분산 프록시를 사용한 VOD 서비스

김영준, 김익수
시립 인천대학교 정보통신공학과
airfight@hanmail.net iskim@incheon.ac.kr

VOD Service using Distributed Proxy

Young-june Kim, Ik-soo Kim
*Dept of Information and telecommunication Engineering,
University of Incheon

요약

본 논문은 인터넷상에서 다양한 매체를 이용한 서비스를 제공할 때 생길 수 있는 긴 사용자-서버간 지연, 엄청난 서버의 부하, 네트워크 자원의 비효율적 사용과 같은 문제점들을 위한 개선된 방법으로 사용자에 가까운 복수의 노드들을 두어 서버에서 전송된 데이터를 분산 저장하는 방법에 대해 다루고 있다.

새로운 분산 Proxy 기법은 VOD 서비스를 원하는 사용자들이 동일한 아이템에 대한 요청이 많은 서비스 패턴을 분석해서, 중복된 네트워크 부하는 줄이는 과정을 수행한다. 사용자가 요청한 영화는 서버에서 전송을 받는데 이때, HEN(Head-End-Node)에 구현된 여러 Proxy에 아이템의 일부를 나누어 저장하고 이에 대한 정보 보관과 제어를 SA(Switching Agent)가 하게 된다.

사용자가 서비스를 요청할 경우 SA의 제어 하에 분산 Proxy에 교호적으로 접속을 하여 저장되어 있는 부분적인 데이터들을 서비스 받도록 한다. 이때 Proxy에 새로운 데이터 블록을 저장할 때는 부족한 저장공간으로 인해 LRU(Least Recently Used), LFU(Least Frequently Used), 또는 이들을 복합한 Hybrid 정책을 사용한다.

1. 서론

인터넷 환경이 성숙함에 따라서 사용자들이 바라는 서비스는 점차 복잡해지고 용량이 커지며, 빠른 데이터 전송 시간을 요구하고 있다. 하지만 미디어의 다양화로 사용자들이 원하는 데이터는 기존의 텍스트에서 벗어나 사운드, 이미지, 비디오 데이터 등 다양한 매체를 사용해 정보를 얻길 원한다. 따라서 이러한 데이터를 요구하는 사용자가 있을 경우 적절히 전송해 주는 기술이 주문형 서비스(On-Demand Service)이다. 그러나 사용자가 원하는 시간에 데이터를 전송해야 하므로 빠른 전송 속도를 보장해야하고 다수의 사용자 요구를 빠르게, 보다 정확히 처리하기 위한 강력한 서버 능력이 요구된다. 또한, 대량의 멀티미디어 데이터를 전송하기 위해 고속의 광대역 망이 필수적으로 구성되어야만 하는 어려운 문제점들을 갖고 있기 때문에 이를 해결하기 위한 다양한 연구가 진행중이다.[1,2,3]

위의 문제들을 해결하기 위한 방법들 중 한가지로서 클라이언트들로부터의 서비스 요청 시점이 비슷할 경우 이들을 그룹화 하여 서비스하는 멀티캐스트 전송 기법이 사용되고 있고, 서버와 사용자 사이에 임시 저장법을 사용한 Proxy caching기법도 사용되고 있다.[6,7,8] 멀티캐스트 전송방식을 사용하면 동일한 데이터에 대해서 사용자들이 그룹을 구성하여 하나의 스트림으로 서비스를 받게 되므로 Server의 부하를 줄이고 네트워크의 자원을 낭비하지 않게 할 수 있으나, 현재 이를 뒷받침 해줄 인프라스트럭처가 완비되지 않아 보완이 필요한 실정이다. Proxy caching은 이전에 요청했던 데이터를 전송만 하는데 그치는 것이 아니라 Proxy에 저장하여 나중에 동일한 데이터를 요청한 다른 사용자에게 다시 전송을 해서 서버에 접속하지 않고 서비스를 제공받을 수 있도록 하여 서버의 부하를 줄일 수 있다.[4,5]

그러나. 기존의 Proxy 구조는 일반적으로 서버 부

근에 위치하기 때문에 End-to-End delay는 다소 줄일 수 있지만 네트워크의 부하는 물론, 특정 item에 대한 Proxy 부하를 줄일 수 없는 단점이 있을 수 있다. 따라서 최근에는 그림 1.에서와 같이 Proxy를 클라이언트 부근에 두어 End-to-End delay를 크게 줄일 수 있을 뿐만 아니라 네트워크의 부하도 줄일 수 있으며, 분산 Proxy를 두어 원활하게 보다 양질의 서비스를 구현하고 있다. 그러나 분산 Proxy를 구현하여 서비스를 제공하더라도 인기 있는 item을 저장하고 있는 Proxy는 상대적으로 다른 Proxy에 비해 트래픽이 폭주하는 단점이 내포되어 있다.[6,7]

본 논문에서 제안하는 새로운 Web-caching 기법은 클라이언트들이 접속되어 있는 다수의 단말-노드(Head-end-node : HEN)들에 소량의 버퍼를 장착하여 Proxy로 활용하고, 모든 사용자들의 서비스를 관할하도록 한다.

