TCP 흐름제어를 이용한 인터넷 게이트웨이에서의 예측기반 버퍼관리 및 조기혼잡예측기법

정회원 여 재 영*, 최 진 우*

On the efficient buffer management and early congestion detection at a Internet gateway based on the TCP flow control mechanism

Jae-Yung Yeo*, Jin-Woo Choe* *Regular Members*

요 약

본 논문에서는 TCP 흐름제어 기법을 전제로 하여, 가까운 미래에 발생할 가능성이 있는 인터넷 입구에서의 혼잡을 미리 예측함으로써 TCP 흐름제어의 효과를 배가시킬 수 있는 QR-AQM이라는 버퍼관리 기법을 제안한다. QR-AQM 버퍼관리기법은 라우터로 전송되어 오는 단위시간당 TCP 흐름량을 추정하여 이를 이용함으로써 RED 기법과 같이 평균버퍼 수위에만 의존하는 조기혼잡제어기법에 비해 미래에 네트워크에 일어날 혼잡에 보다 민첩하게 대처 할 수 있다. 모의 실험을 통한 성능 검증결과 인터넷 입구에서 QR-AQM 기법은 Adaptive RED 기법과 비교하여 간단한 변수의 설정만으로 버퍼를 목적한 수준으로 유지함이 가능함으로, 다양한 응용프로그램의 요구에 맞추어 직관적이고 쉽게 최적의 버퍼수위를 설정할 수 있다. 또한 QR-AQM 기법은 시간에 따른 버퍼수위의 변화가 목적한 버퍼수위를 기준으로 작은 변동폭을 유지함으로써, 패킷들이 라우터의 버퍼를 지나면서 겪게되는 지연시간의 등락현상을 효과적으로 억제할 수 있음을 확인하였다.

key Words : TCP; RED; AQM; flow control.

ABSTRACT

In this paper, we propose a new early congestion detection and notification technique called QR-AQM. Unlike RED and it's variation, QR-AQM measures the total traffic rate from TCP sessions, predicts future network congestion, and determine the packet marking probability based on the measured traffic rate. By incorporating the traffic rate in the decision process of the packet marking probability, QR-AQM is capable of foreseeing future network congestion as well as terminating congestion resolution procedure in much more timely fashion than RED. As a result, simulation results show that QR-AQM maintains the buffer level within a fairly narrow range around a target buffer level that may be selected arbitrarily as a control parameter. Consequently, compared to RED and its variations, QR-AQM is expected to significantly reduce the jitter and delay variance of packets traveling through the buffer while achieving nearly identical link utilization.

*서강대학교 전자공학과 고속통신망연구실

논문번호 : 030240 - 0602, 접수일자 : 2003년 6월 2일

I. 서 론

인터넷(Internet) 사용인구의 폭발적 증가와 다양한 인터넷기반 응용프로그램의 등장으로 인터넷 기간망(backbone)의 부하는 포화상태로 접근하고 있다. 따라서, 인터넷을 통해 보다 많은 사용자가 만족스러운 네트워크서비스를 받기 위해서는 대역폭효율(bandwidth efficiency)을 높이는 것이 매우 중요하며, 이를 위한 대규모 연구노력이 세기를 걸쳐 지속되고 있다. 그러나, 인터넷은 기본적으로 단순히 패킷(packet)을 도착지까지 전달하기 위한 매우 제한적인 기능만을 제공하므로 대역폭 활용도를 높이기 위한 수많은 이론적 연구결과가 발표되었음에도 불구하고 극히 소수만이 실제 인터넷에서 보편적으로 구현되고 있으며, 그 대표적인 예로 Transmission Control Protocol (TCP)에 구현되어 있는 흐름제어(flow control)기법을 들 수 있다[1]. TCP에서의 흐름제어기법에는 여러 형태가 있으나, Tahoe, Reno와 같이 패킷유실을 네트워크 혼잡(congestion)의 함축적인 되먹임(feedback)으로 간주하여 이에 반응하는 기법들이 현재까지도 널리 사용되고 있다[1].

TCP 흐름제어 기법은 종단간(end-to-end) 흐름제어기법으로 분류될 수 있으며 송신단에서 아직 확인응답(acknowledgement)을 받지 못한 정보의 크기를 제한하는 방식으로 이루어진다. 즉, 송신단에서 전송된 후 성공적인 전송이 확인되지 않은 정보의 양이 전송창(congestion window)의 크기 W를 넘지 않는 범위에서 패킷을 전송하며, 전송된 패킷에 대한 확인응답을 받을 때마다 W를 증가시킴으로써 실질적인 전송속도를 증가시킨다. 반면, 전송된 패킷 중에 유실된 패킷이 감지될 경우 이를 네트워크의 혼잡의 징후로 판단하여 전송창의 크기 W를 감소시킴으로써 실질적인 전송속도를 낮추고 네트워크 내의 혼잡을 해소한다. 패킷유실의 감지는 패킷이 송수신단을 왕복하는데 필요한 평균시간보다 약간 크게 설정된 마감시한(time-out period) 내에 확인응답이 도착하지 않는 경우 또는 같은 패킷에 대한 확인응답이 반복되는 경우 등을 통해 간접적으로 이루어지며, 따라서, 실제로 네트워크의 임의의 지점에서 패킷이 유실되는 시점과 송신단에서 이를 인지하게 되는 시점간에는 상당한 차이가 있게 된다. 특히 그림 1과 같은 일반적인 인터넷 상황에서, 패킷의 유실이 인터넷의 입구, 예를 들어 송신단이 속한 근거리망의 게이트웨이(gateway)에서 발생할 경우에 이 시간차이는 최대가 되며, 이는 네트워크 혼잡이 송신단에 의해 감지되기 위해 필요한 시간이 패킷이 송신단과 수신단을 왕복하는데 필요한 시간에 근접할 수 있음을 의미한다. 즉, 인터넷의 입구에서 발생하는 혼잡은 송신단에 의해 감지되어 해소될 때까지 최소, 전달지연시간(propagation delay)에서 최대, 전달지연시간의 두배 만큼의 시간을 필요로 하고, 다수의 재전송(Automatic Retransmission Request, ARQ)을 유발함으로써 전체 네트워크에서의 대역폭 효율을 저하할 수 있다.

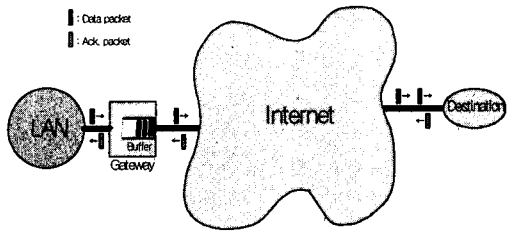


그림 1. 일반 인터넷환경에서의 TCP 패킷전송

이에 본 논문에서는 Reno 또는 Tahoe와 같이 패킷의 유실에 반응하여 동작하는 TCP 흐름제어기법을 전제하고, 가까운 미래에 발생할 가능성이 있는 인터넷 입구에서의 혼잡을 미리 예측함으로써 TCP 흐름제어의 효과를 배가하는 버퍼관리기법을 제안한다. 그러나, 제안하고자 하는 버퍼관리 기법의 적용범위는 근거리망 등을 인터넷으로 연결하는 게이트웨이로만 국한되는 것은 아니며, 인터넷 내부의 어느 라우터나 스위치에서도 적용이 가능하고, Diff-Serve[2]와 같은 차세대 인터넷 환경에서도 간단한 수정을 통하여 최선형 트래픽(best-effort traffic)에 적용할 수도 있는 일반적이고 기초적인 개념의 버퍼관리기법을 제안하는 것이 본 연구의 목적이다.

