다수의 병렬 TCP Flow를 가진 스테이션에 의한 대역폭 독점을 감소시키는 History-Aware RED

정회원 전 경 구*

History-Aware RED for Relieving the Bandwidth Monopoly of a Station Employing Multiple Parallel TCP flows

Kyungkoo Jun* ***Regular Member***

요 약

본 논문에서는 다수의 병렬 TCP flow들을 가진 소수의 스테이션들이 링크 대역폭을 독점하는 불공평성 문제에 대해 randam early detection (RED)을 수정한 history-aware RED (HRED)를 제안한다. BitTorrent와 같은 peer-to-peer방식의 파일 공유 애플리케이션들은 파일 다운로드를 위해 다수의 병렬 TCP flow들을 이용한다. 만약 파일 공유 애플리케이션을 수행하는 스테이션들이 다른 스테이션들과 링크를 공유할 경우 대역폭을 독점하는 문제가 발생한다. 이 경우 개별 TCP flow들 간의 공평성 지원을 위해 개발된 RED를 적용하더라도 불공평성은 개선되지 않는다. 제안하는 HRED는 RED와 유사하게 도착하는 패킷들에 대해 확률적으로 drop여부를 결정하되, 스테이션의 링크 점유율에 따라 drop 확률을 조정할 수 있어, 대역폭을 독점하는 스테이션들의 패킷들에 drop 패널티를 부과할 수 있다. 여러 가지 상황을 가정한 시뮬레이션을 통해 HRED가 RED에 비해 스테이션 차원에서의 throughput 공평성 지원 측면에서 최소 60%이상, 전송 효율성 측면에서 4%이상 개선되었음을 확인하였다.

Key Words : TCP fairness, Random Early Detection, History-Aware, Drop Probability

ABSTRACT

This paper proposes history-aware random early detection (HRED), a modified version of RED, to lessen bandwidth monopoly by a few of stations employing multiple parallel TCP flows. Stations running peer-to-peer file sharing applications such as BitTorrent use multiple TCP flows. If those stations share a link with other stations with only a small number of TCP flows, the stations occupy most of link bandwidth leading to undesirable bandwidth monopoly. HRED like RED determines whether to drop incoming packets according to probability which changes based on queue length. However it adjusts the drop probability based on bandwidth occupying ratio of stations, thus able to impose harder drop penalty on monopoly stations. The results of simulations assuming various scenarios show that HRED is at least 60% more effective than RED in supporting the bandwidth fairness among stations and at least 4% in utilization.

Ⅰ. 서 론

링크를 공유하는 TCP flow들 간의 불공평성은 다양한 조건과 인자들 때문에 발생하기 때문에 여러 연구들이 진행되어 왔다. 우선 TCP congestion control 설계 자체의 결함에 의해 초래되는 불공평성이 있다. TCP Reno의 경우에는 짧은 round trip time (RTT)을 갖는 flow가 긴 RTT를 갖는 flow보

* 인천대학교 멀티미디어시스템공학과 (kjun@incheon.ac.kr)
논문번호 : KICS2009-04-149, 접수일자: 2009년 4월 9일, 최종게재논문통보일자: 2009년 10월 27일

다 더 많은 대역폭을 차지하는 불공평성이 나타난다 [1].

네트워크 환경에 따라 초래되는 불공평성 문제는 ad hoc 모드와 infrastructure 모드의 무선 네트워크 환경에서 각각 다루어졌다. 여기에 대해서는[2]-[5]에서 연구되었다.

하지만 이러한 불공평성에 대한 연구들 중에서 하나의 스테이션이 다수의 TCP flow들을 동시에 병렬적으로 이용하여 전송 대역폭을 독점하는 문제를 다룬 것은 거의 없다[6]. Weighted-fair queuing (WFQ)[12]와 이를 응용한 deficit round robin (DRR)[13] 등이 공평성과 자원 효율성을 고려한 큐 관리 방식이지만 이들은 flow 수준이기 때문에 스테이션 수준의 공평성 지원에는 적절치 않다. 또 다른 큐 관리 방식인 REM[14]이나 PI[15] 같은 방식들은 오히려 공평성을 악화시키는 것으로 알려져 있다[16]. RED를 확장한 방식들 중에서, weighted RED[17]는 flow 수준의 QoS 지원에 관한 것이며, scalable fair RED (SFRED)[18] 방식은 flow 수준에서의 공평성 지원을 다루고 있어 스테이션 수준 적용에 적합치 않다.

