JANT 한국항행학회논문지 Journal of Advanced Navigation Technology

기반 기술

J. Adv. Navig. Technol. 18(5): 461-467, Oct. 2014

다양한 네트워크 환경에 자동적으로 적응하는 RED 알고리즘

A New RED Algorithm Adapting Automatically in Various Network Conditions

김 동 춘
한국폴리텍대학 강릉캠퍼스 전자통신학과

Dong-Choon Kim

Department of Electronics and Communication, Korea Polytechnic College Gangneung Campus, Gangwon-do 210-932, Korea

[요 약]

AQM은 라우터에서 사용되고 있으며 큐의 순간적 길이나 평균길이를 관찰하여 혼잡을 예측하고 탐지하는 알고리즘이며, 평균 큐 길이가 임계 값을 초과하면 혼잡이 발생하였다고 추론하고 라우터로 유입되는 패킷을 미리 폐기시키거나 패킷 전송 노드에게 혼잡을 미리 알림으로써 전송노드에서 전송률을 줄여 혼잡을 피하도록 한다. AQM 알고리즘의 하나인 RED (random early detection)는 패킷을 무작위로 폐기시켜 혼잡제어를 하는 큐 기반의 혼잡 제어기술이다. RED는 TCP의 시간경과 및 큐 지연을 줄이며 링크 활용도를 높이고 버스트한 트래픽이 유입되는 것을 관리하는 기술로써 널리 사용되어지고 있으며 혼잡도의 정도를 큐의 평균 길이로 추정한다. 그러나 파라미터($P_{\max}$, $TH_{\min}$)값들이 고정되어 있어서 특정한 네트워크 환경에서는 동작을 잘 못한다. 본 논문에서는 네트워크 상태를 지속적으로 감지함으로써 $P_{\max}$ 와 $TH_{\min}$ 의 값을 네트워크 상태에 적합한 값으로 자동적으로 바꾸어줌으로서 다양한 환경에서도 동작을 잘할 수 있는 확장 RED를 제안한다.

[Abstract]

Active queue management (AQM) algorithms run on routers and detect incipient congestion by typically monitoring the instantaneous or average queue size. When the average queue size exceeds a certain threshold, AQM algorithms infer congestion on the link and notify the end systems to back off by proactively dropping some of the packets arriving at a router or marking the packets to reduce transmission rate at the sender. Among the existing AQM algorithms, random early detection (RED) is well known as the representative queue-based management scheme by randomizing packet dropping. To reduce the number of timeouts in TCP and queuing delay, maintain high link utilization, and remove bursty traffic biases, the RED considers an average queue size as a degree of congestions. However, RED do not well in the specified networks conditions due to the fixed parameters($P_{\max}$ and $TH_{\min}$) of RED. This paper addresses a extended RED to be adapted in various networks conditions. By sensing network state, $P_{\max}$ and $TH_{\min}$ can be automatically changed to proper value and then RED do well in various networks conditions.

Key words : AQM, RED, Congestion control, Queuing delay, Link utilization.

http://dx.doi.org/10.12673/jant.2014.18.5.461

Received 1 September 2014; **Revised** 22 October 2014
Accepted (Publication) 6 October 2014 (30 October 2014)

***Corresponding Author; Dong-Choon Kim**

Tel: +82-33-610-6181
E-mail: kdchoon@kopo.ac.kr

Ⅰ. 서 론

인터넷에서 체증을 해결하는 방법은 송신 호스트의 윈도우 크기(window size)와 재전송시간 (RTO; retransmission time out)을 이용하여 체증을 조절하는 방법과 망 중간 노드인 라우터에서의 버퍼관리를 통해, 체증을 미리 예견하여 라우터의 체증정도에 따라 라우터에 들어오는 패킷을 계산된 확률에 의해 폐기하는 RED(random early detection)방법이 있다[1],[2].

