□기 고□

비선형 확률함수를 가지는 RED 알고리즘†

동국대학교 박종석 · 최병석
한국외국어대학교 정대인
한국통신 이봉영
명지대학교 박현민 · 류성진

1. 서 론

TCP(Transmission Control Protocol) 체증제어의 주목적은 송신 단말의 전송률을 직접 제어하여 체증으로 인해 손실된 데이터를 재전송하기 위함이다. TCP는 OSI 참조 모델의 4계층에 속하는 프로토콜로 송신 단말과 수신 단말사이의 노드의 수에 상관없이 종단간 동작한다. TCP는 송신 단말과 수신 단말을 양끝으로 하는 하나의 루프를 형성하여 이 루프에 수신 측이 전송하는 Ack(acknowledge) 정보와 윈도우 그리고 타임아웃 기능을 이용하여 체증제어 메커니즘을 구현하고 있다. 이때 송신단이 패킷의 손실을 인식하기 위해서는 손실된 패킷의 재전송 타임아웃(RTO, Retransmission Time Out)까지 기다려야 한다. 체증 빈도에 따라 차이가 있지만 일반적으로 RTO는 RTT(Round Trip Time)보다 최소 2배 이상임으로 전송 단말이 체증을 인식하고 체증에 응답하기 위해서는 오랜 시간이 소요되어 네트워크의 체증이 더욱 심각해진다. 체증상태가 장시간 유지되면 병목구간의 라우터들은 체증 회피 메커니즘에 따라 현재의 네트워크 사용을 감소시키기 위해 입력 패킷들을 모두 폐기한다. 손실된 패킷의 전송 단말들은 RTO 후 체증윈도우(*cwnd*, congestion window)를 1로 줄이고 슬로우-스타트를 실행한다[2].

종단간 경로가 체증된 라우터를 경유하는 모든 전송 단말들이 패킷 손실에 응답하여 동시에 *cwnd*를 1로 줄이고 슬로우-스타트를 실행하면 링크의 사용량이 갑자기 줄어든다. 그러나, TCP의 슬로우-스타트에 의해 곧 링크 사용량이 증가하여 체증을 일으킨다. 계속되는 체증에 의해 라우터는 주기적으로 모든 패킷을 폐기하여 TCP 전역 동기화(global synchronization) 현상을 일으킨다[3]. 이와 같은 현상이 반복되면 링크 및 라우터의 자원 효율이 떨어져 각 전송 단말의 수율(throughput)은 떨어지고, 지연(delay)은 증가하여 사용자가 원하는 서비스 품질(QoS)을 만족해 주지 못한다.

RED(Random Early Detection) 알고리즘은 체증제어와 체증회피를 제공하기 위해 제안된 방식이다[4]. RED 게이트웨이는 버퍼의 평균 큐(queue) 길이를 이용하여 체증을 초기에 감지한다. 게이트웨이는 체증된 연결에게 게이트웨이에서 입력 패킷을 폐기하거나 또는 패킷 헤더의 특정 비트를 설정함으로써 체증을 알려준다. 평균 큐의 길이가 설정된 한계값을 초과할 때, 게이트웨이는 특정 확률로 각 입력 패킷을 마크하거나 폐기한다. 이때, 확률은 평균 큐 길이의 선형함수로써 구해진다. RED 게이트웨이는 약간의 버스티한 패킷들의 입력을 허용하면서 낮은 큐 길이를 유지한다. 체증동안 마크 확률은 게이트웨이를 통과하는 각 연결의 대역 공유에 비례하여 그들의 윈도우를 감소시키기 위해 사용된다.

RED 알고리즘은 단순한 평균함수와 선형 확률함수를 사용하여 구현이 단순하고, 명시적 또는 묵시적으로 체증상황을 전송 단말에 알려줄 수 있기 때문에

† 본 고는 한국통신의 2000년도 HAN/B-ISDN NTB 탁연구 과제로 수행되었습니다.

TCP 프로토콜을 사용하는 현재의 네트워크와 가장 쉽게 연동할 수 있다. 그러나 RED 알고리즘에서 사용하는 매개변수들은 실험을 통해 최적화된 값들로 네트워크의 급격한 변화에 쉽게 적응시킬 수 없는 값이다. RED의 확률함수 역시 단순한 선형함수를 사용함으로 급격한 네트워크 변화에 대응하지 못하는 단점이 있다.

이 글에서는 이러한 RED 알고리즘의 근본적인 문제점을 해결하기 위해 기존의 선형 확률함수를 비선형 확률함수로 대처하여 급변하는 네트워크의 상황에 유연하게 대처하도록 하고, ARED(Adaptivc RED)에서와 같이 매개변수들은 재계산을 통한 네트워크 적응기법을 사용하는 대신[5], 비선형 확률함수의 값을 네트워크 환경에 따라 변화하도록 하여 실험적으로 최적화된 매개변수들의 재계산하는 복잡도를 없앴다.

이 글은 2장에서 RED 알고리즘에 대한 분석과 RED 알고리즘의 문제점을 자세히 지적하고 이를 극복하기 위한 노력들을 살펴본다. 3장에서는 현재까지 제시된 많은 해결방법과 다른 본 논문의 관점에 대해 설명하고 새로운 알고리즘 제시한다. 이를 바탕으로 4장에서는 RED 성능측정을 위해 가장 많이 이용되고 있는 Ns 시뮬레이터를 통해 제안된 알고리즘의 성능을 비교, 분석한다. 마지막으로 결론에서는 제시된 알고리즘의 실험결과를 통해 개선할 점을 지적한다.

