Chunk 분포도에 따른 선택적인 Push/Pull 을 이용한 P2P 스트리밍 성능향상 기법

조두환*, 이춘화**
*한양대학교 일반대학원 전자컴퓨터통신공학과
E-mail : joduhwan@gmail.com,

**한양대학교 공과대학 컴퓨터공학부
E-mail : lee@hanyang.ac.kr

P2P Streaming Enhancement Using Selective PUSH/PULL Based on Chunk Distribution

Duhwan Jo* and Choonhwa Lee**
*Dept. of Electronics Computer Engineering, Hanyang University
**Dept. of Computer Science and Engineering, Hanyang University

요 약

주문형 비디오 서비스와 라이브 스트리밍 서비스는 보편화와 급격한 성장을 바탕으로 보다 나은 품질의 스트리밍 서비스를 제공하기 위한 연구가 진행되어 왔다. 이 과정에서 P2P 시스템을 이용한 비디오 스트리밍은 서비스 공급자에게 비용상으로 큰 이점을 안겨 주었을 뿐만 아니라 사용자에게는 보다 나은 다운로드 속도를 제공하는 등 많은 장점을 가져왔다. 본 논문에서는 P2P 비디오 스트리밍의 대표 방식인 push 와 pull 을 혼합하여 시기 적절하게 사용하는 새로운 push/pull 방식을 제안한다. 본 방식은 push 와 pull 을 chunk 분포도에 기초하여 선택적으로 사용하기 때문에 push 와 pull 방식의 장점을 모두 포함하게 된다

1. 서론

IPTV 등 인터넷 프로토콜을 기반으로 비디오 스트리밍을 제공하는 서비스는 현재의 미디어와 엔터테인먼트 산업을 바꿀 것으로 예상되는 강력한 기술로써 주목 받고 있다. 실제로 미국의 유명한 비디오 스트리밍 사이트인 YouTube 에서 비디오의 재생횟수는 하루 평균 2 천만번을 넘었으며 YouTube 가 소유한 컨텐츠들의 총 재생시간은 만년이 넘는다[1].

IPTV 시스템은 인프라스트럭처 기반과 p2p 기반으로 분류될 수 있다. 인프라스트럭처 기반의 시스템은 비디오 서버로부터 전략적으로 배치된 멀티캐스트 노드(예> CDN 등)를 거쳐서 일반 클라이언트에게 비디오 스트리밍이 이루어진다. 인프라스트럭처 기반의 시스템은 스트리밍 속도 향상을 위해 컨텐츠를 멀티캐스트 노드에 반복해서 저장하기 때문에 설립 비용이 비싸며 유지보수에 어려움이 많다. 반면 p2p 기반의 시스템은 각각의 클라이언트가 다른 클라이언트들에게 컨텐츠를 멀티캐스팅하기 때문에 클라이언트가 잠재적인 서버의 역할을 한다. P2P 기반의 시스템에서는 클라이언트들이 오버레이 네트워크를 구성하면서 자신의 업로드 용량을 이용해서 비디오 컨텐츠의 분배를 돕기 때문에 비용, 확장성 등의 측면에서 유리하며 따라서 인프라스트럭처 기반의 시스템보다 주목받고 있다.

P2P 스트리밍 시스템은 일반적으로 오버레이의 구조에 따라서 트리-push 방식과 메쉬-pull 방식으로 구분 할 수 있다. 트리-push 방식은 오버헤드가 적고 작은 딜레이를 가지지만 높은 churning rate 에 탄력성 있게 동작하기 힘들며 트리의 상단에 위치한 노드들의 충분한 업로드 용량이 보장되어야 충분한 스트리밍 성능을 보이는 단점이 있다[2]. 한편 메쉬-pull 방식은 디자인이 단순하며 높은 churning rate 에서도 훌륭한 성능을 보인다[3]. 하지만 메쉬-pull 방식은 높은 시그널링 오버헤드와 긴 시작 지연, 현저한 비디오 채널 전환 딜레이를 보이는 단점이 있다.