RED 방법은 중간 노드인 라우터에서 체증이 발생하기 전에 능동적인 버퍼관리를 통하여 global synchronization에 의해 순간적으로 링크의 이용률이 감소하는 것을 방지하고 라우터 내의 큐 길이를 가능한 작은 크기로 유지하게 하는 방법이며, 망 중간 노드에서의 큐잉 지연이 전체 패킷 처리율(throughput)에 매우 큰 영향을 미치기 때문에 중간 노드의 큐 길이를 작게 유지하는 것도 체증 제어의 중요한 목적의 하나로 인식되고 있다[3].

RED는 가중 평균 큐 길이에 따라 망의 체증정도를 결정하여 라우터에 들어오는 패킷을 폐기(drop)하는데 적용되는 확률이 결정되며, 이 RED 방식을 사용할 경우 drop tail 방식과 비교하여 링크의 사용효율을 높일 수 있다는 것은 이미 여러 문헌에서 보여주고 있다[3]-[5].

그러나 RED에서 사용하는 파라미터들은 망의 상황과 상관없이 동일한 값으로 설정되기 때문에 다양한 트래픽 상황에 적용하게 되면 여러 가지 문제점이 발생한다.

일반적으로 버스트한 데이터의 특성을 갖는 망에서는 최소 큐 한계 값($TH_{\min}$)을 크게 하면 링크의 사용효율을 높일 수 있으며, 지속적으로 망에 과부하가 가해지고 있을 때는 최대 확률($P_{\max}$)값을 높게 설정하여야 효율적인 체증관리가 가능하다고 알려져 있다[5]. 현재의 망의 특성은 버스트한 데이터의 특성과 지속적인 과부하의 망 특성을 동시에 갖고 있어 동일한 파라미터의 값을 설정하면 상이한 두 환경에서 잘 적용되기란 불가능하다.

본 논문에서는 long-term 가중 평균을 새로이 정의하여 망의 전체적인 부하를 감지하는 변수로 사용하고, 망의 상태에 따라 각각의 파라미터의 값을 자동적으로 조절함으로써 버스트한 환경에서는 최소 큐 한계 값($TH_{\min}$)을 크게 하여 링크의 사용효율을 높이고 지속적으로 과부하가 가해질 때는 최대 확률($P_{\max}$)값을 높게 하여 큐의 평균길이를 작게 유지하는 알고리듬을 제안하고자 한다.

Ⅱ. RED 알고리즘

2-1. RED(random early detection)알고리즘

이 알고리즘은 FIFO 출력 큐에 새로운 패킷이 도착할 때마다 큐가 비어있는지의 여부를 파악하여 큐가 비어있지 않으면

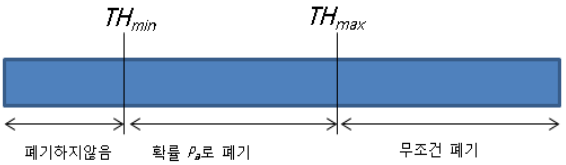

그림 1. RED 버퍼
Fig. 1. RED buffer.

평균 큐 길이 Q_{avg}는 EWMA(exponential weighted moving average)를 통한 low-pass filter를 사용하여 과 같이 계산한다.

$$Q_{avg} \leftarrow (1-w_q)Q_{avg} + w_q Q_{size} \tag{1}$$

Q_{avg} : average queue size
w_q : queue weight
Q_{size} : current queue size

평균 큐 길이(Q_{avg})와 두 임계 값을 그림 1과 같이 비교한다. Q_{avg}가 하한 값 $TH_{\min}$보다 작으면 체증이 없는 것으로 가정하여 이 패킷을 큐에 넣는다. Q_{avg}가 상한 값 $TH_{\max}$보다 크면 체증이 발생하였다고 간주하고 패킷을 버린다. Q_{avg}가 하한 값 $TH_{\min}$와 상한 값 $TH_{\max}$에 있을 때는 확률 P_a로 패킷을 폐기하고 확률$(1-P_a)$로 큐에 들어가게 한다. 기존 RED 알고리즘이 패킷을 폐기하는 방법은 식1과 식2를 이용하여 확률 P_a를 적용하여 패킷을 처리한다.