2. RED 알고리즘

RED는 버퍼에서의 평균 큐 길이(Q_{avg})를 이용하여 네트워크 체증제어와 체증회피를 제공한다[4]. RED는 네트워크 체증을 전송 단말에게 알려주기 위하여 패킷의 특정비트(체증 알림 비트, congestion indication bit)를 정하여 Q_{avg}가 정해진 한계값(*threshold*)를 넘는 경우 랜덤하게 마크 또는 패킷을 폐기한다. 종단 단말에서는 마크된 패킷의 비트를 Ack 패킷에 복사하여 전송 단말에 전송한다. 이를 수신한 전송 단말은 체증 알림 비트가 설정되어 있으면 슬로우-스타트 한계값(*ssthreshold*)을 현 체증윈도우(*cwnd*)를 1로 한 후 슬로우-스타트를 실행한다[2,3,4].

RED 게이트웨이는 패킷이 도착할 때마다 평균 큐의 길이를 구하여 정해진 최소 큐 한계값(min_{th})과 최대 큐 한계값(max_{th})을 비교한다. Q_{avg}가 min_{th}보다 작을 경우에는 네트워크가 정상임을 나타내므로 모든 패킷들을 마크하지 않고 큐에 저장하고 서비스한다. Q_{avg}가 min_{th}과 max_{th} 사이에 있을 경우에는 아직 체증은 발생하지 않았지만 체증회피를 위해 마크 확률 p_a을 구하여 p_a에 의해 랜덤하게 마크 또는 폐기한다. p_a는 Q_{avg}의 함수로써 Q_{avg}가 증가하면 p_a도 증가한다. 이때 랜덤하게 마크를 하더라도 전체 대역폭에서 특정 사용자의 대역폭에 비례하여 마크되므로 많이 보내는 전송단이 적게 보내는 전송단보다 마크될 확률이 높다.

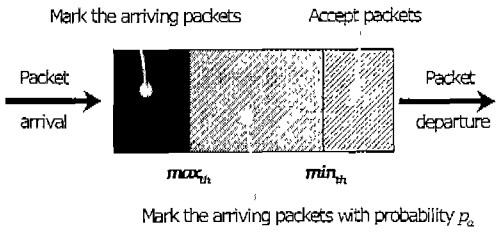


그림 1 RED 큐의 상태

RED 게이트웨이는 Q_{avg}를 계산하기 위하여 저역필터의 일종인 EWMA(Exponential Weighted Moving Average)를 사용한다. EWMA를 사용하면 순간적인 체증 또는 버스티 트래픽으로 인한 짧은 시간동안의 큐 길이 증가가 Q_{avg}의 값에 큰 영향을 주지 않는다. Q_{avg}는 식 (1)과 같이 구해진다[4].

$$Q_{avg} \leftarrow (1-w_q)Q_{avg} + w_q q \qquad (1)$$

w_q는 큐 가중치로 과거 Q_{avg}가 현재 Q_{avg}를 구하는데 있어서 어느 정도의 영향을 주는가를 결정하는 상수이다. 즉, w_q가 큰 값을 가지면 현재 Q_{avg}는 과거 Q_{avg} 보다는 현재 큐 길이 q에 많은 영향을 받지만, 작은 값을 가질 경우에는 과거 Q_{avg}에 영향을 많이 받으므로 일시적으로 큐 길이가 증가하여도 Q_{avg}는 크게 증가하지 않게 된다.

w_q을 결정함에 있어서 주의해야 한다. w_q가 작으면 과거 Q_{avg}가 현재 Q_{avg}를 구하는데 큰 영향을 주어 일시적인 큐 길이 증가에 효율적으로 동작할 수 있다. 그러나 Q_{avg}가 max_{th}를 초과하여 게이트웨이가 입력 패킷 전체를 마크할 때 w_q가 작으면 큐 길이가 줄어들어도 Q_{avg}는 감소하는 비율이 너무 느려 큐 길이가 max_{th} 보다 적어도 Q_{avg}가 max_{th} 보다 크기 때문

에 계속 마크를 한다. 반면에 w_q가 크면 큐 길이 변화에 Q_{avg}도 빠르게 변화하지만, 일시적인 큐 길이에 같이 변화하므로 일시적인 버스티 트래픽을 수용하지 못한다.

min_{th}, max_{th} 값의 선택은 네트워크의 특성에 따라 결정될 수 있다. min_{th}가 너무 작으면 Q_{avg}가 작은 값을 가지는데도 Q_{avg}가 min_{th} 보다 크게 되므로 RED 게이트웨이는 입력 패킷들을 랜덤하게 마크한다. 이때 Q_{avg}가 적기 때문에 마크 확률도 작아 RED 게이트웨이가 실제로 마크하는 횟수는 많지 않겠지만 게이트웨이 자원과 마크된 패킷에 응답하는 전송 단말의 수율 저하를 초래할 수 있다. 그러나 min_{th}가 너무 큰 값을 갖게 되면, RED 게이트웨이의 목적인 체증회피를 제공하는데 문제가 발생한다. 이는 네트워크가 체증상태에 임박하였음에도 RED 게이트웨이의 min_{th}가 Q_{avg} 보다 크기 때문에 마크하지 않고 버퍼에 저장되어 서비스되어 진다. 패킷이 RED 게이트웨이에서 마크되어도 RTT 이후 전송 단말이 슬로우-스타트를 실행하기 때문에 min_{th}가 너무 크면 체증회피 메커니즘이 적용되는 구간이 짧아 네트워크는 체증상태로 빠져들게 된다. 이와 마찬가지로 max_{th}의 선택도 매우 중요하다. Q_{avg}가 max_{th}를 초과하면 입력 패킷 모두를 마크하기 때문에 너무 작은 값을 가지면 할당된 버퍼공간을 사용하지 못하게 되므로 버퍼공간의 효율성이 떨어지고 너무 큰 값을 갖게 되면 DropTail 게이트웨이와 같은 동작을 수행하므로 max_{th} 값의 선택이 매우 중요하다. 그림 2에서는 RED 알고리즘에 최적화된 적절한 매개변수들의 값을 도식하였다[6].

