대용량 파일 전송의 성능 향상을 위한 다중 가상 소스 응용계층 멀티캐스트[1]

이수전°†, 강경란‡, 이동만†

한국정보통신대학교†
{galgyeony, dlee}@icu.ac.kr†

아주대학교‡
korykang@ajou.ac.kr‡

Throughput enhancement for large file delivery using overlay multicast with multiple virtual sources

Soojeon Lee°†, Kyungran Kang‡ and Dongman Lee†
School of Engineering, Information and Communication University, †
College of Information & Computer Engineering, Ajou University‡

요 약

다자간 데이터 송수신을 필요로 하는 다자간 화상 회의, 일대다 스트리밍, 일대다 파일 전송 등의 응용에서는 멀티캐스트를 사용하여 데이터 전송 부담을 줄일 수 있다. 그러나 IP 멀티캐스트가 인터넷 상에서 널리 지원되지 않고 있기 때문에, 멀티캐스트를 응용 계층에 구현하려는 연구가 활발히 진행되고 있다. 본 논문에서는 일대다 대용량 파일 전송에서 효과적으로 전송시간을 단축할 수 있는 응용계층 멀티캐스트 기법을 제시한다. 신뢰적 전송을 위하여 Forward Error Correction (FEC)를 사용하며, 전송 시간 단축을 위해 데이터 소스 외에 수신자들이 가상적으로 소스의 역할을 수행하도록 한다. 즉, 임의의 수신자는 데이터 소스뿐만 아니라, 다중 가상 소스로부터도 데이터를 수신할 수 있으므로 파일 수신 시간을 단축시킬 수 있다.

1. 서 론

다자간 화상 회의, 일대다 스트리밍, 일대다 파일 전송 등의 응용에서는 멀티캐스트를 사용하여 데이터 전송 부담을 줄일 수 있다. IP 멀티캐스트가 가장 효율적으로 네트웍 자원을 활용하여 멀티캐스트를 지원할 수 있지만, 오랜 기간의 노력에도 불구하고, 인터넷 상에서는 널리 지원되지 않고 있는 실정이다. 따라서, 멀티캐스트 기능을 응용 계층에서 구현하려는 연구가 활발히 진행되고 있다. 응용 계층 멀티캐스트는 IP 멀티캐스트에 비하여 제작 및 적용이 용이하다는 장점을 갖기 때문이다.

응용계층 멀티캐스트에서 데이터를 전송할 때 소스와 전체 수신자들 사이에 소스가 루트가 되는 스패닝 트리를 구성하는 것이 일반적이다. 그런데, 이러한 트리 구조는 크게 두 가지 약점을 갖는다. 첫째, 트리의 중간 노드 중 네트웍 환경이나 컴퓨팅 성능이 안 좋은 노드에 의해 그 자손 노드들이 체감하는 효율이 떨어질 수 있다. 둘째, 각 멤버는 단 하나의 부모 노드로부터 데이터를 받아야 하기 때문에 부모 노드에게 갑작스런 오류 (failure) 가 발생하면 데이터 수신에 실패하게 된다. 본 논문은 이러한 트리 구조의 약점들을 보완하기 위하여 다중 가상 소스로부터 데이터를 수신하는 기법을 제안하며 이때 발생할 수 있는 동일 패킷의 중복 수신 문제를 해결한다.

본 논문의 구성은 다음과 같다. 2장에서 관련 연구들을 살펴보고, 3장에서 대용량 파일의 효율적인 전송을 위한 응용계층 멀티캐스트 기법을 고안할 때 고려할 사항들에 대해서 논의한다. 4장에서 다중 가상 소스 응용계층 멀티캐스트를 제안하며, 5장에서 결론 및 향후 계획을 제시하는 것으로 본 논문을 마친다.

2. 관련 연구

[3,4,6]은 멀티미디어 스트리밍이나 파일전송과 같이 하나의 송신자가 다중 수신자에게 데이터를 전달할 경우에 사용될 수 있다. 각 노드들은 "store-and-forward" 방식으로 자식 노드에게 데이터를 전달하므로, 수신 버퍼에 도착한 모든 패킷들은 모든 자식 노드로 포워드 되어야 한다. 따라서 응용계층 멀티캐스트의 전송속도는 가장 느린 링크에 의하여 제한된다. 하지만 [5]가 제안한 "forward-when-feasible" 방식에서는 수신한 패킷에 대해 그 패킷을 즉시 받을 수 있는 자식 노드로만 전달한다. 따라서 가장 느린 링크가 전체 멀티캐스트 세션의 병목이 되지 않는다는 장점을 가진다. 데이터를 Forward error correction 기법에 따라 encoding을 하므로 패킷들을 순차적으로 모두 수신해야만 할 필요가 없기 때문에 가능하다.

