TCP와 유사한 RTP/RTCP 흐름제어 알고리즘

나승구[T], 윤성덕, 안종석
동국대학교 컴퓨터공학과

A TCP-like flow control algorithm for RTP/RTCP

Seung-Gu Na, Sung-duck Yun, Jong-Suk Ahn
Dept. of Computer Engineering Dongguk university

요 약

최근, 멀티캐스트 기법을 사용하는 멀티미디어 응용 프로그램들이 인터넷에 등장하고 있다 이들 응용 프로그램들의 성공 여부는 수신자들에게 전송되는 음성/영상의 품질에 의해 좌우된다. 인터넷은 응용프로그램의 QoS(Quality Of Service)에 대한 요구를 보장할 수 없기 때문에 멀티캐스트 트래픽(multicast traffic)을 위하여 인터넷의 성능을 최대한 효율적으로 이용할 수 있도록 흐름제어에 대한 많은 연구가 진행되고 있다. 그 중 IVS(INRIA Video conferencing System)에서 제안한 멀티캐스트 트래픽 흐름제어 알고리즘은 수신자가 주기적으로 전달하는 RTCP의 패킷손실 정보에 의해 송신자가 전송율을 조절하는 것이다 그러나 이 알고리즘은 네트워크 상태가 부부하(unload)임에도 불구하고 느린 피드백으로 인하여 가용 네트워크 대역폭을 빠르게 파악하지 못하기 때문에, TCP 트래픽과 경쟁상태에서 네트워크 대역폭을 불공정(unfairness)하게 사용하게 되고 네트워크 상태에 알맞는 전송율을 결정하지 못한다 본 논문에서는 더욱 공정하게 대역폭을 공유할 수 있고 전체 링크 이용율을 높이는 두 가지 기법을 제안한다. 첫째 측정된 네트워크 혼잡 상태에 따라 RTCP 피드백의 전송 빈도를 동적으로 조절하는 것이다. 둘째, TCP와 같이 전송율을 증가/감소시킴으로써 공정하게 네트워크를 공유하도록 하는 것이다 본 논문에서는 이 두 가지 기법들이 TCP 트래픽에 영향을 주지 않고 또한 RTCP 피드백의 양을 증가시키지 않으면서도 공정하게 네트워크 대역폭을 공유함으로써 링크의 이용율을 높일 수 있다는 것을 시뮬레이션을 통하여 보여준다

1. 서 론

최근 들어 네트워크가 급속도로 성장하고 다양한 서비스들을 제공하고 있는 가운데 멀티캐스트 기법들을 사용한 많은 멀티미디어 응용 프로그램들이 소개되고 있다. 특히 인터넷의 인기로 인하여 여러 참가자들 사이에 멀티미디어 데이터를 교환하는 많은 응용프로그램들이 인터넷상에 등장하기 시작했다. 예를들면, Mbone상에 개발된 vic[1]이나 vat과 같은 다자간 회의시스템을 들 수 있다. 이러한 툴들은 마치 가정의 전화처럼 우리 생활 속으로 급격하게 확산될 것으로 기대된다.

인터넷과 같이 최선의 노력(best-effort)으로 전송하는 네트워크에서 보다 좋은 품질을 제공하기 위한 방법은 멀티캐스트(multicast)를 위한 종단간(end-to-end)의 흐름제어이다. 이 방법은 가용 네트워크 대역폭을 측정하여 전송율을 동적으로 조절함으로써 네트워크 성능을 최대한 높이는 것이다. 이 방법에 속하는 것으로 두 가지 대표적인 알고리즘이 제안되고 있다.

첫째, IVS(INRIA Video-conferencing System)[2]에 의해 제안된 방법은 송신자가 수신자들의 평균 혼잡상태를 측정하고 평균 혼잡상태에 따라서 송신자가 전송율을 조절한다.