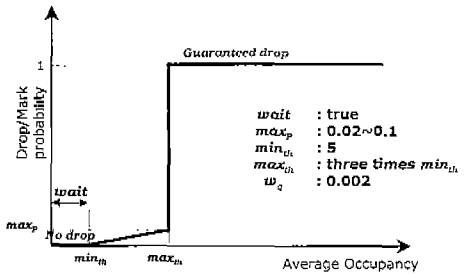


그림 2 RED 매개변수 권고치

3. 관련연구

RED 게이트웨이의 첫 번째 문제점은 체증상태가 아님에도 불구하고 소량을 전송하는 송신 단말의 패킷을 마크할 수 있다는 것이다. RED 게이트웨이는 각 단말이 사용하는 대역에 비례하여 마크하기 때문에 많이 전송하는 단말은 그만큼 마크될 확률도 증가된다. 이때 Q_{avg}값을 이용하여 p_a를 결정하기 때문에 마크 확률은 단지 Q_{avg}에 의존한다. 따라서 Q_{avg}의 증가에 의해 전송량이 적은 단말도 낮은 확률이지만 마크되어진다. 적은 양을 전송하는 단말이 마크되면, RED 게이트웨이는 *count*를 0으로 재설정하여 다음 도착하는 패킷의 마크를 결정하기 때문에 많이 전송하는 전송 단말의 마크 기회가 줄어든다. 이렇게 되면 연결간의 공정성, 지연 등을 보장하지 못하게 된다.

FRED(Flow RED)[7]에서는 첫 번째 문제점에 대한 해결책을 제시하고 있다. FRED 게이트웨이는 서로 다른 체증에 반응하는 패킷들의 흐름을 세 가지로 재정의 하고 이들 흐름의 혼합된 형태로 게이트웨이를 통과할 때 각각에 따른 서로 다른 RED 알고리즘을 적용하는 방식이다. FRED에서 구분 흐름은 다음과 같다.

- ▶ 적응성이 없는 흐름(Nonadaptive flows)- 패킷의 폐기를 무시할 수 있는 전송계층의 프로토콜들
- ▶ 강건한 흐름(Robust flows)-짧은 RTT를 가짐으로 폐기된 패킷을 빠르게 복원할 수 있는 TCP 연결들
- ▶ 허약한 흐름(Fragile flows)-긴 RTT를 가짐으로 폐기된 패킷의 복원에 느리게 대응하는 TCP 연결들

FRED는 복잡한 흐름별 큐잉이 필요하지 않으며 구별하는 흐름의 종류가 단순하여 구현이 용이한 편이다. 그러나 FRED 메커니즘을 사용하는 라우터는 FRED의 흐름의 정의에 따라 라우터로 들어오는 흐름을 새롭게 구별지어야 한다. 또한 각 흐름에 맞는 RED 매개변수들의 값을 유지해야 함으로 다른 RED 버전들에 비해 상대적으로 복잡도가 더욱 높다.

RED 게이트웨이의 두 번째 문제는 매개변수들의 값을 결정하는 정확한 알고리즘이 없다는 것이다. 앞서 w_q, min_{th}와 max_{th} 값을 어떻게 정해지는가에 따라서 RED 알고리즘은 패킷들을 공격적으로 마크할 수도 있고 소극적으로 마크할 수도 있다는 것을 설명하였다. 그러나 라우터의 일반적인 패킷 손실률과 버퍼의 크기 최소화 등을 고려하여 min_{th}와 max_{th} 값의 선

택하였다 하더라도 w_q값의 설정은 입력 트래픽의 특성과 RED 게이트웨이의 구현된 위치마다 다르게 작용해야 그에 따른 Q_{avg}의 변화율도 달라진다. 그러나 고정된 w_q값을 사용하면 체증상황에서 능동적으로 대처할 수 없다.

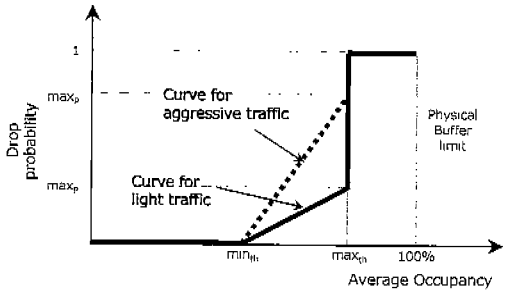


그림 3 ARED의 마크확률

ARED(Adaptive RED)는 기존 RED 게이트웨이의 w_q값에 따른 Q_{avg}의 변화로는 체증상황에서 적극적이지 못하고 RED 게이트웨이에 연결된 TCP 흐름의 수가 증가함에 따라 공격적으로 마크하지 못한다는 것에 착안하여 최근의 체증내력에 따라 매개변수들을 수정한다. ARED는 Q_{avg}의 최근 움직임에 근거하여 max_p를 조절함으로써 마크 확률 함수값을 조절하도록 한다.

ARED 게이트웨이는 Q_{avg}가 max_{th}를 넘어서게 되면 입력 패킷들을 공격적으로 마크하기 위하여 max_p를 증가시킨다. Q_{avg}가 max_{th} 주위에서 진동하게 되면 패킷 폐기만으로는 충분히 공격적이지 못하다고 판단하여 계속적으로 max_p를 증가시킨다. 반대로 Q_{avg}가 min_{th} 아래로 떨어지게 되면 max_p를 감소시킨다[5]. 이러한 ARED의 동작을 그림 3에 도식하였다.

ARED는 TCP 흐름들에 대한 어떠한 외부적인 정보 없이도 ARED 게이트웨이를 통과하는 TCP 흐름의 증감에 따라 큐의 부하량을 조절할 수 있다는 장점이 있다. 그러나 계속적으로 체증 내역을 추적하고 계산해야 함에 따라 알고리즘의 복잡도가 증가되었으며, max_p의 증감 정도를 어떻게 정의하느냐는 것은 또 하나의 매개변수가 될 수 있다.

RED 게이트웨이의 일반적인 문제점과 그에 해결책은 기존 RED 알고리즘보다 우수한 성능을 보이지만 구현의 복잡도를 증가시키고 매개변수들을 변화시킴으로써 또다시 각 해결책에 적합한 매개변수값을 제시해야 한다는 문제점을 안고 있다.

