멀티미디어 스트림을 위한 웹 캐쉬의 설계

윤철주**○ 장성민** 박순동* 원유헌**
**홍익대학교 컴퓨터공학과 *승의여자대학 전자계산학과
**{cjyoon, smjang, won}@cs.hongik.ac.kr *sdpark@sewc.ac.kr

Design of Web Cache for Multimedia Stream

Chul-Joo Yoon**○ Sung-Min Jang** Soon-Dong Park* Yoo-Hun Won**
**Dept. of Computer Engineering, HongIk University
*Dept. of Computer Science, SoongEui Women's College

요 약

오늘날 초고속 통신망의 발달로 인해서 비디오와 오디오와 같은 멀티미디어 스트림의 요청이 많아지고 있다. 하지만 기존에 사용되고 있는 웹 캐싱 기법들은 아직까지 멀티미디어 스트림과 같이 크기가 큰 파일을 적절하게 처리하지 못하고 있다. 본 논문에서는 멀티미디어 스트림을 효율적으로 처리하여 캐쉬의 적중률을 높이고 사용자의 지연 시간을 줄이는 캐싱 기법을 제안한다.

1. 서론

최근 몇 년 동안 인터넷은 빠른 속도로 성장을 하여왔다. 급격한 인터넷 사용자의 증가로 인하여 인터넷 성능의 저하가 발생하였다. 그러나 웹 캐쉬를 사용으로 서버들의 부하를 줄이고, 네트워크의 트래픽을 감소하고, 서버 지연 시간의 줄이는 등의 많은 향상을 가지고 왔다.

그렇지만 기존의 웹 캐쉬에서는 아직까지 비디오나 오디오 같은 멀티미디어 스트림에 대한 처리가 부족하다. 얼마전까지만 해도 비디오와 오디오 스트림같은 멀티미디어 스트림이 큰 비중을 차지하지 않았다. 그렇기 때문에 크기가 큰 멀티미디어 스트림을 웹 캐쉬에 저장하지 않거나 저장을 하더라도 캐쉬에서 빨리 없애는 것이 웹 캐쉬 성능 향상에 도움이 된다는 것이 연구되었다[1]. 그러나 초고속 통신망이 일반화되고 있는 현실에서는 비디오와 오디오 같은 멀티미디어 스트림의 비중이 점점 커지고 있다. 앞으로 5년 이내에 인터넷 데이터의 50%이상이 멀티미디어 스트림이 될 것이라고 예상하고 있다[2]. 그렇기 때문에 현재 나와있는 웹 캐쉬를 사용하게 될 경우 인터넷의 성능은 점점 떨어질 수밖에 없을 것이다.

그래서 본 논문에서는 멀티미디어 스트림을 효과적으로 처리하기 위한 웹 캐쉬 정책과 효율적인 교체 알고리즘을 제안한다.

본 논문의 구성은 2장에서 웹 캐쉬와 관련된 연구들을 살펴본다. 3장에서는 이 연구가 필요한 연구배경에 대해 알아보고, 4장에서는 본 논문에서 제안한 캐쉬 정책과 교체 알고리즘을 설명한다. 마지막으로 5장에서 결론과 향후 연구 과제에 대해서 논하기로 하겠다.

2. 관련 연구

교체 알고리즘은 다음에 들어올 데이터를 예상해서 캐쉬의 적중률을 높임으로 인해서 캐쉬의 성능을 향상시키는 것을 목적으로 하고 있다.

다음은 웹 캐쉬에서 많이 사용하고 있는 교체 알고리즘이다. LRU(Least Recently Used)와 LFU (Least Frequently Used)는 예전부터 캐쉬에 많이 사용되는 알고리즘이고, SIZE와 LRU-Threashold는 웹 캐쉬에서 파일의 크기가 큰 파일을 처리하기 위한 알고리즘이다.

- LRU(Least Recently Used) - 가장 오래전에 접근한 데이터를 캐쉬에서 먼저 없앤다.
- LFU(Least Frequently Used) - 자주 쓰이지 않는 데이터를 캐쉬에서 먼저 없앤다.
- LRU-Threshold - LRU와 기본적으로는 동일하지만 파일의 크기가 정해진 한계(threshold)보다 큰 것은 저장하지 않는다[4].
- SIZE - 크기가 큰 데이터를 먼저 없앤다[1].

LRU-Threashold와 SIZE라는 알고리즘이 나왔을 때는 멀티미디어 스트림과 같이 파일의 크기가 큰 데이터의 비중은 매우 작았다. 그렇기 때문에 멀티미디어 스트림처럼 파일의 크기가 큰 것을 저장해서 캐쉬의 공간을 낭비하는 것보다 다른 데이터를 저장할 수 있도록 캐쉬를 비워두는 것이 더 효율적이었다. 그렇지만 멀티미디어 스트림이 많아지고 있는 현실에서는 그러한 방법이 캐쉬 성능의 향상을 가지고 올 수는 없을 것이다.