$$P_a = \frac{P_b}{1 - count \times P_b} \tag{2}$$

$$\left(F = \frac{Q_{avg} - TH_{\min}}{TH_{\max} - TH_{\min}}, \quad P_b = F \times P_{\max} \quad 0 \le F \le 1 \right)$$

식2의 $count$는 초기 값은 -1로 하고 확률을 적용하여 패킷을 패기하면 0으로 설정하고, 확률을 적용하더라도 패기 되지 않으면 1씩 증가시키는 변수이다. 즉 마지막 폐기 후 들어온 패킷수가 된다. 패킷이 도착하지 않는 idle 기간 동안 식3을 이용하여 Q_{avg}를 계산하고 큐가 비어 있는 시간을 고려한다.

$$m \leftarrow f(time - q_{time})$$
$$Q_{avg} \leftarrow (1-w_q)^m Q_{avg} \tag{3}$$

($time$: current time q_{time} : start of queue idle time,
$f(t)$: a linear function of time t)

2-2 RED 파라미터에 대한 동작특성

RED에서 정의되어 있는 4개의 파라미터는 가중치(w_q), 최소 큐 한계 값($TH_{\min}$), 최대 큐 한계 값($TH_{\max}$), 최대 확률($P_{\max}$)이 있으며 이 파라미터들의 특성은 아래와 같다.

1) 큐 가중치(w_q)와 평균 큐 길이(Q_{avg})

평균 큐 길이(Q_{avg})를 구할 때 실제 큐 길이가 Q_{avg}가 변하는데 미치는 영향을 나타내는 값이 큐 가중치(w_q)이다. w_q가 작으면 작을수록 현재의 큐의 실제 길이가 Q_{avg}가 변하는데 미치는 영향이 작게 된다. 이런 경우에는 짧은 시간 동안 큐의 길이가 증가되더라도 Q_{avg}가 증가하는데 미치는 영향이 작기 때문에 일시적인 트래픽의 증가로 인한 오류를 피할 수 있다.

w_q는 0.001이상을 쓰도록 권하고 있고 일반적으로 0.002를 사용하고 있다.

2) 최대 확률($P_{\max}$)

$P_{\max}$의 값을 조절할 경우 가장 영향을 받는 부분은 Q_{avg}이다. $P_{\max}$가 커질 경우에 패킷 폐기 확률이 커져 폐기하는 횟수가 많아지므로 실제 큐 길이나 Q_{avg}가 전체적으로 줄어들게 된다. 망의 부하가 증가하면 $P_{\max}$ 값을 높여 주여야 한다[5].

3) 최소 큐 한계 값($TH_{\min}$), 최대 큐 한계 값($TH_{\max}$)

실제 망에 RED를 적용했을 때 $TH_{\min}$과 $TH_{\max}$ 값을 설정하는 것은 상당히 중요한 일이지만, 이것은 수학적으로나 시뮬레이션만으로 정할 수 있는 파라미터는 아니다. [4]에서 이 파라미터의 범위를 다음과 같이 정하고 있다.

트래픽이 버스트한 특성을 가질수록 $TH_{\min}$는 크게 주어 링크의 사용효율을 유지할 수 있도록 해준다. 그러나 정상적인 환경에서는 $TH_{\min}$는 가급적 작게 주는 것이 유리하다 $TH_{\max}$는 Drop Tail의 최대 버퍼크기와 비슷하게 동작하므로 이 파라미터의 값이 클 경우는 큐 길이가 전제적으로 높게 유지되기 때문에 큐잉 지연시간(queueing delay time)이 길어진다. $TH_{\min}$와 $TH_{\max}$의 차이는 왕복시간(round trip-time)동안에 계산되는 평균 큐길이의 추정치보다 커야 RED 라우터는 효율적으로 동작할 수 있다. 대략 $TH_{\max}$는 최소한 $TH_{\min}$의 두 배 이상으로 할 것을 권고하고 있다[4].