[3,6]은 서론에서 언급한 트리 구조가 가진 약점을 보완함으로써 전송 속도를 향상시킨다. [6]은 네트워크 코딩 기법과 두 개의 잉여 수신 플로우 (two-redundant incoming flow) 를 사용함으로써 전송속도 향상을 꾀한다. [3]의 경우는 기존의 트리 구조 위에 응용계층 메쉬를 생성하고 이 메쉬를 기반으로 peer-to-peer 기법을 사용한다. 즉, 복수개의 송신자로부터 데이터 수신을 가능케 함으로써, 전체 데이터를 수신하는데 걸리는 시간을 줄인다. 그러나, 복수개의 멤버들로부터 데이터를 수신하기 때문에 패킷의 중복

[1] 본 연구는 과학기술부가 주관하는 국가지정연구실사업 (NRL: M1-0318-00-0130) 의 지원으로 수행되었음

수신이 발생한다는 단점이 있으며, 자신이 필요로 하는 데이터를 가지고 있는 멤버를 찾는 과정을 지속적으로 반복해야 하는 부담을 지닌다.

3. 설계 시 고려 사항

3.1 사용자의 가입/탈퇴 패턴

파일 전송의 경우는 사용자의 가입 시각의 예측이 매우 어렵다. 인터넷 상에 대용량 파일 (영화, 소프트웨어 배포판)을 배포하는 유명 서버가 존재하고 응용계층 멀티캐스트를 통해 수 많은 사용자가 파일을 수신하는 경우를 가정해보자. 이때 이 파일을 수신하고자 하는 참여자가 언제 세션에 가입할 것인지를 미리 예측하기란 어렵다. 탈퇴를 언제 할 것인지도 예측하기 어렵다. P2P 파일 공유 시스템[9]의 경우를 보면 알 수 있듯이, 파일 수신이 끝나자마자, 즉시 세션에서 탈퇴하는 사용자도 있을 것이고, 파일을 다 받았다 할 지라도 오랫동안 세션에서 나가지 않는 사용자도 있을 것이다. 다만, 임의의 사용자가 세션에 가입했을 때, 이미 수 많은 기존 멤버들이 데이터를 수신 중이거나 혹은 수신을 마치고 세션에 남아있을 것이라 예측할 수 있다. 여기서 수신을 마치고도 세션에 남아서 가상 소스로 활동하는 기간을 봉사 시간 (sacrificing time) 이라 한다. 따라서 기존 멤버들이 수신한 데이터를 최대한 활용하면, 결과적으로 자신의 파일 수신 시간을 단축시킬 수 있다.

3.2 루트 노드 (파일 서버)

3.2.1 Forward Error Correction Coding

네트워크를 통한 패킷 전송시 패킷 손실 (loss) 의 문제를 극복하기 위하여 tornado codes [10]를 사용한다고 가정한다. 하나의 파일을 구성하는 패킷의 수를 k 이라 했을 때, 이를 인코딩하여 $n=(k+l : l$은 잉여(redundant) 패킷의 개수) 개의 패킷을 만든다. 이 n개의 인코딩 패킷 중 임의의 서로 다른 $(1+\varepsilon)k$개의 패킷을 수신하면 디코딩을 통하여 파일을 복구할 수 있다. $n=Ck$ 일 때 C는 *stretch factor* 라 불리며 C가 커질수록 인코딩/디코딩 시간이 증가한다. Cyclone server architecture [2]의 경우 $C=10$ 을 사용하고 있다.
예를 들어, $n=10,000$, $C=10$, $\varepsilon=0.05$ 이라 가정해보자. 이때 서버는 그림 1과 같이 circular queue를 이용하여 *[0, 9999]* 번 패킷을 반복적으로 보내게 된다. 만약 9999 번 패킷의 전송이 끝나면, 다시 *0*번 패킷부터 전송을 시작한다. 따라서, 수신자는 이중 *1,050* 개의 서로 다른 패킷을 받음으로써 파일을 완성시킬 수 있다. 전송은 [5]의 forward-when-feasible 방식을 따른다.

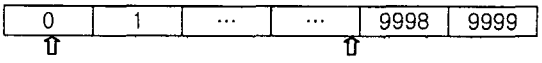
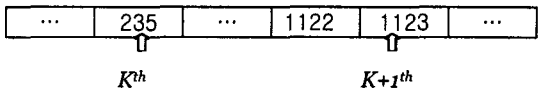


그림 1. 서버의 패킷 전송

3.2.2 Bootstrap 노드

1	*Addr_1*	*Key_seq_1*
2	*Addr_2*	*Key_seq_2*
…		
n	*Addr_n*	*Key_seq_n*

(a) 튜플



(b) 가입 순서

그림 2. 셰딩 인터벌

임의의 멤버가 세션에 가입하기 위해서는 bootstrap 노드에 접속해야 한다. Bootstrap 노드는 그림 2 (a) 와 같이 가입한 멤버들의 *<멤버 주소, 그 멤버가 처음 받은 멀티캐스트 패킷의 시퀀스 번호>* 튜플을 가입 순서대로 유지한다. 본 논문에서는 루트 노드인 파일 서버가 그 역할을 대신한다.