3. 연구 배경

멀티미디어 스트림은 일반적으로 다음과 같은 특징을 갖는다.

우선 멀티미디어 스트림은 일반적인 웹 데이터보다 파일의 크기가 크다는 점이다. 따라서 멀티미디어 스트림을 일반 데이터처럼 캐쉬에 저장을 하게 되면 캐쉬의 대부분이 멀티미디어 스트림으로 저장되므로 다른 데이터들이 저장될 수 있는 공간이 상대적으로 줄어들게 되어서 캐쉬의 적중률이 떨어지게 된다. 그렇기 때문에 멀티미디어 스트림을 많은 공간이 필요없이 저장할 수 있는 방법이 필요하다.

또 다른 문제점은 웹에서 멀티미디어 스트림을 보기 위해서는 여러 단계가 필요하기 때문에 발생한다[5]. 먼저 클라이언트와 서버를 연결하고 원하는 멀티미디어 스트림을 선택한다. 그러면 클라이언트에서는 멀티미디어 스트림을 볼 수 있는 브라우저를 생성을 한 뒤에 서버와 연결해서 실행을 하기에 충분한 양만큼의 스트림을 클라이언트의 버퍼에 저장을 한 뒤에 멀티미디어 스트림을 화면에 보여주게 된다. 이러한 과정을 거치게 되므로 멀티미디어 스트림을 접근하려 할 때마다 상당한 지연 시간이 요구된다. 그러므로 처음 멀티미디어 스트림을 시작할 때의 지연 시간을 줄일 수 있는 방법이 필요하다.

그리고 마지막으로 멀티미디어 스트림은 특정한 인기있는 것에 집중이 된다. VOD(Video-On-Demand) 시스템에서 사람들이 영화를 선택할 때 인기있는 영화를 많이 선택하게 되는 것처럼 멀티미디어 스트림은 인기있는 것에만 집중적으로 요청이 생긴다. 그렇기 때문에 인기 있는 스트림을 캐쉬에 오랫동안 저장을 하고 있으면 적중률을 높일 수 있을 것이라 생각한다.

현재 나와있는 웹 캐쉬들은 이와 같은 사항을 너무 간과하고 있으므로 멀티미디어 스트림 처리에 대한 성능 향상을 기대 할 수 없다.

4. 캐싱 기법

4.1 웹 캐쉬 정책

현재 나와있는 웹 캐쉬에 대한 연구에서는 멀티미디어 스트림을 하나의 객체로 생각을 해서 캐쉬에 넣게 되었다. 파일의 크기가 큰 멀티미디어 스트림을 캐쉬에 넣으므로 인해서 데이터가 캐쉬의 공간을 많이 차지하게 되고 상대적으로 다른 데이터들이 저장될 공간이 줄어들게 되었다. 그러므로 인하여 웹 캐쉬의 성능을 떨어뜨리는 결과를 가지고 왔다.

그렇다고 멀티미디어 스트림을 저장을 하지 않으면 클라이언트가 서버와 연결하고 클라이언트 버퍼에 일정한 양만큼의 스트림을 저장 한 뒤에 멀티미디어 스트림이 실행된다. 이렇게 서버에서 클라이언트의 버퍼까지 저장하는 데 걸리는 지연 시간은 상당히 크다. 그러므로 지연 시간을 줄일 수 있는 방법이 필요하다.

그렇기 때문에 본 논문에서는 멀티미디어 스트림의 첫 부분만을 캐쉬에 저장하고자 한다. 이 방법은 전통적인 웹 캐쉬에서 텍스트나 이미지 데이터를 처리하는 방법과 크게 차이가 나지 않는다. 그렇지만 멀티미디어 스트림을 저장하기 위한 많은 공간이 필요하지 않게 된다. 그리고 클라이언트의 요청을 받으면 웹 캐쉬에서는 캐쉬에 저장된 첫 부분을 클라이언트에게 보내주게 된다. 그리고 나서 스트림의 나머지 부분에 대한 요청을 서버에게 하고 서버에서 나머지 스트림을 읽어 들인다. 멀티미디어 스트림을 처음 읽어 올 때 서버에서 가지고 오지 않고 캐쉬에서 원하는 스트림을 가지고 오기 때문에 지연 시간을 상당히 줄일 수 있을 것이라 생각한다.