그림 1은 다수의 병렬 TCP flow들을 이용한 대역폭 독점 현상을 시뮬레이션으로 재연하기 위한 시나리오이다. 총 6개의 TCP flow가 라우터 A와 B 사이의 링크를 공유하고 있고, 이 중 5개가 하나의 스테이션 STA_{multi} 속하고, 나머지 하나는 스테이션 STA_{single}에 속한다. STA_{multi}는 동시에 다섯 개의 FTP 연결을 통해 스테이션 STA1~5들로부터 각각 한 개씩의 파일을 다운로드 받는 반면, STA_{single}은 단 하나의 FTP 연결을 이용해서 STA6로부터 파일 다운로드를 받는다.

위 시나리오를 Qualnet 시뮬레이터[7]를 이용하여 실험하였다. 라우터 A와 B 사이를 포함한 모든 링크들은 10 Mbps 최대전송속도를 갖고, FTP 연결을 통해 전송되는 파일 다운로드 트래픽은 512 바이트 데이터 패킷을 지속적으로 30초간 전송한다.

그림 2는 시간에 따른 STA_{multi}와 STA_{single}의 수신 throughput의 변화를 보여준다. 공평한 대역폭 분배가 이루어지려면 STA_{single}은 전체 스테이션 수(2개)를 고려할 때 대역폭의 1/2, 50%를 차지해야만 한다. 하지만 실험 결과 STA_{multi}가 평균 6.8Mbps을 차지하여 대역폭을 독점한 반면, STA_{single}은 2 Mbps만을 할당받아, 전체 throughput 8.8Mbps의 22.7%만을 차지하는 불공평성 문제가 발생한다.

이러한 독점문제는 스테이션들 간의 서로 다른 TCP flow 개수에 의해 발생하지만, 라우터에서 패킷들을 임시 저장하는 큐의 drop-tail 방식에 의해 악화된다. Drop-tail 방식은 lock-out 문제를 일으키는데, 이것은 일부 flow들의 패킷들이 큐 공간을 독점하여, 다른 flow들의 패킷 전송을 방해하는 것이다. 앞의 시나리오에서는 STA_{multi}의 패킷들에 의해 STA_{single}의 패킷들이 lock-out되어 불공평성이 심화되었다.

이렇게 하나의 스테이션이 다수의 병렬 TCP flow들을 이용하여 파일을 다운로드 받는 방식을 swarmping이라고 하는 데, BitTorrent[8] 같은 peer-to-peer 파일 공유 프로토콜에서 사용된다. 최근 P2P기반 파일공유 프로그램은 동시에 10개에서 100개의 TCP flow를 사용한다. 웹 브라우징과 비교할 때 P2P 파일공유의 개별 flow는 단위시간당 throughput은 20배 더 크고, flow 개수는 5 ~ 50배가 더 많아져 약 100 ~ 1000배의 대역폭을 더 사용하게 된다. 그 결과 전체 10% 미만의 P2P 파일공

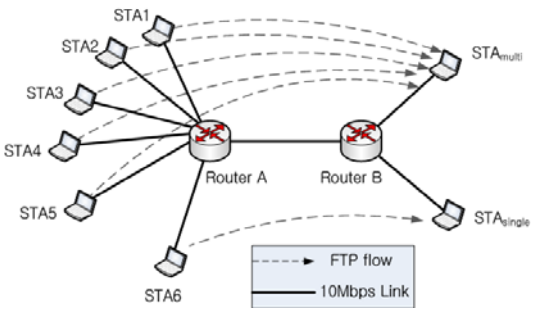

그림 1. 다수의 병렬 TCP flow를 가진 STAmulti에 의한 전송대역폭 독점 시나리오

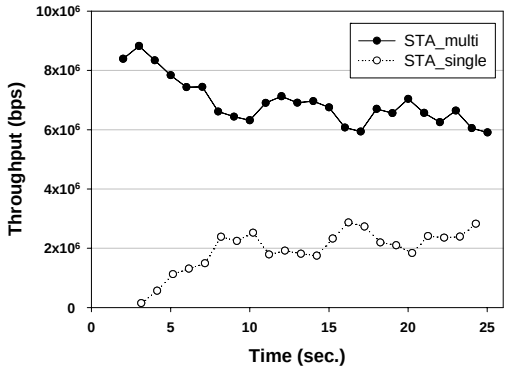

그림 2. STAmulti와 STAsingle의 전송 throughput 비교

유 사용자들이 네트워크 트래픽의 75%를 차지하는 심각한 불공평성 문제가 발생한다[7].

