인터넷에서 TCP/IP 동작 개선을 위한 부하 적응형 DRED 알고리즘

장정식° 이동호

광운대학교 컴퓨터과학과
{jsjang°, dhlee}@cs.gwu.ac.kr

A Load Adaptive DRED for Improving TCP Behavior in Internet

Jung-Sik Jang° Dong-Ho Lee
Dept. of Computer Science, Kwangwoon University

요 약

지금까지 TCP 혼잡 제어를 위한 여러 종류의 메커니즘들이 Connection을 적응성 있게 제어하기 위하여 사용되어 왔지만, TCP 혼잡 제어 메커니즘들은 성능상의 여러 문제점을 가지고 있는게 사실이다. 이에 IETF에서는 AQM(Active Queue Management) 메커니즘으로 RED 알고리즘을 권고했다. 그러나 이 또한 상이한 네트워크에서는 파래메터 설정에 따른 문제점이 있어, 네트워크 상황에 적절하게 대응하지 못하는 단점이 있다. 이러한 RED 알고리즘의 문제점을 극복하고, 효율성을 개선하기 위해서 SRED, BLUE, FRED, DRED 등 다양한 AQM 메커니즘들이 제시되고 있다.

본 논문에서는 네트워크 트래픽 상황에 따라 적응성을 갖고 Threshold의 변경에 사용되는 패킷 손실율을 구하는데 있어 트래픽을 고려한 가중치를 줌으로써 트래픽 상황을 반영하도록 했고, Threshold 설정에 있어 적응성 있는 단계를 통하여 큐 안정성을 개선하도록 하였다. 제안한 알고리즘의 성능 분석은 NS 시뮬레이터를 사용하였고, 제안한 Load Adaptive DRED 알고리즘과 DRED 알고리즘의 버퍼 관리 기법의 성능 비교 분석을 통하여 큐 안정성의 개선된 성능을 확인하였다.

1. 서 론

지금까지 TCP 플로우(flow) 및 혼잡 제어를 위한 여러 종류의 메커니즘들이 Connection을 적응성 있게 제어하기 위하여 사용되어 왔다. 그러나, Drop Tail 패킷을 처리하는 네트워크 상에서 TCP 혼잡제어 메커니즘은 플로우의 동기화(synchronization of flow) 문제와 플로우 사이의 불규칙한 패킷 손실율, 네트워크 자원의 낮은 이용율 등의 문제점을 안고 있다.

이러한 이유로 종단 시스템(end system)은 Congestion Avoidance, Slow Start, Fast Retransmit, Fast Recovery 메커니즘[1] 등을 사용하고 있지만, 현재의 Drop Tail 네트워크에서의 TCP 혼잡 제어는 여전히 만족스럽지 못한게 사실이다.

이에 IETF(Internet Engineering Task Force)는 이러한 TCP 성능상의 문제점들을 보완하기 위해 AQM(active queue management) 메커니즘을 권고했다.[2]

IETF에서 최초로 제안된 AQM 메커니즘이 RED (Random Early Detection) 알고리즘이다.[3]

AQM의 기본 아이디어는 혼잡상황의 발생으로 인하여 큐 오버플로우와 지속적인 패킷 손실이 일어나기 전에, 초기에 그 상황을 TCP 종단 지점(end point)에 알려서 재전송 비율을 줄이는 것이다.

RED를 사용하는 큐는 Drop Tail 큐보다 더 좋은 성능을 가지고 있다. 그러나, RED 알고리즘은 상이한 네트워크에서는 파라메터 설정에 따른 문제점이 있어, 네트워크 상황에 적절하게 대응하지 못하는 단점이 있다.

이러한 RED 알고리즘의 문제점을 극복하고, 효율성을 개선하기 위해서 Stabilized RED(SRED)[4], BLUE[5], FRED[6] 등 많은 AQM 메커니즘이 제시되고 있는데, 이중에서 높은 링크 이용률(link utilization) 동안에 패킷 손실율을 제어하고 큐 크기를 안정화시키는데 효과적인 알고리즘이 DRED(Dynamic RED)이다.[7]

DRED 알고리즘은 네트워크 트래픽 로드에 독립적으로 미리 정해진 목표 큐 사이즈에서 실제적인 큐 사이즈를 안정화시키기 위한 알고리즘인데, 이러한 DRED 알고리즘을 기반으로 네트워크 상황에 적응성 있게 큐 크기를 제어하기 위해 제시된 것이 DRED 알고리즘을 사용한 버퍼 관리 기법[8]이다.

