빠른 손실 감지를 통한 TCP NewReno의 Fast Recovery 개선 알고리듬

정회원 김 동 민*, 김 범 준**, 김 석 규*, 이 재 용*

Enhancements to the fast recovery algorithm of TCP NewReno using rapid loss detection

Dongmin Kim*, Beomjoon Kim**, Seoggyu Kim*, Jaiyong Lee* ***Regular Members***

요 약

국내 무선 네트워크 환경은 사용자의 서비스 요구 수용과 시장 성장으로 인해 빠르게 변화하고 있다. 이에 따라 무선 구간에서 TCP(transmission control protocol)를 이용한 신뢰성 있는 데이터 전송도 늘어날 전망이다. TCP는 유선 네트워크에서 사용함을 가정으로 만들어졌기 때문에 무선에서 발생할 수 있는 비 혼잡 손실에 의해 많은 성능 저하를 겪을 수 있다. 특히 RTO(retransmission timeout)은 TCP의 성능에 많은 영향을 미친다. 본 논문에서는 송신단에서 fast recovery과정 중에 발생한 패킷 손실을 빠르게 감지하여 RTO없이 복구함으로써 성능 저하를 줄일 수 있는 DAC+(Duplicate Acknowledgement Counting)와 EFR(Extended Fast Recovery)을 제안한다. 제안 알고리듬을 TCP NewReno와 비교했을 때 정상 상태에서 fast recovery 확률이 높고, 이에 따른 RTO 감소로 인해 response time이 줄어드는 것을 확인할 수 있다.

키워드: TCP, fast recovery, loss detection, wireless networks

ABSTRACT

Domestic wireless network environment is changing rapidly while adapting to meet service requirements of users and growth of market. As a result, reliable data transmission using TCP is also expected to increase. Since TCP assumes that it is used in wired network, TCP suffers significant performance degradation over wireless network where packet losses are not always result of network congestion. Especially RTO imposes a great performance degradation of TCP. In this paper, we propose DAC+ and EFR in order to prevent performance degradation by quickly detecting and recovering loss without RTO during fast recovery. Compared with TCP NewReno, proposed scheme shows improvements in steady-state in terms of higher fast recovery probability and reduced response time.

Ⅰ. 서 론

최근 들어 모바일 뱅킹, 모바일 방송, 무선 인터넷, 공중 무선 LAN 서비스 등 무선에서 사용자에게 편의성을 증대하는 새로운 서비스들이 개발되고 있고, 사용자들의 이용도가 증가함으로써 무선 네트워크에서 데이터 트래픽이 많이 증가하고 있다. 따라서 TCP(transmission control protocol) 트래픽 또한 증가할 것으로 예상할 수 있다. 무선 링크는 유선과 달리 multipath fading, shadowing, mobility

* 연세대학교 전기전자공학과 고성능 멀티미디어 네트워크 연구실({danny, sgkion, jyl}@nasla.yonsei.ac.kr),
** LG전자 이동통신기술연구소 표준화그룹(beom@lge.com), 논문번호 : 040141-0408, 접수일자 : 2004년 3월
※본 연구는 퀄컴-연세 합동 CDMA 연구 센터 (Qualcomm Yonsei CDMA Joint Research Center) 지원으로 수행되었음.(This work was supported by Qualcomm Incorporated through Qualcomm Yonsei CDMA Joint Research Center.)

등으로 인해 높은 패킷[1] 손실율과 burst한 손실이 발생하는 특성을 가지고 있기 때문에 현재 인터넷에서 유선링크를 가정하고 설계한 TCP를 수정 없이 사용할 경우에는 많은 성능 저하를 감수해야 한다[1],[2],[3]. 이는 TCP가 네트워크에서 발생하는 모든 패킷 손실이 혼잡(congestion)에 의한 것이고 이를 해결하기 위해서 혼잡 제어를 수행하게 되는데 무선에서의 비혼잡 손실은 혼잡 제어 대상이 아님에도 불구하고 혼잡 제어를 수행하기 때문에 나타나는 현상이다. 특히 무선에서는 비혼잡 손실로 인해 불필요하게 윈도우(congestion window) 크기를 감소시키고 또한 burst한 특성으로 인해 한 윈도우에서 다수개의 패킷 손실이 발생한 경우에는 RTO(retransmission timeout)을 발생시켜 전송율을 떨어뜨린다. RTO의 경우는 재전송 타이머가 만료되기 이전까지 패킷을 더 이상 전송하지 못하고 slow start 모드로 전송을 시작하고, 연속된 RTO가 발생하면 타이머 값을 두 배로 계속 증가시키기 때문에 링크의 이용도를 낮게 할 수 있다. 한 연구에서 TCP의 재전송 중에 56%가 RTO에 의해 발생하고, 나머지 44%가 fast retransmit에 의해 발생한다는 결과가 있다[4]. 때문에 패킷 손실이 RTO없이 복구될 수 있다면 TCP의 성능을 개선할 수 있다.

RTO 발생원인은 윈도우의 크기[2]가 작아서 fast retransmit을 위한 충분한 중복 승인 패킷(duplicate acknowledgement)을 수신하지 못하는 경우, 한 윈도우에서 다수개의 패킷 손실이 발생하는 경우, 재전송한 패킷이 다시 손실되는 경우로 나눌 수 있다. 작은 윈도우로 인한 RTO는 Limited Transmit을 이용하여 막을 수 있다[5]. 다수개의 패킷 손실이 발생하는 경우는 TCP Reno를 대신해 TCP NewReno를 이용하여 RTO없이 복구할 수 있다. 하지만 TCP NewReno도 fast recovery과정 중에 패킷 손실이 발생하는 경우 RTO가 발생하는 문제를 가지고 있다. TCP NewReno는 fast recovery 과정 중에 손실된 패킷을 재전송하거나, 중복 승인 패킷의 수신으로 윈도우를 증가시킬 때 윈도우 범위에 새로 포함된 *new packet*[3]을 전송할 수 있다. 재전송 패킷이 다시 손실될 때 RTO가 발생하고, 이 경우가 전체 타임아웃의 약 4%를 차지한다[6],[7]. 또한 *new packet* 손실 시에도 대부분의 경우 줄어든 윈도우 크기로 인해 충분한 중복 승인 패킷을 수신할 수 없기 때문에 fast retransmit에 의한 복구가 불가능하고 RTO이 발생한다.

