
실시간 비디오 스트림의 전송지연 축소를 위한 TCP 친화적 하이브리드 혼잡제어 기법

김형진* · 조정현* · 나인호*

A TCP-Friendly Congestion Control Scheme using Hybrid Approach for Reduction of Transmission Delay for Real-Time Video Stream

Hyoung-Jin Kim* · Jeong-Hyun Cho* · In-Ho Ra*

이 논문은 정보통신부 정보통신연구진흥원에서 지원하고 있는 정보통신기초기술연구지원사업의 연구결과입니다.

요 약

최근 인터넷의 발전으로 디지털 오디오 및 비디오와 같은 멀티미디어 스트림에 대한 요구가 증가하고 있다. 이러한 멀티미디어 스트리밍을 UDP로 전송할 경우 TCP와 같은 혼잡제어를 수행하지 않기 때문에 동일한 경로에 TCP 트래픽 공핍을 일으켜 혼잡붕괴 및 막대한 전송지연을 초래한다. 이러한 문제점으로 인하여 실시간 멀티미디어 스트림의 전송지연 축소와 혼잡제어를 위한 새로운 전송기법과 프로토콜에 대한 다각적인 연구가 수행되고 있다. TCP 친화적 혼잡제어 기법은 크게 일반적인 혼잡원도우 관리기능을 이용하는 윈도우 기반 혼잡제어와 TCP 모델링 방정식 등을 이용하여 전송율을 직접 조절하는 율 기반 혼잡제어로 나눌 수 있다. 본 논문은 윈도우 기반과 율 기반을 복합적으로 다룬 하이브리드형 TCP-friendly 혼잡제어 기법에서 전송율 개선을 위한 알고리즘을 제안하였으며, NS를 사용하여 제안된 TEAR의 성능을 실험하였다. 실험 결과를 통해 제안된 TEAR가 TCP보다 적은 율 변동과 공정성을 동시에 제공할 수 있음을 보였다.

ABSTRACT

Recently, the needs for multimedia stream such as digital audio and video in the Internet has increased tremendously. Unlike TCP, the UDP protocol, which has been used to transmit streaming traffic through the Internet, does not apply any congestion control mechanism to regulate the data flow through the shared network. And it leads to congestion collapse of the Internet and results in long-term transmission delay. To avoid any adverse effect on the current Internet functionality, a new protocol of modification or addition of some functionality to perform congestion control and to reduce huge transmission delay in transmitting of multimedia stream are in study. TCP-friendly congestion control mechanism is classified into two : one is window-based congestion control scheme using general window management functionalities, the other is rate-based congestion control scheme using TCP modeling equation. In this paper, we propose an algorithm for improving the transmitting rate on a hybrid TCP-friendly congestion control scheme combined with widow-based and rate-based congestion control for multimedia stream. And we also simulate the performance of improved TEAR implementation using NS. With the simulation results, we show that the improved TEAR can provide better fairness and lower rate fluctuations than TCP.

키워드

Congestion Control, TCP-Friendly, TFRC, TEAR

* 군산대학교 전자정보공학부

접수일자 : 2003. 12. 26

1. 서론

최근 정보통신기술과 인터넷의 발전으로 다양한 멀티미디어 서비스에 대한 수요가 증가하고 있다. 멀티미디어 스트림을 실시간에 전송하기 위해서는 높은 전송대역폭과 낮은 전송 지연을 필요로 한다. 이러한 특성 때문에 실시간 멀티미디어 스트리밍을 지원하기 위해서는 패킷 손실이나 지연, 대역폭과 같은 조건을 크게 고려하여 전송메커니즘을 설계하여야 한다.