그림 2에서처럼 특정 단말-노드(HEN 1)들에 연결된 클라이언트가 서비스를 요청할 경우 서버로부터 전송되는 데이터 블록(블록 1, 2)은 일단 HEN 1의 버퍼에 순차적으로 저장됨과 함께 서비스 요청 클라이언트에 스트리밍 서비스가 이루어진다. 이후 다른 단말-노드(HEN 2)로부터 동일한 item에 대해 서비스를 요청할 경우에는 서버로부터 전송되는 이어지는 데이터 블록(블록 3)은 HEN 2에 저장한다. 따라서 HEN 1의 클라이언트는 데이터 블록 3을 서비스 받기 위해 SA 제어 하에 HEN 2에 접속하여 서비스를 받으며, 이와 동시에 HEN 2의 클라이언트는 데이터블록 1과 2를 서비스 받기 위해 HEN 1에 접속하여 서비스를 개시하고 이어서 자신의 HEN 2로부터 나머지 블록을 서비스 받는 방식을 사용한다.

분산 Proxy 구조에서는 데이터를 다수의 Proxy에 나누어 저장하므로 데이터가 저장된 위치, 현재에도 데이터가 저장되었던 Proxy에 남아 있는지에 대한 정보 등을 포함한 제어가 복잡해진다. 이러한 제어를 서버에게 맡긴다면 사용자의 요구에 따라 데이터를 전송해야 하는 서버의 부하가 더욱 늘어나게 된다. 따라서 이러한 데이터 제어를 맡겨서 다른 구성요소가 전송에만 능력을 할당 할 수 있도록 해주는 장치가 SA(Switching Agent)이다.

이와 같은 새로운 Web-caching 기법은 VOD 서비스에 있어서 인기 있는 영화의 경우 보다 많은 단말-노드들이 서비스를 요청하기 때문에 해당 영화에 대해 일정 분량의 데이터 블록들이 각기 다른 단말-노드에 저장되어 자동적으로 부하의 분산이 이루어질 수 있는 장점을 보유하고 있다.

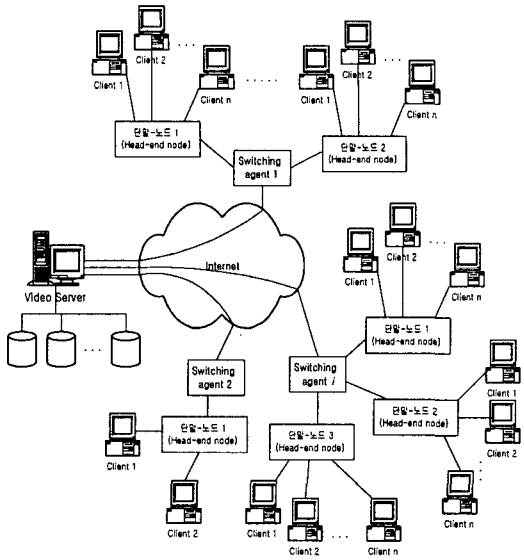

그림 1. 서버, 스위칭-에이전트, 단말-노드와 사용자의 기본 구조

2. 분산 Proxy 네트워크의 구조

VOD 서비스를 제공하는 서버의 부하를 격감시키고, 부족한 네트워크 자원을 보다 효율적으로 사용할 뿐만 아니라 End-to-end delay를 줄일 수 있는 분산 Proxy를 구현할 수 있는 새로운 형태의 Web-caching을 수행하기 위한 구조는 서버, 단말 네트워크와 다수의 단말-노드 그리고 단말-노드 사이의 모든 통신을 제어하는 스위칭 에이전트(SA)로 구성될 수 있다.

우선 서버는 단말-노드에 연결되어 있는 클라이언트들로부터 서비스 요청을 받아들여 해당 영화에 대해 즉각적으로 서비스를 제공하기 위해 스트리밍 서비스를 수행하는 소스 디바이스로서 다수의 클라이언트들로부터 동일한 item에 대해 서비스를 요청할 경우에도 단 1회만 데이터 스트림을 전송하게 되며, 단말 네트워크란 그림 1에 보이는 바와 같이 특정의 스위칭-에이전트에 연결되어 있는 서로 인접해 있는 단말-노드(또는 LAN)들을 서로 연결한 망이라 할 수 있다.