본 논문에서는 TCP NewReno의 fast recovery 과정 중에 발생할 수 있는 RTO의 두 가지 원인을 RTO 없이 복구할 수 있는 방법을 제안한다. 재전송 손실 복구를 위해 수신 예상 중복 승인 패킷 수를 이용한 Duplicate Acknowledgement Counting(DAC)은 이미 제안되었다[8]. 하지만 이 논문에서는 RFC 2581에서 제시하고 있는 '재전송 패킷 손실은 이전 손실과는 다른 혼잡에 의해 손실이다'는 점이 고려되지 않았기 때문에 본 논문에서 DAC$^+$로 이점을 수정하였다. *New packet* 손실은 Extended Fast Recovery(EFR)를 이용하여 복구한다[9]. EFR은 현재의 fast recovery 안에 *new packet* 손실을 복구할 수 있도록 fast recovery를 연장하고 재전송을 통해 복구한다. EFR도 RFC에 따라 '연속적인 윈도우에서의 손실은 서로 다른 혼잡에 의한 것'으로 간주하고 동작하도록 하였다. 두 알고리듬 모두 송신단의 TCP만 간단히 수정하여 동작할 수 있고, 현재 TCP 규격에 맞도록 구현하였다. 따라서 본 논문에서 제안된 알고리듬은 향후 무선 인터넷 또는 공중 무선 LAN 등에서 사용되어질 수 있다.

Ⅱ. 제안 알고리듬

TCP 혼잡 제어는 slow start, congestion avoidance, fast retransmit, fast recovery로 구성된다. TCP NewReno는 TCP Reno를 수정하여 다수개의 패킷 손실이 발생하는 경우에도 부분 승인 패킷(partial acknowledgement)을 이용하여 fast recovery로 복구할 수 있도록 개선된 알고리듬이다. 본 논문의 제안 알고리듬은 개선된 TCP NewReno에 적용 가능하다.

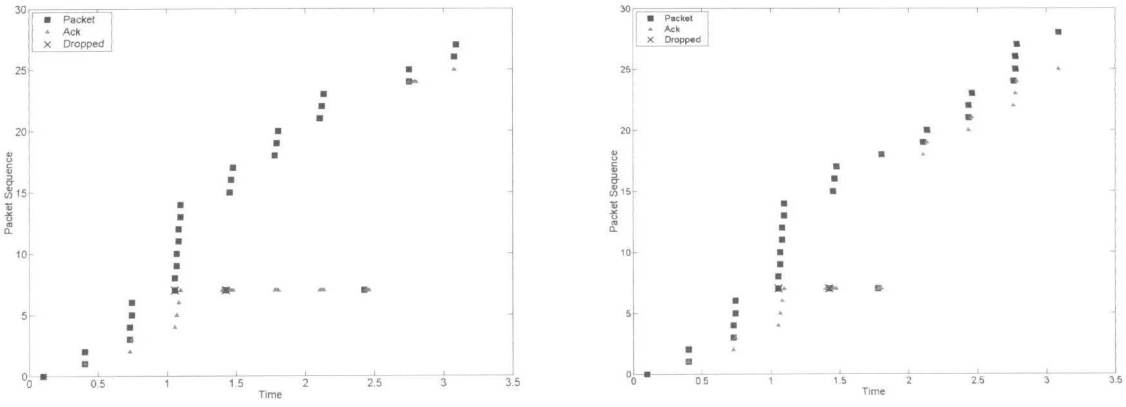
1. DAC$^+$(Duplicate Acknowledgement Counting)

TCP는 패킷 손실이 발생하면 손실된 패킷에 대한 중복 승인 패킷 3개를 수신하고 fast retransmit과 fast recovery 수행을 통해 복구한다. DAC$^+$는 이러한 손실 복구 과정 중에 간단한 계산을 통해 재전송된 패킷이 손실되었을 때 재전송 타임아웃

1) 본 논문에서는 TCP 전송계층에서 사용되는 세그먼트 대신 패킷으로 표시하였다.

2) 윈도우의 크기는 송신단의 혼잡윈도우와 수신단의 광고 윈도우(advertised window)중에 작은 값으로 설정한다.

3) 본 논문에서는 fast recovery 과정 중에 usable window의 증가로 인해 *fast retransmit* 이후에 전송한 새로운 패킷을 *new packet*으로 정의한다.



(a) TCP NewReno (b) DAC$^+$

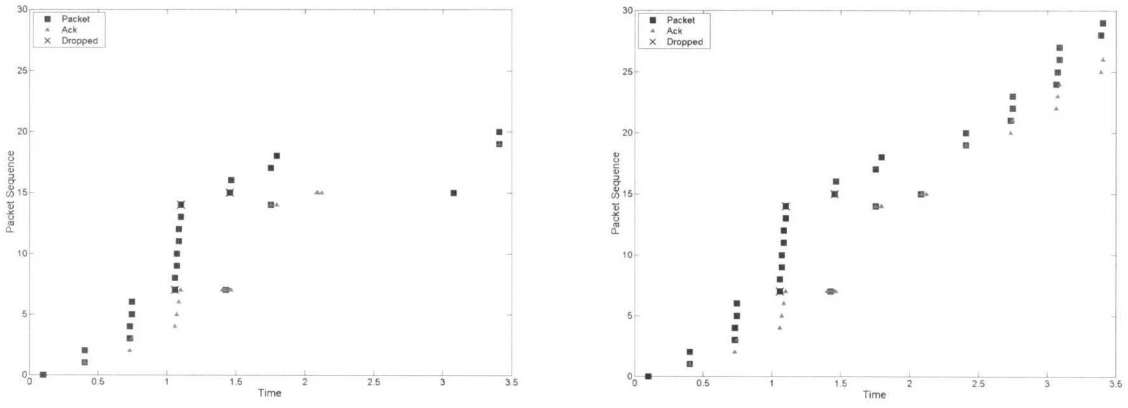
그림 1. 하나의 재전송 패킷 손실이 발생한 경우의 손실 복구 과정 비교

