정체 라우터에서의 선별적 처리 알고리즘

이상민[0] 채현석* 최명렬
한양대학교 전자전기제어계측공학과, 동원대학 인터넷정보과*
sangmin@asic.hanyang.ac.kr, hschae@tongwon.ac.kr*, choimy@asic.hanyang.ac.kr

A Selected Processing Algorithm at the Congested Router

Sang-Min Lee[0] Hyun-Seok Chae* Myung-Ryul Choi
Dept. of EECI, Hanyang University, Dept. of Internet Information & Retrieval

요 약

최근 라우터에서는 정체를 회피하고 전송률을 향상시키기 위한 능동적 큐 관리와 패킷 스케줄링에 대한 많은 논의가 이루어지고 있다. 본 논문은 라우터에서의 전송률 향상을 위한 Random Early Detection (RED) 알고리즘과 최근까지 변형된 RED알고리즘들의 특징을 살펴보고, RED라우터에 적용하여 실제로 종단 호스트(End-to-end)에서 전송 받는 패킷의 양을 향상하기 위한 선별적 처리 알고리즘을 제안한다.

1. 서 론

현재 컴퓨터 네트워크의 중간에서 패킷의 전송과 큐 관리를 담당하는 한정된 버퍼를 갖고 있는 라우터에서의 정체는 패킷의 지연을 증가시키고, 재전송에 의한 대역폭의 낭비를 초래하여 트래픽의 성능을 저하시키는 심각한 문제를 내포하고 있다. 이러한 문제점을 인식하여 라우터의 정체를 회피하기 위해 라우터에서의 능동적 큐 관리와 패킷 스케줄링에 대한 많은 연구가 진행되고 있다.

RED 알고리즘과 변형된 RED 알고리즘들은 한정된 라우터 버퍼에 도착하는 패킷을 로우 패스 필터(low-pass filter)를 이용한 지수적 가중치 평균 이동인 평균 큐 길이(Q_{avg})를 RED에서 정의한 중요한 4개의 파라메터(W_q, MIN_{th}, MAX_{th}, MAX_p)에 따라 망의 체증을 결정한다.[1][2] 이에 따라 RED는 도착하는 패킷을 임의로 확률적 폐기하여 라우터의 큐 길이를 작게 하고 패킷 전송의 지연을 방지한다.

RED와 변형된 RED를 사용 할 경우 Drop-Tail 방식과 비교하여 링크의 대역폭 사용률과 패킷 전송률의 향상을 보이는 많은 논문이 있고, RED를 다양한 스케줄링에 연동하게 하기 위한 연구들이 진행 중이다.

본 논문에서는 네트워크의 종단 호스트간에 전송되는 트래픽 성능을 RED와 변형된 RED, 그리고 기존의 베스트 에포트 방식(Best-effort)라우터 각각에 대하여 비교분석한다. 또한 RED를 사용하는 라우터에 다양한 시나리오를 적용하여, 실제로 종단간에 전송되는 패킷의 양을 향상시키는 선별적 처리 알고리즘을 제안하고, RED를 사용한 시뮬레이션을 통하여 성능을 살펴본다.

2. 선별 처리 알고리즘

2.1. RED와 변형 RED 알고리즘

능동적 큐 관리를 위한 연구들 중 RED는 망의 체증이 발생하여 라우터에 도착하는 패킷을 전부 폐기하기 전에 Q_{avg} 를 이용하여 체증을 회피하고, 송신 호스트의 TCP에 체증 가능성을 미리 통보하여 송신 호스트들의 TCP가 체증 회피 단계로 들어가게 한다.

이와 같은 RED의 목적은 체증 발생시 송신 호스트들의 재전송을 방지하여 링크의 대역폭 낭비를 줄이고, 체증으로 인한 송신 호스트들의 급격한 윈도우 사이즈 줄임 현상에 의한 포괄적 동기화(global synchronization)을 방지하여 대역폭의 활용을 최대로 유지하게 한다.