4. 다중 가상 소스 응용계층 멀티캐스트

다중 가상 소스 응용계층 멀티캐스트는 임의의 트리기반 응용계층 멀티캐스트 기법 위에서 동작한다. 그림 3에서의 실선은 임의의 트리기반 응용계층 멀티캐스트를 통한 데이터 전송을 나타내며, 점선은 가상 소스를 통한 데이터 전송을 나타낸다. 물론, 트리만으로도 정상적인 데이터 전송을 할 수는 있지만, 전송속도를 높이기 위하여 본 논문에서는 트리 기반 기법과는 별도로 동작하는 전송 기법을 제안한다.

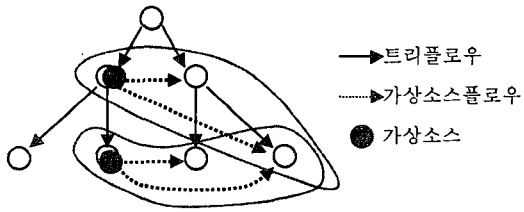


그림 3. 다중 가상 소스 응용계층 멀티캐스트

하나의 가상 소스는 전체 패킷 집합 $S=\{x|x\in[0,n]\}$ 가운데 특정 패킷 영역 (예를 들어, *[13,45]*) 에 대해서만 소스 역할을 담당한다. 기본적으로 모든 멤버가 가상 소스의 역할을 수행할 수 있으며, 멤버들의 가입 (join) 인터벌이 매우 작을 경우에는, 선택된 특정 멤버들만이 가상 소스의 역할을 수행할 수 있다. 이때 모든 가상 소스들은 서로 다른 패킷 영역에 대하여 송신을 담당하므로, 패킷의 중복 수신을 방지할 수 있다. 그림 2 (b) 에서와 같이 멤버 K는 다음 멤버인 $K+1$이 가입하기 전까지 루트로부터 수신한 멀티캐스트 패킷 영역인 *[235, 1122]* 에 대한 가상 소스가 된다.

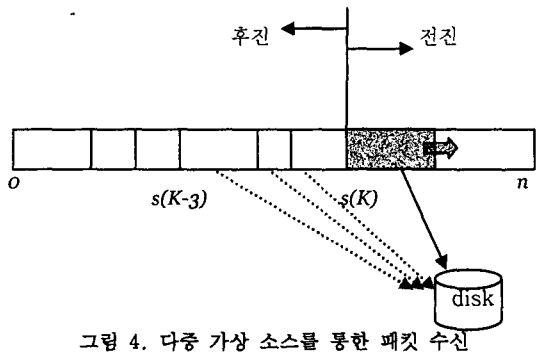


그림 4. 다중 가상 소스를 통한 패킷 수신

따라서, 이후에 새로 세션에 참여한 멤버는 트리를 통하여 루트로부터의 멀티캐스트 데이터를 수신함과 동시에 기존 멤버들 가운데 존재하는 가상 소스들로부터도 데이터를 수신할 수 있다. 그림 4의 *s(K)*는 *K*번째 멤버가 세션에 참가했을 때 트리를 통하여 받은 최초 멀티캐스트 패킷의 시퀀스 넘버이며 수신을 계속 함에 따라 패킷 시퀀스 넘버는 증가할 것이다. 이를 "전진"이라 한다. 이때, *K-1* 번째 멤버는 패킷 영역 *[s(K-1), s(K)-1]* 에 해당하는 가상 소스로 동작한다. 각각의 가상 소스가 제공하는 패킷 영역에 대한 정보는 그림 2 (a) 에서 설명한 bootstrap 노드의 튜플을 통해서 알아낼 수 있다. 결국 멤버 K는 이 영역에 해당하는 패킷을 멤버 *K-1* 로부터 받게 된다. 마찬가지로 기존 멤버들인 *K-2, K-3* 으로부터도 각각에 해당하는 패킷 영역에 대하여 데이터를 수신할 수가 있으며 따라서, 패킷의 중복 수신을 방지하면서 전송속도를 높인다. 이와 같이, 가상 소스를 통하여 패킷 시퀀스 넘버가 감소하는 방향으로 수신을 계속하는 것을 "후진"이라 한다.