다수의 병렬 TCP flow들을 이용하는 것은 인터넷 개발 초기에는 예상치 못했고, TCP의 congestion control 역시 이를 고려하지 못하고 설계되었다. Van Jacobson이 제안한 Additional Increase Multiplicative Decrease (AIMD)[9] 방식의 TCP congestion control은 링크를 공유하는 스테이션들이 동일한 개수의 TCP flow들을 가지고 있을 경우에만 공평하게 대역폭을 분배할 수 있다.

본 논문에서는 이러한 대역폭 독점 문제에 대해 random early detection (RED)[10]을 수정한 history-aware RED (HRED)를 제안한다. RED는 원래 TCP flow들 간의 공평성을 향상시키기 위해 제안된 것으로 이에 대해서는 2장에서 자세히 설명한다. 제안하는 HRED 방식은 기존 RED와 유사하지만 링크를 통과하는 패킷의 drop 확률을 결정할 때, 스테이션별 링크 사용빈도를 반영하도록 하여 스테이션 수준에서의 대역폭 사용 공평성을 지원한다.

본 논문은 다음과 같이 구성된다. 2장에서는 RED를 소개하고, RED만을 적용했을 때 여전히 공평성 문제가 해결되지 않음을 보인다. 3장에서는 HRED 방식을 제안한다. 4장에서는 시뮬레이션을 통해 HRED의 공평성 지원 성능을 평가한다. 마지막으로 5장에서 결론짓는다.

Ⅱ. Random Early Detection

Random early detection(RED)은 능동적인 큐 관리 방식 중의 하나이다[7]. 기존의 drop-tail같은 수동적인 방식은 큐가 다 찰 때까지 기다렸다가 이후에 도착하는 패킷들을 drop한다. 반면 RED는 큐가 다 차지 않아도 확률에 따라 패킷들을 drop할 수도 있다. 이 때 drop 확률은 큐의 길이에 비례한다.

RED의 자세한 동작은 아래와 같다. 큐의 길이에 대해 최소와 최대 threshold, $Q_{\min}$과 $Q_{\max}$을 설정한다. 그리고 평균 큐 길이 q_{len}를 이들 threshold와 비교하여 패킷 drop확률 P_{drop}을 계산한다. 이 때 threshold와 q_{len}는 큐 안에 저장된 패킷 개수를 단위로 한다. P_{drop} 계산에서 $q_{len} < Q_{\min}$인 경우에는 P_{drop}= 0로 하여 도착하는 패킷을 drop하지 않는 반면, $q_{len} > Q_{\max}$경우에는 P_{drop}= 1로 하여 패킷을 drop한다. 그리고 $Q_{\min} \le q_{len} \le Q_{\max}$ 경우에는 다음과 같이 P_{drop}을 계산한다.

$$P_{drop} = P_{\max} \frac{q_{len} - Q_{\min}}{Q_{\max} - Q_{\min}} \qquad (1)$$

여기에서 $P_{\max}$는 패킷의 최대 drop확률로 미리 설정해 놓은 값이다. q_{len}는 과거 큐 길이와 현재 큐 길이를 고려하여 exponentially weighted moving average 방식으로 다음과 같이 계산된다.

$$q_{len} = (1 - q_w)\, q_{len} + q_w\, q_{cur}$$

여기에서 q_w는 평균 가중치이고, q_{cur}은 현재 큐 길이이다.

RED는 공평성 지원에서 효과적이지만, 그림 3과 같은 상황에서 발생하는 대역폭 독점문제에 대해서는 효과적이지 못하다. 라우터 A의 큐에 RED를 적용하고 시뮬레이션을 실행하였다. 사용된 RED는[7]에서 제안된 알고리즘을 구현한 것으로, $Q_{\min}$과 $Q_{\max}$는 각각 5와 15, $P_{\max}$는 0.02, 그리고 q_w는 0.002로 하였다. 시뮬레이션에서는 STA_{multi}가 사용하는 TCP flow의 개수를 5개와 10개인 경우 두 가지 상황을 실험하였다. 두 상황 모두에서 STA_{single}은 항상 1개의 flow만을 사용한다. 이들 상황을 각각 5:1, 10:1로 표시하였다. 나머지 시뮬레이션 환경은 앞선 실험과 동일하다.