DRED 버퍼 관리 기법은 패킷 손실율을 검사하고 Connection의 수에 따라 Threshold를 조정하여 미리 정해진 손실율 수치에 현재의 손실율을 유지시켜서 패킷 손실율과 큐잉 지연, 큐 크기를 효과적으로 제어한다.

본 논문에서는 큐 안정화를 위해 패킷 손실율을 계산하고 Threshold를 조정하는데 있어서, 트래픽 상황을 고려하여 파라메터를 변경하고 적응성을 위한 세부적인 단계를 둠으로써 DRED 버퍼 관리 기법보다도 더욱 트래픽 상황을 반영하고 큐 안정화에 효과적인 부하 적응형(Load Adaptive) DRED 알고리즘을 제안한다.

본 논문은 총 5장으로 기술되었으며, 2장에서는 DRED 알고리즘의 버퍼 관리 기법에 대하여 기술하였고, 3장은 제안한 부하 적응형 DRED 알고리즘을 기술하였다. 4장에서는 제안한 알고리즘의 성능 분석 및 기존 연구와의 비교를 위한 시뮬레이션 모델을 설명하고, 마지막으로 5장에서는 결론 및 향후 과제를 제시하였다.

2. 관련 연구

2.1 DRED 알고리즘의 버퍼 관리 기법

DRED 버퍼 관리 기법은 Active Connection의 수가 변함에 따라 동적으로 Threshold를 설정함으로써 라우터 큐를 안정화시키고 RED 알고리즘의 효율성을 개선시킨 AQM 알고리즘이다

이 기법은 많은 타임아웃을 발생시키지 않는 정해진 수치 값에서 손실율을 유지하기 위해 Threshold를 변화하고 패킷 손실율을 검사하여 Connection의 수를 조정한다.[8]

즉, 현재의 손실율을 검사하고 AQM 파라메터인 Threshold를 그에 맞게 변경하여, 손실율이 목표치(Target level) 주위에 유지하도록 한다. 그래서 Connection의 수가 감소하면 라우터에서 보다 적은 지연을 유지하기 위해 Threshold를 하향으로 조정하여 큐를 안정화 시킨다.

2.2 DRED 버퍼 관리 기법의 Block Diagram

DRED 버퍼 관리 기법을 기능적으로 구분하게 되면, 크게 두 가지로 나눌 수 있다. 패킷 폐기율(Packet drop rate)과 큐 크기 등의 파라메터를 결정하는 Parameter Estimation과 동적으로 Threshold를 조정하고 적용하게 될 폐기율을 결정하는 Controller 부분으로 구성한다. Controller는 High-level과 Low-level Controller 두 단계로 구성된다. 그림 1은 두 단계 Controller의 입력 파라메터 값과 출력 값을 보여준다.

DRED 버퍼 관리 기법은 큐 내의 Connection의 수가 변화함에 따라 Drop Controller의 큐 threshold를 동적으로 변화하기 위해 사용된다. Controller는 큐를 안정화 시키기 위해서 현재의 패킷 손실율을 검사하고 정해진 값에서 손실율을 유지하기 위해 Threshold를 조정함으로써 Connection의 수를 조정한다.

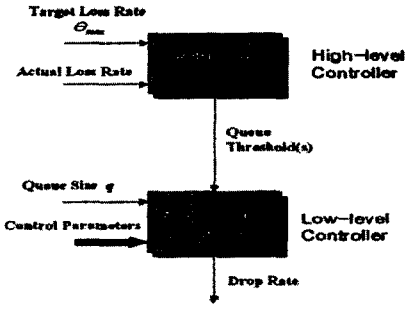


그림 1. 두 단계의 Controller 구성

Controller Design은 Estimation으로부터 측정된 파라메터 값인 실제 손실율(actual Loss Rate)과 목표 손실율을 통하여 Threshold를 설정하게 되고, Low-level Controller는 설정된 Threshold와 Queue Size(q) 그리고 파라메터를 계산하여 패킷 폐기율을 계산하게 된다.

패킷 폐기율은 도착한 패킷과 큐에서의 손실율을 검사하여 계산된다. 계산된 패킷 폐기율은 현재의 손실율을 알수 있는 좋은 측정값이 된다. 그러므로 계산된 패킷 폐기율은 앞으로 나타나는 패킷 손실율을 조정하기 위해서 큐 Threshold를 변경하는데 사용된다.

2.3 버퍼 관리 알고리즘

큐 Threshold는 상위 Threshold(T)와 하위 threshold(L)로 구성되는데, T는 미리 정해진 목표 손실율(θ_{max})에 가깝게 패킷 손실율을 유지하기 위해서 동적으로 변경된다.