II. 혼잡의 조기감지와 효율적 버퍼관리기법

1. TCP 흐름제어를 통한 버퍼관리 기법

앞 절에서 간단히 언급되었던 바와 같이 TCP 흐름제어는 패킷의 재전송기법과 연동하여 동작한다. 한편, 모든 네트워크에서 전송되는 패킷은 송신단을 출발하여 최종 수신단에 도착한 후 다시 성공적 전

송을 확인하기 위한 확인응답이 송신단으로 돌아올 때까지 필연적으로 수 ms에서 수백 ms까지의 시간 지연이 발생한다. 따라서, 최종 수신단에서 송신단으로 전송하는 확인응답에만 의존하는 재전송기법에서는 전송된 패킷의 유실여부를 판단하기 위해 최소 왕복전송지연시간(round trip delay)만큼의 시간을 필요로 하게 된다. 이는 네트워크 내부에서 혼잡이 발생해도 상당한 시간동안은 흐름제어가 전혀 이루어지지 않고 혼잡과 그에 따른 패킷의 유실 및 재전송이 지속되게 됨을 의미하고, 그 결과로 대역폭효율이 크게 저하될 우려가 있다.

전송지연시간으로 인해 발생될 수 있는 이와 같은 문제점을 해결하는 비교적 소극적인 방법으로는 네트워크 혼잡으로 인해 버퍼의 범람이 발생할 우려가 있을 때 버퍼 내에서 가장 오래된 패킷을 폐기(drop)하고 새로 도착하는 패킷을 버퍼에 보관하는 Drop Head-of-Line기법 등이 있다. 그러나, 고속 네트워크에서는 이와 같은 방법을 통해 송신단이 패킷유실을 감지할 때까지 필요한 시간을 크게 단축하지 못하므로, 그 효과는 상대적으로 미미하다.[3] 보다 효과적인 방법으로는 가까운 미래에 발생할 수 있는 혼잡을 조기에 예측하여 미리 일부의 패킷을 고의로 폐기시키는 적극적인 개념의 버퍼관리기법이 있으며, 그 대표적인 예로 Early Random Drop (ERD)[4]과 Random Early Detection (RED)[5]을 들 수 있다.

2. Early Random Drop 버퍼관리기법

ERD 방식은 기존의 소극적인 방법의 버퍼관리 기법보다 발전된 형태를 지닌다. 먼저 버퍼에서 일정한 임계수위를 설정하고 만약 현재 버퍼의 수위가 이를 넘어가면 이를 혼잡이 일어날 징후로 감지하고 일정한 확률로 들어오는 패킷을 폐기함으로써 버퍼가 넘치는 최악의 경우를 조기에 방지하고 송신단에 혼잡의 발생을 미리 예고한다. 이와 같은 조기혼잡감지기법을 도입함으로써 혼잡이 발생하는 시점으로부터 흐름제어가 시작되는 시점까지의 시간차를 줄일 수 있으며, 따라서 대역폭효율의 향상도 기대 할 수 있다.

3. Random Early Detection (RED)버퍼관리기법

ERD의 발전된 형태라 할 수 있는 RED는 Sally Floyd와 Van Jacobson에 의해 1993년 처음으로 제안되었으며, 그 이후 성능개선을 위해 다양한 수정이 이루어 졌다. RED는 앞서 설명한 ERD 기법이 현재 버퍼 수위를 기준으로 혼잡의 정도를 판단하는 것과 달리 기본적으로 평균버퍼수위(average queue length)를 네트워크 혼잡의 지표로 사용하여 확률적으로 패킷을 폐기시키는 버퍼관리기법이다. 즉, 다양한 방법으로 임의의 시간 t에서의 평균버퍼수위를 계산하고 그 결과를 $\bar{q}_t$라 하면,[1] 도착하는 패킷을 혼잡의 해소를 위해 폐기할 확률(packet dropping/marking probability) p_m은 $\bar{q}_t$에 의해 결정된다.

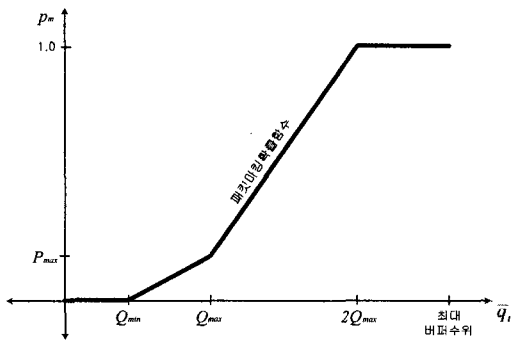


그림 2. RED에서 사용되는 패킷마킹확률함수의 예

$\bar{q}_t$와 p_m의 관계를 정의하는 함수를 패킷마킹확률함수(packet marking probability function)라 통칭하며, 이 패킷마킹확률함수를 다르게 선택함에 따라 실제 RED 버퍼관리기법의 동작은 차이가 있게 된다. RED 버퍼관리기법에서 널리 사용되는 패킷마킹확률함수의 전형적인 형태는 그림 2에 주어져 있다.[6]이 예에서는 제어상수(control parameter)로 Q_{min}, Q_{max}, P_{max}가 주어지며, 이들은 다음과 같은 의미로 해석될 수 있다.

- Q_{min}: 조기 혼잡제어를 위해 위한 고의적 패킷의 폐기가 시작되는 버퍼수위
- Q_{max}: 조기 혼잡제어가 끝나고 적극적인 혼잡제어가 시작되는 버퍼수위
- P_{max}: 조기 혼잡제어의 의미에서 패킷폐기확률이 취하는 최대값

즉, 평균버퍼수위가 Q_{min}이하일 경우에는 가까운 미래에 혼잡의 위험이 없다고 판단하여, 고의로 패

1) RED에서의 평균버퍼수위의 개념은 시간 t직전의 짧은 시간 구간동안에서 버퍼수위의 시간평균을 의미한다

킷을 폐기하지 않고, 버퍼수위가 Q_{min}와 Q_{max}사이에 있을 때는 아직 심각하다고 볼 수 없는 상황이나 앞으로 발생할 가능성이 있는 혼잡을 방지할 목적으로 비교적 작은 비율의 패킷만을 폐기시키며, 평균버퍼수위가 Q_{max}를 넘어서게 되면 심각한 수준의 혼잡이 시작되는 것으로 판단하여 보다 공격적으로 패킷을 폐기하게 된다.