없이 손실된 패킷을 복구할 수 있게 한다. 이를 위해 DAC$^+$는 재전송하기 바로 전의 윈도우의 크기(s_{cwnd})와 재전송된 패킷이 다시 손실되지 않았을 때 수신할 수 있는 예상 중복 승인 패킷의 수를 기억한다. Fast recovery 과정 중에 송신단은 손실된 패킷에 대한 중복 승인 패킷의 개수를 세고 이때 예상 중복 승인 패킷의 수보다 많은 중복 승인 패킷을 수신하면 송신단은 재전송된 패킷이 다시 손실되었다고 판단하고, 재전송 타임아웃을 기다리지 않고 손실된 패킷을 다시 재전송한다. 하나의 윈도우 안에서 여러 개의 패킷 손실이 발생할 수 있으므로 i번째 손실 패킷의 예상 중복 승인 패킷의 개수는 변수 DAC_i에 저장한다. 송신단은 한 윈도우에서 다수개의 패킷이 손실되었다 하더라도 수신되는 중복 승인 패킷에 의해서만 손실 여부를 판단하기 때문에 몇 개가 손실되었는지를 알 수 없다. 다만 최소한 하나 이상의 패킷이 손실되었다는 것만 알 수 있으므로 첫 번째 fast retransmit된 패킷에 대한 DAC_1은 언제나 s_{cwnd} -1이 된다. 두 번째 또는 세 번째 재전송한 패킷 손실도 예상 중복 승인 패킷수를 계산하여 복구할 수 있다.

그림 1은 재전송된 패킷이 손실되었을 때 DAC$^+$와 TCP NewReno를 비교한 그림이다. 송신단은 1.1초에 윈도우의 크기가 8이 되고, 패킷 7-14번까지 전송하지만 패킷 7이 손실되었다[4]. 1.42초에 손실된 패킷에 대한 3개의 중복 승인 패킷을 수신하게 되면 송신단은 fast retransmit을 실행하고 *usable window*를 7(=[8/2]+3), DAC_1도 7(=8-1)로 설정한다[6]. Fast retransmit된 패킷 7은 다시 손실된다. 이후에 패킷 7에 대한 4개의 중복 승인 패킷을 수신하고 *usable window*는 11로 증가하게 되어 새로운 패킷 15, 16, 17번을 전송할 수 있게 된다. 패킷 7번이 수신단에 제대로 도착하지 않았기 때문에 새로 전송한 15-17에 의해 3개의 중복 승인 패킷이 발생하게 되고 이것은 *usable window*를 다시 3만큼 증가시켜 18-20번까지의 패킷이 새로 전송된다. 이러한 과정은 약 2.42초에 재손실된 패킷 7에 대한 재전송 타임아웃이 발생하기 전까지 계속된다. 이것은 재전송된 패킷 7이 손실되어 정상 승인 패킷을 송신단이 수신할 수 없어 fast recovery를 종료할 수 없기 때문이다. 재전송 타임아웃 이후에 송신단은 윈도우를 1로 설정하고 패킷 7을 다시 재전송하고 slow start를 시작하게 된다.

DAC$^+$를 사용하게 되는 경우는 송신단이 패킷 7을 fast retransmit할 때 DAC_1를 7(=8-1)로 설정한다. 패킷 15에 의해 발생하는 8번째 중복 승인 패킷을 수신하면 이 값은 DAC_1보다 크기 때문에 송신단은 재전송 패킷이 손실되었다고 판단하고 패킷 7을 재전송하게 된다. 이때 DAC는 윈도우의 크기를 4(=[8/2])로 유지하지만 DAC$^+$는 현재 윈도우의 크기와 *ssthresh*(slow start threshold)를 2(=[4/2])로 한 번 더 줄인다. 이것은 재전송된 패킷 손실은 처음 손실된 패킷과는 서로 다른 혼잡에 의해 손실된 것으로 간주되어야 하고, 이에 따라 혼잡 제어를 수행해야 하기 때문이다[10]. 또한 DAC$^+$를 사용하지 않는 TCP 연결에서 재전송 패킷 손실을 RTO를 통해 복구하면서 혼잡 제어를 수행하게 되는데, 이와

4) 각각의 패킷을 구별하기 위해 0번부터 시작하는 패킷 시퀀스 번호를 사용하였고, 0번 패킷에 대한 승인 패킷은 1번으로 나타내었다.



(a) TCP NewReno (b) EFR

그림 2. Fast recovery 과정중 *new packet* 손실시 복구 과정 비교

동일하게 DAC$^+$에서도 손실 복구시 윈도우의크기를 줄이는 혼잡 제어를 수행해야지만 공평성(fairness)을 유지할 수 있다. 이후에 다시 재전송된 패킷 7번에 의한 정상 승인 패킷 18번이 도착하게 되면 송신단은 윈도우를 2(*ssthresh*)로 설정하고 fast recovery를 성공적으로 종료하고 congestion avoidance를 수행하게 된다. 위에서의 예처럼 DAC$^+$를 사용함으로써 재전송 손실로 인해 발행하는 RTO를 줄일 수 있게 된다. DAC$^+$는 fast recovery를 종료하지 않고 재전송 손실 패킷을 복구하면서 윈도우와 *ssthresh*를 반으로 줄이기 때문에 같은 경로를 경유하는 다른 TCP 연결에 대해 DAC보다 더 나은 TCP 공평성을 갖는다.