본 논문은 비디오와 같은 대용량의 스트리밍 데이터를 실시간에 IP 프로토콜을 이용하여 전송하고자 할 때 TCP나 UDP와 같이 기존의 전송 프로토콜 특성에 의해 발생하는 트래픽 혼잡을 제어하기 위한 기법에 대한 연구이다. 비연결성 전송 프로토콜인 UDP는 에러제어와 흐름제어를 하지 않기 때문에 네트워크 노드 상에서 보다 빠른 스위칭이 요구되는 멀티미디어 응용에서는 UDP를 사용하는 경우가 많다. 그러나 멀티미디어 스트림을 전송하기 위해 UDP를 이용할 경우, UDP는 TCP와는 달리 혼잡제어를 수행하지 않기 때문에 전송경로에서 트래픽 궁핍을 일으킨다. 이러한 현상을 혼잡붕괴(congestion collapse)라 하며, 혼잡제어를 수행하는 TCP에 비해 상대적으로 UDP에 할당되는 채널 대역폭이 감소하기 때문에 발생하는 현상이다[1]. 이와 유사하게 실시간 멀티미디어 응용에 TCP 혼잡제어를 적용하면 멀티미디어 데이터의 서비스 품질이 크게 저하되는 문제점이 발생하는데, 이것은 송신측의 데이터 전송율이 급격히 변화될 경우에 전송 지연시간이 급속히 증가 또는 감소됨으로서 발생하는 현상이다. 이러한 문제점을 해결하기 위해 실시간 멀티미디어 데이터의 전송에 적합한 TCP 친화적 혼잡제어 기법에 대한 다각적인 연구가 진행되고 있다.

TCP 친화적 혼잡제어 기법은 혼잡윈도우 크기에 대한 관리기능을 이용하여 혼잡제어를 수행하는 윈도우 기반 혼잡제어 기법과 TCP 모델링 방정식 등을 이용하여 전송율을 동적으로 조절하는 율 기반 혼잡제어 기법으로 크게 나눌 수 있다.

본 논문에서는 윈도우 기반과 율 기반을 복합적으로 다룬 하이브리드형 TCP 친화적 혼잡제어

기법을 제안하였다. 본 논문의 구성은 다음과 같다. 먼저 II장에서는 TCP 친화적 혼잡제어 기법의 특징과 방법에 대해 기술하고, III장에서는 본 논문에서 제안한 하이브리드형 TCP 친화적 혼잡제어 기법에 대해 기술한다. IV장에서는 성능 평가 시뮬레이션 방법과 결과에 대해 설명한다. 마지막으로 V장에서는 결론 및 향후 연구방향에 대해 기술한다.

II. TCP 친화적 혼잡제어 기법

2.1 윈도우 기반 혼잡제어 기법

윈도우 기반 혼잡제어 기법은 혼잡윈도우 관리를 통해 전송율을 조절하는 기법이다. 송신측은 혼잡윈도우의 최대크기로 패킷을 전송할 수 있으며, 수신측은 수신한 패킷마다 Ack를 보냄으로써 Ack가 송신측의 전송율을 조절하게 한다. 즉 송신측은 Ack 간격에 의해 패킷을 전송한다. 이와 같은 윈도우 기반 혼잡윈도우의 특성을 'Ack-clocking'라 한다. 이러한 윈도우 기반 혼잡제어 기법에는 AIMD(Additive Increase Multiplicative Decrease), GAIMD(General Additive Increase Multiplicative Decrease), Binomial 등이 있다.

2.1.1 AIMD 혼잡제어 기법

송신측은 가용대역폭을 이용하기 위해 혼잡윈도우 크기를 선형적으로 증가시키고, 혼잡 지시자인 패킷 손실이 발생할 때는 혼잡윈도우 크기를 절반으로 감소시킨다. AIMD 알고리즘은 식 (1)과 같다.

$$\begin{aligned} I: w_{t+R} &\leftarrow w_t + \alpha & ; \alpha > 0 \\ D: w_{t+\delta t} &\leftarrow (1 - \beta)w & ; 0 < \beta < 1 \end{aligned} \quad (1)$$

여기서 I 는 RTT 동안 혼잡윈도우 크기의 증가를 나타내고, D 는 혼잡 윈도우 크기의 감소를 나타낸다. 일반적으로 TCP에서는 $\alpha = 1$, $\beta = 1/2$ 로 설정한다. 이러한 AIMD 기법은 효율적으로 가용대역폭을 사용하기 때문에 대부분의 응용에 사용

되고 있다[2].