4) RED 파라미터의 적용의 문제점

위에서 설명한 바와 같이 RED의 4가지 파라미터들은 그 값을 어떻게 정하느냐에 따라 RED의 성능에 중대한 영향을 미친다. 하지만 다양한 트래픽 상황에서도 범용적으로 적용할 수 있는 파라미터를 결정하는 것은 매우 어렵다[6].

이 파라미터의 최적 값은 망의 트래픽 상황에 따라 달라지기 때문에 결국은 트래픽 상황에 변화에 따라 그 값을 변화 할 수 있어야 할 것이다. RED의 파라미터를 고정시켜놓고 호스트의 수를 증가시키면 어느 순간에는 효율이 극히 나빠지는 상황이 발생할 수 있으며 그 이유는 트래픽의 버스트 기간이 증가됨으로써 심한 체증 현상을 초래하게 되며 이러한 트래픽 상황에 대해서 고정적으로 정한 RED파라미터가 원래의 의도대로 동작하지 못하기 때문이다.

또한 이러한 상황에 맞도록 파라미터의 값을 정하게 되면 트래픽부하가 적을 경우에 throughput에 영향을 주어 성능이 나빠진다.

위의 내용을 간추리면 RED에서는 4개의 파라미터 등을 고정된 값으로 사용하기 때문에 지속적으로 트래픽의 버스트한 상황일 때는 RED가 제대로 기능을 발휘하지 못하는 상황이 발생할 수 있으며 이를 보완하기 위해서는 파라미터의 값을 망의 상태에 따라 조절하여 적응적으로 반응할 수 있도록 하여야 함을 알 수 있다.

Ⅲ. 제안 알고리듬

3-1 RED의 문제점

RED의 고정된 파라미터들은 적정한 트래픽부하일 경우에 잘 적용되도록 정해졌기 때문에 위에서 지적한 트래픽 부하가 지속적으로 가해졌을 때도 문제가 발생하지만 낮은 경우에도 개선의 여지가 있다. 4개의 파라미터 중 $TH_{\min}$의 값은 트래픽 부하가 버스트한 경우 링크 이용률을 높게 유지하기 위해 사용되어지며, 높게 할수록 버스트한 트래픽 상황일 경우에 좋은 링크 이용률을 유지한다.

그러나 이러한 $TH_{\min}$을 높은 트래픽부하일 때에도 동일하게 적용된다면 링크 이용률에는 향상이 없고 불필요한 평균 큐의 크기만 증가시키게 되므로 이 경우에는 $TH_{\min}$은 작게 유지되어야 한다. 또한 지속적으로 과부하가 걸릴 경우에는 $P_{\max}$의 값을 증가시켜 폐기하는 횟수가 많아지도록 하여 실제 큐 길이와 Q_{avg}을 전체적으로 줄어들도록 하여야 하나 RED에서는 불가능 하다[6].

3-2 제안 알고리듬

위에서 기술한 RED의 문제점들은 망의 트래픽 부하 상태를 감지할 수 없다는 특성 때문에 야기되는 문제이다. 본 논문에서는 지속적인 트래픽 상태를 감지할 수 있는 방법을 제시하고 이를 이용하여 최소 큐 한계 값($TH_{\min}$)과 최대 확률($P_{\max}$)을 망 상태에 따라 자동적으로 반응하는 확장된 RED 알고리듬을 제안한다.

1) 트래픽상태 감지방법제시

일반적으로 사용되는 w_q=0.002를 1/20이하로 감소시킨 w_L을 이용하여 식 4와 같이 새로운 값인 long-term 가중평균을 계

산하여 Q_{Long}라 하고 망 트래픽 상태의 척도로 사용한다.

$$Q_{Long} \leftarrow (1-w_L)\times Q_{Long} + w_L \times Q_{size} \quad (4)$$

단, $w_{Long} < \frac{w_q}{20}$

w_q : 가중평균을 구할 때의 가중치

w_L :Long-Term 가중평균을 구할 때의 가중치

이 값이 $TH_{\max}/\beta$일 부근에 있을 경우를 정상적인 상태, 클 경우를 과부하상태, 작을 경우는 낮은 부하상태라 판단한다. 여기서 β는 정상상태의 기준을 결정하는 파라미터로 망의 특성에 따라 달라질 수 있다. 작은 큐잉 지연을 요구되면 크게 하고, 큐잉 지연보다 좋은 링크 이용률을 원한다면 작은 값을 선택할 수 있다. 본 논문에서는 $\beta = 2$로 설정하여 모의실험을 하였다.