2. EFR(Extended Fast Recovery)

TCP NewReno는 fast retransmit을 수행할 때 지금까지 전송한 패킷들 중에 가장 높은 시퀀스 번호를 *recover*에 저장한다[11]. 송신단에서는 저장된 *recover* 값과 수신되는 승인 패킷의 시퀀스를 비교하여 *recover* 보다 작거나 같으면 부분 승인 패킷으로, 그렇지 않은 경우는 정상 승인 패킷으로 분류한다. 부분 승인 패킷인 경우 TCP NewReno는 이 패킷의 시퀀스에 해당하는 패킷이 손실된 것으로 판단하고 재전송한다. TCP NewReno는 fast recovery중에 *recover* 값을 갱신하지 않는다. 따라서 fast recovery 과정 중에 *new packet*이 손실된 경우는 현재의 fast recovery가 종료된 이후에 손실된 패킷에 대한 새로운 3개의 중복 승인 패킷을 수신한 이후에나 복구 가능하다. 하지만 이러한 경우는 *usable window*내에서 손실된 패킷의 위치뿐만 아니라 원래 손실이 발생한 윈도우의 크기와 손실된 패킷의 개수와 관련성을 갖고 있으며 대부분의 경우에 3개의 중복 승인 패킷을 받을 수 없기 때문에 RTO이 발생한다.

EFR을 사용하는 송신단은 fast recovery 과정 중에 손실된 패킷을 재전송할 때마다 *recover*를 현재 전송한 패킷 중에 가장 높은 시퀀스로 갱신하면 재전송 이전에 발생한 *new packet* 손실에 대해 항상 부분 승인 패킷을 수신할 수 있게 되므로 송신단은 fast recovery를 종료하지 않은 상태에서 손실된 *new packet*을 복구할 수 있다. 이 경우에도 DAC$^+$처럼 윈도우와 *ssthresh*를 반으로 줄여야 한다[10].

EFR과 TCP NewReno의 동작 차이를 그림 2에 나타내었다. 윈도우의 크기가 8이고 패킷 7번부터 14번까지 전송하였을 때 7번 패킷과 14번 패킷이 손실되었고, fast recovery 과정 중에 *new packet* 15번이 손실된 경우이다. 첫 번째 7번 패킷을 재전송할 때까지의 모든 과정은 그림 1에 나타낸 DAC$^+$의 경우와 동일하다. 송신단은 7번을 재전송하면서 *recover*를 14로 설정한다. 이후에 3개의 중복 승인 패킷을 더 수신한 후 *usable window*는 10으로 증가되고 *new packet* 15와 16을 전송하지만 15번이 손실된다. 약 1.75초에 송신단은 두 번째 손실 패킷인 14번에 대한 부분승인 패킷을 수신하게 되고 재전송한다. 이때 윈도우는 4(*ssthresh*)로 설정되기 때문에 17번 패킷도 새로 전송된다. 이후에 패킷 16에 의해 발생되는 중복 승인 패킷 14번을 수신하고 *usable window*가 5로 증가하여 패킷 18번을 전송한다. 이후에 재전송한 패킷 14번에 의해 손실된 *new packet* 15번에 대한 승인 패킷을 수신하게 되

면 이 값이 *recover*에 저장된 14보다 크기 때문에 윈도우를 4로 설정하고 fast recovery를 종료하게 된다. 이후에 패킷 17번과 18번에 의해 2개의 중복 승인 패킷 15를 수신하지만 중복 승인 경계 값 3보다 적은 수를 수신하였기 때문에 fast retransmit을 수행하지 못한다. 또한 윈도우의 크기가 4이기 때문에 새로운 패킷을 전송하지 못하므로 NewReno에서는 재전송 타임아웃이 발생한다.

EFR의 경우는 송신단에서 패킷 14번을 재전송할 때 *recover*에 지금까지 전송한 패킷의 가장 큰 시퀀스인 16으로 갱신한다. 2.08초에 갱신된 *recover* 값 16보다 작은 부분 승인 패킷 15번이 수신되면 송신단은 패킷 15번을 재전송한다. 이때 윈도우와 *ssthresh*를 2([4/2])로 한 번 더 줄인다. 이후에 패킷 17과 18번에 의한 중복 승인 패킷을 수신하면 *usable window*를 4로 증가시키지만 이미 패킷을 전송한 상태이므로 새로 전송되는 패킷은 없다. 재전송한 패킷 15에 의해 승인 패킷 19번을 수신하면 이 값은 현재의 *recover* 16보다 크기 때문에 윈도우를 2(*ssthresh*)로 설정하고 fast recovery를 종료하게 된다.

DAC$^+$와 EFR은 각각을 따로 사용하는 경우보다 같이 사용하는 경우가 더 좋은 성능을 나타낸다.

Ⅲ. 실험 환경 및 성능 평가 방법

본 논문의 제안 알고리듬 성능 평가를 위해 ns-2를 사용하였다[12]. TCP 연결은 송신단과 수신단 사이에 10Mbps의 전송속도와 50msec의 전달지연을 갖는 링크를 사용하였고 데이터 패킷의 크기는 1KB이다. 패킷 손실은 p의 확률로 랜덤하게 발생하고 각 패킷 손실당 10^6개 패킷을 전송하는 것으로 가정하였다. 모든 TCP 변종들은 패킷 손실율이 너무 낮거나 높으면 거의 동일한 성능을 갖기 때문에 실험에서는 10^{-3}부터 $3*10^{-1}$까지만 고려하였다. 승인 패킷의 크기는 데이터 패킷에 비하여 크기가 작기 때문에 전송 중에 손실되지 않고, 하나의 TCP연결 설정을 고려하였기 때문에 "ACK-compression" 현상은 발생하지 않는 것으로 가정하였다[13]. 윈도우는 연결 설정시 수신단에서 허용하는 최대 윈도우 크기, W_{max}까지 증가할 수 있다.

제안 알고리듬은 DAC$^+$와 EFR을 같이 사용하였고, 아래와 같이 정의된 4가지 성능 평가 항목에 따라 평가하였다.