그림 3은 측정된 throughput을 RED를 적용하지 않은 경우를 Normal, 적용한 경우를 RED로 각각 표시한다. Normal인 경우 STA_{single}의 throughput은 5:1과 10:1 상황에서 각각 1.9 Mbps와 0.36 Mbps로 STA_{multi}의 6.9 Mbps와 8.46 Mbps에 비해 현저하게 낮아 불공평한 대역폭 독점상황임을 알 수 있다. RED를 적용한 경우 STA_{single}의 throughput이

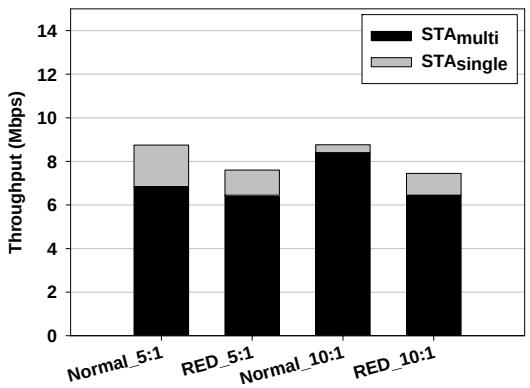

그림 3. 그림 1 상황에 RED를 적용한 결과

각각 1.15 Mbps와 1.0 Mbps로 변화되었다. STA_{single}의 throughput에 약간의 변화가 있었지만 여전히 STA_{multi}가 대역폭의 대부분을 독점하고 있어 RED가 효과적인 해결책이 되지 못함을 알 수 있다. 5:1 경우에는 STA_{single}의 throughput이 RED를 적용할 때 오히려 Normal경우보다 40% 감소하였다. 이것은 RED 적용으로 인한 오버헤드로 전체 throughput이 감소하고, 또한 flow들의 대역폭 사용량이 균등해지는 과정에서 비정상적으로 높았던 STA_{single}의 throughput이 줄어들었기 때문이다.

Ⅲ. 제안방식: History-aware RED

제안하는 history-aware RED (HRED)는 기존 RED를 수정하여 전송대역폭을 독점하는 스테이션을 검출할 수 있고, 이를 이용해 해당 스테이션의 송수신 패킷에 대해 높은 drop확률을 적용함으로써 대역폭 독점 문제에 대응할 수 있다. RED는 임의 drop방식을 사용하기 때문에 대역폭을 독점하는 스테이션에 대해 차별적인 패널티를 부과할 수 없었다.

HRED는 RED의 식(1)에 의해서 계산된 패킷의 drop확률 P_{drop}를 스테이션별 링크 점유율을 반영하여 조정한다. 예를 들어, 링크 점유율이 높을수록 P_{drop}을 증가시키는 방식으로 대역폭 독점을 완화시킨다.

스테이션별로 링크 점유율을 정확히 계산하는 것은 소요자원 오버헤드가 크기 때문에, HRED에서는 추정치를 계산하여 이용한다. 이를 위해 히스토리 리스트를 이용한다. 이것은 도착하는 패킷의 송신 혹은 수신 IP 주소를 FIFO방식으로 저장하는 유한 크기의 공간이다. HRED 방식으로 동작하는 노드의 (라우터, 액세스포인트, 스테이션 등이 노드가 될 수 있다) 큐에 패킷이 도착하면 IP 주소를 추출해서 히스토리 리스트에 저장한다. 송신 혹은 수신 IP 주소 중 어느 것을 저장할 지는 대역폭 독점을 막고자 하는 경우에 따라 달라진다. 본 논문에서는 수신 스테이션에 의한 대역폭 독점을 막고자 하기에 수신 IP 주소를 저장하는 경우를 설명한다. 송신 IP 주소의 경우에도 동일하게 처리된다.

히스토리 리스트에 패킷의 IP 주소를 저장하는 세부내용은 다음과 같으며, 앞서 설명한 RED 방식의 q_{len}, $Q_{\min}$, $Q_{\max}$를 여기서도 사용한다. 패킷 P가 도착했을 때 $q_{len} \geq Q_{\min}$ 경우에는 패킷의 수신 IP 주소를 추출하여 히스토리 리스트의 끝에 추가한다. 저장 공간이 없을 경우에는 FIFO방식으로 가장 먼저 저장되었던 IP 주소를 삭제한 후 추가한다. 반대로 $q_{len} < Q_{\min}$ 경우는 링크 대역폭 사용에 여유가 있다는 의미이며, 이때는 대역폭을 독점하는 스테이션을 검출할 필요가 없기 때문에 히스토리 리스트에 저장된 내용을 모두 삭제해 초기화시킨다.