둘째, McCane에 의해 제안된 RLM(Receiver-driven Layered Multicast)[3]은 송신자가 수신자들의 다양한 멀티캐스트 그룹을 고려하여 하나의 데이터 프레임을 여러 개의 데이터 스트림으로 나누어 전송하고 각 수신자는 자신의 네트워크 상태에 따라 전송받을 데이터 스트림을 결정한다. 송신자가 멀티미디어 데이터를 여러 개의 데이터 스트림으로 나누는 것을 계층화된 코딩(layered coding)이라 하는데 수신자는 데이터 스트림을 많이 받을수록 양질의 데이터를 제공받을 수 있다.

본 논문에서는 IVS의 효율성을 개선시킬 수 있는 두 가지 기법을 소개한다. 첫째, 송신자가 네트워크 상태를 빠르게 파악할 수 있도록 하기 위하여 네트워크 상태에 따라 RTCP 피드백의 빈도를 동적으로 바꾸어 주고, 둘째, IVS가 네트워크 자원을 공정하게 공유하도록 하기 위해 TCP만큼 전송율을 증가시키는 것이다. 이 두 가지 방법을 통하여 IVS가 피드백의 양을 크게 증가시키지 않고 네트워크 자원을 공정하게 공유할 수 있으며 더욱 높은 성능을 발휘한다는 것을 보여준다.

본 논문의 구성은 다음과 같다. 2절에서는 제안된 두 가지 흐름제어 기법에 대해 자세하게 소개하고 3절에서는 시뮬레이션 결과와 인터넷상의 실제 실험 결과를 보여준다. 마지막으로, 4절에서는 실험 결과를 요약하고 향후 연구방향에 대하여 기술한다.

2. 제안된 두 가지 흐름제어 기법

본 절에서는 IVS처럼 RTP[4]를 사용하는 응용프로그램에서 네트워크 자원의 공정성과 효율성을 증진시킬 수 있는 두 가지 기법에 대하여 설명한다. 첫째, 송신자가 수신자의 네트워크 상태를 빠르게 파악할 수 있도록 하기 위하여 네트워크 상태에 따라 RTCP 피드백의 빈도를 동적으로 바꾸어 주는 기법이다.

RTP는 TCP와 같은 공격적인 알고리즘과 같이 사용할 경우 TCP에 의해 대부분의 대역폭을 점령당하게 된다. 왜냐하면 TCP는 슬라이딩 윈도우 프로토콜(sliding-window protocol)을 사용하는데 수신자로부터 ACK가 하나씩 도착할 때마다 윈도우 크기를 1씩 증가시키기 때문에 무부하 상태일때는 전송율이 급격하게 상승하게 된다.

반면에 RTP에서는 수신자로부터 약 5초 간격으로 RTCP 패킷이 전송되는데, 수신자가 현재의 네트워크 상태를 파악한 후 송신자에게 전달되기까지는 지연시간이 너무나 크기 때문에 송신자가 네트워크 상태에 맞는 전송율을 결정하지 못하고 TCP 프로토콜에 대역폭을 빼앗기게 되는 것이다. 그리므로 네트워크 상태에 따라 무부하 상태일 때는 보통으로, 부하 상태일 때는 조금 빠르게, 혼잡 상태일 때는 빠르게 RTCP 피드백의 빈도를 동적으로 바꾸어 주는 것이다.

둘째, IVS가 네트워크 자원을 공정하게 공유하도록 하기 위해 TCP와 유사하게 전송율을 증가시키는 것이다.

현재 IVS는 네트워크가 무부하(unload) 상태라고 판단되었을 때 전송량을 마음대로 결정한다. 만약 증가량이 너무나 크게되면 네트워크는 갑자기 혼잡상태에 이르게 되고 다른 트래픽들은 많은 패킷손실을 경험할 것이다. 한편, 증가량이 너무 작게 되면 네트워크 대역폭이 IVS에 불공정하게 할당된다. 그러므로 [그림 1]의 알고리즘과 같이 TCP와 유사한 방법으로 데이터의 전송량을 증가시키는 것이다.

```
#define BETA            0.1
#define DEC_AMOUNT   0.875  // 3/4
#define MIN_RATE      3000  // 3kbps(최소 전송율)

if (decision == UNLOADED) {     // 무부하 상태
    if (rate_ < rthresh_)
 ①      rate += delta;          // 선형적 증가
    else if (rate_ > rthresh_)
 ②      rate_ -= (cur_rtt/avg_rtt) * MTU; // 지수적 증가
    if (rmax_ < rate_)     // 최고 대역폭 찾기
        rmax_ = rate_;
        congest_flag = 1;
}
else if (decision == CONGESTED) {     // 혼잡 상태
    if (congest_flag)  // 대역폭 상승에서 감소로 전환
 ③      rthresh = (1 - BETA) * rthresh + BETA * rmax_ / 2;
    rate_ = max(rate_ * DEC_AMOUNT, MIN_RATE); // 전송량 결정
    congest_flag = 0;
}
```