이러한 기본적인 동작원리를 바탕으로 RED 버퍼관리 기법은 여러 가지 수정을 거쳐 현재에 다양한 형태의 성능 개선안이 제안되고 있다. 그 예로는 Adaptive RED[7], Blue[8], Adaptive Virtual Queue(AVQ)[9], PI controller[10] 등이 있다. 그러나 이들 변형 버퍼관리기법은 대부분 최초 RED 버퍼관리기법의 기본적인 형태를 유지하고 있으며 많은 경우 특정상황에서 발생할 수 있는 RED의 결함보완이나 성능개선에 초점을 맞추고 있다[11]. 일례로서 Adaptive RED 기법은 처음 W. Feng 등에 의해 제안된 것을 Sally Floyd가 보완하여 발표한 것으로[12], Adaptive RED 기법을 간략히 요약하면 앞에서 언급한 RED 기법에서 P_{max} 값을 가변적으로 변화시키면서 평균버퍼 수위가 Q_{min}과 Q_{max}사이의 값을 유지하도록 하는 것이다. 알고리즘의 동작 형태는 매 특정 시간간격마다 평균버퍼수위 $\overline{q}_t$가 목표로 하는 범위 내에 있는가를 확인하는 형태로 이루어진다. 만일 $\overline{q}_t$가 목표범위보다 크고, P_{max} 값이 0.5보다 작거나 같으면 P_{max}를 $\min\{0.01, P_{max}/4\}$ 만큼 증가시키며, $\overline{q}_t$가 목표범위보다 작고 P_{max}가 0.01 이상이면 P_{max}는 0.9의 비율로 감소된다.

앞에서 설명한 Adaptive RED를 비롯한 RED 기반 변형 버퍼관리기법들은 공통적으로 버퍼수위에 의존하여 미래의 혼잡을 예측하며, 이로 인해 혼잡 시작 및 해소 시점의 감지가 지연된다는 공통적인 문제점을 내포하게 된다. 예를 들어, 인터넷 입구에서 혼잡이 발생하면, RED 버퍼관리기법을 구현하는 게이트웨이가 이를 인지하고 패킷폐기확률을 높이기 위해서는 버퍼의 수위가 특정수준 이상으로 증가하여야 한다. 따라서 혼잡이 시작되고 이를 감지하기까지 지연시간이 존재하게 되고, 혼잡제어가 일어나기 전까지 버퍼의 수위는 필요이상으로 높아지게 된다. 또한 조기혼잡예측 및 해소과정이 시작되고 충분한 시간이 지나게 되면, 송신단에서는 네트워크상의 혼잡을 인지하고 패킷전송속도를 낮추게 된다. 그러나 전송속도의 감소가 버퍼수위와 함께 패킷폐기확률을 낮출 때까지는 또한 상당한 정도의 시간이 추가로 필요하게 되며, 이는 RED 버퍼관리기법 또는 수정기법들이 송신단에서 이미 전송속도를 충분히 낮추어 혼잡이 사실상 해소된 상태에서도 이전의 혼잡으로 인해 높아진 버퍼수위 때문에 일정시간동안 계속 많은 수의 패킷을 폐기하게 된다는 것을 의미한다. 이러한 혼잡이 해소된 상태에서의 패킷폐기는 불필요한 재전송을 유발할 뿐만 아니라, 버퍼의 수위를 필요이상으로 저하시키게 된다. 이렇듯, RED 버퍼관리기법에서는 혼잡이 버퍼의 수위에 반영되기까지의 지연시간으로 인해 버퍼의 수위가 불필요한 등락을 거듭하게 되며 이로 인해 버퍼를 지나면서 패킷들이 겪는 지연시간 지터 역시 증가하게 된다. 이러한 RED 버퍼관리기법 및 그 변형들이 가지는 문제를 해결하기 위해서는 단순히 버퍼수위만이 아닌 다른 지표도 네트워크의 혼잡정도를 판단하는데 사용이 되어야 한다. 다음 절에서는 버퍼수위와 함께 전체 TCP 세션(session)들의 전송속도도 함께 고려하는 새로운 버퍼관리기법을 제안한다.

Ⅲ. QR-AQM 버퍼관리기법

1. QR-AQM의 동작원리

본 논문에서 제안하는 Queue-Rate Active Queue Management (QR-AQM) 버퍼관리기법의 동작원리는 매우 간단하며 또한 직관적이다. 우선, 대역폭이 μ인 링크로 인터넷에 연결된 게이트웨이에서 어떤 시점 t_o에 버퍼수위가 q_{t_o}이고 전체 TCP 세션들의 총 전송속도가 r_{t_o}로 주어졌다고 하자. 이후로 이 전송속도가 변화하지 않는다는 가정 하에서 $t \geq t_o$에 대해 예측버퍼수위 $\hat{q}_t$는 다음과 같이 간단히 유도된다.

$$\hat{q}_t = q_{t_o} + (r_{t_o} - \mu)(t - t_o). \qquad (1)$$

만일, 게이트웨이로 전송되는 TCP 패킷들을 정확히 $p_m \in (0,1)$ 비율의 패킷들을 확률적으로 폐기한다면, 실질적으로 버퍼에 저장될 정보가 게이트웨이에

도착하는 속도는 평균적으로 $p_m r_{t_o}$만큼 감소하게 되고, (1)은

$$\hat{q}_t = q_{t_o} + ((1-p_m)r_{t_o} - \mu)(t - t_o) \quad (2)$$

와 같이 수정될 수 있다. 즉, p_m을 0과 1사이에서 적절히 선택함으로써 q_t의 예측변화율을 $-\mu$와 $r_{t_o} - \mu$사이에서 임의로 조절할 수 있다. 제안하는 QR-AQM은 식 (2)를 이용하여 버퍼수위가 항상 이상적인 버퍼수위 q_{opt}에 최대한 근접하도록 p_m(패킷폐기확률)을 결정하는 것이 그 동작의 기본원리이다.

2. QR-AQM 버퍼관리 세부 알고리즘

상세한 QR-AQM 버퍼관리기법의 설명을 위해서 다음과 같은 확률변수 및 상수를 정의한다.

- τ_i : i번째 패킷이 게이트웨이에 도착하는 시간
- t_i : i번째 패킷과 i-1번째 패킷 사이의 시간차 (즉, τ_i - τ_{i-1})
- s_i : i번째 패킷의 크기 또는 정보량
- r_i : i번째 패킷의 도착 직후에 추정된 전체 TCP 세션의 전송속도
- q_i : i번째 패킷이 도착하기 직전의 버퍼수위
- q_{opt} : QR-AQM 버퍼관기기법이 유지하고자 하는 이상적인 버퍼수위
- α : 전송속도 추정기의 민감도를 결정하는 제어상수
- T_o : 버퍼수위를 이상적인 수준 q_{opt}로 가깝게 접근시키기 위해 허용된 시간

QR-AQM 버퍼관리기법은 새로운 패킷이 게이트웨이에 도착하는 시점에 r_i와 p_m을 갱신하며 그 결과로 0보다 큰 p_m값이 결정될 경우, 이 확률로 도착한 패킷을 폐기하는 과정을 통해 버퍼수위를 조절하게 된다. 따라서, QR-AQM 버퍼관리기법의 정확한 동작은 (가) 각각의 새로운 패킷이 도착하는 시점에 추정 전송속도인 r_i의 결정방법, (나) r_i로부터 p_m을 결정하는 방법을 결정방법을 기술함으로써 정의될 수 있다.

2.1 전송속도 의 추정

i번째 패킷이 도착한 시점에서 추정된 평균 패킷의 크기와 패킷간 시간 간격을 각각 $\bar{s}_i$와 $\bar{t}_i$라 하면, i번째 패킷이 도착한 시점에 추정된 TCP 세션들의 총 전송 속도 r_i는 일반적으로 다음과 같이 나타낼 수 있다.

$$r_i = \frac{\bar{s}_i}{\bar{t}_i} \quad (3)$$

따라서, $\bar{s}_i$, $\bar{t}_i$를 추정하는 방법에 따라 r_i의 추정 방식은 여러 가지로 분류될 수 있으며, 이중 널리 사용되는 방식으로는 우선 window기반의 moving-average 추정방식과 auto-regressive 방식이 있다[13].