4. 제안된 알고리즘

RED 알고리즘의 마크 확률은 min_{th}와 max_{th} 구간에서 확률 0과 max_p를 잇는 선형함수(linear function)이다. 이는 RED 알고리즘에서 패킷을 마크하는 구간이 전체 큐 길이에 비해 짧기 때문에 체증회피 알고리즘의 단순화를 위하여 바람직한 접근법이라 할 수 있다. 일반적인 라우터의 패킷손실률은 2%에서 최대 10%미만이다. 이에 맞추어 RED는 최대 마크확률 max_p을 선택하고 평균 큐의 길이를 최대한 짧게 유지하기 위해 min_{th}을 낮게 설정하고 있다[6].

이와 같은 매개변수들의 선택은 많은 실험과 연구결과들을 통해 입증되고 있어 앞서 서술한 RED 알고리즘의 단점을 극복하기 위해 매개변수들을 조정하는 것은 바람직하지 못하다고 할 수 있다. 본 논문에서는 RED 알고리즘의 장점을 최대한 유지하기 위해 매개변수들을 조정하지 않고 네트워크 상황에 따라 가변적으로 대응하는 새로운 비선형 확률함수의 도입을 제안한다.

RED 알고리즘에서 비선형 확률함수를 도입하게 된 계기는 RED 알고리즘이 패킷을 마크하는 구간은 전체 큐 길이에 비해 짧다는 것에 착안한 것이다. 짧은 구간 내에서 선형 확률함수에 의한 마크 동작은 체증 회피를 위해 바람직한 것이라 할 수 있으나 min_{th}과 max_{th}를 전후하여 그 불연속성이 크다. Q_{avg}가 min_{th}경계에 있는 경우는 앞서 설명한 소량을 전송하는 단말에게 적은 확률이나마 불공정성을 초래할 수 있다. 또한 Q_{avg}가 max_{th}경계에 있는 경우에는 마크 확률의 불연속성에 의해 입력패킷들의 마크확률이 급격히 변화하고 이에 따라 Q_{avg}가 심하게 진동하는 현상을 초래한다. 이러한 진동현상은 결국 Droptail 게이트웨이로 동작하여 TCP 전역 동기화 문제를 야기 시킨다. 이는 다른 RED 버전들이 지적하고 있는 매개변수의 선택문제로 해석할 수도 있다. 매개변수를 적절히 조정한다면 이런 불연속성으로 인한 문제를 해결할 수 있다. 그러나 매개변수의 조절이 불연속점을 늦추거나 당길 뿐 궁극적인 해결책이 되지는 못한다.

이러한 관점에서 본 논문에서는 확률구간과 불연속점이 유연하게 연결되는 비선형 함수를 적용하였

다. 이 함수는 식 (2), (3)과 같다.

$$p = \max_p\left(\frac{Q_{avg} - \min_{th}}{\max_{th} - \min_{th}}\right)^{C_{low}}, \quad \min_{th} \le Q_{avg} < \max_{th-} \tag{2}$$

$$p = (1-\max_p)\left(\frac{Q_{avg} - \max_{th-}}{\max_{th+} - \max_{th-}}\right)^{\frac{1}{C_{high}}} + \max_p \max_{th-} \le Q_{avg} < \max_{th+} \tag{3}$$

p는 마크확률이고, max_{th-}와 max_{th+}는 확률함수가 불연속점과 유연하게 연결되도록 하고 급격한 확률 변화를 흡수하기 위해 새롭게 적용된 수치들로서 max_{th} 전후로 약간의 변화를 준 값이다. C_{left}와 C_{right}는 네트워크 상황에 따라 곡선의 깊이를 조절해주는 역할을 하는 변수이다.

그림 4는 제안된 비선형 확률함수를 개략적으로 도식한 것이다.

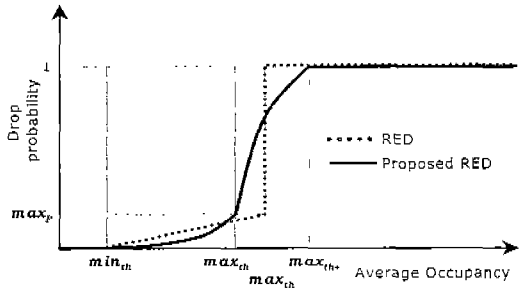


그림 4 비선형 패킷 마크 확률 함수

제안된 확률함수의 특성을 살펴보면, min_{th}과 max_{th-}에서는 식 (2)가 선형확률 함수보다 작은 값을 가지도록 하여 Q_{avg}가 작은 경우에 마크확률을 최소화하였다. 이때 C_{left}와 C_{right}는 비선형 함수의 곡면지수(convex index)로 사용된다. C_{left}는 입력 패킷이 식 (2)에 의해 마크될 때마다 증가하도록 $count$값에 비례하게 설정하여 낮은 확률에서 마크되는 빈도를 더욱 줄일 수 있도록 하였다. 반대로 마크되지 않을 때마다 감소하여 선형에 가깝게 접근하도록 하였다. C_{right}는 식 (3)에서 사용되는 곡면지수로 입력 패킷이 식 (3)에 의해 마크될 때마다 감소하도록 $count$값에 반비례하게 설정하였으며 반대의 경우에는 비례하게 설정되어 있다.