계산된 패킷 손실율 $\hat{P}_l$이 θ_{max}를 위아래로 벗어나 있을 때, 큐 Threshold T는 변경된다. $\hat{P}_l$이 θ_{max}를 보다 크면, 큐 Threshold T는 증가되고, $\hat{P}_l$이 θ_{max}를 보다 작으면 T는 감소된다.

시간 n에서 패킷 폐기 확률 $P_d(n)$은 패킷 손실율 P_l를 알 수 있는 좋은 측정값이다. $P_d(n)$은 현재 손실율에 거의 유사하게 나타나기 때문에, $P_d(n)$을 통해서 현재의 손실율을 알 수 있다. 그러므로 $P_l(n) = P_d(n)$을 T를 변경하는데 사용함으로써 손실율을 줄일 수 있다.

$P_l(n) = P_d(n)$ 값은 High-level controller에서 사용되기 전에 일시적인 현상들을 제거하기 위해 증가치 γ를 갖는 EWMA(Exponential Weighted Moving Average) 필터를 사용하여 필터링 된다.

그 필터링 된 값은 다음으로 구해진다.

$$\hat{P}_l \leftarrow (1-\gamma)\,\hat{P}_l(n-1) + \gamma P_l(n), \quad 0<\gamma<1$$

필터링이 필요 없다면, $\hat{P}_l = P_l$이 된다.

3. Load Adaptive DRED

큐를 안정화 시키기 위해 Threshold의 동적인 변경에 있어 중요하게 사용되는 파라메터가 $\hat{P}_l$이다.

필터링 된 $\hat{P}_l$를 구할 때 현재의 패킷 손실율(P_l)이 $\hat{P}_l$로 변하는데 영향을 나타내는 값이 가중치(γ)이다. 가중치(γ)를 이용하여 일시적인 버스트 트래픽의 영향으로 인해 현재의 패킷 손실율이 짧은 동안 증가하더라도 $\hat{P}_l$은 바로 영향을 받지 않게 된다.

γ가 작으면 작을수록 패킷 손실율 $\hat{P}_l$이 변하는데 미치는 영향이 작게 된다. 이런 경우에 짧은 시간 동안 패킷 손실율이 증가하더라도 $\hat{P}_l$이 증가하는데 미치는 영향이 작기 때문에 일시적인 트래픽의 증가로 인한 오류를 피할 수 있다.

그러나, DRED 버퍼 관리 기법에서는 $\hat{P}_l$를 계산할 때, 가중치가 고정되어 있어 트래픽 상황이 큰 폭으로 변할 때에는 유연하게 대처하지 못하는 문제점이 있다.

본 논문에서 제시한 Load Adaptive DRED 알고리즘은 이러한 현재의 트래픽 로드 상태를 고려한 가중치(γ)를 줌으로써, 보다 네트워크 트래픽 상황을 적응성 있게 반영하도록 하고 라우터에서의 큐 안정성을 높였다.

그림 2는 트래픽 상황을 고려하여 가중치를 변경하기 위해서 계산되는 요약된 알고리즘을 보여준다.

```
for each packet arrival
        calcuate the average queue size Qavg
if  Tmin ≤ Qavg < Tmax
        decrease γ
if  Tmax ≤ Qavg
        increase γ
```

그림 2. 트래픽 상황에 따른 가중치(γ) 변경

현재의 패킷 손실율을 통하여 Threshold를 조정하므로써 트래픽 상황에 따라 큐를 조정하게 되는데, 본 논문에서는 보다 정확한 현재의 트래픽 상황을 고려하여 Threshold를 변경하도록 하기 위해서 지금까지의 평균 패킷 손실율, $\hat{P}_l(avg)$을 구하고 이를 현재의 패킷 손실율, $\hat{P}_l(now)$과 비교하여, 세부적으로 Threshold 값을 변경함으로써, DRED 버퍼 관리 기법보다 트래픽 상황을 더욱 적응성 있게 반영하여 큐를 안정화시켰다.

그림 3은 제안한 Load Adaptive DRED 알고리즘의 패킷 손실율에 따른 Threshold를 변경하는 정리된 알고리즘이다.

```
every δ sec do the following
If p̂l (now) > θmax then
        if  p̂l (now) > p̂l (previous)
                increase T : T <- T+ΔT
        else if  p̂l (now) > p̂l (avg)
                increase T : T <- T+ΔT
        else do not change T
If p̂l (now) < θmax then
        if  p̂l (now) < p̂l (previous)
                decrease T : T <- T-ΔT
        else if  p̂l (now) < p̂l (avg)
                decrease T : T <- T-ΔT
        else do not change T
```

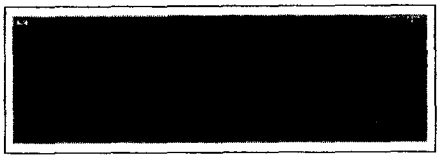
그림 3. Load Adaptive DRED 알고리즘

4. 실험 및 성능 평가

본 장에서는 제안한 Load Adaptive DRED 알고리즘의 큐 버퍼 안정화의 개선 정도를 측정하기 위해서 실험 모델을 설명하고 NS 시뮬레이션을 통한 DRED 버퍼 관리 기법과 비교 분석한 결과를 보인다.