우선, moving-average 방식에서 window란 전송속도 추정을 위해 평균을 취하게 될 시간구간의 크기 또는 패킷의 수를 의미하며, 각 패킷의 도착 시점에서 현재까지의 특정 길이의 시간구간에 도착한 패킷 또는 현재까지 도착한 특정 수의 패킷에 대해 (즉 window 내에서) 패킷의 크기와 시간간격의 평균값을 취함으로써 전송속도를 추정하게 된다. 예를 들어 w를 window의 크기라 하면 가장 최근에 도착한 w개의 패킷정보로부터 $\bar{s}_i$, $\bar{t}_i$를 다음과 같은 식으로 구할 수 있다.

$$\bar{s}_i = \frac{1}{w} \sum_{k=i-w+1}^{i} s_k \quad (4)$$

$$\bar{t}_i = \frac{1}{w} \sum_{k=i-w+1}^{i} t_k \quad (5)$$

한편, auto-regressive 방식에서는 현재까지 도착한 패킷들의 크기와 시간간격을 아래의 식과 같이 지수함수적 평균을 취하는 방식으로 $\bar{s}_i$, $\bar{t}_i$를 추정하게 된다.

$$\bar{s}_i = (1-\alpha) \sum_{k=0}^{\infty} \alpha^k s_{i-k}. \quad (6)$$

$$\bar{t}_i = (1-\alpha) \sum_{k=0}^{\infty} \alpha^k t_{i-k}. \quad (7)$$

이러한 전송속도 추정방식의 장점은 식 (6)와 (7)의 추정식이 아래와 같이 점화적으로 구현될 수 있다는 것이다.

$$\bar{s}_i = \alpha \bar{s}_{i-1} + (1-\alpha) s_i \quad (8)$$

$$\bar{t}_i = \alpha\,\bar{t}_{i-1} + (1-\alpha)\,t_i \qquad (9)$$

위의 점화식을 moving-average 전송속도 추정식 (4), (5)와 비교하면, auto-regressive 방식을 이용함으로써 새로운 패킷이 도착할 때마다 $\bar{s}_i$와 $\bar{t}_i$를 갱신하기 위해 필요한 연산을 절감할 수 있고, 또한 오직 $\bar{s}_i$, $\bar{t}_i$만을 보관함으로써 지속적인 추정이 가능하므로 기억용량도 절감할 수 있음을 쉽게 알 수 있다. 이와 같은 이점을 활용하기 위해 QR-AQM 버퍼관리기법에서는 auto-regressive 방식으로 전송속도를 추정한다. 한편, 식 (8)과 식 (9)에서 예상할 수 있듯이 α의 값이 0에 가까워짐에 따라 추정되는 $\bar{s}_i$, $\bar{t}_i$값은 가장 최근에 도착한 패킷들에 보다 많은 영향을 받게 되고, 따라서 시간에 따라 빠르게 변화하는 특성을 가지게 된다. 즉, α는 auto-regressive 전송속도 추정방식의 민감도를 결정하는 제어상수로 볼 수 있으며, QR-AQM 버퍼관리기법의 성능 또한 α값의 선택에 따라 영향을 받을 수 있다.

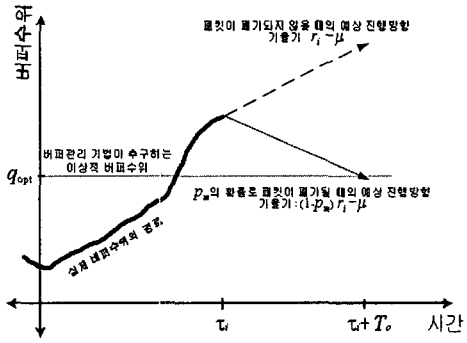


그림 3 QR-AQM에서 패킷폐기확률의 도식적 해석

2.2 패킷폐기확률 의 결정

일단, r_i가 계산되면, 이로부터 식 (2)를 이용하여 p_m을 계산하게 된다. 이 계산을 위해 두 가지 제어상수 q_{opt}와 이에 접근하기 위해 주어진 시간 T_o가 사용된다. 즉, QR-AQM버퍼관리기법에서는 새로운 패킷이 도착하는 시점 τ_i에서부터 T_o의 시간이 경과한 시점 $\tau_i + T_o$에 버퍼수위가 이상적인 수준 q_{opt}에 최대한 근접하도록 p_m을 결정하며, 이를 도식화하면 그림 3과 같다. 따라서, 패킷폐기확률을 계산하기 위한 식은 다음과 같이 유도된다.

$$p_m = \left\langle \frac{q_i - q_{opt} + (r_i - \mu)T_o}{r_i T_o} \right\rangle_0^1 \qquad (10)$$

단, 여기서 $\langle x \rangle_0^1$은 $\max\{0, \min\{1, x\}\}$을 의미한다. 이와 같은 방법 계산된 패킷폐기확률 p_m은 직전에 도착한 i번째 패킷의 폐기 여부를 결정하기 위해 사용되며, 매번 새로이 도착하는 패킷의 폐기여부는 이전 패킷의 폐기 여부에 관계없이 p_m에 의해 독립적으로 결정된다.

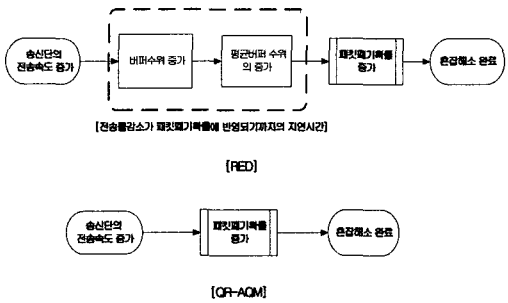


그림 4 혼잡이 시작되는 시점에서의 QR-AQM과 RED의 조기혼잡예측 및 패킷폐기기능 발효과정

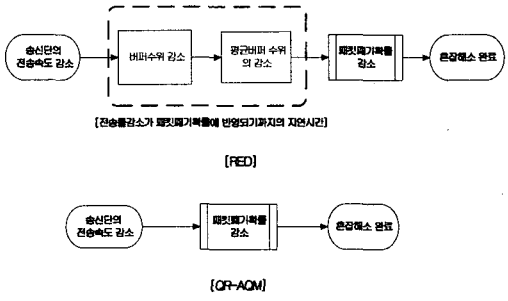


그림 5 혼잡이 해소되는 시점에서의 QR-AQM과 RED의 조기혼잡예측 및 패킷폐기기능 해제 과정

3. QR-AQM과 RED의 비교

RED와 비교하여 QR-AQM의 가장 큰 차이점은, 버퍼수위에 TCP세션의 전송속도도 감안하여 패킷의 폐기여부를 판단한다는 점이다. TCP세션의 전송속도를 추가로 고려함으로써 네트워크 혼잡의 심각성을 보다 적절한 시기에 정확히 판단하게되는 효과가 있으며, 이는 특히 혼잡이 시작되는 시점과 해소되는 시점에 혼잡제어의 효율을 향상하는 요인으로 작용한다. 그림 4와 그림 5는 혼잡이 시작되는 시점과 해소되는 시점에서의 RED와 QR-AQM에서 조기혼잡예측을 통해 버퍼를 관리하는 과정을 비교하