2.1.2 GAIMD 혼잡제어기법

GAIMD는 AIMD 기법을 확장하여 일반화한 것이다. GAIMD가 TCP 친화적이기 위해서는 α 와 β 가 식 (2)와 같은 관계를 가져야 한다.

$$\alpha = \frac{4(1-\beta^2)}{3} \quad (2)$$

GAIMD는 멀티미디어 응용에 적합한 평활화한 전송을 위해 $\beta = 7/8$ 로 설정한다. 따라서 식 (2)에 의해 α 는 0.31의 값을 가지게 된다[3].

2.1.3 Binomial 혼잡제어기법

Binomial 혼잡제어기법은 선형 혼잡제어를 수행하는 AIMD와 달리 비선형 혼잡제어를 수행하여 가용대역폭을 보다 빨리 사용할 수 있는 기법이다. Binomial 알고리즘은 식 (3)과 같다.

$$\begin{aligned} I: w_{t+R} &\leftarrow w_t + \alpha/w_t^k & ; \alpha > 0 \\ D: w_{t+\delta t} &\leftarrow w - \beta w_t^l & ; 0 < \beta < 1 \end{aligned} \quad (3)$$

여기서 Binomial 기법이 TCP 친화적이 위해서는 적절한 α , β 값과 $k+l=1, l \leq 1$ 의 조건을 만족하면 된다. 이 기법은 AIMD 기법에 비해 울 변화는 적지만 멀티미디어 응용에 적용하기에는 어렵다[4].

2.2 울 기반 혼잡제어기법

울 기반 혼잡제어기법은 혼잡을 지시하는 네트워크 피드백 기법을 이용하여 동적으로 전송율을 조절하는 기법으로 TCP 모델링 방정식을 이용하는 TFRC(TCP Friendly Rate Control) 등이 있다.

2.2.1 TCP 모델링 방정식

TCP 처리율은 왕복시간 t_{rtt} , 재전송타임아웃 t_{rto} , 패킷크기 s , 패킷손실율 p 를 근거로 하여 식 (4)과 같이 구할 수 있다.

$$T = \frac{s}{t_{rtt} \sqrt{\frac{2p}{3}} + t_{rto} \min(1.3 \sqrt{\frac{3p}{8}}, p(1+32p^2))} \quad (4)$$

2.2.2 TFRC

TFRC는 TCP 모델링 방정식을 이용하여 전송율을 조절하는 방법이다. 울 기반 혼잡제어를 수행하기 위해 식 (4)를 사용하기 위해서는 패킷 손실율을 정확히 측정하는 것이 매우 중요하다. 이에 따라 TFRC는 연속적인 패킷손실 간의 패킷수를 측정하는 평균손실간격(Average Loss Interval)과 EWMA기법을 이용하여 패킷 손실율을 측정한다. TFRC 수신측은 매 RTT마다 TCP 방정식에 필요한 파라미터를 업데이트하여 송신측에 보내고 송신측은 업데이트된 정보를 이용하여 계산한 전송율로 패킷을 전송한다. TFRC는 지연 기반 혼잡제어(delay-based congestion control)을 사용하여 프로토콜의 성능을 향상시키고 있으며, 대역폭을 놓고 서로 경쟁하는 트래픽에 대해 충분한 반응을 보임으로써 상대적으로 안정된 전송율을 유지할 수 있다[5][6].

III. 하이브리드 TCP 친화적 혼잡제어기법

TEAR(TCP emulation at receivers)는 윈도우 기반과 울 기반을 결합한 하이브리드 TCP-friendly 혼잡 제어 기법 중 대표적인 프로토콜[6][7]이며, 논문에서는 이것을 기반으로 TEAR 전송율 개선을 위한 알고리즘을 제안하였다.

