공한다.

화일 디스크는 대용량 화일을 통한 디스크 블럭의 연속 할당과 범용 화일 시스템의 디스크 입출력 요구 큐의 관리 방식을 활용해 효율적인 입출력 관리를 지원한다. 범용 화일 시스템을 경유한 디스크 블럭의 연속 할당은 범용 화일 시스템의 제어 블럭 때문에 물리적으로 완전한 연속구조라 할 수 없지만, 범용 화일 시스템의 관점에서 논리적인 연속 할당이라 할 수 있다. 화일 디스크를 활용한 디스크의 순차 입출력은 범용 화일 시스템의 최고의 성능이라 할 수 있다. 범용 화일 시스템의 디스크 입출력 요구는 물리적 디스크 드라이브 당 배정된 큐를 통해서 수행되므로 화일 디스크를 이용한 저장 공간의 선점은 디스크 드라이브의 입출력 큐의 선점이고 대역폭의 선점이라 할 수 있다.

제안된 캐쉬 화일 시스템 구조은 그림 3과 같이 캐쉬 관리 블럭과 미디어 저장을 위한 미디어 블럭들로 구성된다. 캐쉬 관리 블럭은 super block, mnode table, indirect block과 미디어 노드를 관리하는 block bitmap, mnode bitmap, indirect block bitmap이 있다. super 블럭은 운영체제의 범용 화일 시스템 정보, 캐쉬 화일 시스템의 상태 정보와 나머지 제어 정보의 위치와 크기를 기술한다. mnode는 유닉스 화일 시스템의 inode와 유사한 구조로 미디어 식별자, 미디어 시간 정보, 캐쉬 정보, 블럭 참조 정보로 구성된다. 미디어 식별자는 캐쉬에 등록된 미디어에 대한 식별자로 세션 설정 과정에서 콘텐츠의 URL 정보와 동일하고, 미디어 시간 정보는 생성과 참조에 대한 시간 정보이다. 캐쉬 정보는 캐쉬된 미디어의 참조를 횟수와 시간 간격을 기록하고 캐쉬 시스템의 캐쉬 관리자에 의해서 캐쉬 적중률을 반영한 등급을 유지한다. 그리고 블럭 참조는 직접 블럭 참조와 확장된 1차 간접 블럭 참조 구조이다. 직접 블럭 참조는 미디어 블럭을 지정하는 일대일 맵핑 구조이며 배정된 블럭 수를 초과하는 경우 간접 블럭 참조에 블럭 참조 테이블을 배정한다. 블럭 참조 테이블은 각각 미디어 블럭에 맵핑되며 블럭 참조 테이블의 초과시에 마지막 블럭 포인터를 통해 다른 블럭 참조 테이블을 할당받는다. indirect block은 mnode에 할당할 미디어 블럭을 참조하기 위한 블럭 포인터의 집합이다. 그리고 각각의 비맵 구조는 미디어 노드와 블럭에 일대일 관계로 설정된다. 미디어 블럭 구조는 헤더와 미디어

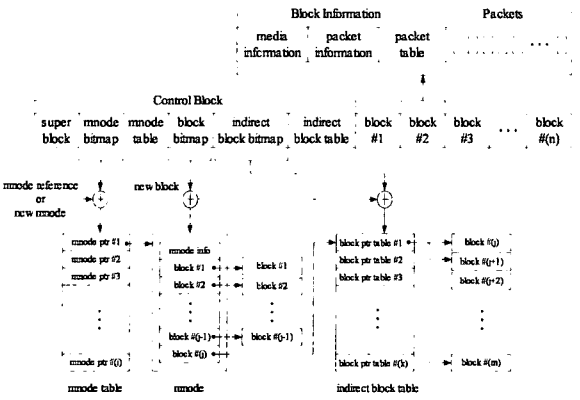


그림 3 캐쉬 화일 시스템 구조

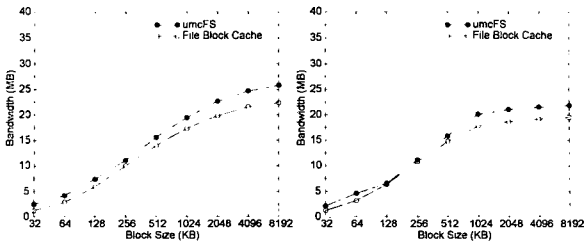
패킷들로 구성된다. 미디어 블럭 헤더는 패킷 정보와 패킷 테이블로 구분된다. 패킷 정보는 전송 프로토콜과 블럭 내에 존재하는 전송 패킷들의 시간 범위를 기록한다. 패킷 테이블은 블럭 내에 패킷의 위치와 패킷에 대한 시간 정보, 패킷 순서를 담는다.

캐쉬 화일 시스템의 API는 캐쉬 시스템을 위한 스트리밍 미디어 입출력 인터페이스다. API는 캐쉬 미디어의 블럭화된 입출력과 미디어 검색 그리고 캐쉬 관리 정책을 적용하기 위한 기능들로 구성되었고 전반적인 API 설계 측면에서 유닉스 화일 시스템의 API와 유사하다. API 콜은 umount(), uopen(), uclose(), uread(), uwrite(), useek(), uremove(), uremove_block(), uplace(), ureplace(), ustatfs(), ulookup()으로 구성된다.