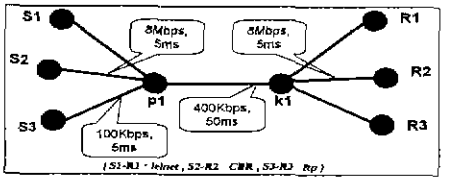
[그림 1] TCP와 유사한 알고리즘

TCP에서는 Slow Start 단계와 Congestion Avoidance 단계로 나누어서 흐름제어를 하는데 네트워크 상태가 혼잡상태로 감지되면 Slow Start 단계로 진입하게 된다. 이 단계에서는 데이터의 전송량을 급격하게 떨어뜨리게 되고 ①경계값(threshold)까지는 일정량 delta(10kbps)를 증가시킨다. 경계값을 넘어서게 되면 Congestion Avoidance 단계로써 ②(cur_rtt/avg_rtt) * MTU만큼씩 증가시키게 된다. cur_rtt는 현재의 전송시간, avg_rtt는 여러 수신자들의 평균 전송시간, MTU는 데이터 전송 단위를 뜻한다. ③경계값은 체증상태를 경험하기 이전의 최대 전송율의 반값을 취한다.

3. 실험

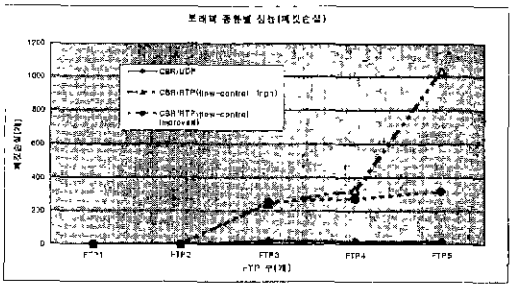
3.1 흐름제어의 필요성에 대한 실험

본 절에서는 인터넷상에서 흐름제어가 과연 필요한가를 증명하기 위하여 트래픽의 종류에 따라 성능을 시뮬레이션하여 측정하였다. 실험환경은 SUN Ultra-I(CPU 174Mhz, 64M 주메모리)워크스테이션에서 Solaris 2.6 운영체제를 이용하였으며 본 실험에 이용된 시뮬레이터는 LBL(Lawrence Berkeley Laboratyry)에서 개발된 ns(network simulator)[5]이다. [그림 2]는 성능 측정을 위한 네트워크 토폴로지를 보여주고 있다. 총 10개의 노드로 구성되어 있고 노드 P1과 노드 K1의 링크 대역폭은 400Kb/s, 50ms의 전파지연을 갖고, 노드 S3-P1의 링크 대역폭은 100Kb/s, 나머지 모든 링크 대역폭은 8Mb/s, 5ms의 전파지연을 갖고, 모든 노드의 큐 크기는 6으로 설정하였다. 각 실험시간은 1800초(30분)이다.



[그림 2] 네트워크 시뮬레이션 구성도(I)

실험은 노드 S1-R1은 telnet을 백그라운드 트래픽으로 설정하고, S2-R2는 CBR의 트래픽을, S3에서 노드 R3로 ftp를 실행시켰다. [그림 3]은 CBR의 트래픽을 3가지로 분류하여 두고 ftp의 개수를 1개씩 점차 늘려가며 패킷손실에 의한 성능을 실험한 결과이다. ①은 CBR/UDP, ②는 Ingo Busse 흐름제어 알고리즘에 의한 CBR/RTP, ③은 개선된 흐름제어 알고리즘에 의한 CBR/RTP 트래픽을 적용한 것이다. CBR/UDP의 성능이 CBR/RTP의 성능보다 더 좋고, CBR/RTP들 중에서는 Ingo Busse의 알고리즘을 적용한 것보다 이를 개선시킨 흐름제어 알고리즘을 적용한 것의 성능이 좋은 것으로 나타났다. 또한 P1과 K1 사이의 링크 병목지점에서 링크 이용율(CBR패킷수/전체패킷수)을 측정한 실험에서는 3가지 트래픽이 거의 비슷한 값을 나타냈다.