일단 히스토리 리스트에 IP주소 정보에 대한 저장이 끝나면, 패킷 P에 적용할 drop확률 P_{drop}을 계산한다. RED와 동일하게 $q_{len} > Q_{\max}$이면 P_{drop}= 1, $q_{len} < Q_{\min}$이면 P_{drop}= 0이 된다. 그리고 $Q_{\min} \leq q_{len} \leq Q_{\max}$인 경우에는 다음과 같이 계산한다.

$$P_{drop} = P_{red} + kP_{red}(e^{\eta - \tau} + P_{red} - 1) \quad (2)$$

여기에서 P_{red}는 기존 RED 방식의 식(1)에 의해서 계산된 drop확률이고, η $(0 \leq \eta \leq 1)$는 히스토리 리스트 내에서 패킷 P의 IP 주소가 차지하는 점유율이다. 이것은 히스토리 리스트에 저장된 IP 주소들 중 패킷 P의 수신 IP 주소와 일치하는 것들의 개수를 히스토리 리스트 전체 개수로 나눈 값이다. 따라서 η는 수신 IP 주소에 해당하는 스테이션이 점유하는 대역폭 비율에 대한 추정치가 된다. 그리고 τ는 히스토리 리스트 점유율에 대한 임계값이다. η가 τ보다 큰 경우에는 P_{drop}을 증가시키고, 그렇지 않을 경우에는 감소시킨다. 링크 대역폭을 독점하는 스테이션일수록 임계값 τ 보다 높은 η값을 가지게 되므로 선택적으로 drop확률을 증가시킬 수 있게 된다. 마지막으로 k는 상수로서 P_{drop} 변화의 폭을 조절한다.

히스토리 리스트를 이용한 점유율 추정치에 대한 정확도 요구수준은 확률에 의한 패킷 폐기라는 점을 고려하면 근사치만으로도 만족시킬 수 있다. 위에서 설명한 히스토리 리스트 방식을 사용하면 congestion 징후 시점에서의 순간 점유율을 효과적으로 계산할 수 있으며, 이 결과의 유효성은 4장의 실험결과에서 확인할 수 있다.

이렇게 계산된 P_{drop}을 이용하면 HRED방식은 대역폭 점유율에 따라 그림 4와 같이 차등적인 패킷 drop확률을 적용을 할 수 있다.

그림 4는 히스토리 리스트 점유율 η의 변화에 따른 RED와 HRED의 P_{drop} 확률을 각각 X축과 Y축에 보여준다. HRED의 식(2)에서 k와 τ는 각각

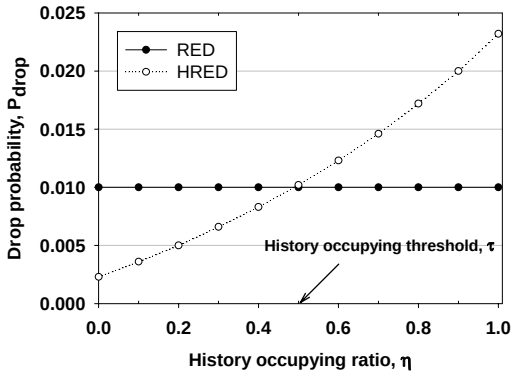

그림 4. HRED방식에서 히스토리 리스트 점유율 η에 따른 패킷 drop확률의 변화

2.0과 0.5로 설정하였으며 기존 RED방식에 의해 계산된 P_{red}값이 0.01이라고 가정한다. 우선, RED의 경우에는 η를 고려하지 않기 때문에 고정된 P_{drop}값 0.01을 갖는다. 하지만 HRED에서는 τ (=0.5) 값을 경계로 η값이 더 클 경우에는 P_{drop}이 RED경우보다 크고, 반대의 경우에는 작다. 따라서 HRED는 대역폭 점유율에 따라 차등적인 P_{drop}을 적용함을 알 수 있다.