2) 최소 큐 한계 값($TH_{\min}$) 결정

최소 큐 한계 값은 적은 망 부하 상태일 때는 높게, 높은 망 부하상태일 때는 낮게 설정할 수 있도록 식 5을 이용하여 구한다.

$$TH_{\min} = \alpha \times (\frac{TH_{\max}}{\beta} - Q_{Long}) + \gamma \quad (5)$$

α : 부하가 적을 때 burst data에 대한 응답을 결정하는 파라미터

γ : 적당한 망부하일 때의 $TH_{\min}$

여기서 α와 γ 값은 최소 큐 한계 값을 결정하는 파라미터로 본 논문에서는 $\alpha = 5, \gamma = 5$로 설정하여 모의실험을 하였다.

3) 최대 확률($P_{\max}$) 결정

망의 트래픽상태에 따라 $P_{\max}$ 값을 식 6과 같이 결정한다. 망의 부하가 적을 경우에는 낮게 하고 많을 경우에는 높게 하여 다양한 망의 환경에 자동적으로 적응할 수 있도록 결정한다.

$$P_{\max} \leftarrow P_{\max} \times (\delta^{Par} - \delta + 1) \quad (6)$$

$$Par \leftarrow \frac{Q_{Long}}{TH_{\max} - Q_{Long} \times (\beta - 1)}$$

δ: 부하가 많아질 때 $P_{\max}$를 변화시키는 파라미터

δ은 최대 확률을 결정하는 파라미터이며 $(\delta^{Par} - \delta + 1)$는 Q_{Long}가 $TH_{\max}$에 가까워질 때 $P_{\max}$를 급격히 증가시키기 위해 사용되며 본 논문에서는 $\delta = 5$ 로 설정하여 모의실험을 하였다.

3) 제안 RED 알고리즘

Initialization:

$Q_{avg} \leftarrow 0$

$Q_{Long} \leftarrow TH_{\max}/\beta$

$P_{\max} \leftarrow P_{inini}$

$count \leftarrow$-1

$TH_{\max} \leftarrow 60$

$TH_{\min} = \alpha \times (\frac{TH_{\max}}{\beta} - Q_{Long}) + \gamma$

for each arrival

if the queue is nonempty

$Q_{avg} \leftarrow$(1-w_q)$\times Q_{avg} + w_q \times Q_{size}$

$Q_{Long} \leftarrow$(1-w_L)$\times Q_{Long} + w_L \times Q_{size}$

else

$m \leftarrow f(time - q_{time})$

$Q_{avg} \leftarrow (1-w_q)^m \times Q_{avg}$

$Q_{Long} \leftarrow (1-w_L)^m \times Q_{Long}$

calculate new $TH_{\min}$

$TH_{\min} = \alpha \times (\frac{TH_{\max}}{\beta} - Q_{Long}) + \gamma$

if $TH_{\min} < Q_{avg}$

if $TH_{\min} < Q_{avg} < TH_{\max}$

increment $count$

calculate P_a

$Par \leftarrow \frac{Q_{Long}}{TH_{\max} - Q_{Long} \times (\beta - 1)}$

$P_{\max} \leftarrow P_{\max} \times (\delta^{Par} - \delta + 1)$

$P_a = \frac{P_b}{1 - count \times P_b}$

($F = \frac{Q_{avg} - TH_{\min}}{TH_{\max} - TH_{\min}}$,

$P_b = F \times P_{\max}, \quad 0 \leqq F \leqq 1$)