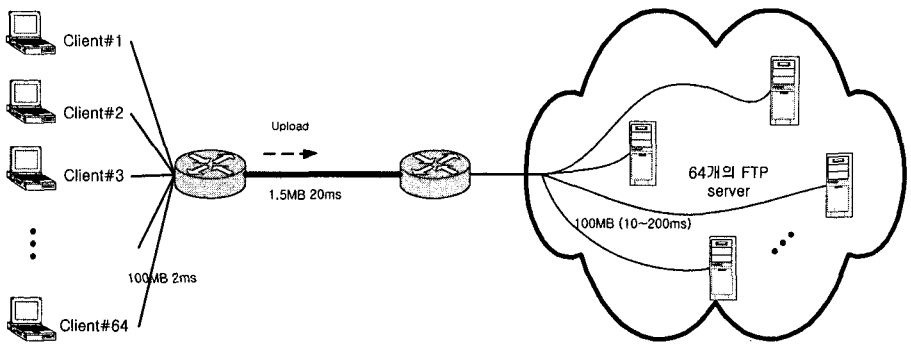


그림 6 RED와 QR-AQM의 성능비교를 위한 Network 모델

여 나타낸 것이다.

우선 혼잡이 시작되는 상황을 고려하면, RED의 경우 혼잡이 감지되기 위해서는 평균버퍼수위가 일정한 수준(즉, Q_{min})까지 증가하여야 한다. 그러나 일반적으로 혼잡의 징후는 전송속도의 증가라는 형태로 우선적으로 나타나고, 이러한 현상이 평균버퍼수위의 증가로 나타날 때까지는 일정시간이 필요하게된다. 즉, RED의 경우 혼잡이 시작됨에 따라 패킷의 고의적 폐기가 일어나는 시기가 혼잡의 일차적 징후인 전송속도의 증가가 나타나는 시기보다 상당히 늦어지게 된다. 따라서 RED 기법에서는 혼잡제어가 시작되기까지의 지연시간 때문에 버퍼수위가 필요이상으로 증가하게 된다. 한편, QR-AQM의 경우 전송속도의 증가가 즉시 패킷폐기를 확률적으로 증가시키는 효과를 가져온다. 예를 들면, 현재의 버퍼수위가 이상적인 수준인 q_{opt}이하인 경우라도 전송속도의 증가로 인해 r_i가 큰 값을 갖게되면, $p_m > 0$으로 설정되어 패킷의 폐기가 시작된다. 따라서, 새로운 TCP 세션의 시작 등으로 인해 급격히 전체 전송속도가 증가하고 네트워크의 혼잡이 시작되는 상황에서 QR-AQM은 즉시 패킷폐기확률을 증가시키게 되고, 버퍼수위가 과도하게 올라가는 현상을 방지한다. 이와 유사한 형태의 개선은 혼잡이 해소되는 상황에서도 발생한다. 그림 5에서 도식적으로 예시되었듯이, RED의 경우 혼잡이 감지되어 일정 시간동안 패킷의 폐기가 일어나고, TCP 흐름제어가 동작하여 전송속도가 충분히 낮아진 상황에서도, 전송속도의 감소가 평균버퍼수위에 반영될 때까지 패킷의 폐기가 계속된다. 그러나 QR-AQM의 경우 현재의 버퍼수위가 이상적인 수위 q_{opt}보다 높다 하더라도 전송속도 추정치인 r_i가 충분히 작은 값을 갖게 되면, 혼잡제어를 위한 패킷의 폐기는 중단되게 된다. 따라서, QR-AQM 기법은 혼잡으로부터 벗어나는 시점에서 불필요한 패킷의 폐기를 억제하는 효과가 있으며, 과도한 패킷 폐기로 인해 버퍼수위가 필요이상으로 저하되는 현상을 완화 할 수 있다.

앞에서 살펴보았듯이 QR-AQM은 혼잡제어의 시점을 개선함으로써, 패킷의 폐기와 회복이 RED에 비하여 신속하게 이루어지게 된다. 따라서, 버퍼수위가 과도하게 등락을 반복하는 것을 방지해주며 또한, 적은 등락폭을 유지함으로써 QR-AQM 스위치나 라우터를 통과하면서 패킷들이 겪는 지연시간의 지터 역시 줄일 수 있을 것으로 예상할 수 있다. 이어지는 절에서는 간단한 실험을 통해 QR-AQM 버퍼관리기법의 성능을 Adaptive RED 버퍼관리기법과 비교함으로써 앞서 기술된 QR-AQM 버퍼관리기법의 도입으로 기대되는 이론적인 성능개선을 실험적으로 확인한다.

Ⅳ. 실험을 통한 QR-AQM의 성능평가

1. RED와 QR-AQM의 실험 및 비교 방법

본 절에서는 실험을 통해 QR-AQM의 성능을 RED와 비교하여 검증한다. 실험은 NS2 2.1b9a version을 통하여 이루어졌으며 그림 6에 주어진 가상의 네트워크 모델에서 RED와 QR-AQM의 성능이 비교되었다. 그림을 통해 쉽게 확인할 수 있듯이 하나의 게이트웨이를 통해 인터넷에 연결된 64개의 클라이언트들이 인터넷상의 다양한 위치에 있는 서버들과 TCP 세션을 형성하고 있는 비교적 간

단한 네트워크 모델 상에서 실험이 이루어 졌으며, 모든 클라이언트들은 반드시 이들을 인터넷으로 연결하는 게이트웨이와 인터넷 입구노드간의 링크를 통하여 서버와 통신을 하도록 설정하였다. 여기서, 모든 클라이언트들은 게이트웨이에 100MB의 대역폭에 2ms의 지연시간을 가지는 링크를 통해 연결되어 있고, 서버들은 인터넷의 입구노드로부터 100MB의 대역폭과 최소 10ms에서 최대 200ms 까지 다른 지연시간[2]을 갖는 링크에 의하여 연결되어 있는 것으로 가정하였으며, 게이트웨이와 인터넷을 연결하는 링크는 1.5MB의 대역폭과 20ms의 지연시간을 갖도록 설정되었다. 즉, 클라이언트들이 서버로 패킷을 전송하기 위해서는 필연적으로 링크사정이 좋지 않은 병목구간, 즉 게이트웨이와 인터넷 입구노드간의 링크를 통해야 하는 상황으로 실험환경을 설정하였으며, 클라이언트에서 서버로 패킷을 전송할 경우, 게이트웨이에서 혼잡으로 인해 패킷 유실이 일어난다면 유실된 패킷은 최소 62ms (=20ms×2+10ms×2+2ms)에서 최대 442ms(=20ms ×2+200ms×2+2ms) 이후에 해당 클라이언트에 의해 감지될 수 있다. 64개의 클라이언트들은 시뮬레이션 시작과 함께 병목구간을 통해 서버에 파일 전송을 시작한다. 이 때 하나의 클라이언트는 각각 하나의 서버와 통신을 하게 되며, 따라서 전체 64개의 ftp 세션을 생성하게 된다. ftp응용은 전체시간 100초 동안 계속되며 클라이언트에서 전송되는 패킷의 크기는 데이터 패킷의 경우 1040byte, 확인응답패킷의 경우 40byte로 두 가지 종류이며 병목구간 라우터의 버퍼용량은 100KByte로 설정하였다.