그림 5는 HRED의 식(2)에서 임계값 τ값 설정이 P_{drop}에 끼치는 영향을 보여준다. τ는 P_{drop} 계산과정에서 대역폭 점유율에 따른 패널티를 부과하기 위한 기준 값이다. 여기서는 η값을 0.5로 고정하였고, X축은 q_{len}을 나타내며, Y축은 그에 따른 P_{drop}을 나타낸다. τ값을 낮출수록 독점 스테이션으로 검출되기 쉽기 때문에 P_{drop}이 높아지게 된다. 따라서 HRED는 τ값 조정으로 대역폭을 독점하는 스테이션 검출 정도를 조절할 수 있다.

Ⅳ. 시뮬레이션을 통한 성능 검증

본 장에서는 시뮬레이션을 통해 HRED의 성능을 검증한다. 비교대상은 아무런 조치를 취하지 않은 것과 RED를 적용한 것으로, 각각을 normal, RED, 그리고 HRED로 표시한다.

시뮬레이션은 Qualnet 4.0을 이용하여 그림 1에서와 동일한 환경을 구축하였다. RED의 경우, 큐 사이즈는 50 KByte이고 나머지 설정은 앞선 시뮬레이션과 동일하다. HRED의 경우에 τ는 0.85, k는 2.0으로 하였다.

그림 6은 STA_{multi}의 flow 갯수를 5개부터 40개까지 증가시키면서 STA_{single}의 수신 throughput을 측정한 결과이다. Normal, RED, HRED 경우에 대해 각각 측정하였다. 우선, normal의 경우, flow의 갯수가 5개일 경우 1.9 Mbps의 throughput에서 10개 일 때 0.365 Mbps, 20개 0.08 Mbps 식으로 급격히 감소하여 40개일 때 0.020 Mbps로 대폭 축소되었다. 이는 STA_{multi} flow들의 패킷들이 router queue를 lock-out하기 때문이다. RED의 경우도 normal과 비슷한 급격한 감소를 보인다. 이는 RED의 drop 방식이 공평성을 고려하지 않고 고르게 모든 flow들의 패킷을 drop하기 때문이다. 반면, HRED를 적용했을 경우에는 flow 갯수가 증가하더라도 약 2.0 Mbps이상의 throughput이 보장됨을 알 수 있다. 이는 HRED가 공평성을 고려하여 스테이션별로 차별적인 drop 확률을 적용하기 때문이다.

그림 7은 앞선 그림 6과 동일한 시뮬레이션 상황에서 STA_{multi}의 throughput을 측정한 것이다.

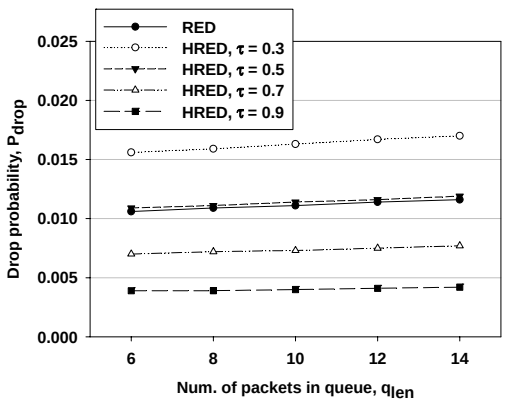

그림 5. HRED방식에서 히스토리 리스트 점유율 임계치 τ의 변화에 따른 drop확률의 변화

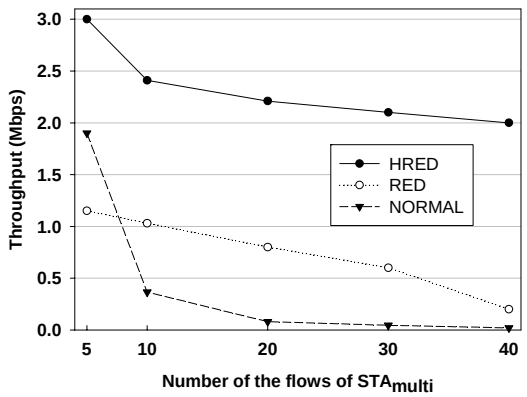

그림 6. STAsingle의 throughput 변화: STAmulti의 flow 수를 5 ~ 40개로 증가

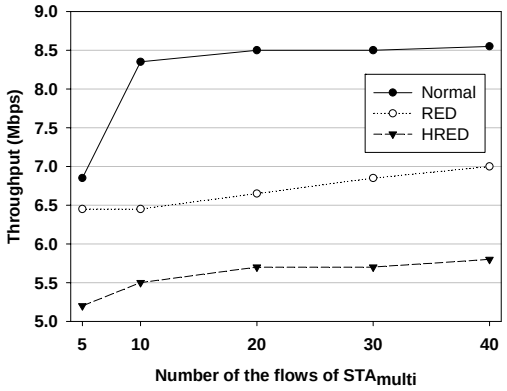

그림 7. STAmulti의 throughput 변화: STAmulti의 flow 수를 5 ~ 40개로 증가.