스위칭 에이전트 SA는 각 단말-노드의 서비스 요청에 따라 서버로부터 전송되어 온 데이터 스트림 블록들이 어느 단말-노드의 Proxy에 저장되어야 하

는가는 물론, 서비스가 종료되기 전에 다른 단말-노드(HEN *i*)로부터 동일 Item에 대한 요구가 있을 경우 순차적으로 이어지는 서버로부터의 데이터 블록을 HEN *i*에 저장하도록 제어하는 역할을 한다. 또한 SA는 사용자가 요구한 모든 단말-노드들의 서비스 요청 상황과 각 단말-노드에 저장되어 있는 데이터 블록들의 목록들을 보유해야 한다. 따라서 각 단말-노드들의 서비스 요청에 따라 자신의 목록들을 조사하여 요청된 영화의 데이터 블록이 인접한 단말-노드에 이미 Caching되어 있는지의 여부를 판단하여 서버로 또는 다른 단말-노드로 연결할 것 인지의 여부 결정과 사용자수를 알린다.

단말-노드는 자신에게 연결되어 있는 클라이언트들의 서비스 요청에 대해 SA를 통해서 서버에 알림과 동시에 자신의 Cache에 요청된 Item의 첫 번째 블록이 저장되어 있는지의 여부를 판단하여 이미 저장되어 있을 경우 서비스를 즉각적으로 개시하지만, 그렇지 않을 경우 SA에 제어를 요청하게 된다.

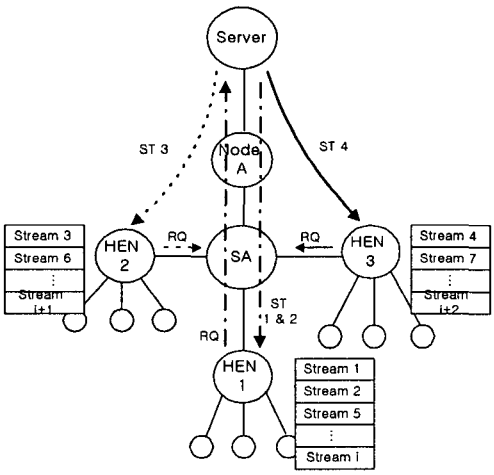

그림 2. HEN의 블록 캐싱.

3. 분산 Web-caching의 동작

클라이언트 부근에 Proxy를 둘 경우 End-to-end delay를 감소시킬 수 있을 뿐만 아니라 네트워크 자원을 효율적으로 사용할 수 있으며, 주문형 서비스인 VOD 서비스에서 인기 있는 하나의 영화를 여러 단말-노드(또는 LAN)의 Proxy에 부분적으로 저장하는 본 논문에서 제안한 분산 Web-caching의 동작은 다음과 같다.

특정 단말-노드에 속해있는 클라이언트가 VOD 서비스를 요청할 경우 단말-노드는 스위칭 에이젼트 SA를 거쳐 서버에 연결되며, 이때 SA는 요청한 item이 다른 단말-노드에 이미 Caching되어 있는지를 자신의 목록을 조사하여 Hit될 경우 서버로의 접속을 끊고 해당 단말-노드로 스위칭을 제공한다. 물론 Miss가 될 경우 서버로부터 전송되는 데이터 스트림을 요청 단말-노드로 전송한다.

3-1. Cache Miss

클라이언트의 특정 Video에 대한 서비스 요청에 대해 접속 단말-노드의 Proxy에 데이터 스트림이 존재하지 않을 경우에 대한 절차는 다음과 같다.

1. 접속 단말-노드는 SA를 거쳐 서버에 서비스 요청과 함께 SA는 목록을 조사
2. SA는 목록 조사 결과 Miss될 경우 서버로의 연결 유지
3. 서버로부터 해당 데이터 스트림을 요청 단말-노드에 전송과 함께 목록에 기입
4. 요청 단말-노드는 Proxy에 저장과 함께 클라이언트에 스트림 전송
5. SA는 다른 단말-노드로부터 동일 item에 대한 서비스 요청이 없을 경우 단계 4 반복. 만약, 다른 단말-노드로부터 서비스 요청이 있을 경우 서버로부터 전송되는 이어지는 데이터 스트림을 요청 단말-노드들에 서비스 요청 순서대로 분산 저장하고 목록에 기입

클라이언트의 특정 item에 대한 최초의 서비스 요청과 Cache miss의 경우는 동일한 상황으로 나타낼 수 있다.

3-2. Cache Hit

클라이언트의 특정 Video에 대한 서비스 요청에 대해 단말-노드의 Proxy에 데이터 스트림이 존재할 경우에 대한 절차는 다음과 같다.

