효율적인 스트리밍 서비스를 위한 가용대역폭 측정 기법에 관한 연구

이희상[O] 이선헌, 최웅철, 이승형, 정광수
광운대학교 전자공학부
{sanglee[O], sunlee}@adams.kw.ac.kr , {wchoi, shrhee, kchung}@daisy.kw.ac.kr

A Study on the Available Bandwidth Measurement Scheme for Efficient Streaming Services

Heesang Lee[O] Sunhun Lee, Kwangsue Chung
School of Electronics Engineering, Kwangwoon University

요 약

인터넷에서의 효율적인 스트리밍 서비스(Streaming Service)를 위해서는 안정된 전송률의 보장이 중요하며, 네트워크의 대역폭이 제한적이기 때문에 네트워크를 공유하는 경쟁 트래픽의 형평성에 대한 고려도 필요하다. 이러한 필요성에 따라 멀티미디어 데이터 전송의 주요 프로토콜인 UDP(User Datagram Protocol)에 혼잡제어 메커니즘 적용에 대한 연구가 활발히 진행되고 있으며, 대표적으로 가용대역폭 측정을 통한 혼잡제어 기법이 있다. 이 기법은 혼잡제어를 하기 위해서 네트워크의 상태에 따라 가변하는 가용대역폭(Available Bandwidth)을 측정하고 이것을 기반으로 전송률을 조절하는 방식을 말한다. 본 논문에서는 네트워크 상태에 따라 혼잡제어를 하기 위해서 가용대역폭을 보다 빠르고 정확하게 측정하고, 이것을 기반으로 스트리밍 서비스에 맞게 전송률을 조절하는 방법을 제안하였다. 실험을 통해 본 논문에서 제안한 방법이 기존의 가용대역폭을 측정하는 스트리밍 프로토콜 보다 성능이 개선이 되었음을 확인할 수 있었다.

1. 서 론

네트워크를 통한 멀티미디어 데이터의 전송은 VOD(Video On Demand), 화상회의(Video Conferencing), 인터넷 방송 등 멀티미디어 서비스의 다양한 분야에서 폭넓게 이용되고 있으며 그 이용량은 급격히 증가하는 추세이다. 이러한 네트워크를 통한 멀티미디어 데이터의 실시간 전송(Real-Time Transmission)을 위하여 필요한 기술이 스트리밍 서비스 기술이다. 현재의 스트리밍 서비스 기술의 연구 방향은 미디어의 품질을 좋게 하며 미디어의 끊김없이 부드러운 재생이 될 수 있게 하고 TCP, UDP, 그 외 다른 트래픽과 가용대역폭을 공평하고 효과적으로 공유하는 형평성을 고려하는 것으로 초점이 모아지고 있다.

가용대역폭은 스트리밍 서비스에 있어서 사용자의 QoS (Quality of Service)를 만족시킬 수 있는 중요한 파라미터이다. 만약 가용대역폭을 정확하고 빠르게 측정할 수 있다면 네트워크의 효율성을 증대시킬 수 있으며 경쟁하는 트래픽과 공정하게 가용대역폭을 공유할 수 있는 기반이 된다.

패킷 페어 메커니즘은 네트워크 경로에서 대역폭이 가장 작은 경로(Bottleneck Link Capacity)를 측정하는 것에서 시작되었다. 하지만 현재에는 병목구간 내에서 같은 경로를 공유하는 경쟁트래픽의 대역폭 점유에 따라 가변 하는 가용대역폭을 측정하는 연구가 더 많이 진행되어 가고 있다.