제안된 비선형 함수에서는 max_{th-}와 max_{th+}를 새롭게 도입하였다. 비선형 함수를 도입함으로써 max_{th-} 이전에는 선형 함수에서보다 낮은 마크 확률을 가지게 됨을 보충하기 위해 max_{th+}을 도입하여 Q_{avg}을 낮게 유지할 수 있도록 하였고, max_{th} 전후의 불연속성을 없애기 위해 비선형 함수는 max_{th-}과 max_{th+} 사이에서 유연하게 변화한다. 그리고 max_{th}와 max_{th-} 사이의 차이가 max_{th-}와 max_{th} 사이의 차이보다 크게 하여 max_{th-}을 도입함으로 커진 마크 확률을 낮추는 역할을 한다. 이를 통하여 통계적인 마크 확률은 크게 달라지지 않는다.

```
01 : Initialization
02 : for each packet arrival
03 :   calculate the new average queue size Qavg
04 :   if minth ≤ Qavg < maxth+
08 :     if Qavg < maxth-
10 :       p ← maxp((Qavg-minth)/(maxth--minth))^Clow
14 :       with probability p :
15 :         mark the arriving packet
05 :           increment count
05 :         else decrement count
04 :         if Qavg > Qprev
05 :           increment count
05 :         else decrement count
11 :     else if Qavg < maxth+
13 :       p ← (1-maxp)((Qavg-maxth-)/(maxth+-maxth-))^
(1/Chigh)+ maxp
14 :       with probability p :
15 :         mark the arriving packet
05 :           increment count
05 :         else decrement count
04 :         if Qavg > Qprev
05 :           increment count
05 :         else decrement count
16 :     Clow ← Clow + count
17 :     Chigh ← Chigh - count
19 :   else if maxth+ < Qavg
20 :     mark the arriving packet
24 :   Qprev ← Qavg
```

그림 5 제안된 RED 알고리즘

또한 Q_{avg}가 min_{th}와 max_{th-} 사이에서 감소할 때는 곡면지수를 증가시키고, 반대로 증가할 때는 곡면지수를 감소시켜 마크 확률을 조절한다. 이는 낮은 확

룰이지만 네트워크 상황에 적극적으로 대응하기 위한 방법으로 *count*를 마크여부에 따라 증감하는 이유와 같다. 이와 마찬가지로 Q_{avg}가 max_{th-}와 max_{th+} 사이에서 감소할 때는 곡면지수를 감소시키고, 반대로 증가할 때는 곡면지수를 증가시켜 마크 확률을 조절한다.

기존 RED 알고리즘에서는 그림 5에서와 같이 마크확률 p_b를 계산하고 이를 다소간 높여주기 위해 p_a를 다시 계산하였으나 제안된 알고리즘에서는 p_a를 더 이상 사용하지 않고 비선형 확률 함수에 의해 계산된 p만을 사용한다.

비선형 확률함수를 도입함에 있어 단순 선형함수를 적용하였을 때보다 구현의 복잡도는 나빠졌다고 할 수 있다. 그러나 본 논문에서는 이점에 중점을 두지 않는다. 본 논문에서는 FRED나 WRED에서와 같은 다중 큐(multiple queue)의 처리를 사용하지 않는 점, RED 게이트웨이에서 패킷을 복잡한 규정에 따라 분류하지 않는 점 등에 더욱 중점을 두었다. 이를 바탕으로 제안된 알고리즘을 적용한 RED 알고리즘에서는 기존 선형 확률함수 대신 식 (2)와 (3)의 비선형 확률함수로 교체하였으며 기존 RED 알고리즘에서 마크되지 않을 때마다 증가하여 마크확률을 다소간 높여주는 역할을 하던 *count*의 용도를 곡면지수를 증감하는 용도로 전환하였다. *count*는 마크확률에 따라 마크되지 않을 때와 평균 큐 길이가 증가할 때 따라 증가하여 곡면지수에 영향을 준다. 마크확률에 따라 마크되거나 평균 큐 길이가 감소할 때는 *count*가 감소하여 곡면지수를 재계산 하도록 한다.

5. 실험 및 결과

본 논문에서는 그림 6과 같은 망 구성에 의해 실험을 수행하였다. 각 TCP 전송단의 패킷 크기는 2Kbytes, 최대 윈도우 크기는 64Kbytes로 하였으며 각 전송단들은 자신이 전송한 패킷에 의해 네트워크가 체증상태임을 통보받은 후에 슬로우-스타트와 체증회피를 수행한다. 먼저 연결된 TCP 전송단의 수에 따른 평균 큐의 길이 변화를 실험하였고 급격한 대역의 변화에 비선형 함수를 적용한 RED 게이트웨이의 적응성에 대하여 실험하였다.

5.1 평균 큐 길이의 변화

시뮬레이션에 사용된 네트워크는 게이트웨이 G_1

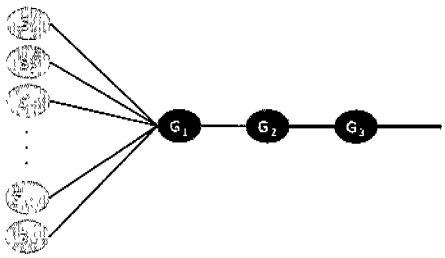


그림 6 시뮬레이션 모델

에 다양한 부하의 변화를 주기 위하여 16~40개의 전송단과 연결되어 있으며, 각 전송단 S_1,…..., S_n과 G_1 사이의 링크는 모두 10Mbps의 대역폭과 5ms의 전파지연을 가지도록 설정하였다. G_1과 G_2에서 병목현상을 유도하기 위하여 10Mbps의 대역폭과 5ms의 전파지연을 갖도록 설정하였고 G_2와 G_3, G_3와 E_1사이에는 45Mbps의 대역폭과 5ms의 전파지연을 갖도록 설정하였다.

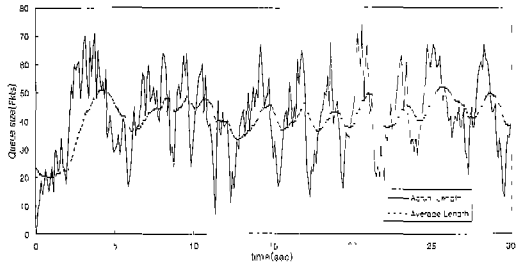
RED의 경우 w_q=0.002, max_p=0.1, max_{th}= 15, min_{th}=5로 하였다[4, 6]. 본 논문에서는 RED의 매개변수를 그대로 사용하고 max_{th+}=18, max_{th-}=14를 추가하였다.