1) Normalized throughput : 각각의 p에서의 성능과 p가 10^{-3}일 때의 성능 비.
2) Fast recovery probability : 재전송을 통해 복구된 패킷[5]과 전체 손실 패킷 수의 비.
3) Timeout probability : RTO에 의해 재전송된 패킷 수와 전체 손실 패킷 수의 비.
4) Response time : 송신단에서 첫 번째 패킷을 전송한 시점부터 마지막 승인 패킷을 받는 시점까지의 경과 시간.

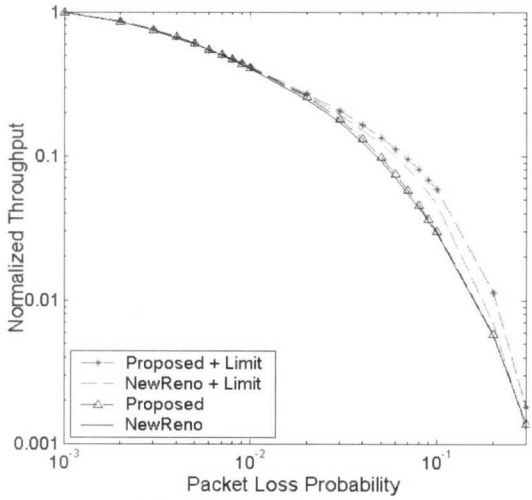
TCP는 사용하는 fast recovery 알고리듬에 따라 재전송을 통한 손실 복구 확률이 서로 다르다는 것이 이미 여러 연구에서 분석되었다[8],[14]. Kumar는 마코프 체인(Markov chain) 분석을 통해 손실 윈도우[6] 크기의 정상 상태 분포와 패킷 손실 확률 분석을 통해 각 손실 윈도우에서의 fast retransmit 확률을 구하였다[14]. 본 논문에서는 제안 알고리듬의 fast retransmit 성공 확률을 식 (1)에 정의된 Kumar 모델의 TCP NewReno fast retransmit 성공 확률과 비교한다.

$$P_{FR} = \sum_{M=1}^{W_{max}} F_M \cdot \pi(M) \tag{1}$$

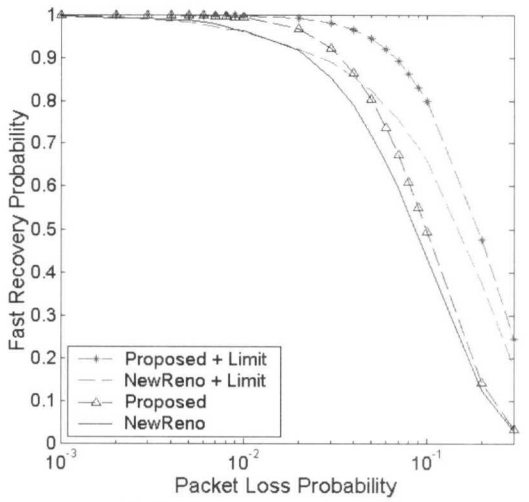
$$F_M = \begin{cases} 0 & 1 \le M \le K \\ \sum_{n=K}^{M-1} \binom{M-1}{n} (1-p)^n p^{M-1-n} & K+1 \le M \le W_{max} \end{cases} \tag{2}$$

M은 손실 윈도우의 크기이고, $\pi(M)$은 정상 상태에서 윈도우 크기의 확률 분포를 나타낸다. 식 (2)의 F_M은 손실윈도우의 크기가 M일 때 fast retransmit 확률이고 K는 fast retransmit에 의해 재전송하기 위해 필요한 중복 승인 패킷의 경계 값으로 보통 3으로 설정된다. F_M은 TCP NewReno가 첫 번째 손실 패킷이 fast retransmit에 의해 복구된다면 어떠한 경우에도 RTO이 발생하지 않는 이상적인 경우를 모델링 하였다.

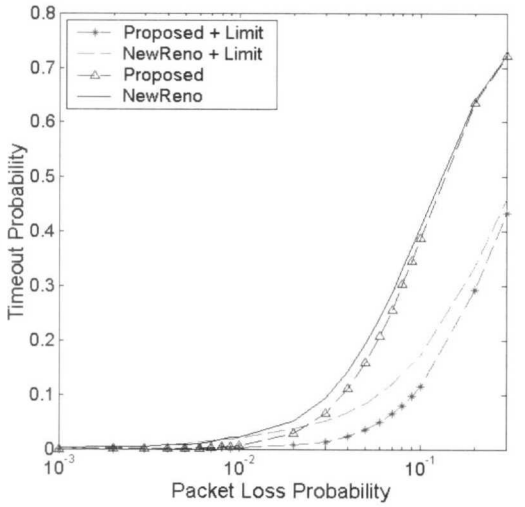
5) fast recovery과정 중에 fast retransmit, 부분 승인 패킷, DAC+, EFR에 의해 복구된 패킷
6) 패킷 손실 시점에서의 윈도우 크기



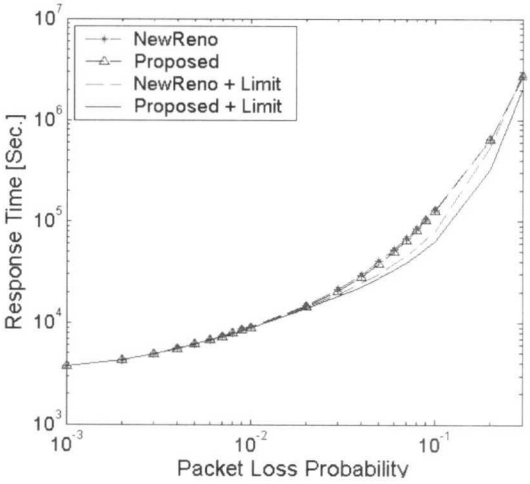
(a) Normalized throughput



(b) Fast recovery probability



(c) Timeout probability



(d) Response time

그림 3. 알고리듬별 성능 평가 비교(W_{max}=32)