1. 접속 단말-노드는 SA를 거쳐 서버에 서비스 요청과 함께 SA는 목록을 조사
2. 접속 단말-노드에 해당 item의 첫 번째 데이터 스트림이 존재할 경우 서비스 개시
3. SA는 목록 조사 결과 Hit될 경우 서버로의 연결 해지
4. SA는 이어지는 데이터 스트림이 저장되어 있는 단말-노드로 스위칭

Video 데이터를 구성하고 있는 많은 데이터 블록들은 서비스를 요청하는 여러 단말-노드들에 걸쳐서 위와 같은 방법으로 데이터 블록이 저장되고 서비스를 제공하기 때문에 동일 item을 요청한 단말-노드들은 이미 다른 노드에 Caching된 데이터 블록을 접근하기 위해선 자신도 일정량을 Caching 하여야 하기 때문에 각 단말-노드의 Proxy들은 부하의 분산이 이루어질 수 있다.

4. 캐쉬 재배치 정책

각 단말-노드에 존재하는 버퍼의 크기는 무한일 수 없다. 따라서, 사용자가 증가하고 서비스를 개시한 시점에서 오랜 시간이 흐를수록 버퍼에 저장되는 비디오 데이터의 양이 커지므로, 버퍼에 저장된 데이터의 일부를 삭제해야 하는 문제가 생긴다. 이 때 사용하는 보편적인 방법으로, 가장 먼저 저장된 데이터를 삭제하는 LRU, 가장 서비스 빈도가 낮은 데이터를 삭제하는 LFU 등이 있는데, 이를 모두 채용한 Hybrid 방식을 사용한다.

5. 시뮬레이션 및 결과

실험에 사용된 표본 영화는 100편으로 하였고, 서비스 요구율(λ)은 1에서 10까지로 설정하였다. λ=10인 경우는 prime time대의 경우로 하였다.

그 중 λ=10이고 단말-노드는 3대라고 했을 때, 100분간 서비스를 하였을 경우, Zipf 분포를 따르는 가장 인기 있는 영화는 각 단말-노드에서 302명의 사용자가 요구하였다. 또한, 인기순위 5위안의 영화를 요청한 사용자가 무려 513명으로 나타나 사용자의 절반 이상이 단지 5편의 영화를 신청하였다.

6. 결론 및 향후 연구과제

새로운 Web-caching 기법은 서버의 부하를 크게 줄어들게 되므로 저가의 시스템으로 원활한 VOD 시스템이 이루어질 수 있으며, 인기 있는 Video item에 대해 각 단말-노드 사이에 멀티캐스팅 기술이 도입될 경우 네트워크의 자원이 고갈되는 것을 효과적으로 개선시킬 수 있다.

현재 VOD 시스템은 충분한 네트워크 대역폭과 고성능 처리를 위한 고가의 서버의 제한 때문에, 성능의 제한이 따르게 된다. 따라서, 올바른 시뮬레이션을 통해 알맞은 재배치 정책을 결정하고, 사용자에게 필요한 가장 적정할 버퍼의 양과 효율 등을 연구한다.

감사의 글

본 연구는 한국 과학 재단 지정 인천대학교 멀티미디어 연구센터의 지원에 의한 것입니다.

참고문헌

[1] T. Little and D. Venkatesh, "Prospects for Interactive Video-On-Demand," IEEE Multimedia, pp 14-23 Fall 1994.

2. K. Almeroth and M. Ammar, "An Alternative Paradigm for Scalable On-Demand Applications: Evaluating and Depolying the Interactive Multimedia Jukebox," IEEE Tanas. On Knowledge and Data Engineering Special Issue on Web Technology, April 1999.

3. S. Acharrya and B.Smith "Middleman : A Video Caching Proxy Server" Tech.Report Cornell, 1999

4. K. almeroth and M. Ammar, "Providing a scalable, interactive Video-On-Demand service using multicast communication", InICCN'94, San Francisco, CA, September 1994

5. D.Eager, M.Vernon and J.Zahojan, "Optimal and Efficient Merging Schedule for Video-On-Demand Service", Proc. ACM Multimedia 99, ACM 1999

6. R. Rajaie, M. Handley, H. Yu and D. Estrin, "Proxy caching mechanism for multimedia playback streams in Internet", in Fourth International WWW Caching workshop, Mar. 1999

7. S. Sheu, Kien. a. Hua, and W. Tavanapong, "Chaining: A Generalized Batching Technique for Video-On-demand", In Proc. of the Int'l conf. On Multimedia computing and System, Ottawa, Ontario, Canada, pp110-117, June 1997

8. 김동수, 김익수 " 효율적인 노래방 서비스를 위한 멀티미디어 분산 프락시 캐싱구조 " 정보처리학회 15회 춘계 학술 대회 2001년 4월

9. C. Shahabi and M.H. Alshayeji, "Super-streaming: a New Object Delivery Paradigm for Continuous Media Servers", Journal of Multimedia Tools and Applications V11, n1, 275-298 May 2000