RED에서 기본적으로 제시한 개념은 크게 Q_{avg} 를 작게 유지하며, 로우 패스 필터를 이용하여 시간 함수를 관리하고 패킷에 적절한 체증 표시를 하는 것이다. 이를 바탕으로 RED에서는 Q_{avg} 로 체증을 결정하기 위한 4개의 파라메터를 제시하였다.

그러나 초기의 RED는 동일한 패킷 폐기 확률을 사용함으로써 종단 호스트들이 링크를 공평하게 사용하지 못하는 경우가 발생할 수 있으며, 접속 노드의 급격한 증감이나 버스트 트래픽에 대하여 제대로 반응하지 못하는 단점이 있다.[3] 이에 따라 여러 가지 변형된 RED알고리즘이 제안 되었으며, 최근에 RED를 개념을 제시한 Sally Floyd는 이러한 약점들을 수렴하고 스케줄링과 연동하기 위한 적응성 RED(RED-PD)를 제안중이다.[4]

2.2. 선별 처리 알고리즘

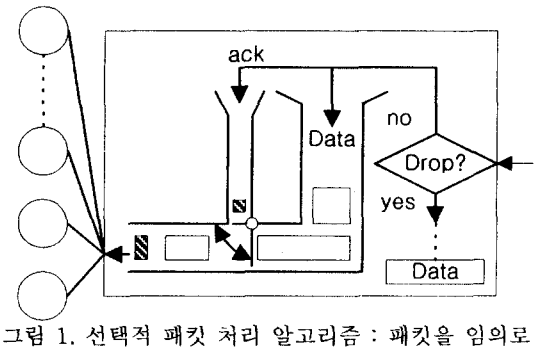

그림 1. 선택적 패킷 처리 알고리즘 : 패킷을 임의로 폐기하기 전에 ACK 패킷과 DATA 패킷으로 구분한다.

본 논문은 RED라우터가 컴퓨터 네트워크의 중간 단에서 전송률을 최대로 활용하면서 종단간 호스트에서 실제적으로 전송되는 패킷의 양을 향상시키기 위한 알고리즘(그림 1, 그림 2)을 제안한다. 본 알고리즘은 기존의 RED라우터에서 임의로 폐기 될 수 있는 패킷들 중, 응답메시지(ACK) 패킷과 같은 TCP의 흐름제어에 영향을 주는 패킷들은 따로 관리하여 폐기되지 않고 전송함으로써 네트워크 정체시 전송효율을 높일 수 있다.[5]

```
for each newly arrival packet, decide the packet
is ACK packet or DATA packet

if (the arriving packet == ACK packet) {
  if (the line is sending DATA or ACK packet)
    transmit ACK packet after sending packet
  else
    transmit ACK packet
}
```

그림 2. 선별 처리 알고리즘

응답메시지 패킷과 데이터 패킷이 다른 변수의 영향 없이 안전히 송수신지 간에 전송된다고 가정하면, 응답메시지 전송이 늦어짐에 따른 재전송이나 대기 시간이 길어지는 것을 방지함으로써 응답메시지 패킷의 처리 시간을 계산할 수 있다.[6]

그림 3의 R2와 R3간에 더 많은 N개의 라우터가 있고, 각 라우터에서 대기하게 되는 큐 버퍼의 크기를 queue-buffer(Kbytes),각 큐 버퍼에서 시간당 처리되는 큐를 t(kbyte/s), 버퍼에 쌓여 있는 큐를 queue-stored, 전송되는 응답메시지 크기가 ack-size(kbytes)일 때, 도착되는 ACK를 먼저 처리하고 각각의 노드에 큐 버퍼의 절반에 패킷이 쌓여 있다고 가정하면, 응답 메시지의 지연은

$$T_{all} = \left(\frac{queue - buffer}{t} \right) \times N_{node} \qquad (1)$$

과 같이 되고, 만약 선택적 처리 알고리즘을 적용하면

$$T_{ack} = \left(\frac{ack - size}{t} \right) \times N_{node} \qquad (2)$$

와 같이 되어서 실제로 지연되는 시간인

$$T_{save} = T_{all} - T_{ack} = \left(\frac{queue_{stored}}{t} \right) \times N_{node} \qquad (3)$$

만큼 대기 시간을 줄일 수 있고, 수식에는 없지만 임의의 ACK패킷 폐기를 방지 할 수 있게 되어 재전송의 시간은 단축되어진다.