Ⅳ. 결과 분석

그림 3에서는 4가지 성능 평가 항목에 대해 제안 알고리듬과 TCP NewReno를 비교하였다. 또한 Limited Transmit을 사용하였을 때의 성능도 같이 비교하였다. 그림 3-(a)의 normalized throughput과 3-(b)의 fast recovery 확률은 예상과 같이 p가 증가할수록 급격히 감소한다. 제안 알고리듬은 fast recovery 과정 중에 발생한 재전송 손실과 *new packet* 손실을 RTO없이 복구할 수 있기 때문에 fast recovery 확률이 TCP NewReno보다 높다. 이로 인해 normalized throughput은 제안 알고리듬이 p가 0.01부터 0.1까지의 범위를 가질 때 TCP NewReno보다 좋은 성능을 보인다. 하지만 성능에 큰 차이가 없는 것은 이 결과가 정상상태에서 얻어진 것이기 때문에 RTO의 영향이 긴 TCP 연결 시간 동안 고르게 분산되기 때문이다. 그러나 짧은 TCP 연결에서 *new packet* 손실이나 재전송 손실이 발생한다면, 제안 알고리듬은 더 좋은 성능을 제공할 수 있다. 제안 알고리듬은 normalized throughput과 fast recovery 확률이 p가 0.05일 때 최고 6.9%와 11.7%의 차이를 보이고 p가 낮거나

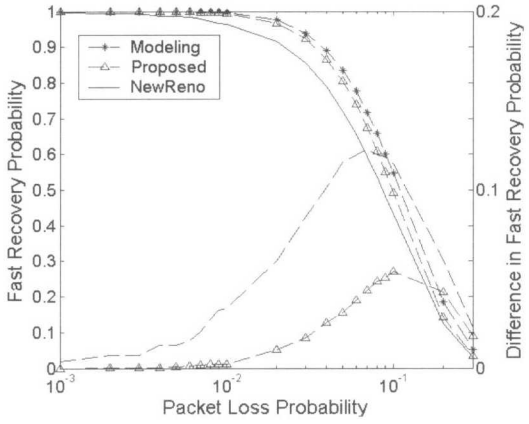


그림 4. Modeling에 의한 fast recovery 확률

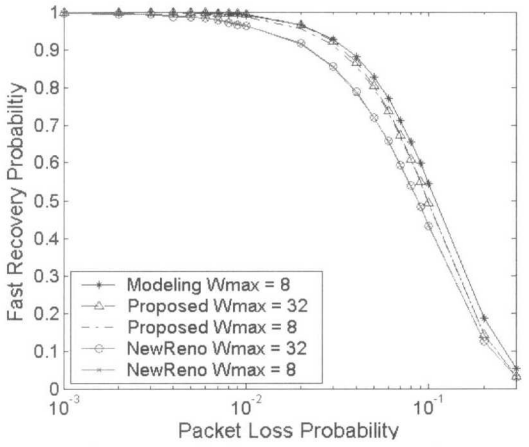


그림 5. W_{max}에 따른 Fast recovery 확률

높을 경우엔 거의 동일한 성능을 갖는다. 이것은 제안 알고리듬이 fast recovery 과정 중에만 동작하는데 p가 작은 경우에는 손실되는 패킷의 수도 작기 때문에 재전송된 패킷 또는 *new packet*이 손실될 가능성이 적어 제안 알고리듬이 동작하지 않고, p가 큰 경우에는 잦은 혼잡 제어로 윈도우 크기를 충분히 큰 값으로 유지하지 못해 fast retransmit을 하지 못하기 때문이다. 참고로 DAC^+ 대신 DAC와 EFR 조합을 사용한 경우는 p가 0.05일 때 최고 9.8%와 12.7%의 차이를 보인다.

그림 3-(c)는 p가 증가할수록 timeout 확률도 증가하는 결과를 보인다. 이 경우도 역시 p가 0.05일 때 제안 알고리듬의 RTO가 TCP NewReno에 비해 3.5% 포인트(18.3%) 감소한다. 실험에서 재전송 손실이 *new packet* 손실보다 많이 발생하고, 재전송 손실은 항상 RTO를 발생시키지만, *new packet* 손실은 앞서 언급한 대로 가능성이 낮긴 하지만 현재 fast recovery가 종료된 이후에 fast retransmit으로 복구될 수도 있기 때문에 DAC^+가 EFR보다 RTO 감소에 대한 기여도가 높다.

그림 3-(d)에서 볼 수 있듯이 제안 알고리듬과 TCP NewReno의 response time의 차이는 각 알고리듬의 모든 RTO의 기간의 합의 차이가 전체 TCP 연결시간에 비해 매우 작기 때문에 크지 않다. 하지만 Round Trip Time(RTT)의 관점에서 보면 제안 알고리듬의 사용으로 인해 많은 RTT(0.1초)가 감소한다.

그림 3-(a)~(d)에서 볼 수 있듯이 Limited Transmit을 TCP NewReno와 사용할 경우 윈도우의 크기가 작을 때 손실 패킷을 RTO없이 복구할 수 있는 효과적인 방법임을 확인할 수 있다. 하지만 Limited Transmit을 사용하는 경우에도 재전송 손실과 *new packet* 손실을 RTO없이 복구할 수 없기 때문에 제안 알고리듬과 Limited Transmit을 같이 사용하는 경우가 TCP NewReno와 같이 사용하였을 때보다 훨씬 좋은 성능을 보인다. 특히 전송하려는 데이터 양이 적거나, 수신단의 버퍼 제한 또는 종단간 혼잡 제어로 인해 윈도우의 크기를 크게 할 수 없는 환경에서는 제안 알고리듬과 Limited Transmit을 같이 사용하는 것이 효과적이다.