패킷 페어 메커니즘을 응용하여 가용대역폭을 측정하는 기존의 많은 연구들 중에 TCP-Westwood에서 제안된 대역폭 측정(Bandwidth Estimation)기법이 있다 [1]. 이 기법은 기존 TCP의 혼잡제어의 성능을 향상시키기 위해 가용대역폭의 측정을 기반으로 전송률 조절을 하기 위해서 제안된 기법이다. TCP-Westwood의 송신자는 수신자가 보낸 ACK 패킷이 들어오는 간격으로 가용대역폭을 반복적으로 측정한다. TCP-Westwood의 장점은 가용대역폭의 측정의 정확성과 New Reno와 같은 기존 TCP 프로토콜과 형평성을 보장한다는 것이다. 스트리밍 프로토콜인 SMCC(Streaming Media Congestion Control using Bandwidth Estimation)는 TCP-Westwood의 대역폭 측정방식을 수신자 측에서 그대로 사용하게 된다 [2]. 그러므로 SMCC는 TCP-Westwood의 대역폭 측정방식의 장점은 그대로 계승할 수 있었지만 필터방식을 사용하므로 초기가용대역폭을 측정하는 시간이 오래 걸리게 된다. 이러한 문제점을 보완하기 위해서 본 논문에서는 개선된 알고리즘을 제안하고자 한다. 스트리밍 서비스를 위해서 패킷 페어 메커니즘을 응용하는 알고리즘을 사용함으로써 스트리밍 서비스의 전송률의 변화를 안정화 시키고 네트워크의 효율성을 증가시켰다.

본 논문의 2장에서는 패킷 페어 방식을 응용하여 가용대역폭을 측정하는 기존의 관련 연구에 대해 소개하고 그 문제점을 기술하였으며 3장에서는 스트리밍 서비스를 위해 개선된 세부 알고리즘을 소개하였다. 4장에서는 실험을 통해 제안한 알고리즘의 성능을 평가하였으며 마지막으로5장에서는 결론을 맺었다.

2. 관련 연구

2.1 SMCC

SMCC는 수신자 측에서 네트워크 경로 중 병목구간 안에서 TCP-Westwood의 대역폭 측정 방식을 사용하여 가용대역

* 본 연구는 한국과학재단 특정기초연구 [R01-2005-0000-10934-0(2005)]의 지원에 의해 수행되었음.

폭을 측정한다. 기존 방식의 ACK 패킷 대신 데이터 패킷으로 대역폭을 측정하며, 패킷손실이 발생한 경우 NACK(Negative Acknowledgment)패킷으로 송신자에게 패킷 손실을 알리게 되고 이 상황을 대역폭 측정에 반영하도록 하고 있다.

$$b_k = d_k /(t_k - t_{k-1}) \quad (1)$$

$$BE_k = \alpha_k BE_{k-1} + (1-\alpha_k)\left(\frac{b_k + b_{k+1}}{2}\right) \quad (2)$$

$$\alpha_k = \frac{2\tau - \Delta t_k}{2\tau + \Delta t_k} \quad (3)$$

식 (1)은 수신자가 수신하는 패킷 페어의 첫 번째 패킷인 $k-1$ 번째와 두번째 패킷인 k 번째 사이의 도착하는 시간의 간격에 의해서 구해지는 가용대역폭 값을 의미한다. d_k 는 패킷 사이즈를 나타내며 t_k 는 k 번째 패킷이 수신되는 현재의 시간을 의미한다. 식 (2)는 식 (1)에서 구해진 가용대역폭 샘플값들을 필러링 기법을 사용하여 더해가면서 가용대역폭을 계산하는 것을 나타내고 있다. 식 (3)은 식 (2)에서 사용되는 필터링의 이득을 의미하게 된다.

식 (1)에서 구해지는 가용대역폭의 샘플값들이 네트워크의 큐잉딜레이(Queueing Delay)나 경쟁 트래픽에 의해 계속적으로 변하게 되며 이 값들은 식 (2), (3)에 의해 구하게 된다. 하지만 이런 필터링기법은 초기 가용대역폭까지 측정하는 시간이 오래 걸리게 된다. 또, 네트워크 상태가 변하게 되면 가용대역폭의 변화가 급격하게 이뤄지게 되고 필터링 이득에 따라 불필요하게 가용대역폭 측정값을 낮추게 되는 현상도 발생하게 된다. 이러한 문제는 지연에 민감한 스트리밍 서비스에서는 부적합하고 효율면에서 떨어지게 된다.