임의의 멤버가 세션에 가입하여 파일을 다 받는 데에 걸리는 시간을 *t*라 하고, 루트로부터 수신되는 멀티캐스트 스트림의 속도를 V_m 이라 하자. 이때 기존의 트리기반 응용계층 멀티캐스트 프로토콜의 파일 수신시간 *t*는 아래의 식으로 계산 가능하다.

$$V_m \times t = File\_size \ldots (1)$$

임의의 멤버가 세션에 가입한 순간에 존재하고 있던 가상 소스의 개수를 *n*이라 하고, 각각의 가상 소스가 루트로부터 수신하는 (또는 했던) 멀티캐스트 스트림의 속도를 $V_m(i), (i=1,\ldots,n)$ 라 하자. 그리고, 가상 소스 *i* 와 *i+1* 간의 세션 가입 간격 (join interval) 을 *h(i)* 라 하면, 가상 소스 i의 패킷 영역 크기는 $V_m(i) \times h(i)$ 가 된다. 가상 소스가 세션에 충분히 오랫동안 머무르고, 모든 가상 노드를 동시에 이용할 수 있다고 가정해보자. 멤버 자신과 가상 소스간의 전송 속도를 *V(i)* 라 했을 경우, 제시한 기법의 파일 수신 시간 *t* 는 식 (1)을 이용하여 아래와 같이 계산된다.

$$\sum_{i=1}^{n} \min(V(i) \times t, V_m(i) \times h(i)) + V_m \times t = File\_size \ldots (2)$$

5. 결론

제시한 기법의 전송속도는 기존 멤버들이 많을수록, 또 그들의 봉사 시간이 길수록 증가한다. 또한, 가상 소스들은 패킷 시퀀스상의 특정 영역에 대하여만 소스 역할을 담당한다. 이때 임의의 가상 소스가 담당하는 패킷 영역은 다른 가상 소스의 영역과 서로소이다. 또한, 가상 소스로부터 수신하는 데이터는 루트 노드로부터 수신하는 데이터와 서로소이다. 임의의 노드가 한 싸이클 즉, *n(=Ck)* 시퀀스가 지나기 전에 *(1+ε)k* 개의 패킷을 받을 수 있다면 패킷의 중복수신은 일어나지 않는다. 다만, 극도로 느린 멤버는 한 싸이클을 넘어서서 패킷을 받게 되므로 패킷 중복 수신이 가능한데, 이 경우에도 [2]의 pure random transmission 을 통하여 패킷 중복률을 크게 낮출 수 있다. 따라서, [3]에서 측정되는 10% 이상의 패킷 중복률을 크게 낮출 수 있을 것으로 예상된다. 또한, [3]에서는 자신이 필요로 하는 데이터를 가지고 있는 멤버 (송신자) 를 찾는데 시간이 걸리는데 반해, 본 논문에서 제시한 기법을 사용하면 송신자를 찾는 과정을 거치지 않고 즉각적으로 가상 소스로부터 데이터를 수신할 수 있기 때문에, 전송속도 향상을 기대할 수 있다.

본 논문에서 제시된 기법은 MACEDON[8] 을 이용하여 구현중이며, [3]과 성능 비교를 할 예정이다. 특히, 실제 인터넷 상에서의 실험 결과를 도출해 내기 위해 플레닛랩 [7] 을 이용할 것이다.

6. 참고 문헌

[1] Y.-H.Chu, S.G.Rao, S.Seshan, and H. Zhang, "A case for end system multicast,", in IEEE JSAC, Sp. Issue on Network Support for Group Communication, 2003.

[2] Rost, S., Byers, J., and Bestavros, A. The Cyclone Server Architecture: Streamlining Delivery of Popular Content. In Intl Workshop on Web Caching and Content Distribution (Boston, MA, June 2001).

[3] D.Kostic, A. Rodriguez, J. Albrecht, and A. Vahdat, "Bullet: High bandwidth data dissemination using an overlay mesh", ACM (SOSP 2003), Oct. 2003.

[4] Z.Li and P.Mohapatra, "Hostcast: A new overlay multicasting protocol", in Proc. IEEE 2003 Intl. Conf. On Communications (ICC 2003), May 2003.

[5] Gu-In Kwon and John Byers, "ROMA: Reliable Overlay Multicast with Loosely Coupled TCP Connections", in Proc. of IEEE INFOCOM 2004.

[6] Ying Zhu, Baochun Li, and Jiang Guo, "Multicast with Network coding in application-layer overlay networks", IEEE JSAC, January 2004.

[7] http://www.planet-lab.org

[8] A.Rodriguez, S.Bhat, C.Killian, D.Kostic, and A.Vahdat. MACEDON: Methodology for automatically Creating, Evaluating, and Designing Overlay Networks. Technical Report CS-2003-09, Duke University, July 2003.

[9] http://www.gnutella.com

[10] J.byers, M.Luby, M.Mitzenmacher, and A.Rege, A digital fountain approach to reliable distribution of bulk data. In ACM SIGCOMM, Vancouver, Canada, 1998.