실험은 같은 수의 노드로 대칭적으로 구성된 구조와 비대칭적으로 구성된 구조로 나누어 실험을 하였다.

그림 4는 노드가 대칭적으로 구성된 구조에서의 시뮬레이션 결과이다. (a)와 (b)의 성능을 비교했을 때, 제안한 Load Adaptive DRED를 사용한 경우, 시간당 큐의 변화가 (b)의 경우보다 적고 평균 큐의 변화가 완만함을 알 수 있다.



(a) Load Adaptive DRED의 버퍼 크기 변화

(b) DRED 버퍼 관리 기법의 버퍼 크기 변화

그림 4 라우터에서 버퍼 크기의 변화에 따른 성능 비교 1 (노드의 대칭 구성)

그림 5는 노드가 비대칭적으로 구성된 경우의 결과로서 제안한 Load Adaptive DRED를 사용한 경우, 대칭적으로 구성된 경우와 마찬가지로 평균 큐 크기의 변화가 완만하고 시간당 큐의 변화가 적음을 알 수 있다.

(a) Load Adaptive DRED의 버퍼 크기 변화

(b) DRED 버퍼 관리 기법의 버퍼 크기 변화

그림 5 라우터에서 버퍼 크기의 변화에 따른 성능 비교 2 (노드의 비대칭 구성)

위의 실험에서 제안한 Load Adaptive DRED 알고리즘이 라우터에서의 큐 안정성 면에서 개선된 결과를 보여주는데, 이는 트래픽 상황에 따라 가중치를 반영하고 Threshold의 설정에 있어 보다 적응성 있는 단계를 줌으로써 가능했다. 본 실험을 통해서 제안한 알고리즘이 보다 적절하게 트래픽 상황을 반영하고 큐 안정성이 개선되었음을 확인했다.

5. 결론 및 향후 과제

본 논문에서는 네트워크 트래픽 상황을 반영하고 큐의 안정성에 있어서 개선된 성능을 갖는 Load Adaptive DRED 알고리즘을 제안하였다.

Load Adaptive DRED 알고리즘은 네트워크 트래픽 상황에 따라 적응성을 갖고 Threshold의 변화에 중요하게 영향을 미치는 패킷 손실율을 구하는데 있어 트래픽 상황을 고려하여 가중치를 설정하고, Threshold의 설정에 있어 보다 적응성 있는 세부 단계를 통하여 큐 안정성의 개선된 성능을 가질 수 있었다.

향후 과제로는 실험한 결과에 대해서 수식적인 증명을 통하여 제시된 알고리즘의 타당성과 효율성을 검증하고 이를 토대로 실제 네트워크 적용에 대한 연구가 수행되어야 할 것이다.

참고문헌

[1] W. Richard Stevens. "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms," IETF RFC 2001, Jan. 1997.

[2] B. Braden et al.,"Recommendation on Queue Management and Congestion Avoidance in the Internet", IETF RFC 2309, Apr. 1998.

[3] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," IEEE/ACM Trans. Net., vol. 1, no. 4, pp. 397-413, Aug. 1993

[4] T.J.Ott, T.V.Lakshman, and L.H.Wong, "SRED: Stabilized RED", Proc. IEEE INFOCOM '99, pp. 1346-55, Mar. 1999

[5] W.Feng et al., "BLUE:A New Class of Active Queue Management Algorithms, "Technical Report CSE-TR- 387-99, Dept.EECS, Univ. MI, Apr. 1999

[6] D. Lin and R. Morris, "Dynamics of Random Early Detection", Proc. SIGCOMM '97, Cannes, France, pp127-137, Sept.1997

[7] Chengyu Zhu et al., "A Comparison of Active Queue Management Alogorithsm Using the OPNET Modeler", IEEE Communications Magazine, pp. 158-167, June 2002.

[8] James Aweya et al., "An adaptive buffer Management Mechanism for Improving TCP Behavior under Heavy Load", IEEE ICC, Vol. 10, pp. 3217-3223, 2001.