RED 시뮬레이션

RED의 경우에는 보다 공정한 비교를 하기 위하여 초기의 RED가 아닌 Sally Floyd에 의해 최근에 제안된 Adaptive RED 알고리즘을 사용하였다.[7] Adaptive RED 알고리즘 역시 다른 RED 변형 알고리즘과 마찬가지로 기본적으로는 버퍼수위를 기반으로 한 혼잡제어를 한다. 그러나 내부변수의 설정이 까다로운 기존의 RED 알고리즘과 비교하여 그 성능이 내부변수 설정에 의해 크게 영향을 받지 않고 또한 다양한 네트워크 환경에서도 목적한 수준의 평균버퍼 수위를 유지 할 수 있는 장점을 가지고 있다. 실험에서 사용한 Adaptive RED 알고리즘의 내부변수들은 NS2에서 제공하는 최적화된 값들을 이용하였으며 또한 Adaptive RED의 성능을 최대화하기 위하여, 내부변수를 변화시킬 경우는 Sally Floyd에 의해 제안된 권고안에 따라 실험을 진행하였다.[12]

QR-AQM 시뮬레이션

QR-AQM의 경우에도 다양한 설정을 통해 얻은 결과 중 버퍼수위를 가장 안정적으로 유지하는 것을 택하여 RED와 비교하였다. 즉, 다양한 값의 q_{opt}에 대하여 α와 T_o를 변화시켜가며 실험한 후 목적한 버퍼수위를 가장 효과적으로 유지하는 설정 ($\alpha=0.99$, $T_o=40$ms)을 택하여 모든 실험에 사용하였다. 참고로 QR-AQM의 성능은 $1/(1-\alpha)\approx 100$, $T_o\approx 50$ms를 만족하는 α, T_o 값에 대하여 큰 변화를 보이지 않았으며, 따라서 이들을 최선의 값으로 선택하는데 큰 어려움은 없었다.

2. RED와 QR-AQM의 성능비교 결과

먼저 그림 7~9에서는 Adaptive RED를 게이트웨이에 적용했을 때 전체 실험시간 100초 동안의 시간에 따른 버퍼수위의 변화를 그래프로 도시하였다. 그림 7은 NS2 2.1b9a에 내장되어 있는 Adaptive RED의 최적 내부변수를 사용한 경우이며, 그림 8과 그림 9는 내부변수 중 Q_{min}, Q_{max}의 값을 증가시키면서 버퍼수위의 변화를 추적한 경우다. 이들 실험결과로부터 패킷의 전송이 안정적으로 이루어지는 10초부터 90초 사이의 구간에서 평균버퍼수위를 구했으며 그 결과를 표 1에 명시하였다. QR-AQM을 동일한 네트워크 모델에 적용한 실험에서는 RED를 적용한 실험에서 측정한 평균버퍼수위를 q_{opt} 값으로 설정함으로써 동일한 수준의 평균버퍼수위를 유지하도록 하였으며, 역시 10초부터 90초 사이의 평균버퍼수위를 구하여 표 1에 나타내었다. QR-AQM이 적용되었을 때의 시간에 따른 버퍼수위의 변화는 그림 10~12에 도시하였다.

2) 인터넷의 입구노드에서 서버까지의 지연시간은 NS2에 내장된 균일분포 난수발생기를 이용하여 10ms에서 200ms사이의 값을 무작위로 할당하였다.

실험설정	내부변수 설정	평균 버퍼수위
RED1	자동설정	10175.18Byte
RED2	Q_{min}=7500Byte, Q_{max}=22500Byte	27428.72Byte
RED3	Q_{min}=22500Byte, Q_{max}=67500Byte	73541.44Byte
QR-AQM1	q_{opt}=10175.18Byte	9361.688Byte
QR-AQM2	q_{opt}=27428.72Byte	26256.51Byte
QR-AQM3	q_{opt}=73541.44Byte	73615.99Byte