최근 push 와 pull 을 혼합하여 시그널링 오버헤드, diffusion delay, diffusion ratio, churning rate, 충돌 등에서 좋은 성능을 내기 위한 연구가 많이 진행되고 있다[5][6]. 본 논문에서도 push 와 pull 방식을 시기 적절하게 사용하는 새로운 push-pull 방식을 제안하고자 한다.

이 논문은 2009 년도 정부(교육과학기술부)의 재원으로 한국과학재단의 지원을 받아 수행된 연구임(No. 2009-0075682)

2. 관련 연구

Chunk 를 기반으로 하는 p2p 오버레이에서 컨텐츠의 전달방식은 일반적으로 오버레이의 구조에 따라서 트리 기반 방식과 메쉬 기반 방식으로 구분 할 수 있다. 트리기반 방식은 p2p 의 IP multicast 방식과 유사하며 어플리케이션 계층에서 소스 서버를 루트로 하는 트리가 형성된다. 트리기반 방식에서 피어는 기본적으로 하나의 트리내에 오직 한 명의 부모만 소유한다. 이 경우에 부모가 시스템을 떠나면 그 피어 뿐만 아니라 그 피어의 자손들 모두가 컨텐츠를 얻지 못하는 single point of failure 를 초래하게 된다. 피어가 시스템을 떠나는 것은 peer churn 이라고 일컬어지며 트리 방식에서 큰 난제로 자리잡고 있다. 따라서 많은 p2p 스트리밍 시스템이 메쉬 기반의 방식을 채택하고 있다. 메쉬 기반 방식은 먼저 파일 공유 방식에서 주로 사용되었다. 메쉬 기반 방식에서 피어들의 이웃관계 형성과 종료는 동적으로 행해지며 피어들은 다수의 피어와 관계를 맺고 그들과 동시에 컨텐츠를 주고받는 것이 가능하다 메쉬 기반 방식에서는 한 이웃이 시스템을 떠나도 다른 이웃으로부터 컨텐츠를 제공받을 수 있기 때문에 peer churn 이 문제 되지 않는다. 하지만 기존의 파일 공유에 사용된 메쉬는 데이터의 시간 의존성을 고려하지 않기 때문에 스트리밍 서비스에 바로 적용이 불가능했다. 따라서 이를 해결하고자 [7]등에서 피어들이 자신의 재생시점에 우선권을 부여한 방식이 제안되었다.

3. New Push-Pull Protocol

Push 와 pull 은 각각의 고유한 특성으로 인해서 chunk 의 분포도에 따라 성능상의 차이가 있다. Push 는 불필요한 딜레이가 없지만 충돌을 발생시키는 특성을 지니기 때문에 분포도가 낮은 chunk 를 분배할 때 유리하다. 반대로 pull 방식은 충돌이 없다는 장점을 가졌기 때문에 시스템에 널리 퍼진 chunk 를 분배할 때 유리하다. 따라서 본 연구진은 chunk 의 분포도에 따라서 push/pull 을 선택적으로 사용하는 방식을 제안하는 바이다.