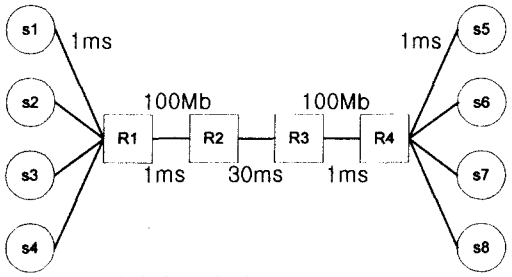

그림 3. 모의실험 구성 망

RED의 4개의 파라메터 값을 어떻게 정하느냐 혹은 동적으로 변화를 주느냐에 따라 RED의 성능에 중대한 영향을 주고, 그러한 파라메터를 어떠한 네트워크 망에서 시뮬레이션 하는가에 따라 그 결과도 상이하게 나타난다. 이에 본 논문은 기존 논문이 제시하고 있는 네트워크 망의 기본이 되는 그림 3 과 같은 구성을 기본으로 리눅스 Kernel 2.4.2에서 ns-2 시뮬레이터로 모의 실험 한다.

현재 변형된 RED알고리즘들에서 보여주듯이 어떠한 트래픽 상황에서도 적용되고 반응하는 파라메터를 찾기 어렵고, 따라서 Sally Floyd가 제시하고 있는 RED의 매개변수의 수정이 적절히 이루어 졌다고 가정한다.

출발노드(S1~S4)와 목적지노드(S5~S8)에 사용되는 TCP버전은 가장 보편적이며, 일반적인 Slow-start, 정체 제어(Congestion control), 빠른 재전송(Fast retransmit), 빠른 회복(Fast recovery)을 추가한 Reno의 문제점을 향상시킨 NewReno와 Reno의 재전송, 정체 제어, 느린 출발, 시간경과(Time out)을 수정한 Vegas알고리즘을 선택하여 모의실험 한다.

왼쪽 4개의 노드의 TCP버전은 처음에 NewReno 4개에서 매 실험시 NewReno를 Vegas로 변화를 주었고, 오른쪽의 4개의 노드는 모두 NewReno 혹은 모두 Vegas와 이들의 조합으로 한다. 각 링크속도는 매 실험시 변화를 주어 RED R2와 R3에서 정체가 일어나게 하였다.

본 논문에서는 ACK 와 DATA 패킷으로만 구분을 하였으나, RFC2309에서 제시하고 있는 무응답(unresponsive) 흐름들이나 TCP와 호환 되지 않는 흐름들과 같은 패킷에 대한 처리를 구분하고 처리하여 좀 더 지능적으로 개선해야 할 부분들이 있다.