그림 4는 식 (1)~(2)에서 정의된 이상적인 TCP NewReno 모델과 제안 알고리듬, 그리고 TCP NewReno의 fast recovery확률이다. 오른쪽 y축은 모델링과 각 알고리듬의 차이를 표시하였다. 파선은 모델링과 TCP NewReno와의 차이를, 삼각형과 같이 표시된 파선은 제안 알고리듬과의 차이를 표시하였다. 제안 알고리듬은 이상적인 TCP NewReno 모델링에서 고려하지 않은 재전송 손실과 *new packet* 손실을 복구하기 때문에 fast recovery 확률은 거의 차이가 없다.

그림 5는 W_{max}값에 따른 복구 확률이다. TCP NewReno는 W_{max}가 8에서 32로 증가하여도 fast recovery 확률에는 큰 변화가 없다. 이것은 fast recovery 확률이 W_{max}보다 현재 윈도우 크기와 윈도우 안에 발생한 손실 패킷의 수에 결정되기 때문에 손실 복구 관점에서 큰 W_{max}로 인해 얻는 이득이 없다. 즉 TCP NewReno는 하나의 손실 패킷이 있는 경우 윈도우의 크기가 4이상이면 fast retransmit으로 복구되고, 다수개의 패킷 손실이 있는 경우라도 첫 번째 손실 패킷만 fast retransmit으

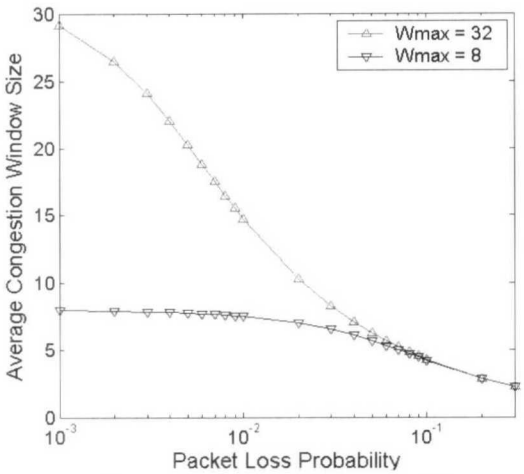


그림 6. W_{max}에 따른 평균 윈도우 크기

로 복구되면 RTO를 피할 수 있다. 이에 비해 제안 알고리듬은 W_{max}가 8인 경우 보다 32인 경우에 더 높은 fast recovery 확률을 갖는다. 이것은 비교적 큰 W_{max} 값에 의해 fast recovery 과정 중에 전송한 *new packet*의 개수와 손실의 개수가 달라졌기 때문이다. 즉 하나의 패킷이 손실되었을 때 윈도우의 크기가 8인 경우 송신단은 fast retransmit 이후에 3(=[8/2]+7-8)개의 *new packet*을 전송할 수 있고, 윈도우의 크기가 32인 경우는 17(=[32/2]+31-32)개의 *new packet*을 전송할 수 있다. 즉 같은 p에서 많은 수의 *new packet*은 많은 *new packet* 손실을 발생시킬 수 있고, EFR은 W_{max}가 32 일 때 더 많은 *new packet* 손실을 RTO없이 복구할 수 있다고 추정할 수 있다. 그러나 p가 증가할수록 윈도우의 크기는 최대값인 W_{max}에 도달하기 전에 크기를 계속 줄인다. 비록 패킷 손실이 존재할 경우 윈도우의 크기가 작게 유지되더라도 제안 알고리듬의 W_{max}가 8인 경우가 TCP NewReno의 W_{max}가 32일 때보다 큰 fast recovery 확률을 갖는다. 이것은 제안 알고리듬이 윈도우의 크기가 작은 경우에도 효과적으로 패킷손실을 복구하는 것을 나타낸다. DAC*의 경우는 중복 승인 패킷의 수에 의해 동작하기 때문에 W_{max}의 값과는 관련이 없다.

그림 6은 W_{max}가 8과 32일 때 p의 증가로 인한 잦은 혼잡 제어 호출로 감소한 윈도우의 평균 크기를 모델링한 것이다. W_{max}가 32일 때 p가 증가할수록 평균 윈도우의 크기가 급격하게 줄어드는 것을 보인다. 이것은 W_{max}의 값이 크더라도 p가 작을 때에만 큰 값에 의한 이득을 얻을 수 있고, p가 일정 값이상이 되면 W_{max}의 값에 상관없이 동일한 윈도우 값을 가지게 된다.

제안 알고리듬은 fast recovery 과정 중에 네트워크이 혼잡 상황이라면 수신되지 않을 중복 승인 패킷을 수신함으로써 손실된 패킷을 재전송하고, 윈도우를 반으로 줄인다. 따라서 제안 알고리듬은 동일한 경로의 자원을 공유하는 다른 TCP 연결에 비해 공격적으로 패킷을 전송하거나(aggressive) 공평도를 해치지 않는다. 특히 짧은 데이터 전송에서 송신단이 어떤 패킷이 손실되었는지 빠르고 정확하게 알 수 있다면 제안 알고리듬의 재전송 방법을 통해 RTO 이전이라도 복구하는 것이 효율적인 복구 방법일 뿐 아니라 실제 혼잡 상황에서도 추가적인 혼잡을 초래하지 않는다. 전송 손실이 빈번하게 발생하는 환경에서 제안 알고리듬은 TCP의 fast recovery를 효율적으로 동작하도록 만들어 준다. 특히 Limited Transmit을 같이 사용하는 경우에는 더욱 효율적으로 손실된 패킷을 RTO 없이 복구할 수 있다.