2.2 패킷 페어 트레인(TOPP : Train Of Packet Pair)

패킷 페어 트레인은 패킷 페어를 하나씩 따로 구분하지 않고 연속적으로 나열하여 전체 구간을 하나의 크고 긴 패킷 페어로 보는 관점이다 [3]. 패킷 페어 트레인은 모든 패킷 사이즈가 같다는 것을 가정하고 측정을 시작한다. 패킷 페어의 쌍이 연결되어 있으므로 네트워크의 가용대역폭의 변화가 그대로 반영이 되며 하나의 패킷 페어가 네트워크의 큐잉 딜레이에 의해 잘못 측정이 되어도 전체적인 값에는 큰 변화가 일어나지 않는다. 이러한 특성으로 가용대역폭의 변화를 안정하고 부드럽게 측정할 수 있다는 장점을 갖는다.

$$b(k) = \frac{(k-1)d}{\sum_{n=1}^{k-1}\delta^k} \quad (4)$$

식 (4)는 패킷 페어의 간격의 합으로 패킷 사이즈의 합을 나눠줌으로써 가용대역폭을 구하는 식이다. δ^k 는 패킷 페어에서 첫 번째 패킷 도착시간과 두번째 패킷의 도착시간의 간격을 의미하며 d 는 패킷 사이즈, k 는 패킷의 수를 의미한다. 식 (4)의 값이 평균을 의미하는 것이므로 실제 가용대역폭을 계산하는데 정확하지 못하다는 단점이 있다.

3. 효율적인 스트리밍 서비스를 위한 가용대역폭 측정기법

본 논문에서는 효율적인 스트리밍 서비스를 위해 가용대역폭의 변화를 안정화시켜서 전송률을 조절하고 네트워크의 대역폭을 효율적으로 사용하기 위해서 제안하였다. 그대로 사용함으로써 다른 가용대역폭의 측정 방식보다 측정의 정확성을 향상시킬 수 있었고 가변 하는 가용대역폭에 민감하게 반응을 할 수 있었다. 하지만 안정된 전송률이 필요한 스트리밍 서비스에서는 적합하지 않은 가용대역폭 측정 방식이기 때문에 이 문제점을 개선시키기 위해서 다음과 같은 방법을 사용한다.

3.1 가용대역폭의 초기값을 설정

SMCC는 가용대역폭을 측정함에 있어서 필터링 기법을 사용하므로 초기값을 ' 0' 으로 설정하게 된다. 이런 메커니즘은 안정된 전송률이 필요한 스트리밍 서비스에서는 부적합하며 초기값을 현재의 전송률로 설정하게 되면 가용대역폭을 측정하는 시간은 빠르게 단축시킬 수 있다. 스트리밍 서비스를 시작하게 되면 수신자측에서 SMCC와 패킷 페어 트래인을 동시에 사용하여 가용대역폭을 측정한다. 처음 두 개의 패킷 페어를 받게 되고 두 방식에서 산출되는 가용대역폭 값을 측정한 다음 두 방식 모두 가용대역폭이 증가추세를 보이고 있다면 현재의 가용대역폭 값이 실제 가용대역폭보다 작다는 것으로 인식하게 된다. 그 이후에 초기값을 패킷 페어 트레인이 산출한 값으로 설정하게 되고 그 값에서부터 다시 SMCC의 방식으로 가용대역폭을 측정하기 시작한다. 패킷 페어 트레인은 가용대역폭 측정을 평균으로 계산하기 때문에 실제 전송률 조절에서는 SMCC보다 상대적으로 정확성이 떨어진다. 하지만 이 방식이 가용대역폭에 근접한 값을 의미하기 때문에 SMCC가 스트리밍 서비스가 시작될 때 가용대역폭 측정의 문제점을 보완해 줄 수 있다.

3.2 가용대역폭의 변화에 따른 적응성 개선

SMCC와 패킷 페어 트레인의 값이 감소 추세로 전환되면 SMCC는 급격하게 값들이 떨어지는 반면 패킷 페어 트레인은 상대적으로 안정하고 완만하게 줄어들게 된다. 이것은 패킷 페어 트레인이 간섭에 강하기 때문이다. 이점을 이용하여 가용대역폭이 감소 추세를 보이면 패킷 페어 트레인을 사용함으로써 SMCC가 실제 가용대역폭보다 낮게 떨어지는 문제점을 해결할 수 있다. 그러므로 스트리밍 서비스의 전송률을 완만하게 줄일 수 있게 되며 전송률의 안정화를 가질 수가 있다.