4.2 캐쉬 교체 알고리즘

VOD(Video-On-Demand) 시스템에서 인기 있는 영화

를 보려고 하는 사람이 많은 것처럼 멀티미디어 스트림은 특정 인기 있는 스트림에만 요청이 집중적으로 들어오게 된다. 이러한 멀티미디어 스트림의 특성을 고려하지 않고 전통적인 캐쉬 교체 알고리즘을 사용하면 캐쉬의 적중률을 높일 수 없을 것이다. 그래서 본 논문에서는 그러한 멀티미디어 스트림의 특징을 살려서 자주 참조되는 것은 오랫동안 캐쉬에 남아 있도록 하려고 한다. 즉, 서버에 있는 멀티미디어 스트림마다 가중치 값을 가지고 있고, 클라이언트에서 요청이 있을 때마다 그 가중치를 증가시킴으로써 요청이 많은 스트림은 큰 가중치를 가지게 돼서 오랫동안 캐쉬에 남아있게 될 것이다.

```
while (요청이 있으면) {
  S ← 요청된 스트림의 첫 부분
  if(S가 캐쉬에 있으면) {
    클라이언트에 S 전달
    접근시간 저장
    Ws 증가 요청
  }
  else {
    서버에서 클라이언트에 S 전달
    while(캐쉬에 공간이 없으면) {
      Ws가 가장 작은 것 캐쉬에서 삭제
      Ws가 같으면 오래된 접근 시간 것 삭제
    }
    캐쉬에 S 저장
    접근시간 저장
    Ws 증가 요청
  }
}
```

[그림 1] wLRU 알고리즘

[그림 1]은 본 논문에서 제안하고 있는 wLRU 알고리즘을 나타낸 것이다. wLRU는 많이 참조가 되는 멀티미디어 스트림의 가중치를 고려한 다음 가중치가 같으면 LRU 기법을 적용하는 방법이다. S는 캐쉬에 저장할 멀티미디어 스트림의 첫부분이고 W_s는 서버에 있는 멀티미디어 스트림이 몇 번이나 참조되었는 지를 가지고 있는 가중치 값으로 요청이 있을 때마다 가중치가 증가하게 된다. 캐쉬 교체를 할 때 W_s가 작은 것부터 캐쉬에서 먼저 없애고 W_s가 같은 것에 대해서는 LRU 방식을 사용하게 된다.

이와 같은 캐쉬 알고리즘을 사용함으로써 요청이 많은 멀티미디어 스트림을 캐쉬에 남겨 놓게 되므로 캐쉬의 적중률을 높일 수 있을 것이다.

5. 결론 및 향후 연구 과제

본 논문에서는 멀티미디어 스트림에 적합하게 동작하기 위한 웹 캐쉬를 설계하였다. 멀티미디어 스트림의 첫 부분만을 저장함으로써 캐쉬의 공간을 많이 차지하지 않고서 지연 시간을 줄여 줄 수 있다. 그리고 여기에서 제안한 새로운 교체 알고리즘을 사용함으로써 기존의 웹 캐쉬보다 적중률을 높일 수 있을 것이라 기대한다.

하지만 멀티미디어 스트림의 첫 부분을 저장 할 때 어느 정도의 크기를 저장하는 것이 가장 효율적인지를 더 연구해야할 필요가 있다. 저장하는 스트림의 크기가 작으면 지연이 생길 수 있고 너무 크면 캐쉬의 공간이 낭비가 될 수 있으므로 적절한 크기를 저장하는 것이 중요하다. 그리고 본 논문에서는 가중치를 클라이언트 측의 특징을 고려하지 않고 서버 측의 특징만을 고려해서 가중치를 주었는 데, 클라이언트 측의 특징도 고려해서 가중치를 주는 방법이 필요할 것이라 생각한다.

[참고 문헌]

[1] Stephen Williams, Marc Abrams, Charles R. Standridge, Ghaleb Abdulla, Edward A. Fox, Removal Policies in Network Caches for World-Wide Web Documents, Proceedings of Sigcomm'96.

[2] Renu Tewari, harrick M. Vin, Asit Dan and Dinkar Sitaram, Resource-based Caching for Web Servers, Tech report of CS of University of Texas at Austin, 1997.

[3] Jia Wang, A Survey of Web Caching Schemes for the Internet, ACM CCR Vol. 29, Nov. 1999.

[4] M. Abrmas, C. R. Strandridge, G. abdulla, S. Williams, and E. A. Fox, Caching proxies: limitations and potentials, Proceedings of the 4th International WWW Conferece, Boston, MA, Dec. 1995.

[5] K. Ross and J. Kurose, Computer Networking and Internet Protocols. Online Document, 1998. http://www.seas.upenn.edu/~ross/book/Contents.htm.