표 1 Adaptive RED와 QR-AQM 내부변수 변화에 따른 평균버퍼수위

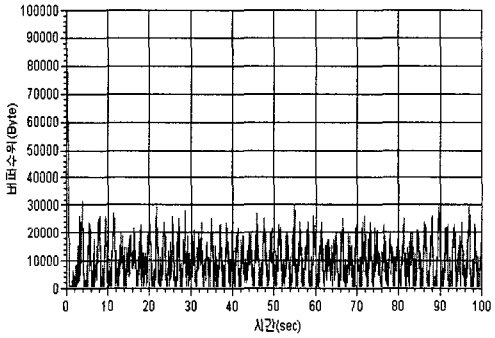


그림 7 자동설정시의 Adaptive RED에서 시간에 따른 버퍼수위의 변화

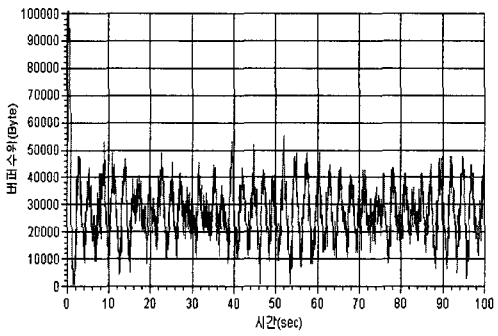


그림 8 Q_{min}은 7.5Kbyte, Q_{max}는 22.5Kbyte로 설정 후 Adaptive RED에서 시간에 따른 버퍼수위의 변화

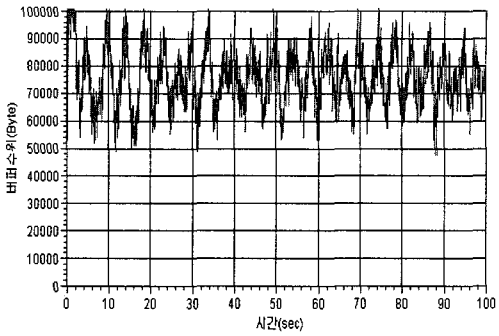


그림 9 Q_{min}은 22.5Kbyte Q_{max}는 67.5Kbyte로 설정 후 Adaptive RED에서 시간에 따른 버퍼수위의 변화

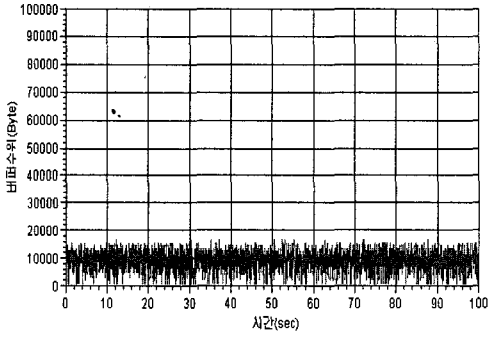


그림 10 q_{opt}=10175.18Byte일 때 QR-AQM에서 시간에 따른 버퍼수위의 변화

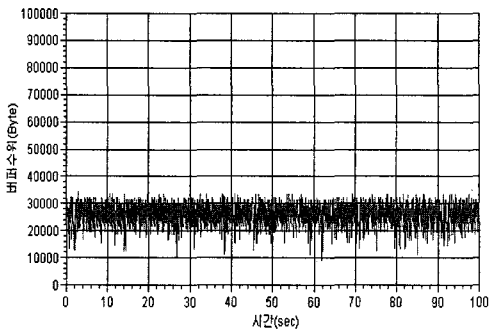


그림 11 q_{opt}=27428.72Byte일 때 QR-AQM에서 시간에 따른 버퍼수위의 변화

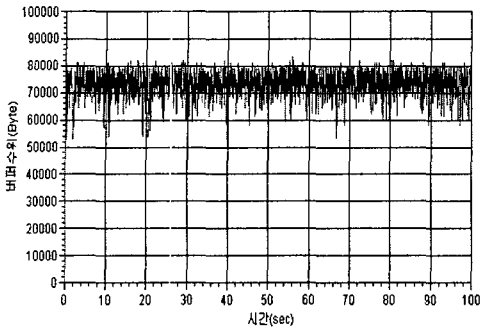


그림 12 q_{opt}=73541.44Byte일 때 QR-AQM에서 시간에 따른 버퍼수위의 변화

표 1에서 q_{opt}이 3가지 다른 값으로 설정되었을 때의 QR-AQM의 평균버퍼수위를 살펴보면, QR-AQM은 목표로 하는 버퍼수위 q_{opt}부근에서 버퍼수위를 유지할 수 있음을 확인할 수 있다. 이는 QR-AQM을 적용할 경우 제어상수인 q_{opt}를 통해 평균버퍼수위를 직접 제어하는 것이 가능함을 보여주고 있으며, 따라서 Q_{min}과 Q_{max}를 통해 간접적으로 평균버퍼수위를 제어할 수 있는 RED 방식에 비해 QR-AQM이 보다 쉽고 정확하게 평균버퍼수위를 제어할 수 있음을 알 수 있다. 일반적으로 평균버퍼수위가 높으면 전송률(throughput)은 증가하는 반면 버퍼에서 패킷들이 겪는 평균지연시간이 증가하는 단점이 있다. 반대로 평균버퍼수위가 낮으면 패킷의 지연시간은 감소되지만 전송률이 저하된다. 따라서, 버퍼관리기법은 응용프로그램이 요구하는 패킷지연시간을 보장할 수 있는 수준에서 평균버퍼수위를 최대한 증가시킴으로써 지연시간과 전송효율간의 최적 균형점을 찾아야 하며, 이러한 상황에서 QR-AQM은 RED에 비해 손쉽게 다양한 응용프로그램의 요구에 맞추어 최적의 버퍼수위를 유지할 수 있는 기능을 제공한다.

한편, 그림 7~12에서 시간에 따른 버퍼수위의 등락폭을 비교해보면 비슷한 수준의 평균버퍼수위를 유지하는 상황에서 Adaptive RED에 비해 QR-AQM은 확연히 작은 등락폭을 보여주며 버퍼수위가 q_{opt}값에 근접한 범위에서 유지되는 것을 볼 수 있다. 이러한 등락폭의 감소는 앞 절에서 언급되었던 QR-AQM의 개선된 혼잡예측 및 해소 능력으로부터 얻어진 효과라 할 수 있으며, 버퍼수위의 변동폭을 감소시킴으로써 버퍼를 통과하면서 패킷들이 겪는 지연시간의 지터[13]를 크게 개선할 수 있다. 이를 확인하기 위해 표 2에 Adaptive RED와 QR-AQM가 적용된 다양한 상황에서의 실효전송효율[3]과 지연시간의 평균과 표준편차를 패킷전송이 안정화되는 10초에서 90초 사이의 구간에서 측정하여 나타내었다.

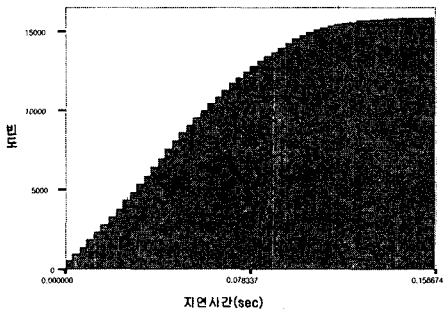


그림 13 자동설정시의 Adaptive RED 라우터에서 패킷들이 겪는 지연시간의 누적 히스토그램

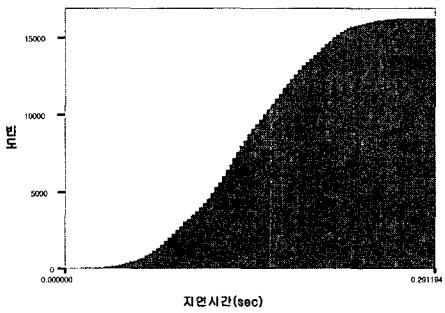


그림 14 Q_{min}은 7.5Kbyte Q_{max}는 22.5Kbyte로 설정 후 Adaptive RED 라우터에서 패킷들이 겪는 지연시간의 누적 히스토그램

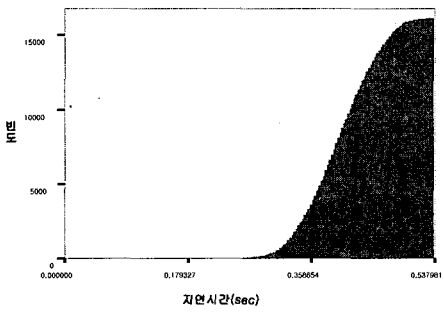


그림 15 Q_{min}은 22.5KByte Q_{max}는 67.5Kbyte로 설정 후 Adaptive RED 라우터에서 패킷들이 겪는 지연시간의 누적 히스토그램

3) 본 논문에서는 실효전송효율을 병목 링크의 전체 대역폭에서 패킷전송을 위해 (재전송을 포함하여) 사용된 대역폭의 비율로 정의한다.