mark the arriving packet with P_a

$count \leftarrow 0$

else if $TH_{\max} < Q_{avg}$

mark the arriving packet

$count \leftarrow 0$

else $Q_{avg} < TH_{\min}$

Queue Packet

when Queue become empty

$q_{time} \leftarrow time$

※ Saved Variables

Q_{avg} : average queue size

Q_{Long}:Long term average size

$count$: packets since last marked packet

$P_{\max}$: maximum value

$TH_{\min}$: minimum threshold for queue

※ Fixed parameters

w_q : queue weight

w_L : long term queue weight

α, β, γ, δ : parameter setting $P_{\max}$ and $TH_{\min}$

$TH_{\max}$:maximum threshold for queue

P_{inini}:the initial value of $P_{\max}$

※ Other

P_a : current packet-marking probability

Q_{size} : current queue size

$time$: current time

q_{time} : start of queue idle time

$f(t)$: a linear function of time t

Ⅳ. 모의실험 결과 및 분석

4-1. 모의실험 환경

본 논문에서 사용한 network 환경은 그림 2와 같이 구성하였으며 다음과 같은 전제조건을 두었으며, 일반적인 사항은 고려하지 않았다.

① 모든 링크의 대역폭은 100 Mbps
② 링크와 gateway 사이의 delay time은 1, 2, 3, 5, 4, 4 … msec으로 반복적용
③ sink 노드와 router 사이의 delay time : 2 msec

모의실험은 링크 수를 늘여가면서 gateway 형태를 RED와 제안 확장 RED의 방식을 비교하였으며, 모의실험 시간은 10 초, gateway 큐 길이는 100 패킷, 1 패킷은 1000 byte로 하였고 ack packet은 40 byte로 하였다.

체증인지 방법은 gateway가 패킷을 폐기하였다는 것을 링크 송신노드가 10 msec 지나서 알 수 있도록 하였다. 각 호스트들은 자신이 전송한 패킷이 체증의 영향으로 폐기되었을 경우 TCP의 체증제어 방식인 slow start와 congestion avoidance를 행하게 된다. 그리고 송신 호스트에서 수신 호스트로 단 방향 전송만을 가정하였다.

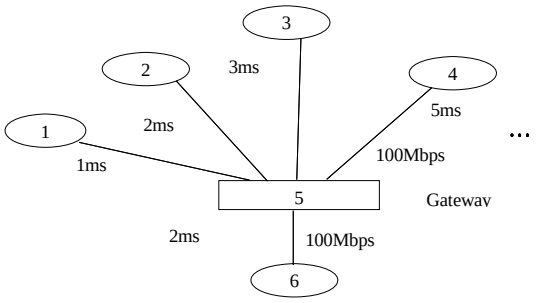

그림 2. 모의실험 네트워크
Fig. 2. Simulation network.

사용되어진 파라미터들은 RED인 경우 $P_{\max}$는 0.02, w_q는 0.002, $TH_{\max}$는 60, $TH_{\min}$는 9, 패킷 크기는 1000, 윈도우 크기는 1024를 사용하였다.

제안 알고리듬의 w_q는 0.002, w_L는 0.00004, P_{inini}는 0.02, α는 5, β는 2, γ는 5, δ는 5로 설정하고, $TH_{\max}$ 값이 50, 60, 70에 대하여 모의실험을 하였다.

4-2. 모의실험결과 및 분석

버퍼에 연결된 노드수를 5개에서 80개로 변화시키면서 제안 알고리듬과 RED의 시뮬레이션을 하였으며, 링크 이용률은 전송 가능한 최대 전송 패킷 수와 실제 전송한 패킷 수와 비율이고, 평균 큐 길이는 패킷이 도착할 때의 큐 길이의 평균으로 나타내었다. 평균 큐 길이는 도착한 패킷이 처리될 때까지의 대기시간과 같으므로 가능한 작아야하고 링크이용률은 가능한 100%이어야 좋은 성능이라 할 수 있다.

TCP망에서는 연결 노드수가 적을 경우는 데이터의 전송은 일반적으로 버스트한 특징이 나타난다. 모의실험결과 연결노드수가 5개인 경우를 살펴보면 기존 RED의 링크이용률이 70%인 반면 $TH_{\max}$= 60인 제안 알고리듬은 85.7%의 링크사용효율을 보여주고 있어 제안 알고리듬이 버스트한 망에서 링크효율이 약 15.7%의 향상됨을 그림3을 통해 알 수 있다.