3.1 TEAR

TEAR 수신측은 혼잡윈도우 크기를 관리하고 이를 토대로 처리율을 계산한다. 또한, 계산된 처리율을 피드백을 통해 송신측에 전송하고 송신측은 이를 토대로 전송율을 결정한다.

3.1.1 TEAR 상태 전이

TEAR의 상태전이는 그림 1과 같은 형태로 구성되어 있다.

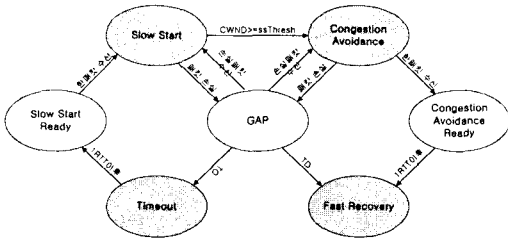


그림 1. TEAR 상태전이
Fig. 1 State transition of TEAR

3.1.2 혼잡윈도우 증가 알고리즘

Slow-start, Congestion-avoidance 상태에서 하나의 패킷을 수신하는 경우 혼잡윈도우 크기의 증가는 그림 2와 같은 동작에 의해 수행된다.

```

Void TEARSinkAgent::IncreaseWindow()
{
    switch(state){
        case SLOW_START:
            cwnd=cwnd+1;
            if(ssThresh<=cwnd)
                cwnd=ssThresh;
        case CONGESTION_AVOIDANCE:
            cwnd+=1/lastcwnd;
    }
}
    
```

그림 2. 혼잡윈도우 증가 알고리즘
Fig. 2 Increase window algorithm

여기서 lastcwnd는 매 round가 시작할 때마다 다음 round의 혼잡윈도우 증가를 계산하기 위해 현재 혼잡윈도우로 업데이트 된다.

3.1.3 혼잡 윈도우 감소 알고리즘

패킷 손실이 발생하면 Slow-start, Congestion-avoidance 상태는 Gap상태로 전이된다. Gap 상태에서는 혼잡윈도우크기가 변하지 않는다. Gap 상태는 수신측이 패킷 손실이 타임아웃에 인한 것인지, 세 개의 중복 Ack에 인한 것인지를 결정하는 중계상태이다. 그림 3은 혼잡윈도우크기 감소 알고리즘을 나타내고 있다.

① Fast recovery

손실된 패킷을 l 이라고 가정하자. 수신측이 패킷 손실 이후 $l+lastcwnd$ 패킷을 수신하기 전에 적어도 2개의 패킷을 수신하게 되면 Gap상태에서

Fast-recovery 상태로 전이하게 된다. 이 상태에서 수신측은 혼잡 윈도우와 lastcwnd를 절반으로 감소한다.

② Timeout

수신측이 손실된 패킷 l 을 인지한 후 $T_{timeout}$ 동안 Gap상태에서 Fast recovery상태로 전이되지 못하면 Timeout 상태로 전이하게 된다. 이 상태에서 수신측은 혼잡윈도우와 lastcwnd를 1로, $T_{timeout}$ 는 두 배로 설정한다.

```

Void TEARSinkAgent::DecreaseWindows()
{
    switch(state){
        case FastRecovery:
            cwnd=cwnd/2;
            ssThresh=cwnd;
        case Timeout:
            cwnd=1;
            ssThresh=cwnd;
    }
}
    
```

그림 3. 혼잡윈도우 감소 알고리즘
Fig. 3 Decrease window algorithm

3.2 TEAR 전송율 개선을 위한 알고리즘

TEAR 수신측은 TCP 톱니형태의 전송율 변화를 피하기 위해 매 RTT 대신 epoch단위로 처리율을 계산한다. Epoch는 연속적인 율 감소 사건간의 시간으로 정의된다.