그림 7과 8은 같은 조건하에서 RED와 비선형 확률함수를 적용한 RED 게이트웨이에 16개, 32개의 연결을 주고 큐 길이와 평균 큐 길이의 변화를 나타낸 것이다. y축은 패킷 단위의 큐 길이이다.

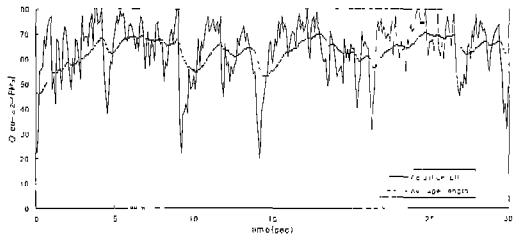
RED 게이트웨이와 마찬가지로 비선형 확률함수를 적용한 RED 게이트웨이에서도 유사한 패턴의 큐 길이의 변화와 평균 큐 길이를 가지는 것을 알 수 있다. 시뮬레이션에서 G_1에 심한 병목현상을 유발하도록 조절하였기 때문에 큐는 대부분 높은 점유율을 보인다. 연결이 2배로 증가하면 더욱 높은 점유율을 보이는 것을 알 수 있다.

RED 게이트웨이의 경우는 연결의 수가 증가함에 따라 큐 길이의 진동이 심하다. 그러나 비선형 확률함수를 적용한 RED 게이트웨이가 좀 더 적극적으로 마크하여 전체적 평균 큐 길이가 큰 변화없이 유지된다는 것을 알 수 있다. 그러나 16개의 연결에서 전체 평균 큐 길이의 경우 RED 게이트웨이는 41.76패킷, 제안된 RED 게이트웨이는 45.78패킷이고, 32개의 연결에서 전체 평균 큐의 길이는 RED 게이트웨이의 경우 62.94패킷, 제안된 RED 게이트웨이는 65.89패킷으로 기존의 RED 알고리즘에 의한 큐 길이의 편

차가 심하지만 평균 큐 길이를 낮게 유지하는 데에는 기존 RED 알고리즘이 조금 더 유리한 것을 알 수 있었다.

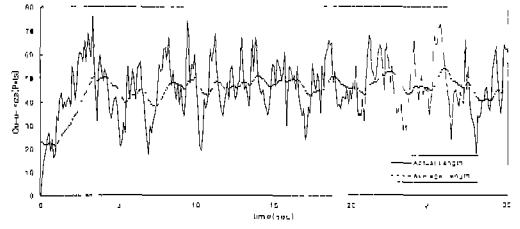


(a) 16 Connections

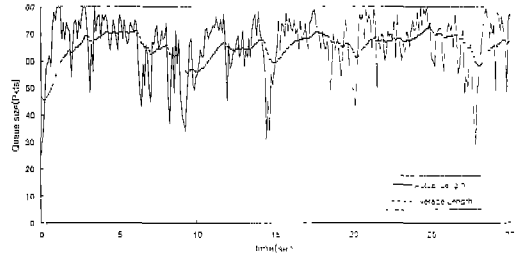


(b) 32 Connections

그림 7 RED 게이트웨이의 큐 길이 변화



(a) 16 Connections



(b) 32 Connections

그림 8 비선형 함수를 적용한 RED 게이트웨이의 큐 길이 변화

5.2 대역 변화의 적응성

RED 알고리즘은 많은 실험을 거쳐 현재 사용 중인 매개변수들로 최적화 되었으나 근본적으로 고정된 매개변수들을 사용함에 따라 급격한 대역의 변화에 대처하는 능력이 떨어진다. 그림 9와 그림 11에서와 같이 급격한 대역의 변화를 겪는 RED 게이트웨이는 큐 길이의 심한 진동과 패킷 손실을 유발하여 안정화되기까지 오랜 시간이 소요된다.

이러한 문제점을 개선하기 위해 ARED 알고리즘은 최대 마크 확률을 평균 큐 길이의 변화에 따라 가변적으로 사용한다[5]. 그러나 앞서 언급한 바와 같이 최적화된 매개변수를 조정함으로 인한 알고리즘 복잡도의 증가와 새로이 max_p의 증감정도를 정의해야 한다는 문제점을 안고 있다.

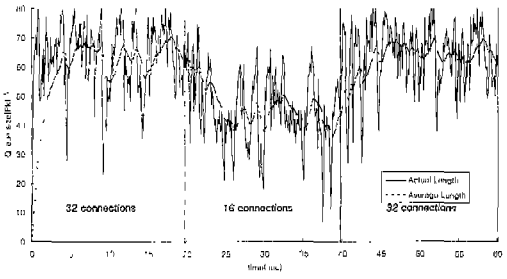


그림 9 RED 알고리즘의 적응성 (1)

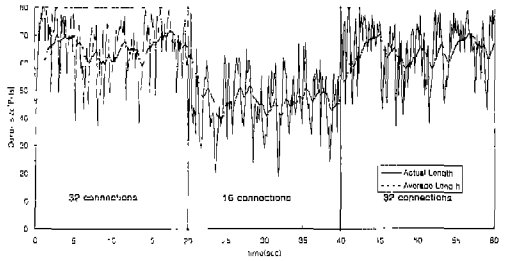


그림 10 비선형 함수를 적용한 RED 알고리즘의 적응성 (1)

이에 비해 비선형 확률함수를 적용할 경우에는 기본적으로 RED 알고리즘과 같은 매개변수를 사용하고, 마크여부에 따라 마크확률이 적응성을 가지고 변화하기 때문에 RED 알고리즘의 마크확률에서 크게 벗어나지 않는다는 장점을 가지고 있다.