지속적으로 과부하가 가해지는 경우는 연결노드수가 많을 경우이며, 모의실험결과에서 연결 노드수가 50개인 경우를 보면 RED의 결과는 링크효율 99.7%, 평균 큐길이 55.0이며, $TH_{\max}$= 60인 제안 알고리듬의 결과는 링크효율 99.7, 평균 큐 길이 37.9로 링크효율의 저하 없이 평균 큐 길이를 31%이상 감소시킬 수 있음을 그림5를 통해 알 수 있으며 이는 제안 RED 알고리듬이 지속적인 과부하가 걸릴 경우에도 효과적으로 동작하고 있음을 알 수 있다.

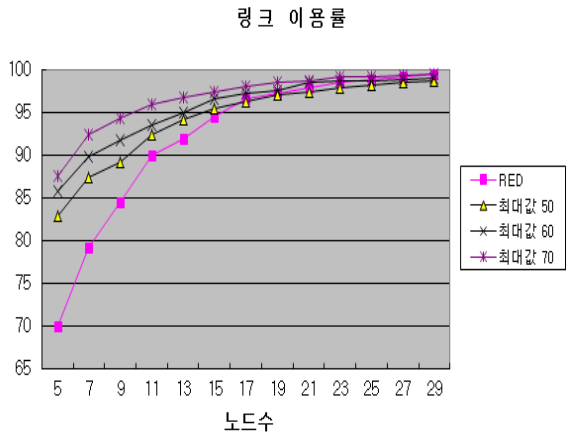

그림 3. 링크 이용률
Fig. 3. Link utilization.

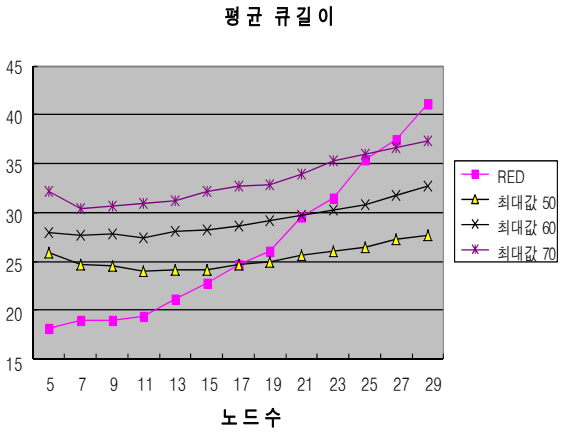

그림 4. 평균 큐길이
Fig. 4. Average queue length.

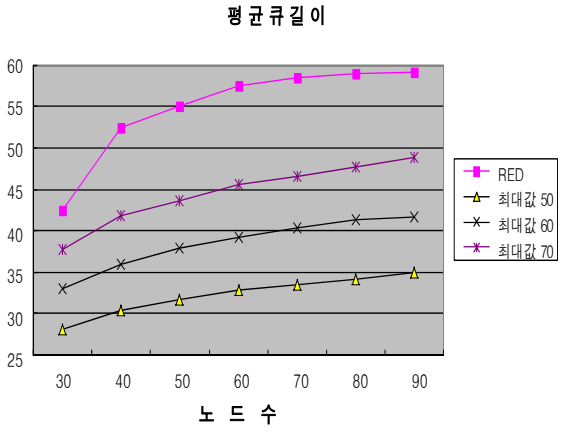

그림 5. 과부하일 경우의 평균 큐길이
Fig. 5. Average queue length when overloaded.

전체적인 모의실험결과는 제안 알고리듬은 버스트한 경우는 평균 큐 길이는 약간 증가하지만 많은 링크효율의 향상을 보여주고 있으며, 지속적인 과부하가 걸릴 경우에는 링크효율이 저하 없이 평균 큐 길이를 감소시킬 수 있음을 보여주고 있어 기존 RED에서 지적되는 문제점이 해결됨을 모의실험을 통하여 확인할 수 있다.