Normal과 RED의 두 경우 모두 throughput이 증가함을 알 수 있다. 특히 normal의 경우, flow가 5개 일 때 6.85 Mbps에서 40개 일 때 8.55 Mbps로 급격히 증가하였으며 이후에는 완만하게 증가한다. RED의 경우, flow가 5개 일 때 6.45 Mbps에서 40개 일 때 7.0 Mbps로 완만히 증가하였다. 이와 같이 두 경우 모두 대부분의 throughput을 STA_{multi}가 차지하여 다수 flow 사용에 의한 불공평성이 발생한다. 반면 HRED의 경우, flow가 5개 일 때 5.2 Mbps에서 40개 일 때 5.8 Mbps로 STA_{multi}의 throughput이 일정 수준이하로 제한되는 것을 볼 수 있다. 그림 6과 7을 통해 HRED가 스테이션 수준의 공평성을 지원하는 데 효과적임을 알 수 있다.

다음 실험에서는 STA_{multi}타입의 스테이션이 다수인 상황에서 HRED 성능을 검증한다. 이를 위해 STA_{multi} 타입 스테이션을 3개로 늘리고 각각이 5개의 flow를, STA_{single}은 1개의 flow를 갖도록 하였다. 나머지 시뮬레이션 환경은 이전과 동일하다.

그림 8에서 STA_{single}의 throughput은 HRED를 적용할 때 0.72 Mbps로 normal이나 RED의 0.3 Mbps, 0.45 Mbps 경우보다 각각 140%와 60% 증가하였다. 전체 throughput에 있어서는 HRED는 7.78 Mbps로 normal의 8.8 Mbps의 88.4%에 해당해 크게 떨어지지 않으며, RED의 5.98 Mbps 보다는 30.1% 증가하였다. 이를 통해, HRED 방식이 STA_{multi} 타입 스테이션들이 다수인 상황에서도 효과적으로 스테이션간 공평성을 유지할 수 있음을 알 수 있다. 결과에서 주목할 만한 특징은 HRED의 경우 STA_{multi} 스테이션들의 throughput이 normal이나 RED보다 훨씬 균등해져서 각각 2.53, 2.38, 2.15 Mbps를 차지하는 것을 볼 수 있다. 이는

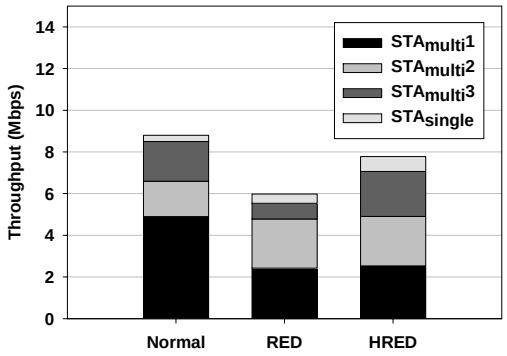

그림 8. STAmulti와 STAsingle이 각각 3개와 1개 상황에서 평균 throughput 비교

HRED가 TCP flow간 공평성 유지 측면에서도 RED보다 우수함을 보여준다.

또 다른 실험에서는 앞 실험과 반대로 STA_{single} 타입 스테이션이 다수인 상황에서 HRED의 성능을 검증한다. TCP flow를 1개만 가지는 STA_{single}타입의 스테이션 개수를 5개로 늘이고, 5개의 flow를 가지는 STA_{multi} 타입의 스테이션을 1개로 하였다. 나머지 시나리오 환경은 앞선 실험들과 동일하다.

그림 9에서 STA_{multi}의 throughput은 HRED 방식을 적용할 때 3.0 Mbps로 normal이나 RED의 6.3 Mbps, 4.0 Mbps 경우보다 52.3%와 25% 각각 감소하였다. 이는 HRED 방식이 다수의 STA_{single} 타입 스테이션들 가운데서도 STA_{multi} 스테이션을 효과적으로 검출할 수 있음을 보여준다. HRED 적용할 때의 전체 throughput은 7.99 Mbps로 normal의 8.72 Mbps의 91.6%에 해당해 크게 떨어지지 않으며, RED의 7.6 Mbps보다는 4.3% 증가하였다.