V. 결 론

본 논문에서 fast recovery 과정 중에 발생한 패킷 손실을 RTO 없이 복구하는 알고리듬을 제안하고, TCP NewReno의 이상적인 모델과 성능 비교를 보였다. 이 알고리듬을 이용하는 TCP NewReno는 대부분의 재전송 손실과 *new packet* 손실을 빠르게 감지하고 복구할 수 있다. 어떤 상황에서는 RTO가 발생하고 slow start부터 새로 시작하는 것이 성능에 더 좋을 수 있다. 즉 RTO를 무조건 발생하지 않게 하는 것이 항상 성능에 좋지는 않지만, 어떤 패킷이 손실되었는지를 정확하게 아는 경우에 재전송을 하지 않고 RTO를 기다리는 것은 비효율적이다. 실험 결과는 제안 알고리듬이 혼잡 제어의 원칙을 위배하지 않으면서 빠르게 손실된 패킷을 재전송하여 정상 상태에서 TCP NewReno의 성능을 개선한 것을 보인다. 재전송 손실과 *new packet* 손실이 많이 발생하는 사건이 아니기 때문에 이들을 빠르게 복구하는 것에 의한 성능 개선은 중요하게 생각되지 않을 수 있다. 그러나 RTO가 TCP의 성능에 미치는 심각한 영향을 고려한다면 간단한 제안 알고리듬을 구현하는 것은 충분한 가치가 있는 일이다. 특히 제안 알고리듬은 무선 상에서 web과 같은 응용에서 짧은 데이터 전송 중에 발생하는 재전송 손실과 *new packet* 손실로 인한 성능 저하를 방

지할 수 있다. 특히 Limited Transmit과 함께 사용된다면 효율적으로 모든 RTO의 원인에 대처할 수 있게 된다. 추후에 제안 알고리듬의 모델링과 무선에서 발생할 수 있는 burst한 손실에 대한 고려가 필요하다.

참 고 문 헌

[1] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links", *IEEE/ACM Transactions on Networking*, vol. 5. no. 6, pp. 756-769, 1997.

[2] M. Zorzi, A. Chockalingam, and R. R. Rao, "Throughput Analysis of TCP on Channel with Memory", *IEEE Journal on Selected Areas in Communications*, vol. 18. no. 7, pp. 1289-1300, 2000.

[3] C. Barakat, E. Altman, and W. Dabbous, "On TCP Performance in A Heterogeneous Network: A Survey", *IEEE Communications Magazine*, pp. 40--46, 2000.

[4] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, M. Stemm, and R. H. Katz, "TCP Behavior of a Busy Internet Server: Analysis and Improvements", *IEEE INFOCOM'98*, 1998.

[5] M. Allman, H. Balakrishnan, S. Floyd, "Enhancing TCP's Loss Recovery Using Limited Transmit", January 2001.

[6] K. Fall and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno, and SACK TCP", *ACM Computer Communication Review*, vol. 26. no. 3, pp. 5-21, 1996.

[7] Dong Lin and H. T. Kung, "TCP Fast Recovery Strategies: Analysis and Improvements", *IEEE INFOCOM'98*, pp. 263-271, 1998.

[8] 김범준, 김동연, 이재용, "임의 패킷 손실에 대한 TCP의 손실 복구 과정 모델링 및 분석", *한국통신학회논문지*, 28권 4B호, 2003.

[9] Dongmin Kim, Beomjoo Kim, Jechan Han and Jaiyong Lee, "Enhancements to the Fast Recovery Algorithm of TCP NewReno", *ICOIN 2004*

[10] M. Allman, V. Paxson, W. Stevens, "TCP Congestion Control", RFC 2581, April 1999.

[11] S. Floyd and T. Henderson, "The NewReno Modification to TCP's Fast Recovery Algorithm", RFC 2582, 1999.

[12] http://www.isi.edu/nsnam/ns/index.html

[13] J. Mogul, "Observing TCP dynamics in real networks," in *Proc. SIGCOM'92*, pp. 305-317.

[14] Anurag Kumar. "Comparative Performance Analysis of Versions of TCP in a Local Network with a Lossy Links," *IEEE/ACM Transactions on Networking (ToN)*, vol. 6, no. 4, pp. 485-498, Aug. 1998

김 동 민(Dongmin Kim) 정회원

1999년 2월 : 건국대학교 전자공학과 졸업
2001년 8월 : 연세대학교 전기전자 공학과 석사
2001년 9월~현재 : 연세대학교 전기전자공학과 박사과정

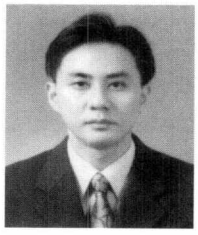
<주관심분야> TCP 모델링 및 성능 분석, 무선 TCP, 무선 랜, IPv6, Mobile IP, 센서네트워크

김 범 준(Beomjoon Kim) 정회원

1996년 2월 : 연세대학교 전자 공학과 졸업
1998년 8월 : 연세대학교 전자 공학과 석사
2003년 8월 : 연세대학교 전자 공학과 박사
2003 년 6월~2004년 1월 : 연세대학교 전기전자공학과 전자정보통신 연구소 박사후과정 연구원
2004 년 2월~현재 : LG전자 이동통신기술연구소 표준화그룹

<주관심분야> IEEE 802.16, IEEE 802.21, B3G Convergence Network, TCP 성능 분석, 무선 링크상의 TCP 성능 향상 방안, IP기반 유무선 통합 네트워크, IP 트래픽 엔지니어링.

김 석 규(Seoggyu Kim) 정회원



1990년 2월 : 연세대학교 전자 공학과 졸업
1992년 8월 : 연세대학교 전자 공학과 석사
1997년 8월 : 연세대학교 전자 공학과 박사
1997 년 9월~2004년 3월 : SK 텔레콤 선임연구원
2004년 3월~현재 연세대학교 전기전자공학과 전자정보통신 연구소 박사후과정 연구원
<주관심분야> 유,무선 통합망에서 QOS Architecture, B3G Convergence Network, TCP 성능 분석, 무선 링크상의 TCP 성능 향상 방안, IP기반 유무선 통합 네트워크

이 재 용(Jaiyong Lee) 정회원

1977년 2월 : 연세대학교 전자 공학과 졸업
1984년 5월 : IOWA State University 공학 석사
1987년 5월 : IOWA State University 공학 박사
1987년 6월~1994년 8월 : 포항공과대학 교수
1994년 9월~현재 : 연세대학교 전자공학과 교수
<주관심분야> Protocol Design for QoS Management, Network Management, High Speed Networks