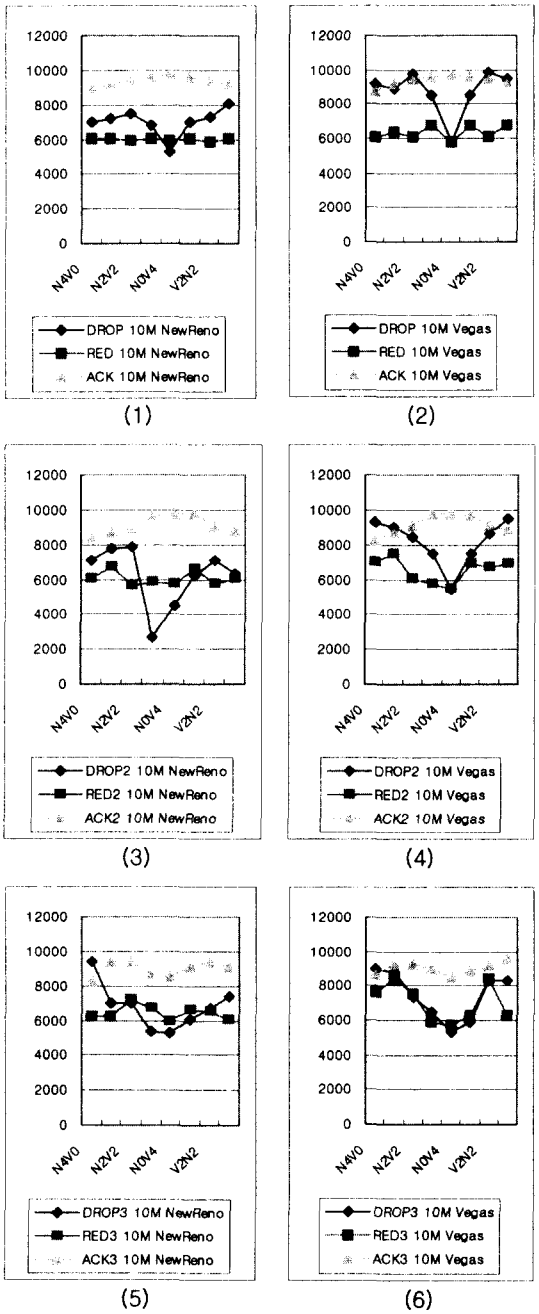

그림 4. RED와 선택 처리 알고리즘을 적용한 RED의 성능 비교 : (1)(3)(5)은 목적지의 노드가 모두 NewReno일 때, (2)(4)(6)은 목적지의 노드가 모두 Vegas일 때

차후 RED에서 흐름들에 대한 지능적인 파라메터와 선택적인 패킷 폐기와 조절을 통하여 전송률을 향상 시키고 스케줄링과 연동 된다면, 종단에 연결되는 TCP 버전이 갖고 있는 특성에 따라 적절한 라우터에서의 트래픽 관리가 이루어지고 실제로 종단에서 처리되는 패킷의 양을 향상 시킬 수 있다.

3. 결 론

본 논문은 종단간에 실제로 전송되는 데이터의 전송률을 향상하기 위한 방법을 제시하였다. 선택적 처리 알고리즘은 TCP 프로토콜이 갖고 있는 기본 개념에 부합하지 않으며, RED와 같은 정체제어 알고리즘이나 RED에 스케줄링 기법들을 이용한 변형된 RED에서 아무런 문제 없이 효율적으로 적용됨을 보았다. 제안한 알고리즘은 변화가 많은 종단간의 시간경과를 추정하는 것이 아니라 라우터에서 선별적으로 패킷을 처리하고 정렬하는 알고리즘이다. 즉, 현재는 응답 메시지 패킷이 도착하면 폐기되지 않고 데이터 패킷보다 먼저 처리한다.

향후의 연구 과제로는 개선되어진 RED와 같은 능동적 큐 관리와 스케줄링과 연동되어진 라우터에서 각 노드에 대한 특성을 파악하고 패킷 버림 확률에 영향을 받지 않고 선택되어지고 처리되는 패킷의 분석과 차후 컴퓨터 네트워크의 다양한 서비스에 지능적으로 대처 하는 것이다.

참 고 문 헌

[1] Sally Floyd and Van Jacobson. "Random Early Detection Gateways for Congestion Avoidance". IEEE/ACM Transactions on Networking, vol. 1 no.4, pp. 397-413, August 1993.

[2] http://www.aciri.org/floyd/red.html

[3] May, M.;Bolot, J.;Diot, C.;Lyles, b., "Reasons not to deploy RED", IWQoS '99. 1999 Seventh International Workshop on, pp. 260-262, 1999.

[4] Sally Floyd, Ramakrishna Gummadi, and Scott Shenker, "Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management", August 1, 2001.

[5] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the internet", IEEE/ACM Transactions on Networking, vol. 7, no. 4, pp. 458-472, 1999.

[6] Jing WU, Tao DU, Shiduan CHENG, Jian MA. "Using Random Early Detection to ACK Delay Control of TCP Traffic". Fourth Optoelectronics and communications Conference. pp 166-170, 1999.