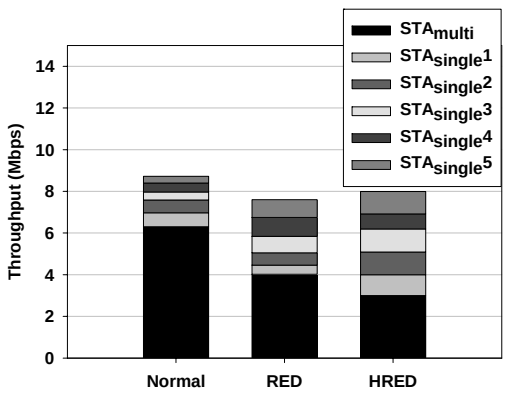

그림 9. STAmulti와 STAsingle이 각각 1개와 5개 상황에서 평균 throughput 비교

V. 결 론

다수의 병렬 TCP flow들을 가진 스테이션에 의한 불공평한 대역폭 독점 문제에 대해 RED를 수정한 HRED를 제안하였다.

제안방식의 효과적인 적용범위는 로컬 서브넷의 edge 게이트웨이 라우터가 될 것이다. 실제 다수 flow 사용에 의한 불공평성을 야기하는 소수 사용자 스테이션에 위치적으로 근접하여 효과적으로 트래픽을 조절할 수 있기 때문이다.

참 고 문 헌

[1] T. Henderson, E. Sahouria, S. McCanne, and R. Katz, "On Improving the Fairness of TCP Congestion Avoidance," In Proc. of Globecom 1998.

[2] N. Blefar-Melazzi, A. Detti, I. Habib, A. Ordine, S. Salsano, "TCP Fairness Issues in IEEE 802.11 Networks: Problem Analysis and Solutions Based on Rate Control," IEEE Transactions on Wireless Communications, Vol. 6, No. 4, April 2007.

[3] D. Leith, P. Clifford, D. Malone, A. Ng, "TCP Fairness in 802.11e WLANs," IEEE Communications Letters, Vol. 9, No. 12, December 2005

[4] J. Ha, C. Choi, "TCP Fairness for Uplink and Downlink Flows in WLANs," in Proc. IEEE Globecom 2006

[5] S. Pilosof, R. Ramjee, Y. Shavitt, P.Sinha, "Understanding TCP fairness over wireless LAN," in Proc. IEEE INFOCOM 2003

[6] B. Briscoe, "A Fairer, Faster Internet," IEEE Spectrum, Dec. 2008.

[7] Qualnet. Network simulator. Available at http://www. scalable-networks.com

[8] B. Cohen, "BitTorrent - a new P2P app". Yahoo eGroups.

[9] Van Jacobson, "Congestion Avoidance and Control," In Proc. of ACM SIGCOMM '88, Aug. 1988

[10] S. Floyd and V. Jacobson, "Random early detection gateways for contestion avoidance," IEEE/ACM Transactions on Networking. 1(40, Aug. 1993

[11] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," ACM SIGCOMM Computer Communications Review, vol. 19, no. 4, pp. 1-12, 1989

[12] M Shreedhar and G. Varghese,''Efficient fair queueing using deficit round robin," in Proc. of ACM SIGCOMM, 1995

[13] S. Athuraliya, S. Low, V. Li and Q. Yin, "REM: Active queue management," IEEE Network, vol. 15, 2001

[14] C. Hollot, V. Misra, D. Towsley, and W. Gong, "On designing improved controllers for AQM routers supporting TCP flows," in Proc. of IEEE INFOCOM, 2001

[15] M. Shin, S. Chong, and I. Rhee, "Dual-Resource TCP/AQM for Processing-Constrained Networks," IEEE/ ACM Transactions on Networking. April 2008.

[16] Cisco Systems Document, "Class-based Weighted Fair Queueing"

[17] X. Lin, K. Zhou, H. Wang, and G. Su, "Scalable Fair Random Early Detection," in Proc. of WiCOM 2006.

전 경 구 (Kyungkoo Jun) 정회원

한국통신학회 논문지 제31권 제4B호 참조
현재 인천대학교 멀티미디어시스템공학과 조교수