그림 3은 링크이용률, 그림 4는 평균 큐의 길이의 결과비교를 나타내었고, 그림 5는 과부하로 생각되는 상태의 망의 평균 큐의 길이를 나타내었다. 과부하일 경우, RED의 평균 큐 길이는 $TH_{\max}$의 값으로 수렴하여 drop tail과 유사하게 동작하여 RED의 기능을 상실하지만 제안 알고리즘은 과부하가 걸리더라도 RED의 기능을 하고 있음을 볼 수 있다. 또한 제안 알고리즘은 평균 큐의 길이가 $TH_{\max}/\beta(\beta$=2로 설정)근처에 있음을 보여주고 있어 β값을 조정함으로써 평균 큐의 길이를 조절될 수 있음을 알 수 있다.

Ⅴ. 결론

본 논문에서는 RED 알고리듬의 문제점을 제시하고 문제점을 보완하고 확장할 수 있는 방안을 제시하였다.

RED 알고리듬은 버스트한 데이터의 특성을 가질 경우에는 링크이용률이 낮으며, 과부하가 걸릴 경우에는 큐 관리를 못하여 평균 큐의 길이가 $TH_{\max}$의 값으로 수렴하는 문제점이 있다.

본 논문에서는 long-term 가중평균을 이용하여 지속적인 망 상태를 검색하고, 이러한 가중평균을 최소 큐 한계 값($TH_{\min}$)과 최대 폐기확률($P_{\max}$) 결정에 활용함으로써 부하의 변동이 심한 인터넷 환경에 잘 부합될 수 있는 확장 RED를 제안하였다.

제안 확장 RED의 설계결과는 기존 RED와 비교하여 볼 때, 버스트한 데이터의 특성을 가질 경우 링크 이용률을 증가시켰으며 부하가 많을 때에는 링크이용률의 저하 없이 평균 큐의 길이를 감소시켰고, 특히 부하가 아주 많은 경우에, 큐의 관리가 기존 RED에서는 불가능하였으나 제안 확장 RED는 적절한 관리가 가능함을 보여주었다. 또한 평균 큐의 관리도 가능하여 다양한 환경에서 라우터에서의 지연시간도 적절히 제어할 수 있음을 보여주었다.

향후 연구과제로는 새로이 제안된 확장 RED에서 사용된 파라미터에 대한 연구가 다양하게 이루어져야 할 것이다.

참고문헌

[1] V. Jacobson, “Congestion avoidance and control,” *Computer Communication Review*, Vo1. 18 No. 4, pp 314-329, Aug. 1988.

[2] W. Steven, Standard: TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms, Internet Engineering Task Force (IETF), RFC 2309, 1997.

[3] B. Braden et al., Recommendations on queue management and congestion avoidance in the internet, Internet Engineering Task Force (IETF), RFC2309, 1998.

[4] S. Floyd and V. Jacobson, “Random early detection gateways for congestion avoidance,” *IEEE/ACM Transaction on Networking*, Vo1. 1 No. 4, pp. 397-413, Aug. 1993.

[5] Y. S. Yoo, “Analysis of the parameters of the random early detection (RED) algorithm for congestion avoidance,” *The Journal of Korea Information and Communications Society*, Vo1. 16 No.3, pp. 41-44, Jan.1997.

[6] S. Woo, Stability analysis of random early detection active queue management scheme for congestion control in next-generation networks, Ph. D. dissertation, Gwangju Institute of Science and Technology, Gwangju: Korea, Feb, 2011.

김 동 춘(Dong-Choon Kim)

1993년 2월 : 제주대학교 전자공학과(공학사)
1995년 2월 : 연세대학교 대학원 전자공학과(공학석사)
2002년 2월 : 제주대학교 대학원 전자공학과(박사수료)
1997년 2월~현재 : 한국폴리텍대학 강릉캠퍼스 전자통신학과 교수
※관심분야 : 컴퓨터네트워크, 네트워크 토폴로지 설계분야