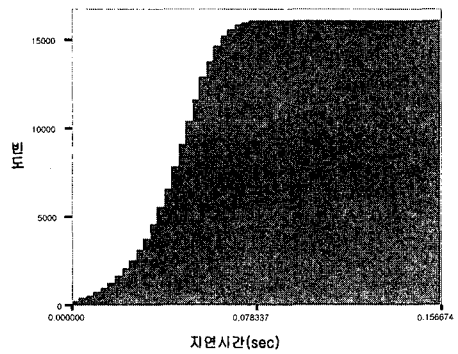


그림 16 q_{opt}=10175.18Byte일 때 QR-AQM 라우터에서 패킷들이 겪는 지연시간의 누적 히스토그램

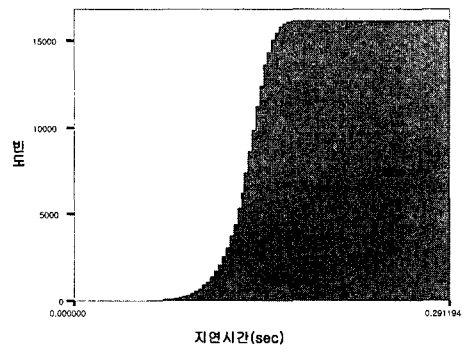


그림 17 q_{opt}=27428.72Byte일 때 QR-AQM 라우터에서 패킷들이 겪는 지연시간의 누적 히스토그램

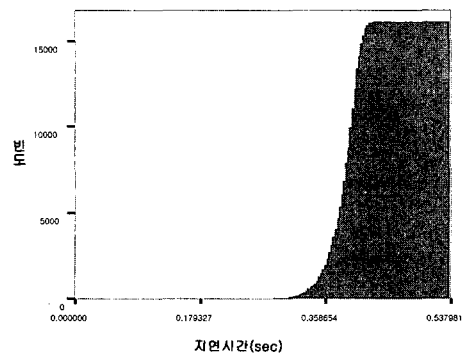


그림 18 q_{opt}=73541.44Byte일 때 QR-AQM 라우터에서 패킷들이 겪는 지연시간의 누적 히스토그램

실험설정	내부변수 설정	실효전송 효율(%)	평균지연 시간(sec)	지연시간 표준편차
RED1	자동설정	97.8	5.083E-02	3.210E-02
RED2	Q_{min}=7500Byte Q_{max}=22500Byte	99.99	14.31E-02	4.829E-02
RED3	Q_{min}=22500Byte Q_{max}=67500Byte	99.99	39.98E-02	5.012E-02
QR-AQM1	q_{opt}= 10175.18Byte	99.28	4.357E-02	1.597E-02
QR-AQM2	q_{opt}= 27428.72Byte	99.99	13.39E-02	1.760E-02
QR-AQM3	q_{opt}= 73541.44Byte	99.99	38.62E-02	2.184E-02

표 2 Adaptive RED와 QR-AQM의 내부변수에 따른 지연시간의 분포

표 2에서 유사한 수준의 평균버퍼를 유지하는 각각의 실험설정 즉, RED1과 QR-AQM1, RED2와 QR-AQM2, 그리고 RED3와 QR-AQM3를 각각 비교해보면 QR-AQM이 적용된 경우가 비슷한 평균지연시간과 실효전송효율을 유지하면서도 패킷들이 겪는 지연시간의 표준편차가 확연히 작은 것을 알 수 있다. 이는 다양한 실험설정에서 패킷들이 겪는 지연시간의 누적 히스토그램을 나타낸 그림 13~18에서도 확인된다. 즉, Adaptive RED의 경우 누적 히스토그램이 완만하게 누적되어 있는 것을 볼 수 있으며, 이는 패킷들의 지연시간이 광범위하게 분포함을 의미한다. 반면 QR-AQM 경우 누적분포가 평균지연시간 부근에서 가파르게 상승하는 것으로 볼 수 있으며, 이는 대다수 패킷들의 지연시간이 평균 지연시간을 중심으로 하는 좁은 구간에 분포함을 의미한다.

V. 결 론

본 논문에서는 TCP에 구현된 흐름제어를 응용한 기존의 적극적인 개념의 조기혼잡제어기법인 RED와 그 개선기법들의 취약점에 대해 알아보고, 그 문제점이 버퍼수위에만 근거하여 패킷폐기확률을 산출하는데 있음을 확인하였다. 이러한 기존 버퍼관리기법의 취약점을 개선하기 위해 전체 TCP 세션의 전송속도를 추정하여 미래의 혼잡상황을 예측하고, 또한 패킷폐기확률을 계산하는데 반영하는 QR-AQM기법을 제안하였으며, 모의실험을 통해 QR-AQM이 RED보다 목표로 하는 버퍼수위를 유지함에 있어 제어상수의 설정이

용이하고 또한 적절한 시기에 혼잡제어를 시작하고 중단함으로써 패킷들이 버퍼에서 겪게되는 지연시간의 지터를 줄일 수 있음을 입증하였다.

QR-AQM 기법에서는 목표로 하는 버퍼수위를 나타내는 제어상수 q_{opt}와 함께 전송속도를 추정하는데 있어 중요한 제어변수인 α와 폐기확률의 계산에 영향을 미치는 T_0값이 QR-AQM의 성능에 상당한 영향을 미칠 수 있다. 본 논문에서는 시행착오를 통해 이들 두 제어상수의 최적값을 도출하였으나, QR-AQM과 관련한 추후 연구에서는 QR-AQM의 성능개선과 다양한 네트워크 상황에 대한 적응성의 향상에 초점을 맞출 필요가 있으며, 특히 α와 T_0값을 네트워크의 구조와 혼잡정도에 따라 적응적으로 설정하는 알고리즘을 개발함으로써 QR-AQM을 실제의 네트워크에 적용하는데 있어 편의성을 크게 향상할 수 있을 것으로 예상된다.

참고문헌

[1] L. Peterson et. al., *Computer Networks a Systems Approach*, Morgan Kaufmann, pp464-474, 2000

[2] S. Blake et. al., "An Architecture for Differentiated Services," *RFC2475*, Dec. 1998.

[3] B. Braden et. al., "Recommendations on Queue Management and Congestion Avoidance in the Internet," *RFC2309*, April 1998.

[4] S. Keshav, *An Engineering Approach to Computer Networking*, Addison-Wesley, pp.258-259, 2001

[5] S.Floyd and V.jacobson, "Random Early Detection gateways for congestion avoidance," *in Proc. of IEEE/ACM Trans. Net., Vol. 1*, pp. 397-413, Aug 1993.

[6] C. Brandauer et. al., "Comparison of tail drop and active queue management performance for bulk-data and Web-like Internet traffic," *in Proc. of IEEE Symp. on Comp. and Comm.*, p.p. 122-129, July 2001.

[7] Sally Floyd et. al.. "Adapive RED: An Alogorithm for Increasing the Robustness of RED's Active Queue Management," *pre-report*, (available at http://www.icir.org/floyd/papers.html,) August 2001.

[8] W.Feng et. al., "Blue: An Alternative Approach To Active Queue Management Algorithms," *NOSSDAV 2001*, August 2001.

[9] S. Kunniyur et. al., "Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm fir Active Queue Management," *SIGCOMM 2001* , August 2001.

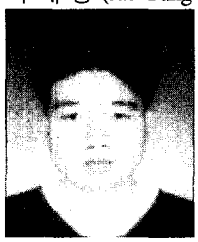
[10] C. Hollot et al., "On Designing Improved Controllers for AQM Routers Supporting TCP Flows," *INFOCOM 2001*, April 2001.

[11] M. Christiansen et. al., "Tuning RED for Web traffic," *IEEE/ACM Trans. Net., Vol. 9*, p.p. 249-264, June 2001.

[12] Sally Floyd et. al.. "RED: Discussion of Setting parameters from email message" (available at http://www.icir.org/floyed/REDparameters.txt,) November 1997.

[13] Z. Wang, *Internet QOS*, Morgan Kaufmann, pp.17-18, 2001

여 재 영 (Jae-Yung Yeo) 정회원

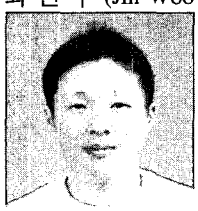


2002년 2월 : 서강대학교 전자공학과 졸업

2002년 3월~현재 : 서강대학교 전자공학과 석사과정

<주관심분야> 인터넷 QOS, 통신망성능해석

최 진 우 (Jin-Woo Choe) 정회원



1990년 2월 : 서울대학교 제어 계측공학과 학사

1992년 2월 : 서울대학교 제어 계측공학과 석사

1998년 12월 : Purdue University 전기컴퓨터공학과 박사

1998년 11월~2001년 1월 : University of Toronto 전기컴퓨터공학과 조교수

2001년 3월~현재 : 서강대학교 전자공학과 조교수

<주관심분야> 통신망 설계 및 성능해석, 광통신망, 멀티미디어 캐싱