그림 9와 그림 10은 대역 변화의 적응 능력을 측정하기 위해서 20초 간격으로 전송단을 16개에서 32개로, 다시 16개를 연결하여 연속적인 변화를 주었다. 이때 연결의 수가 변하고 난 뒤 10초 이후부터는 변화구간에서 평균 큐 길이가 안정화된 것으로 볼 수 있다. RED 게이트웨이에서는 최초 32개의 연결에서 16개의 연결로 바뀌어 4.72초만에 30초에서 40초간의 평균 큐 길이인 43.88패킷에 도착하였다. 그러나 비선형 함수를 적용한 RED 게이트웨이는 1.4초만에 30초에서 40초간의 평균 큐 길이인 45.81패킷에 도달함을 알 수 있다. 또한 16개의 연결에서 32개의 연결로 바꾸었을 경우, RED 게이트웨이는 4.3초만에 50초에서 60초 동안의 평균 큐 길이인 64.54패킷에 도달하였으나 비선형 함수를 적용한 RED 게이트웨이는 2.64초만에 50초에서 60초 동안의 평균 큐 길이인 64.55패킷에 도달함을 알 수 있다.

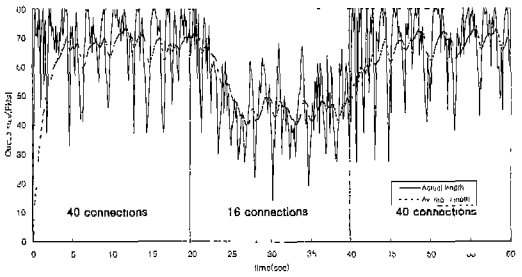


그림 11 RED 알고리즘의 적응성 (2)

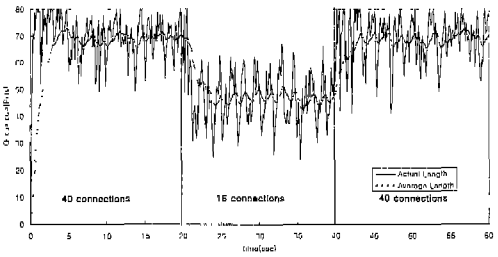


그림 12 비선형 함수를 적용한 RED 알고리즘의 적응성 (2)

그림 11과 그림 12는 앞선 실험에서 보다 좀 더 가혹한 환경을 조성하기 위하여 20초 간격으로 전송단을 16개에서 40개로, 다시 16개를 연결하여 연속적인 변화를 주었다. 이때 역시 연결의 수가 변하고 난 뒤 10초 이후부터는 변화구간에서 평균 큐 길이가 안정화된 것으로 가정한다. RED 게이트웨이에서는 최초 40개의 연결에서 16개의 연결로 바뀌어 5.90초만에 30초에서 40초간의 평균 큐 길이인 44.85패킷에 도착하였다. 그러나 비선형 함수를 적용한 RED 게이트웨이는 3.82초만에 30초에서 40초간의 평균 큐 길이인 46.01패킷에 도달함을 알 수 있다. 또한 16개의 연결에서 40개의 연결로 바꾸었을 경우, RED 게이트웨이는 4.42초만에 50초에서 60초 동안의 평균 큐 길이인 68.71패킷에 도달하였으나 비선형 함수를 적용한 RED 게이트웨이는 2.90초만에 50초에서 60초 동안의 평균 큐 길이인 69.04패킷에 도달함을 알 수 있다.

이 실험에서 한 가지 더 살펴볼 것은 큐 길이가 안정화되는 동안 패킷 손실정도이다. 앞선 실험에 비해 가혹한 조건을 부과함에 따라 급격한 큐 길이의 변동으로 인하여 연결이 변화하는 동안 패킷 손실이 많이 발생함을 알 수 있다. RED 게이트웨이에서는 최초 40개의 연결에서 16개의 연결로 바뀌어 안정화되는 5.90초 동안 패킷손실률이 3.6%였으며 비선형 함수를 적용한 RED 게이트웨이는 같은 5.90초 동안 2.17%의 손실을 발생하였다. 또한 16개의 연결에서 40개의 연결로 바꾸었을 경우, RED 게이트웨이는 안정화되는 4.42초 동안 7.65%의 패킷 손실이 발생하였으나 비선형 함수를 적용한 RED 게이트웨이는 같은 시간 동안 5.19%의 손실이 발생하였다.

두 실험을 통해서 기본적인 RED 알고리즘은 대역의 급격한 변화에서 큐 길이의 심한 진동을 겪는다는 것을 알 수 있다. 그러나 비선형 확률 함수를 적용하였을 경우에는 급격한 변화에 적응하여 진동구간이 빠르게 안정되는 모습을 볼 수 있다. 이는 RED 알고리즘이 정적인 매개변수들을 사용함으로 급격한 변화에 빠르게 대응하지 못하는 것에 비하여 비선형 함수를 적용할 경우는 대역의 변화에 적응성이 빠르고 또한 패킷의 손실도 줄일 수 있음을 보여주는 것이다.

6. 결 론

RED 알고리즘은 게이트웨이 입장에서 큰 부하없이 구현할 수 있는 알고리즘이다. 단순한 확률함수와 복잡한 계산이 필요없는 매개변수들로 체증상황을 효과적으로 회피할 수 있다는 점에서 RED 알고리즘은 큰 장점을 가지고 있다. 그러나 RED 알고리즘의 단점을 보완하기 위한 많은 연구들은 복잡한 트래픽

의 분류나 매개변수를 변화시키는 방법을 채택하여 RED 본래의 장점을 잃어버릴 수 있다.

본 글에서는 RED 알고리즘의 단점을 보완하기 위한 다른 RED 알고리즘들의 접근법과는 달리 RED 알고리즘의 기본적인 매개변수를 바꾸지 않고 선형적인 확률함수를 단순한 비선형 함수로 전환함으로써 RED 알고리즘의 단점을 보완하고자 하였다. 새로이 제안된 비선형 함수는 기존의 알고리즘을 크게 해치지 않으면서 네트워크의 상황에 따라 능동적이면서도 유연하게 대처할 수 있도록 설계되었다.