3.1 Chunk-wise Push/Pull

Chunk 의 분포도에 따라서 시기 적절한 push/pull 을 선택하는 방법으로 chunk 별로 push/pull 을 선택하는 방안을 제안한다. 본 push/pull 방식에서의 피어는 기본적으로 playback buffer, pull requester, pull signal queue, pull signal dispatcher, push scheduler, chunk delivery queue 등의 모듈을 소유한다. 먼저 재생되기 전의 chunk 들을 모으고 저장하며 미디어 플레이어에 전달하는 역할을 하는 playback buffer 가 존재한다. Pull requester 는 필요한 chunk 를 이웃에 요청하는 역할을 한다. Pull signal queue 모듈은 이웃들로부터 요청된 pull request 를 처리해주는 역할을 하며 전달 받은 pull signal 을 queue 에 차례대로 저장하고 pull signal dispatcher 에 전달한다. Pull signal dispatcher 는 pull signal 이 가리키는 해당 chunk 를 chunk delivery queue 에 추가하여 chunk delivery queue 가 해당 이웃에게 전달하도록 돕는다. 한편 push scheduler 모듈은 그림. 1 과 같이 playback buffer 내에서 어떤 chunk 를 push 방식을 이용해서 전달할 것인지를 선택하는 역할을 한다. Push scheduler 는 전달 받은 이웃들의 버퍼 맵을 기반으로 각 chunk 별로 희귀도를 측정한다. 이후 희귀도 값이 높은 즉, chunk 의 분포도가 낮아서 충돌이 적게 발생할 것으로 예상되는 chunk 들이 push 로 전달할 대상이 되어 chunk delivery queue 에 전달된다. 왜냐하면 첫째로 희귀도가 낮은 chunk 의 경우에는 push 로 전송하면 충돌에 따른 자원낭비가 발생하고, 둘째로 push 방식을 통해 분포도가 낮은 chunk 의 분포도를 높이면 후에 해당 chunk 를 얻지 못한 피어들이 pull 방식으로 그 chunk 를 획득할 기회가 높아지기 때문이다. 마지막으로 chunk delivery queue 는 전달된 chunk 들을 차례차례 이웃에게 전달하면서 본 시스템을 완성한다.

본 방식은 현재 PeerSim 시뮬레이터를 이용하여 구현하였으며 다른 p2p 스트리밍 방식들과의 비교연구를 진행 중에 있다.

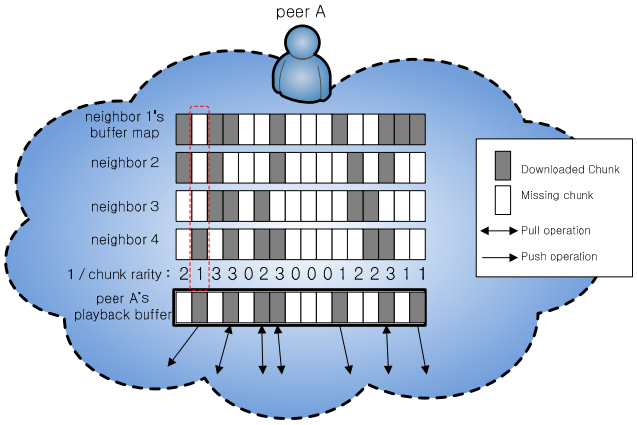

그림. 1 Chunk-wise Push/Pull(Push Scheduler)

4. 결론

본 논문에서는 push 와 pull 방식을 시기 적절하게 사용하는 새로운 push-pull 방식을 제안하였다. 본 방식의 장점은 chunk 분포도에 기초하여 push 와 pull 을 조율하기 때문에 push 방식과 pull 방식의 장점을 모두 포함한다는 것이다.

참고문헌

[1] L. Gomes, "Will All of Us Get Our 15 Minutes On a YouTube Video?" in Wall Street Journal, August 30, 2006.

[2] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "Splitstream: High-bandwidth multicast in cooperative environments", in ACM SOSP 2003, Lake Bolton, New York, October, 2003

[3] F. Pianese, D. Perino, J. Keller, and E. Biersack, "PULSE: An Adaptive, Incentive-Based, Unstructured P2P Live Streaming System", in IEEE Transactions on Multimedia, Vol. 9, No. 8, December 2007

[4] M. Zhang, Q. Zhang, and S. Yang, "Understanding the Power of Pull-based Streaming Protocol: Can We Do Better?", in IEEE Journal on Selected Areas in Communications, special issue on Advances in Peer-to-Peer Streaming Systems, Vol. 25, No. 8, December 2007

[5] M. Zhang, J. Luo, and S. Yang, "A peer-to-Peer Network for Live Media Streaming– Using a Push-Pull Approach", in ACM Multimedia 2005

[6] A. Vlavianos, M. Iliofotou and M. Faloutsos, "BiToS: Enhancing BitTOrrent for Supporting Streaming Applications", in IEEE INFOCOM 2006, Barcelona, April 23-29, 2006