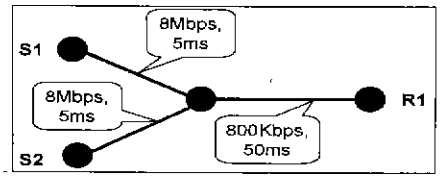


[그림 3] 트래픽 종류별 성능(패킷손실)

3.2 피드백 빈도 조절에 대한 실험

본 절에서는 IVS에서 네트워크 상태에 따라 RTCP 피드백 정보의 전송빈도를 조절하였을 때 영상회의 시스템의 성능을 시뮬레이션하여 측정하였다. 실험환경은 위 실험과 동일하며 [그림 4]는 성능 측정을 위한 네트워크 토폴로지를 보여주고

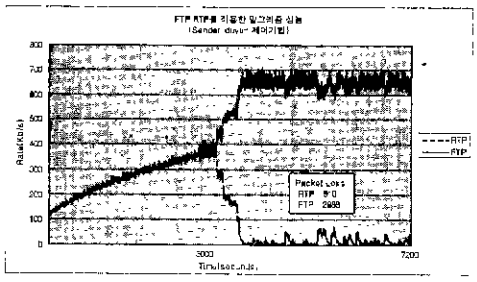
있다. 총 4개의 노드로 구성되어 있고 노드0와 노드2의 링크 대역폭은 8Mb/s, 5ms의 전파지연을 갖고, 노드1과 노드2의 링크 대역폭은 8Mb/s, 5ms의 전파지연을 갖고, 노드2와 노드3의 링크 대역폭은 800Kb/s, 50ms의 전파지연을 갖고, 각 노드의 큐 크기는 6으로 설정하였다 각 실험시간은 7600초(2시간)이다.



[그림 4]네트워크 시뮬레이션 구성도(II)

먼저 IVS에서 RTP 프로토콜이 TCP 프로토콜과 같이 사용되었을 때 네트워크 대역폭을 공정하게 사용하지 못한다는 것을 보이기 위한 실험을 하였다. 이 실험에서는 S1이 R1으로 화상회의의 RTP 데이터를 먼저 전송하고 있다가 3000초 후에 S2가 R1으로 FTP 전송을 시작하도록 하여 전송율의 변화를 측정하였다.

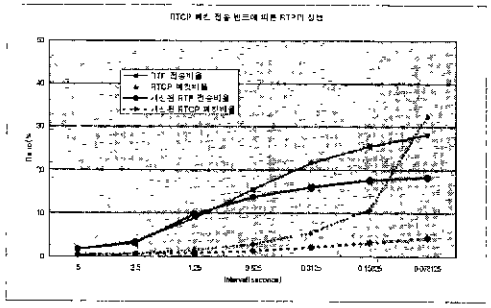
[그림 5]는 두 가지 프로토콜(RTP, TCP)의 성능을 나타낸 것이다 S1의 RTP 전송율은 링크의 대역폭을 향해 계속 증가하다가 3000초 지점에 이르러서는 FTP전송이 시작되면서 급격히 하강하게 되고 반면에 FTP의 전송율은 급상승하게 된다 TCP프로토콜을 사용하는 FTP는 매우 공격적인 알고리즘이기 때문에 재빠르게 RTP의 대역폭을 점령해버리기 때문이다. 이와 같은 현상은 네트워크 자원을 불공정하게 사용하는 결과가 되기 때문에 매우 비효율적이라 할 수 있다.