비선형 확률을 사용함에 있어서 가변적인 다차원 함수를 사용해야한다는 점은 구현의 복잡도에서 가장 큰 단점이다. 기존의 많은 RED알고리즘들은 알고리즘을 단순화한다는 점에서 비선형 확률 함수는 고려의 대상이 아니었다. 본 논문이 비선형 함수를 사용한 것은 선형함수만을 고집하는 RED 알고리즘의 많은 개선책들 역시 성능향상을 위해 구현의 복잡도가 증가한다는 것에서 기존 알고리즘들과 다른 RED 알고리즘의 성능을 향상하기 위한 접근법으로 해석할 수 있다.

시뮬레이션의 결과를 통하여 비선형 확률함수를 적용한 RED 게이트웨이의 경우에 평균 큐 길이 특성과 전역동기화를 방지하는 특성이 RED 게이트웨이와 유사하게 측정됨을 보았고, 체증상황에서 좀 더 능동적으로 마크를 하여 네트워크의 효율을 높일 수 있음을 알 수 있었다. 또한 급격한 네트워크 부하의 변화에서도 빠르게 적응할 수 있음을 알 수 있었다. 빠른 적응 능력은 심한 대역의 변화를 겪는 동안 패킷의 손실을 줄인다는 것을 알 수 있었다.

실험을 통하여 비선형 함수를 적용하였을 때 평균 큐 길이는 RED 게이트웨이에 비해 다소 높게 측정되었다. 짧은 평균 큐 길이를 유지할 수 있는 능력은 버스트한 트래픽을 흡수할 수 있다는 점에서 기존의 RED 알고리즘이 좀 더 우수한 것으로 나타났다. 큐의 사용효율 측면에서는 그 반대의 경우로도 생각할 수 있다. 이러한 점에서는 평균 큐 길이의 차이가 RED 게이트웨이에 비해 크게 다르지 않기 때문에 단점으로만 생각할 수는 없다.

본 논문은 추가적으로 새로이 제시된 비선형 확률함수가 좀더 능동적으로 네트워크 환경에 적용할 수 있도록 단순한 지수함수 형태에서 벗어나 체증정도에 반응하여 곡선의 중심을 이동할 수 있도록 할 것이다. 또한 평균 큐 길이를 낮게 가지게 하기 위한 연구가 추가적으로 필요할 것이다.

참고문헌

[1] Cui-Qing Yang and Alapati V.S. Reddy. "A Taxonomy for Congestion Control Algorithms in Packet Switching Networks", IEEE Network Magazine July/August 1995, Vol. 9, No. 5.

[2] W. Stevens. "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms", Internet RFC2001, January 1997.

[3] Raj Jain, K.K. Ramarkrishnan, and Dah-Ming Chiu. "Congestion Avoidance in Computer Networks With a Connectionless Network Layer", DEC-TR-506, June, 1997.

[4] V. Jacobson, and S. Floyd. "Random Early Detection Gateways for Congestion Avoidance", IEEE/ACM Transactions on Networking, Vol. 1, No. 4., August 1993.

[5] Feng, W., D. Kandlur, D.Saha, and K. Shin. "A Self-Configuring RED Gateway", IEEE INFOCOM99, March 1999.

[6] S. Floyd and K. Fall. "NS Simulator Tests for Random Early Detection(RED) Gateways", available via http://www-nrg.ee.lbl.gov/nrg-papers.html, 1997.

[7] Lin, D., and R. Morris. "Dynamics of Random Early Detection", IEEE/ACM SIGCOMM97. Cannes, France. (October 1997), pp127-137.

박 종 석

1999 대구대학교 정보통신공학부 학사
2001 동국대학교 정보통신공학과 대학원 석사
관심분야:고속통신망, ATM통신, 통신프로토콜
E mail:color@lily.dongguk.ac.kr

최 병 석

1985 서울대학교 전자공학과 학사
1987 Fairleigh Dickinson University EE 공학석사
1989 Polytechinc University EE 공학석사
1994 Polytechinc University EE 공학박사
2000~현재 동국대학교 정보통신공학과 부교수
관심분야:ATM 교환기 성능 분석, ATM/B-ISDN망 설계 및 성능 분석, 네트워크 트래픽 관리
E-mail:bchoe@dgu.ac.kr

정 대 인

1984 서울대학교 제어계측공학과 학사
1986 서울대학교 제어계측공학과 석사
1997 Polytechnic University 전기공학과 박사
2001~현재 한국외국어대학교 디지털정보학과 조교수
관심분야:Traffic Management in ATM Networks, QoS in Internet, Next Generation Internet
E-mail:djeong@maincc.hufs.ac.kr

이 봉 영

1978 고려대학교 물리학과 입학
1985 고려대학교 물리학과 졸업
1989 일본 오사카대학 전기공학분야 물리계 석사
1992 일본 오사카대학 전기공학분야 물리계 박사
2001~현재 한국통신 통신망연구소 선임연구원
관심분야:광통신 기술(TDM, WDM Coherent 광통신, Optical Control 등,), Network Engineering(IP over WDM, IP over SDH, IP over ATM, Voice over IP, Voice and Telephony Over ATM)
E-mail:yiby@kt.co.kr

박 현 민

1985 서울대학교 전자공학과 공학학사
1988 North Carolina State University Electrical and Computer Engineering 석사
1988 North Carolina State University Electrical and Computer Engineering 박사
2001~현재 명지대학교 공과대학 컴퓨터학부 조교수
관심분야:ATM 교환기 성능 분석, ATM/B-ISDN망 설계 및 성능 분석, 네트워크 트래픽 관리 및 보안 기술 연구
E-mail:hpark@mju.ac.kr

류 성 진

2000 대진대학교 전자공학과 학사
2001~현재 동국대학교 정보통신공학과 석사과정
관심분야 고속통신망, ATM통신, 통신망 분석 및 모델링, 인터넷
E-mail:sjryu@dongguk.edu