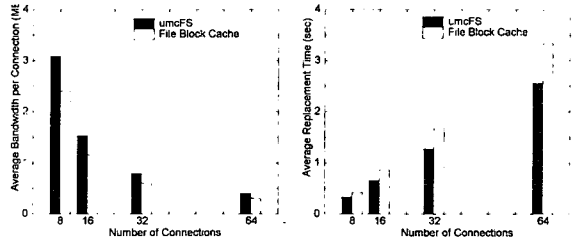
4 실험 결과

캐쉬 화일 시스템(umcFS)의 개발 환경은 다음과 같다. 개발 호스트는 Compaq Proliant ML370이며 Dual P-III 866 MHz에 메모리 512MB를 장착하였다. 사용된 운영체제는 대용량 화일 시스템(LFS)[3]을 지원하는 리눅스 시스템이다. 범용 화일 시스템은 SGI의 xFS[4]를 사용하였고, 스트리밍 환경은 Apple's QTSS 3.0[5] 서버와 Quick Time Player Pro 5.0이며 실험에서 사용한 스트리밍 미디어 포맷은 Apple에서 제공하는 Hinted Track화된 MOV를 활용하였다. 실험을 위한 비교 모델은 웹 기반의 Squid 캐쉬 입출력 체계와 umcFS의 관리 체계를 도입하여 성능 비교를 위해 개발된 화일 블럭 캐쉬 입출력 시스템이다. 실험은 블럭 크기와 수용된 커넥션 수에 따라 저수준과 API 수준에서 입출력 성능을 측정하였다.

저수준 입출력 실험은 umcFS와 화일 블럭 캐쉬의 캐쉬 블럭에 대한 범용 화일 시스템에 기반한 디스크 입출력 성능을 측정한 것이다. 그림 4(a)는 저수준의 임의의 입출력 대역폭을 측정한 것으로 umcFS가 512KB, 1024KB에서 7~13%의 성능 향상을 보였다. 성능 차이의 요인은 umcFS의 자원 선점 구조와 범용 화일 시스템에 의존적인 화일 블럭 캐쉬의 자원 비선점 구조에서 발생한다. API 수준의 입출력은 사용자 세션의 캐쉬 입출력하고 uread와 uwrite 콜을 통해 수행되며 캐쉬 입출력 시스템을 제외한 실험 구성을 umcFS와 화일 블럭 캐쉬에 동일하게 적용하였다. 콘텐츠 당 10개의 블럭을 할당하여 캐쉬 전체를 채웠으며 각 스트리밍 커넥션은 다른 커넥션과 중복 없이 임의로 콘텐츠를 선택한다. 그림 4(b)의 캐쉬 교체 대역폭은 캐쉬 블럭 크기에 대해 캐쉬 교체 시간으로 나눈 커넥션 당 입출력 대역폭이고, 캐쉬 교체 시간은 커넥션에 발생한 캐쉬 입력에 대해서 기존에 캐싱된 블럭을 캐싱 아웃시키고 새로운 블럭



(a) 저수준 : 임의 입출력 대역폭



(b) API : 캐쉬 교체 대역폭과 평균 시간

그림 4 캐쉬 입출력 성능

입력에 걸리는 시간이다. 화일 블럭 캐쉬는 umcFS와 비교해 1024KB 일 때 약 76% 성능을 차지하였고, 성능 차이는 저수준 입출력 경향이 주요인이다.

5 결론

본 논문에서는 스트리밍 미디어의 효율적인 캐쉬 저장과 재전송을 목적으로 범용 화일 시스템에 기반한 사용자 수준의 스트리밍 미디어 캐쉬 화일 시스템인 umcFS를 설계하고 구현하였다. umcFS는 스트리밍 미디어의 패킷 단위 네트워크 입출력에 대한 전형화된 디스크 기반의 캐쉬 블럭 입출력 구조를 제시하고 캐싱 서버의 성능 향상을 위한 핵심 부분을 이루며 캐싱 미디어의 형식과 활용 방식을 제시하고 있어 스트리밍 미디어 캐싱의 연구 분야에 파급효과가 있을 것으로 예상된다. 또한, 화일 디스크는 사용자 수준의 입출력 제어의 가능성을 보여주며 멀티미디어와 데이타 베이스 등의 비슷한 응용에도 쉽게 적용할 수 있을 것이다.

참고문헌

[1] Yeuwei Wang, Zhi-Li Zhang, David H.D. Du, and Dongli Su, 'A Network Consious Approach to End-to-End Video Delivery over Wide Area Networks Using Proxy Servers', IEEE Infocom, April 1998, pp.660-667.

[2] R. Rejaie, M. Handley, H. Yu, and D. Estrin,'Proxy caching mechanism for multimedia playback streams in the Internet', in Fourth International WWW Caching Workshop, Mar. 1999.

[3] Large File Support in Linux, URL : http://www.suse.de/~aj/linux_lfs.html.

[4] SGI's xFS File System for Linux, URL : http://oss.sgi.com/projects/xfs.

[5] QuickTime Streaming Server, URL : http://www.apple.com/quicktime/products/qtss.