그림1은 기존의 SMCC가 사용하는 가용대역폭 측정에 대한 문제점을 표시한 것이다. 초기 가용대역폭 측정이 약 9초 이상 소요되는 것을 확인할 수 있으며 가용대역폭이 크게 변하게 되는 경우 측정값이 불필요하게 가용대역폭보다 더 낮게 떨어지는 것을 확인할 수 있다.

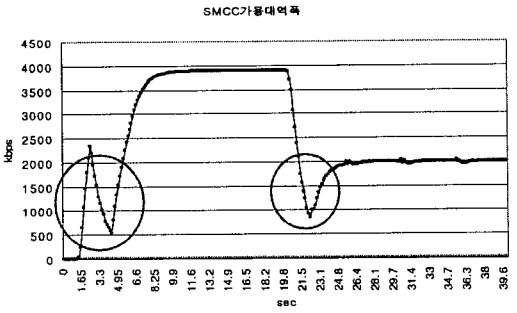

그림1. SMCC 가용대역폭 측정시 문제점

```
If (Current Rate ≤ Available Bandwidth)
                                    /* Increase Phase */
  Available Bandwidth Estimation = SMCC ();
Else                                /* Drease
Phase */
  Available Bandwidth Estimation = TOPP();
```

기존의 SMCC는 TCP-Westwood의 가용대역폭 측정 방식을

그림2. SMCC을 개선한 알고리즘

위 그림2는 본 논문에서 제안한 가용대역폭 측정의 알고리즘을 나타낸다. 두 방식에서 측정되는 가용대역폭의 변화에 따라 네트워크 상태를 인식하고 현재 전송률을 조절하게 된다. 즉, 가용대역폭이 현재 전송률보다 크다면 SMCC방식을 사용하게 되고 가용대역폭이 현재 전송률보다 작다면 TOPP방식을 사용하게 된다.

4. 실험 및 성능 평가

4.1 실험 환경

제안한 알고리즘의 성능 평가를 위해 LBNL(Lawrence Berkely National Laboratory)의 ns-2(network simulator)를 사용하여 실험을 수행하였다 [4]. 그림 3 은 제안한 알고리즘의 성능평가를 위한 실험환경을 보여준다. 가변하는 가용대역폭에 대한 성능을 측정하기 위해 전체 20 초의 시뮬레이션 구간에서 가용대역폭에 변화를 주었다. UDP1 은 CBR(Constant Bit Rate)로 5 초에 2Mbps 로 트래픽을 발생시키고 10 초에 멈추게 된다. 그 다음 UDP2 는 CBR 로 10 초에 1Mbps 로 트래픽을 발생시키고 15 초에 멈추게 된다. 제안된 알고리즘이 적용된 New SMCC 와 기존의 SMCC 는 송신자측에서 전송을 시작하게 되고 수신자 측에서 가용대역폭의 측정값을 산출하게 된다. New SMCC 가 가변하는 가용대역폭의 변화에 따라 가용대역폭을 측정하는 성능을 알기 위해 기존의 SMCC 과 비교를 하였다.