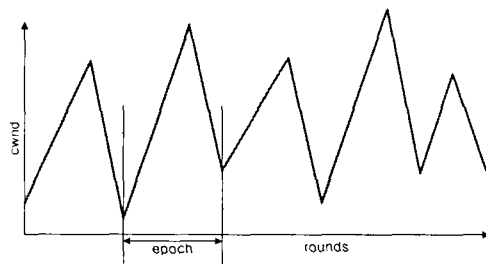


그림 4. 구간
Fig. 4 Epoch

그림 4는 epoch를 나타내고 있다. 노이즈에 의한 불필요한 율 변화를 방지하기 위해 지난 w 개 epoch의 처리율에 대한 가중치 평균을 사용하여 평활화한 처리율을 계산하고, 계산된 처리율은 피드백을 통해 송신측에 보낸다. 송신측은 이를 토

대로 전송율을 결정하기 앞서 인터넷과 같은 다양한 네트워크 상태에 적응하도록 지연기반 혼잡제어 기법을 도입하여 전송율 T 를 식(5)와 같이 조절하였다.

$$T = T_{estimated} \times \frac{\sqrt{SRTT}}{RTT_{current}} \quad (5)$$

여기서 $T_{estimate}$ 는 수신측에서 계산된 처리율이다. 제안된 기법은 현재의 네트워크 상태를 예측하는 가장 좋은 방법인 네트워크 지연을 토대로 혼잡회피를 수행하면서 다양한 네트워크 환경에 적응 가능하도록 한 것이다. 즉, 현재의 네트워크 지연이 과거의 지연보다 크면 현재 네트워크 상태가 혼잡하다는 것을 인식할 수 있도록 전송율을 감소하게 하였고, 그 반대의 경우 전송율을 증가하도록 하여 상대적으로 안정된 TEAR 전송율을 유지하도록 하였다. 그림 5는 전체적인 동작 메커니즘이다.

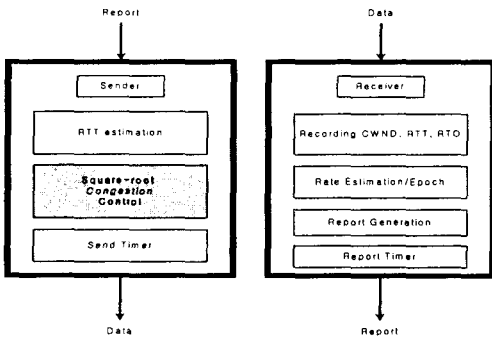


그림 5. 전체적인 동작 메커니즘
Fig. 5 Mechanism

IV. 시뮬레이션 결과

그림 6과 같은 시뮬레이션 토폴로지 환경에서 NS-2[8]를 사용하였고, 제안한 TEAR와 TCP의 총 전송율과 순간 전송율을 측정하여 공정성과 평활성을 측정하여 비교하였다.

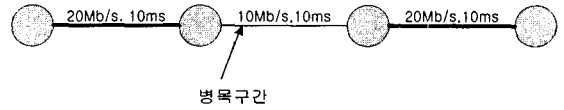


그림 6. 시뮬레이션 토폴로지
Fig. 6 Simulation topology

그림 7, 그림 8에 보여진 것과 같이 본 논문에서 제안한 지연기반 혼잡제어 알고리즘을 TEAR에 적용하였을 경우 TCP와 비교하여 대역폭 경쟁에 따른 공정성을 높이면서 상대적으로 평활화된 전송이 이루어져 혼잡을 피할 수 있음을 알 수 있으며, 제안된 기법을 통해 네트워크 상태의 변화에 잘 반응함을 알 수 있다.