[그림 5] IVS에서 RTP와 FTP를 적용한 알고리즘의 성능

이 같은 문제점을 개선하기 위해 본 논문에서는 다음과 같은 실험을 하였다 RTCP 전송간격을 각각 5초, 2.5초, 1.25초, 0.625초. 0.3125, ... 로 변화시켜 가면서 TCP 트래픽과 RTP 트래픽을 동시에 전송하였을 때 RTP의 대역폭 사용비율과 RTCP 패킷의 발생비율을 측정하였다. [그림6]에서 보는 바와 같이 RTCP의 전송 빈도를 높일 경우 TCP에 대한 RTP의 전송율은 20%이상 증가하였다

이와 같이 RTCP 패킷의 발생빈도를 높일수록 네트워크 상태를 신속하게 전달하기 때문에 TCP 트래픽과 정쟁관계를 유지할 수 있다 그러나 네트워크 상태를 전혀 고려하지 않고 RTCP 패킷의 발생빈도를 내내 고정된 값으로 설정하였기 때문에 RTCP 패킷 전송 간격이 짧을수록 피드백 정보의 양이 급증하는 문제점이 발생하였다.



[그림 6] RTCP 패킷 전송 빈도에 따른 RTP의 성능

이 점을 개선하기 위해 [그림 6]과 같이 네트워크 상태에 따라 RTCP 패킷의 발생빈도를 동적으로 조절하는 방법에 의해 실험하였다. X축의 1.25초 지점을 예로 들면, RTCP 패킷의 발생빈도를 네트워크 상태에 따라서 부부하상태일 때는 5초로, 부하상태일 때는 2.5초로, 혼잡상태일 때는 1.25초로 전송하는 것이다. 그 결과 네트워크 상태에 따라 RTCP 패킷의 전송 빈도를 동적으로 빈도를 조절할 경우 기존의 방식에 비해 20%정도 전송비율이 증가하였고 피드백 정보의 증가량은 거의 변화가 없다는 것을 알 수 있었다.

4. 결론 및 향후 연구방향

본 논문에서는 RTP를 이용한 응용 프로그램에서 네트워크 상태에 따라 피드백 정보의 전송 빈도를 동적으로 조절하고, TCP와 유사하게 전송율을 증가시킴으로써 다른 프로토콜을 사용하는 응용프로그램과 네트워크 대역폭을 공정하고 효율적으로 이용할 수 있다는 제안을 하였다.

본 연구자는 위 실험을 토대로 앞으로 다음 다섯 가지에 대한 연구를 지속해 나갈 것이다 첫째, CBR 트래픽을 통하여 흐름제어에 대한 실험을 여러 가지 트래픽과 다양한 토폴로지를 적용하여 실험할 것이다. 둘째, 2절의 TCP와 유사한 흐름제어 알고리즘에 대한 시뮬레이션과 실제 네트워크에도 적용하여 실험할 것이다. 셋째, FTP, Telnet 등 다양한 TCP 백그라운드 트래픽을 적용할 것이며, 100개 이상의 노드를 가진 네트워크 토폴로지로 확장하여 실험할 것이다. 넷째, 종단간에 양방향(순방향/역방향)의 네트워크 상태가 다를 것이라는 판단 아래 네트워크의 이질도(heterogeneity)를 파악하는 실험을 할 것이다 다섯째, IVS뿐만 아니라 RLM에서도 공정성(fairness)을 향상시키기 위한 연구를 계속할 것이다.

참고문헌

[1] Steven McCanne, Van Jacobson, vic: a flexible framework for packet video, ACM Multimedia, November 1995

[2] T Turletti, "The INRIA Videoconferencing System (IVS)", conneXions - The Interoperability Report, Vol 8 , No 10, pp. 20-24, Oct 1994

[3] McCanne, S. and Jacobson, V., Receiver-driven Layered Multicast, Sigcomm, October 1996, pp.117-130

[4] Schulzrinne, Casner, Frederick, and Jacobson, V RTP: A transport protocol for real-time applications. March 21, 1995, [draft-ietf-avt-rtp-07.ps]

[5] McCanne, S., and Floyd, S. THe LBNL Network Simulator. Lawrence Berkly Laboratory Software on-line(http://www-nrg.ee.lbl.gov/ns/)