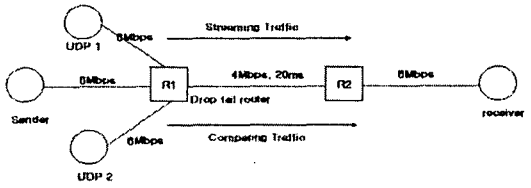

그림3. 실험 환경

4.2 개선된 SMCC의 성능 검증

그림4는SMCC를 개선한 알고리즘이 적용된 가용대역폭 측정 성능이 향상됨이 검증되는 실험 결과 이다. 실제 가용대역폭, 기존의 SMCC와 논문에서 제안된 알고리즘을 적용한 New SMCC를 비교하였다. 경쟁 트래픽이 없을 때 대역폭이 가장 작은 병목구간의 가용대역폭은4Mbps이다. 개선된 New SMCC가 추정된 가용대역폭을 초기값으로 가지고 실제 가용대역폭을 기존의 SMCC보다 더 빨리 측정하므로 네트워크를 효율성이 높아지게 되었다. 네트워크 상태가 변하여 가용대역폭이 현 가용대역폭 보다 작게 추정이 되면 New SMCC가 완만하게 변하면서 가용대역폭을 측정하는 반면 기존의 SMCC는 급격하게 가용대역폭 측정값이 떨어지는 것을 확인할 수 있었다. SMCC에서의 이 같은 변화는 스트리밍 서비스를 하고 있는 도중에 갑작스런 전송률 변화를 가지고 오게 되고 이런 변화는 사용자에게 스트리밍 서비스를 제공하는 것에 막대한 지장을 주게 된다. 시뮬레이션 후에 수신자측에서 측정한 전송률을 비교하면 New SMCC가 3.4Mbps이며 기존의 SMCC는 2.9Mbps가 나왔다. SMCC보다 New SMCC의 전송률이 향상되었으며 네트워크의 효율성이 더 높다는 것을 확인할 수 있었다.

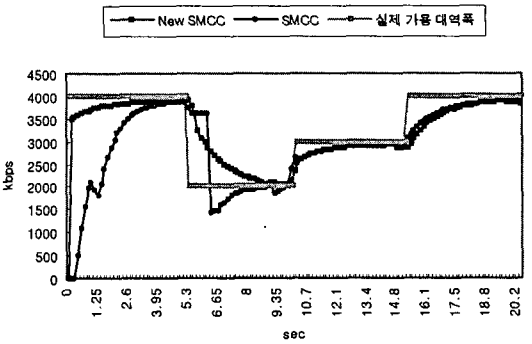

그림4. New SMCC, SMCC와 실제 가용대역폭의 비교

5. 결론

비디오 스트리밍에서 패킷 전달은 대부분 UDP 를 통해 이루어지고 있다. 이러한 UDP 기반의 트래픽은 기본적으로 혼잡제어 메커니즘이 없으며 현재 인터넷의 주요 트래픽인 TCP 와의 형평성을 보장하지 않는다는 문제점을 갖는다. 이러한 문제를 해결하기 위해서 가용대역폭을 측정하고 전송률을 조절하는 SMCC 가 제안되었으나 가용대역폭을 측정하기 시작하는 부분과 네트워크의 상태가 가변할 때 가용대역폭을 효율적으로 측정하지 못 한다는 한계가 있었다.

본 논문에서는 이러한 SMCC 의 문제점을 개선하기 위해서 패킷 페어 메커니즘을 응용한 패킷 페어 트레인으로 보완하였다. 실험을 통해 제안된 알고리즘을 적용한 New SMCC 의 성능을 검증하였으며 그 결과로 기존의 SMCC 보다 네트워크의 효율성이 증가하고 전송률이 안정화가 되었음을 확인하였다.

향후 과제로는 실험환경을 확장하여 경쟁 트래픽을 다양하게 주어지고 이런 환경에서 가용대역폭 측정을 검증과 함께 TCP 트래픽에 대한 형평성 문제를 개선하는 방법에 대해 연구가 수행되어야 할 것이다.

참고 문헌

[1] S. Mascolo, C.Casetti, M. Gerla, M. Sanadidi, and R. Wang. " TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links." *In Proceedings of Mobicom 2001*

[2] Nadeem Aboobaker, David Chanady, Mario Gerla, M.Y. Sanadidi, "Streming Media Congestion Control using Bandwidth Estimation", *the 5th IFIP/IEEE International Conference 2002*

[3] B. Melander, M. Bjorkman, and P. Gunningberg. "A New End-to-End Probing and Analysis Method for Estimation Bandwidth Bottlenecks", *In Global Internet Symposium, 2000*

[4] The network simulator ns-2, http://www.isi.edu/nanam/ns/