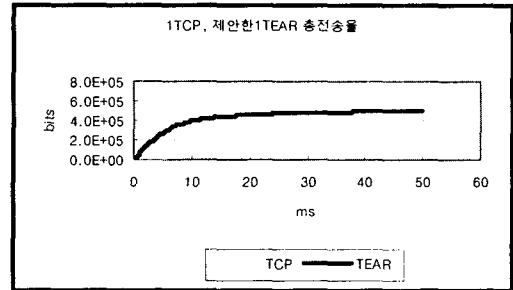


그림 7. 1TCP, 제안한 1TEAR 총전송율
Fig. 7 1TCP, proposed 1TEAR aggregate throughput

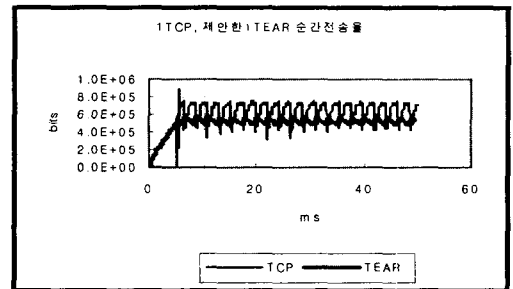


그림 8. 1TCP, 제안한 1TEAR 순간전송율
Fig. 8 1TCP, proposed 1TEAR instantaneous throughput

V. 결론 및 향후 연구방향

본 논문은 멀티미디어 스트리밍을 위한 TCP 친화적 혼잡제어기법을 다루었다. TCP와 경쟁시

대역폭을 공정하게 나누어 사용할 수 있게 TCP 동작을 적용하였고, 갑작스런 유휴 감소에 의한 멀티미디어 응용의 사용자 인지 품질이 저하되는 것을 피하기 위해 평활화한 전송을 할 수 있도록 epoch단위로 처리율을 계산하였다. 또한 네트워크 지연으로 인한 혼잡을 피하기 위한 방법을 도입하여 상대적으로 안정된 TEAR 전송율을 유지하도록 하였다. 시뮬레이션을 통해 제안한 TEAR의 성능을 측정된 결과로부터 평활성과 공정성을 유지함을 알 수 있었다.

향후에는 제안된 알고리즘을 멀티캐스트에 서도 동작할 수 있도록 확장하여, 그 성능을 분석하고 평가하고자 한다.

참고 문헌

- [1] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the internet", IEEE/ACM Transactions on Networking, Aug 1999.
- [2] S. Jin, L. Guo, I. Matta, A. Bestavros, "A Spectrum of TCP-friendly Window-based Congestion Control Algorithms", July 2002.
- [3] Y. Richard Yang and Simon S. Lam. "General AIMD congestion control", In Proceedings of ICNP, November 2000.
- [4] D. Bansal and H. Balakrishnam, "Binomial congestion control algorithms", In Proceedings of IEEE INFOCOM, April 2001.
- [5] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications", Technical Report, ACIRI, Feb 2000.
- [6] J. Widmer, R. Denda, M. Mauve, "A Survey on TCP-Friendly Congestion Control(extended version)", Feb 2001.
- [7] I. Rhee, V. Ozdemir, and Y. YI, "TEAR : TCP Emulation At Receivers - flow control for multimedia streaming", Technical Report, NCSU, April 2000.
- [8] Network Simulator - ns-2
[http:// www.isi.edu/nsnam/ns](http://www.isi.edu/nsnam/ns)

저자 소개



김형진(Hyung-Jin Kim)

1997년 호원대학교 전자계산학과 졸업
 1999년 군산대학교 정보통신공학과 석사

2000년~ 현재 군산대학교 정보통신공학과 박사과정
 ※ 관심분야 : 멀티미디어DBMS, 멀티미디어 시스템



조정현(Jeong-Hyun Cho)

2000년 조선대학교 공과대학 고분자공학과 졸업
 2002년~ 현재 군산대학교 공과대학 전자정보공학부 석사과정.

※ 관심분야 : Qos, Congestion Control



나인호(In-Ho Ra)

1988년 울산대학교 전자계산학과 졸업
 1991년 중앙대학교 전자계산학 석사
 1995년 중앙대학교 전자계산학 박사

1995. 9 ~ 현재 군산대학교 전자정보공학부 부교수
 ※ 관심분야 : 멀티미디어 통신시스템, 초고속 통신망